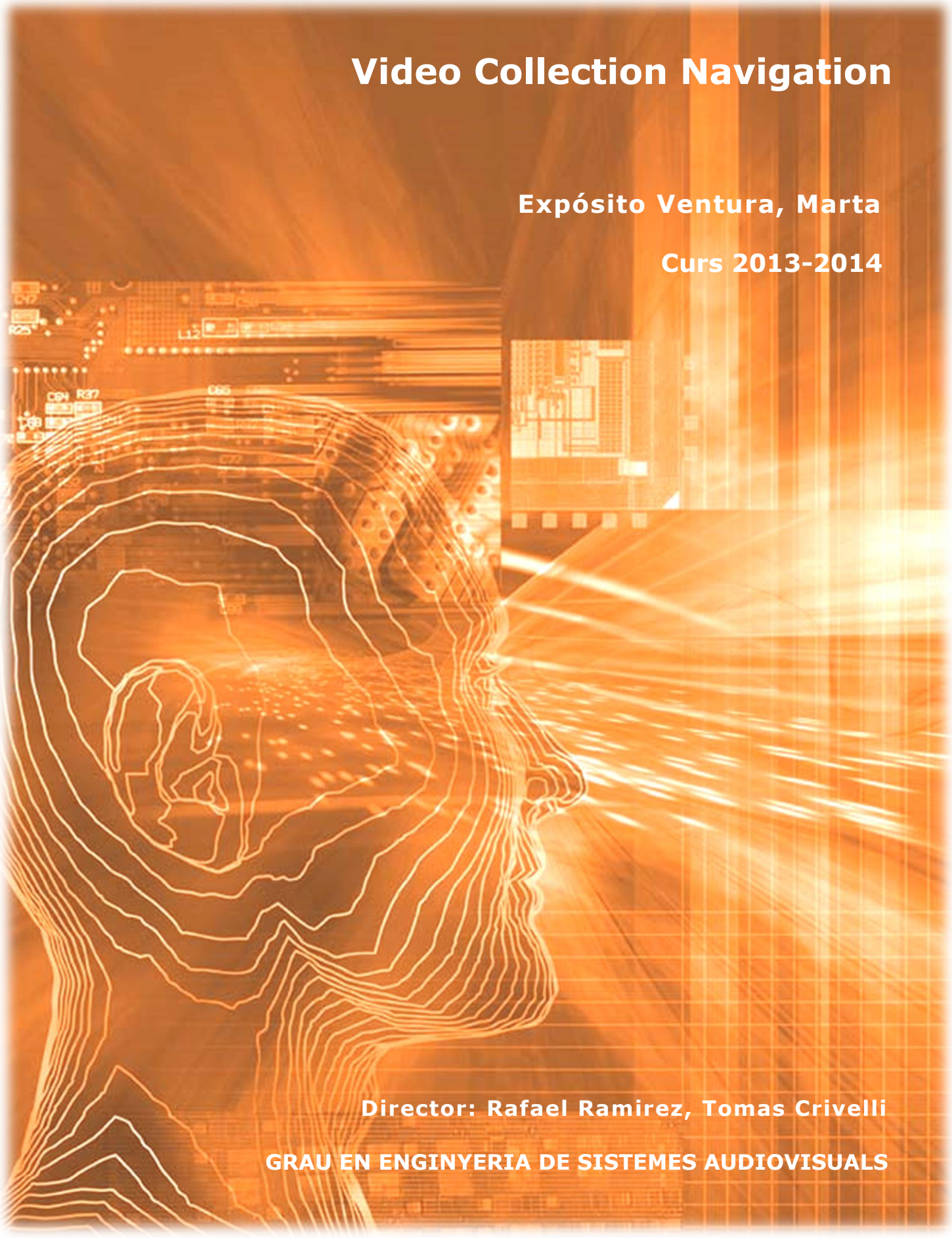


Video Collection Navigation

Expósito Ventura, Marta

Curs 2013-2014

Director: Rafael Ramirez, Tomas Crivelli

GRAU EN ENGINYERIA DE SISTEMES AUDIOVISUALS

upf. Universitat Pompeu Fabra Barcelona
Escola Superior Politècnica

Treball de Fi de Grau

About Technicolor

Technicolor, a worldwide technology leader in the media and entertainment sector, is at the forefront of digital innovation. Our world class research and innovation laboratories and our creative talent pool enable us to lead the market in delivering advanced services to content creators and distributors. We also benefit from an extensive intellectual property portfolio focused on imaging and sound technologies, supporting our thriving licensing business.

About Technicolor Rennes – A pole of excellence in creation, distribution and content access technologies.

Technicolor Rennes is the largest technology centre of Technicolor Group. Pole of excellence in the whole image chain, the Rennes centre is the supplier of innovative solutions for Media & Entertainment Industries. A part of the activities is focused on the compression, protection, transmission, production and management of high definition contents while other technological solutions are developed to respond to the demands of network operators and to meet the needs of Media & Entertainment customers. Technicolor Rennes has built partnerships with top academic institutes in Europe, public institutes, industrial partners and is involved in all French and European research programs such as RNRT, RNTL, RIAM, MEDEA, ITEA, IST, CNES, ESA …

Research and Development themes addressing Technicolor Group strategy

- Compression/Decompression with audio/video processing
- Signal and image acquisition and processing
- Mobile and broadband technology
- Content security on the whole image chain
- 3D
- Licensing
- Digital decoder platforms including the most recent technologies such as Personal Video Recorder, MPEG4 coding, video over Internet Protocol, high definition…

Activities

Technicolor's activities are organized into three operating divisions:

- Technology
    - Research & Innovation
    - Licensing
    - Media Navi
- Connected Home (formerly Digital Delivery)
- Entertainment Services
    - Creative Services (visual effects, animation & post production services) and
    - DVD Services
    - IZ-ON Media (provide digital place-based media services)

Supported by Corporate functions:

- Corporate
- Finance
- Human Resources
- Marketing & Branding

Technicolor Key Figures

- Employees end of 2012: 14,500 employees worldwide
- 2012 key financial figures :
    - Group revenues: €3.5 billion
    - Adjusted EBITDA: €512 million
    - Free cash flow generation: €106 million

Acknowledgement

I would like to express my gratitude to Tomas Crivelli and Franck Thudor, for all their help throughout this project and for everything they have taught me, the good practices and their trust in me. It has been an honour to work with them and furthermore, now I am closer to become the researcher I have always wanted to be. For all these things: thank you very much.

On the other hand, I would also like to thank my family for their unconditional support, no matter the distance, and also to my best friend Marta, who has always believed in me. Thank you.

Finally, my deepest gratitude to Coloma Ballester, Felipe Calderero and Edoardo Provenzi. They made me love science and Image Processing and they are and will be my role model for the next years. I wish one day I can inspire as many people as they do.

Abstract - English

Given the ubiquitous camera devices at hand for practically everybody and the ease to share media, the amount of video data around us is growing faster and faster. However, navigation systems for these data have not evolved very much over the years. When we have thousands of clips, navigate through that video collection could become a tedious task. In this project we present two different solutions to that problem. The first one consists in navigating through the collection in a temporal way: by showing videos only when there is an overlapping period of time previously found. The second one consists in letting the user to move himself through the video space: to change from his current video to another one recorded from a different relative position. We have explored different video analysis techniques such as interest point matching, point tracking and spatial transformations, while applying graph analysis techniques to build meaningful navigation schemas. Results on novel human-machine interfaces are presented to validate the workflow.

Abstract - Spanish

Dada la gran cantidad de dispositivos con cámaras y las facilidades de compartir cualquier tipo de media en las redes sociales o en internet, el volumen de videos a nuestro alrededor es cada vez mayor. Aun así, los sistemas para reproducir videos no han evolucionado durante todos estos años y cuando tenemos una colección de videos con más de mil ficheros, explorarlos puede convertirse en un trabajo pesado. Nosotros proponemos dos soluciones para este problema. La primera consiste en mostrar al usuario un video central todo el tiempo y los otros videos cuando haya una superposición temporal (previamente encontrada). La segunda propuesta consiste en navegar la colección de videos espacialmente, cambiando de un video a otro que esté grabado en una dirección relativa diferente a la actual. Hemos explorado diferentes técnicas de análisis de video como por ejemplo emparejar puntos de interés, seguimiento de puntos y transformaciones espaciales. También hemos usado grafos para estructurar información y hemos analizado diferentes posibilidades de interfaces.

Index

1. INTRODUCTION

1.1. Motivation and objectives

Given the ubiquitous camera devices at hand for practically everybody and the ease to share media on social networks, the amount of video data around us is growing faster and faster. However, navigation systems and interfaces for these data have not evolved accordingly over the years. We still have to change from one video to another manually in order to explore a video collection and, when we have thousands of videos from an event, this could become a tedious task.

We aim to setup a functional navigation interface which integrates automatic content organization algorithms and visualization schemes into an intuitive, easy-to-use, end-user oriented system.

Our system has to be able to find meaningful relations into different videos, interpret these relations and as an output give to the user a tool to watch those videos in an easy and friendly way. This means it will improve the human-machine interaction: the user will always be the one who decides what he wants to play, while the system will give him the tools to watch a collection of videos in an organized and non-redundant way.

In what follows, we study what information of the video is useful to the user, and we discuss different ideas for representing that information.

Two solutions to navigate a video collection are proposed and analysed, as well as different analysis techniques such as interest point matching, point tracking and spatial transformations, while applying graph analysis techniques in order to structure video set's information.

1.2. Related Work

Data analysis using graphs. Finding relations between images and dispose that information in a structured way so they can be easily explored and analysed has already been studied before. Good examples of this are the so-called Videoscapes, [1]. James Tompkin et al. propose to build a graph whose edges are videos and whose nodes are the portals (transitions opportunities) between them. For them, a portal is a span of video frames in either video that shows the same spatial location (notably common landmarks in the scene) even if they are filmed at different times or view-points. Given that graph, a user can explore a set of videos by walking through

the graph through the portals or by exploiting additional metadata as for example, GPS information embedded in a geographic map (Figure 1).

FIGURE 1: ON THE LEFT: INTERFACE TO EXPLORE VIDEOS AT DIFFERENT VIEW-POINTS OR TIMES. ON THE RIGHT: INTERFACE TO EXPLORE VIDEOS BY GPS INFORMATION, ACCORDING TO [1]

A similar idea has been used in [2] by Heath et al., where a graph is built in order to reflect the large-scale connectivity between still images. They have also implemented a visual hyperlink browser that lets users navigate to related images by clicking in visual hyperlinks and shows at the same time the local neighbourhood of the web (Figure 2).

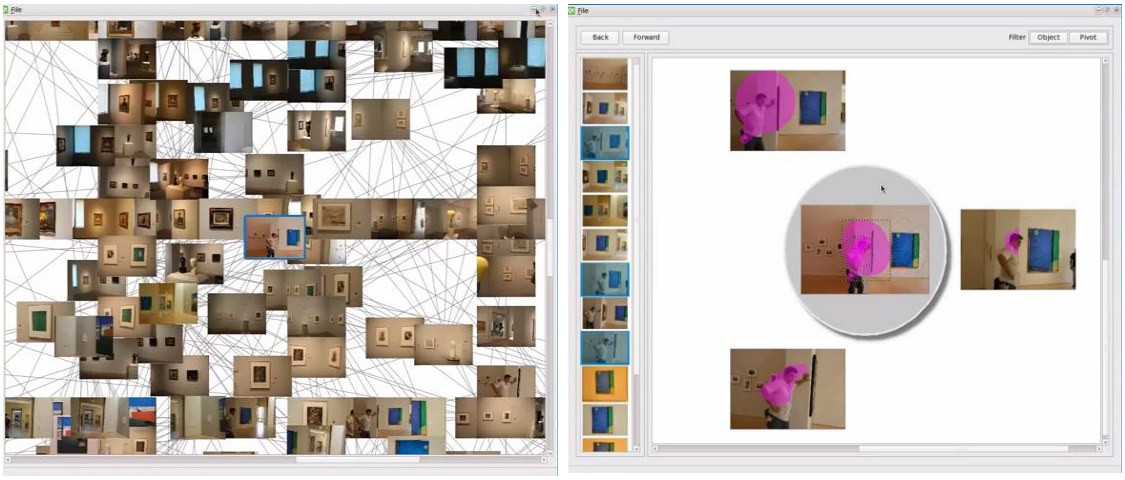


FIGURE 2: GRAPH (LEFT) AND THE HYPERLINK BROWSER (RIGHT). WHEN THE USER PICKS A NODE, HE SEES THE IMAGE IN THE BROWSER INTERFACE WITH ALL ITS NEIGHBORS IN A PANEL, ACCORDING TO [2]

A different approach to browse image collections was proposed in [3]. Jörg A. Walter et al. find spatial layouts of objects (in their particular case, of images, but it could be applied for any kind of multimedia content) in a hyperbolic plane based on pair-wise dissimilarity of data. They built an interface where the hyperbolic plane is displayed with the whole image set in it (see Figure

3). It allows human-computer interaction: the user can navigate through the images by moving its "focus" (centre) in the hyperbolic plane via his mouse, and without losing the "context" around the centre, as the other objects appears with gradually lower resolution and size depending on its "image similarity" level.

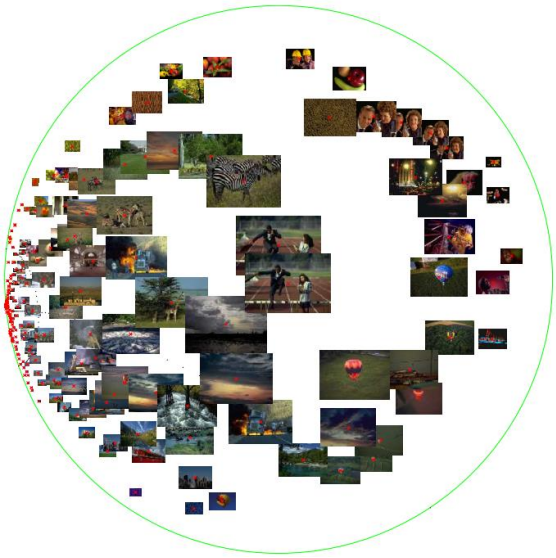


FIGURE 3: HYPERBOLIC IMAGE BROWSER USING A COLOR SIMILARITY METRIC FOR 100 PICTURES, ACCORDING TO [3].

In [4] Lyndon Kennedy and Mor Naaman also worked on finding a way to navigate a video collection. They also construct a graph where nodes are video clips and edges are representations of temporal overlapping between a pair of clips (Figure 4, left). Their interface would allow the user to watch videos recorded at the same time of the same event. The audio chosen while the user is watching the videos is the best quality one (Figure 4, right).

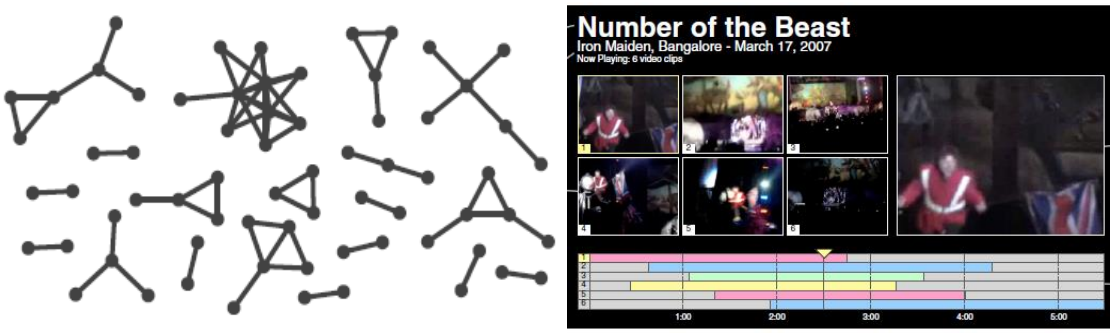


FIGURE 4: AT LEFT, GRAPHS STRUCTURES EMERGING FROM SYNCHRONIZING AND LINKING CLIPS BASED ON OVERLAPPING AUDIO CONTENT. AT RIGHT, POSSIBLE INTERFACE TO WATCH THESE CLIPS, ACCORDING TO [4].

Graphs structures are also used for other goals than data navigation, such as video summarization. In [5], in order to get a good video summarization, they try to find the most representative parts of a video. Chong-Wah Ngo et al. represent a video as a complete

undirected graph, whose nodes are shots linked temporally by its edges, and then they do an optimal partition (by a global criterion, normalized cut [11]) of it into different clusters. These clusters form a directed temporal graph and its shortest path detects video scenes. They assign attention values to each scene, and with this values they can inherently describe the evolution and perceptual importance of a video.

Our system is more user-oriented. We found that work in [2] may be confusing for some users in the sense of how to navigate the graph. Moreover, in a large collection, this graph may become huge, and one might get lost in going from one video to another one. We aim to set up an interface that the user has not to think how it works: intuitive and friendly.

In our first version, we do something similar to [4]. We have created a demo that shows a main video and the others connected temporally to it. However, by looking at the results in [4] we thought that an interface where the user visualizes several videos simultaneously, even if they are happening at the same time, may become overwhelming, prone to missing important details. Normally, when someone watches a movie, he needs to focus in one video at a time. A system where more than one video clip is playing might distract the user. Thus, in section 4 we propose a system where there is one main video, so the user only needs to fix his attention to this one, and when something important happens in another video to a current time of the main video, it will appear in the interface gradually so the user can watch and switch to that secondary video.

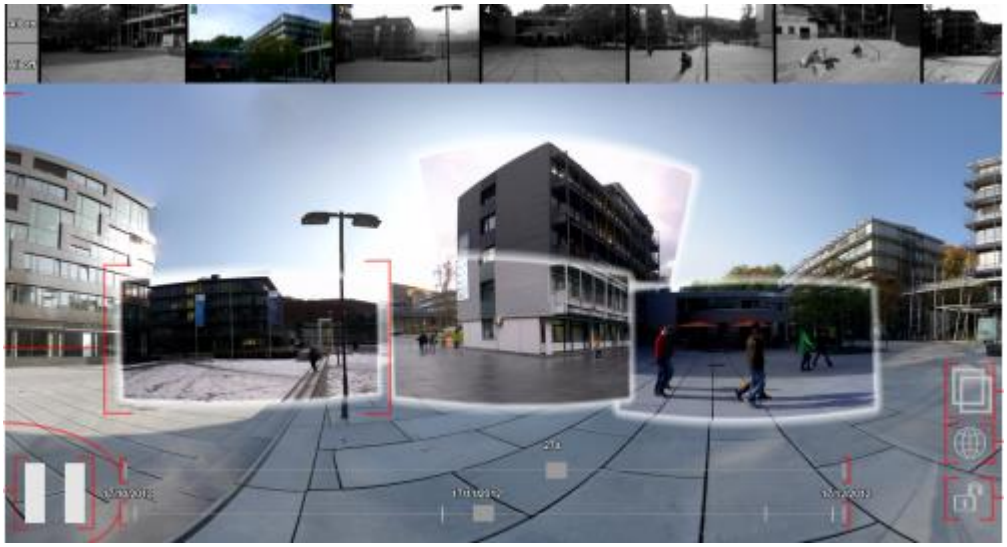


FIGURE 5: FROM [12] THE VIDEOCONTEXT INTERFACE. DIFFERENT VIDEOS ARE ADDED TO ONE IMAGE TO RECONSTRUCT THE VIDEO SET'S SPACE.

In section 5, we will present our second demo, which is similar to [6]. We will try to build the space of the event by looking at the relative positions from each video in our collection. In [6], they try to find videos locations and orientations by optimizing a graph connected by interest

points between frames, looking for a coherent context result. Our system, however, would not aim to reconstruct the scene as it is done in [12] (as we can observe in figure 5), but let the user change from one video to another by a coarse relative positioning. It would also differ from work in [1] because the user will be able to move himself freely into the event space: he will be able to change his position even if the system did not find any video with a transformation in that direction thanks to the transitivity condition, which will be further explained later.

2. SYSTEM DESIGN

As said before, our goal consists in finding a new intuitive and friendly way to navigate through a video collection i.e. to let the user change from one video to another one in an easy, practical and smart way so that, at the end, he is able to visualize all his collection. In order to start planning how to face that problem, we have analysed the standard signal processing system scheme and we have thought about different approaches of navigation.

2.1. System analysis

Usually, the general schema for video processing systems is defined like in figure 6. First, videos are analysed in order to find features. These features will be used to create new metadata such as matches, clusters, similarity coefficients and different types of information. Thanks to that metadata, the system can take decisions and generate a desired output.

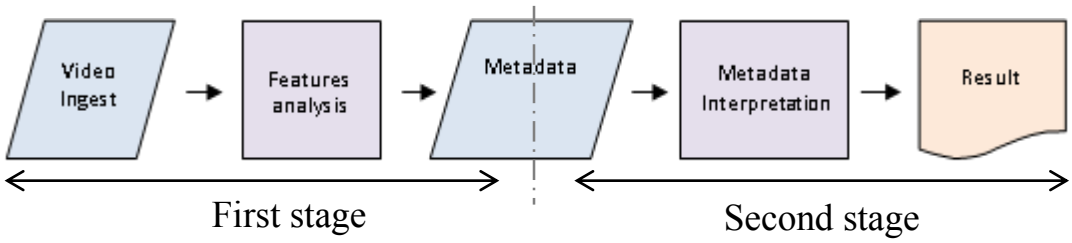


FIGURE 6: STANDARD WORKFLOW OF A VIDEO SIGNAL PROCESSING SYSTEM

However, in our case we needed to start for the last step: the result. Until we know exactly what we want to show to the user we are not going to be able to think about what algorithms or methods we need to perform.

2.2. Analysis of possible interfaces

2.2.1. Action detection

This idea is represented in Figure 7, where we can see that the main idea is to display a principal video in the centre, the one which displays the scene in the most representative way, and when the system detects action in a particular side of that video, it displays also another video to show that action at one side of the main video. In Table 1, we can see the advantages and disadvantages of that solution for navigation through a video collection.

FIGURE 7: A SAMPLE FOR ACTION DETECTION VISUALIZATION OF A VIDEO COLLECTION

Pros	Cons
It is very informative, it shows the most important information to the user	It is not very interactive, the user does not select what he considers interesting
We could make the user to select what he wants to see: goals in a match, a player, a singer...	To let the user specify the action could be very complex (due to bad quality of data, occlusions…)

TABLE 1: PROS & CONS OF NAVIGATING A VIDEO COLLECTION BY ACTION DETECTION

2.2.2. Navigation by viewpoints

This approach is inspired on Google's street view[1]. As we can see in Figure 8, while the user is watching a specific video, he will have the opportunity to change his point of view by clicking on the arrows displayed on the video, each time that this is possible (when it exists a video recorded from this direction). Table 2 points out the problems and benefits of using that type of visualization.

[1] http://www.google.com/maps/about/behind-the-scenes/streetview/

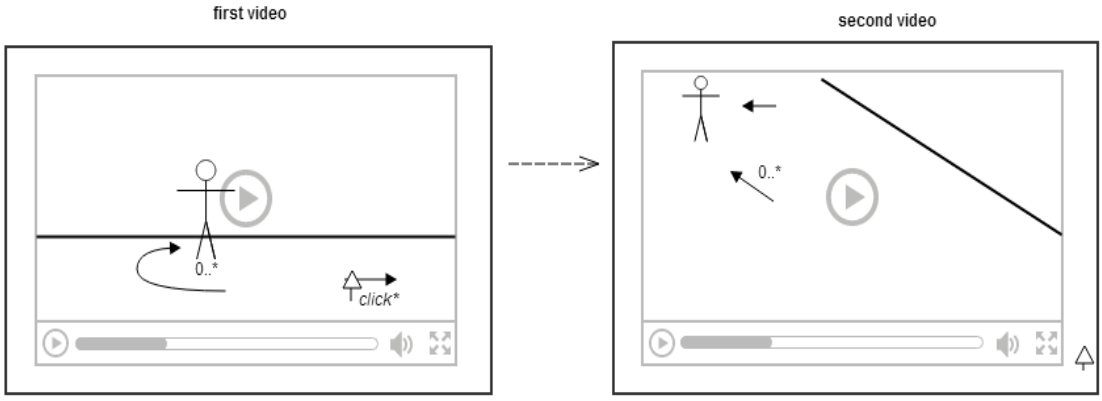


FIGURE 8: A SAMPLE FOR NAVIGATING A VIDEO COLLECTION BY VIEWPOINTS

Pros	Cons
Nice interface, very intuitive. It allows the user to locate himself in the scene.	Very hard to make a 3D reconstruction on videos, but we could find a simpler way to represent that positions.

TABLE 2: PROS & CONS OF NAVIGATING A VIDEO COLLECTION BY VIEWPOINTS

2.2.3. Semantic search

The basic idea is to provide an easy way to find videos related to what the user is looking for. As it is shown in Figure 9, the user writes on search space what he would like to watch (for example, a singer) and below it appears a list of videos with that data.

That could be done by tagging each video with their main content. In Table 3 we can see its advantages and disadvantages.

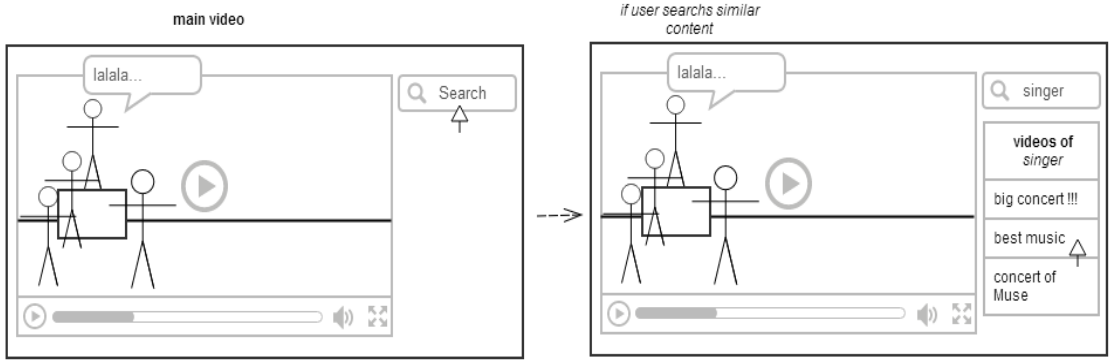


FIGURE 9: A SAMPLE FOR NAVIGATING A VIDEO COLLECTION BY VIEWPOINTS

Pros	Cons
It makes it easy to find what you are looking for	We need to trust the users to tag correctly the data
	It does not differ too much to the standard video navigation

TABLE 3: PROS & CONS OF NAVIGATING A VIDEO COLLECTION BY VIEWPOINTS

2.2.4. Object detection

The idea is similar to the one before, but here the user can select over the video the object that he wants to watch in a different video, instead of writing it (see Figure 10). The pros and cons of this kind of navigation are exposed in Table 4.

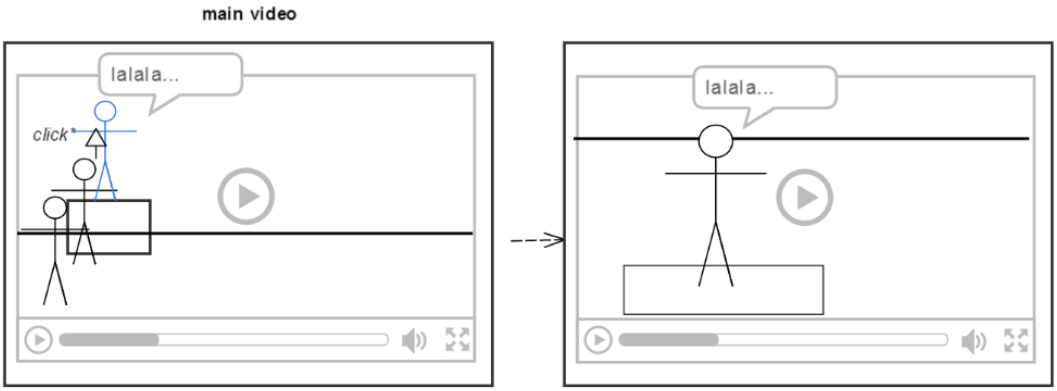


FIGURE 10: A SAMPLE FOR NAVIGATING A VIDEO COLLECTION BY OBJECT DETECTION

Pros	Cons
Very intuitive for the user	It does not differ too much to the [1] approach
It allows the user to find what he is looking for	User may need to specify good enough what he wants to watch (for instance, another object with same colour, shape…)

TABLE 4 : PROS & CONS OF NAVIGATING A VIDEO COLLECTION BY OBJECT DETECTION

2.2.5. Temporal and spatial search

This approach is inspired on the work in [3]. It allows the user to watch all the video data of a specific place (or around) in a specific time, in a way that he can get an overall idea of what happened in that time. The visualization could display the videos in different sizes, so the smaller ones make reference to videos in a further moment to the one desired (see Figure 11).

This approach could be interesting too if we consider a hyperbolic space to represent that data. That could help when we have thousands of videos, so we would be able to place them all.

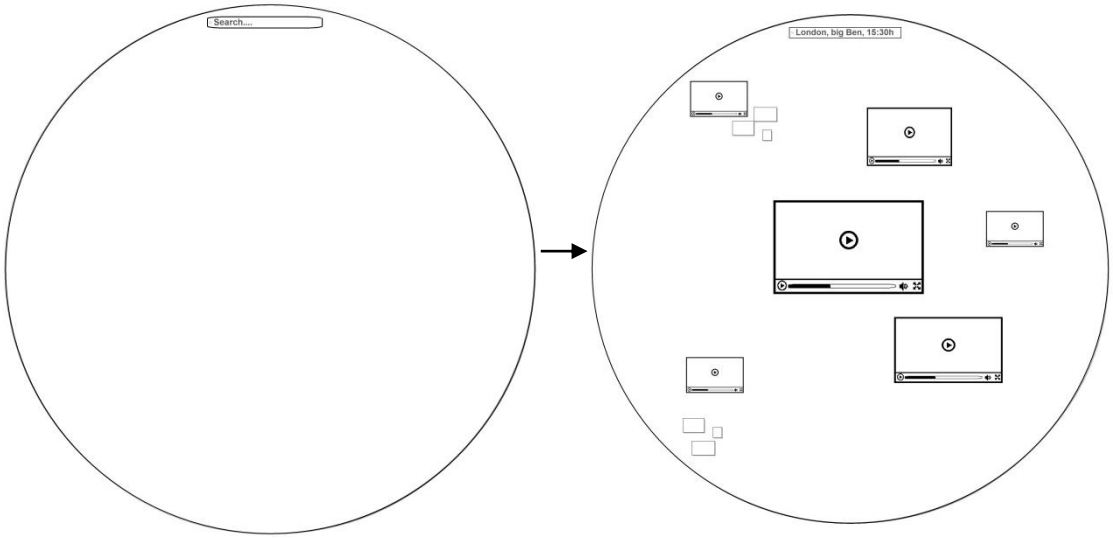


FIGURE 11 : A SAMPLE FOR NAVIGATING A VIDEO COLLECTION BY A SPATIAL-TEMPORAL SEARCH

Pros	Cons
It is a different approach to link the video data	We have to trust in the camera device information
It allows the user to get an overall idea of a moment	It is complex to display the data in a synchronized way

TABLE 5: PROS & CONS OF NAVIGATING A VIDEO COLLECTION BY A SPATIAL-TEMPORAL SEARCH

2.2.6. Album visualization

It's based on the above idea but it tries to show the temporal linked data in a more friendly way to the user (look at Figure 12). It allows him to build his own video-album (or one shared with

more users or friends) sorted by time, so he can navigate through his data in an easier way. In Table 6, the pros and cons of that approach are presented.

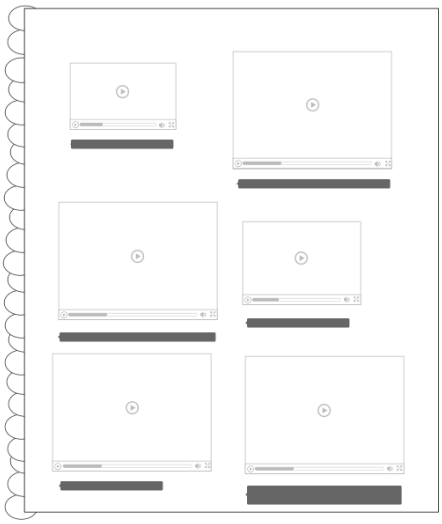


FIGURE 12: A SAMPLE FOR NAVIGATING A VIDEO COLLECTION BY USING AN ALBUM

Pros	Cons
It could be more interesting as application, for the user point of view	The same ones as the idea in 2.2.5
	With a huge data set of videos, the album can be very big and uncomfortable to navigate through it.

TABLE 6: PROS & CONS OF NAVIGATING A VIDEO COLLECTION BY USING AN ALBUM

2.2.7. Navigation by suggestions

This idea is very similar to the one presented in [1], in a way that while the user is watching a specific video, the screen will display an information icon (see Figure 13). If the user clicks it, it will show to the user suggestions of similar videos to the one he is looking at. We can observe in Table 7 the advantages and disadvantages of that solution to video collection navigation.

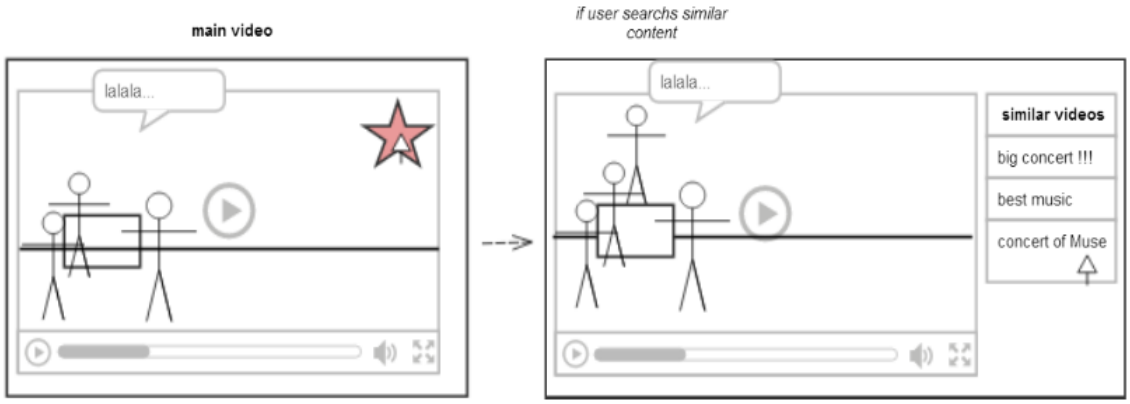


FIGURE 13: A SAMPLE FOR NAVIGATING A VIDEO COLLECTION BY SUGGESTIONS

Pros	Cons
It does not disturb the user while he is watching his video, letting him decide if he wants to know about more similar videos.	Very similar to the Videoscapes approach

TABLE 7: PROS & CONS OF NAVIGATING A VIDEO COLLECTION BY SUGGESTIONS

3. VIDEO COLLECTION

Throughout our project we work with our own video collection ,called “Soccer”, which consists in 4 different video clips extracted from a bigger collection with about 75 video clips (around 8 hours of video), recorded in the same place over a period of time. It is useful for us because they are clips about the same event (we have taken a collection of videos that record a soccer match but we could have used perfectly another types of content such as concerts, races or weddings). They are recorded from different positions over the same place and by using different sensors (smartphones, GoPro cameras, and digital ones). Thus, they have some spatial and temporal overlapping, which makes a collection very appropriate for us when we need to find connections between video clips. Moreover media is UGC (User-generated content) i.e. recorded with not professional devices, letting us to test our implementations with the kind of data that our future users will use to have.

Moreover, some videos have a lot of movement and other ones are rather static, what gives us variety and the possibility to study different types of scenarios. However, there is one important disadvantage: there are some objects repeated over the same space (for instance, the letters “www.soccer.fr” or the lights), what may be a problem when we need to use matching techniques.

Information about our dataset is given in Table 8. However, in order to reduce time calculation, we have decreased their resolution by half of them, and even like that while testing we have seen that it took quite a lot of time. Thus, from these 4 videos, we have created one set called High Resolution (which contains half original resolution) and Low Resolution (who contains half the half of original resolution). In Table 9 and Table 10 we can see the final parameters that we have used and in Figure 14, Figure 15 and Figure 16 some examples from our dataset.

Name	Size	Frame Rate	Length	Frame height	Frame width
10350-100000004-00003.mp4	555 MB	25 frames/second	26 min 03 sec	1080	1920
10362-100000004-00003.mp4	2.13 GB	25 frames/second	34 min 42 sec	1080	1920
10382-100000004-00003.mp4	591 MB	25 frames/second	17 min 25 sec	960	1280
10702-100000004-00001.mp4	79.0 MB	25 frames/second	01 min 14 sec	1080	1920

TABLE 8: VIDEOSET INFORMATION

Name	Size	Frame Rate	Length	Frame height	Frame width
10350-100000004-00003_LR.mp4	52.2 MB	25 frames/second	26 min 03 sec	270	480
10362-100000004-00003_LR.mp4	128 MB	25 frames/second	34 min 42 sec	270	480
10382-100000004-00003_LR.mp4	65.5 MB	25 frames/second	17 min 25 sec	240	320
10702-100000004-00001_LR.mp4	5.54 MB	25 frames/second	01 min 14 sec	270	480

TABLE 9 VIDEOSET LOW RESOLUTION INFORMATION

Name	Size	Frame Rate	Length	Frame height	Frame width
10350-100000004-00003_HR.mp4	52.2 MB	25 frames/second	26 min 03 sec	540	960
10362-100000004-00003_HR.mp4	427 MB	25 frames/second	34 min 42 sec	540	960
10382-100000004-00003_HR.mp4	175 MB	25 frames/second	17 min 25 sec	480	640
10702-100000004-00001_HR.mp4	15.5 MB	25 frames/second	01 min 14 sec	540	960

TABLE 10: VIDEOSET HIGHT RESOLUTION INFORMATION

FIGURE 14: FRAMES FROM 10702-100000004-00001.MP4

FIGURE 15: FRAMES FROM 10382-100000004-00003.MP4

FIGURE 16: FRAMES FROM NAME: 10362-100000004-00003.MP4

FIGURE 17: FRAMES FROM 10350-100000004-00003.MP4

4. VIDEO COLLECTION NAVIGATION BY TEMPORAL OVERLAPPING

After discussing some approaches of how we are going to face our problem of visualizing several video data, we have opted to implement a first version of our project in order to analyse it and to find out what are we interested to see. Thus, as we can see in Figure 18, we are going to focus on the second stage.

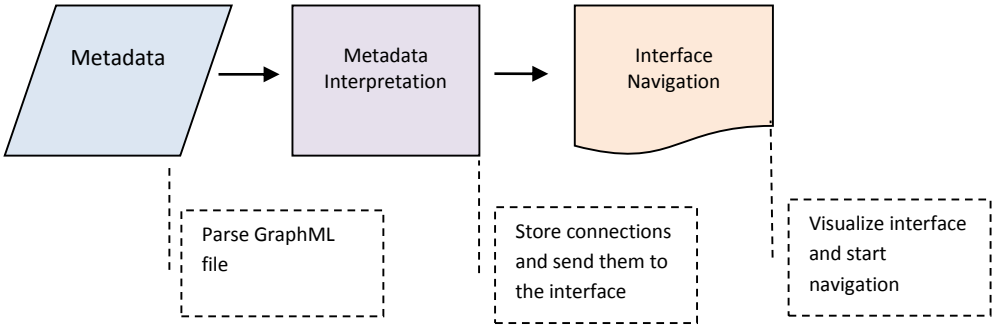


FIGURE 18 SCHEMA OF VIDEO COLLECTION NAVIGATION BY TEMPORAL OVERLAPPING

This first version, Video collection navigation by temporal overlapping or so-called V0, offers a minimal/basic interaction to the user. The idea consists in studying an example of interface in order to understand what is important to show and in order to take a first glance to its possible complications. The interface that we have coded, basically, displays a graph whose nodes are video-clips and where they are connected by the frame number where a temporal overlapping exists (see Figure 19). After showing this graph to the user, he can select which video he wants to visualize. Afterwards, that video is played in the centre of the screen and its neighbours are played in the background, in a 3D space (see Figure 20).

The navigation proceeds as follows: when there is a temporal overlapping between the main video and one of its neighbours in the background, that video comes to first plane and it is played too, synchronously. When the overlapping period is finished, it returns to the background.

4.1. Representation of our data

4.1.1. Graph representation

In order to develop this initial version of our project, we have analysed 2 minutes of a video from our collection and we have selected some key instants for this time slot. We have searched for these key instants in four different videos from the same collection and we have created our temporal overlapping links.

Afterwards, we have written that information in a Graphml file (see Appendix).

4.1.2. What is a node, what is an edge?

We have considered the easiest and intuitive matching criteria: the temporal one. In our graph, we can see four video clips in grey, and several sub-clips. These sub-clips represent the frame where there is a temporal overlap. In our case, we can see how clip 1 is connected three times with clip 2, whereas it only does it once with clip 4.

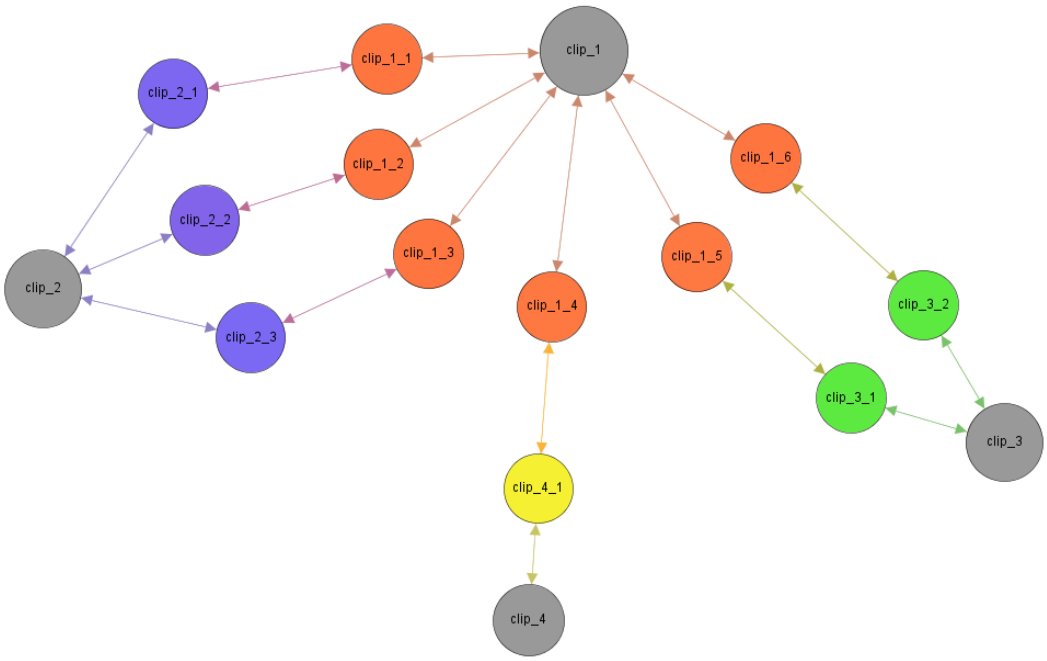


FIGURE 19: GRAPH USED IN VIDEO COLLECTION NAVIGATION BY TEMPORAL OVERLAPPING

Keeping this in mind we can define a vertex as a whole video clip, a sub-vertex as a landmark or portal, defining as a portal a time slot where there is a temporal overlapping, and an edge as the bridge between two clips.

4.2. Our code

4.2.1. Parse function

We have stored the graph's attributes such as names of the vertices, paths and key-frames in a class called "Graph". Those key-frames indicate the first and the last frame where the overlap exists.

Once we have the graph's information, we display it and we wait until the user selects the video (vertex) that he wants to visualize.

4.2.2. Visualization function

The system displays the video selected by the user in the centre and all its video's neighbours in the background. The user can move himself in the temporal axes by using a seek-bar and thus he can choose the video reproduction time. He can also pick another vertex in the graph and change the main video.

When the main video arrives to one of its key-frames, the video linked to that key-frame comes into the first plane and both videos are displayed synchronously.

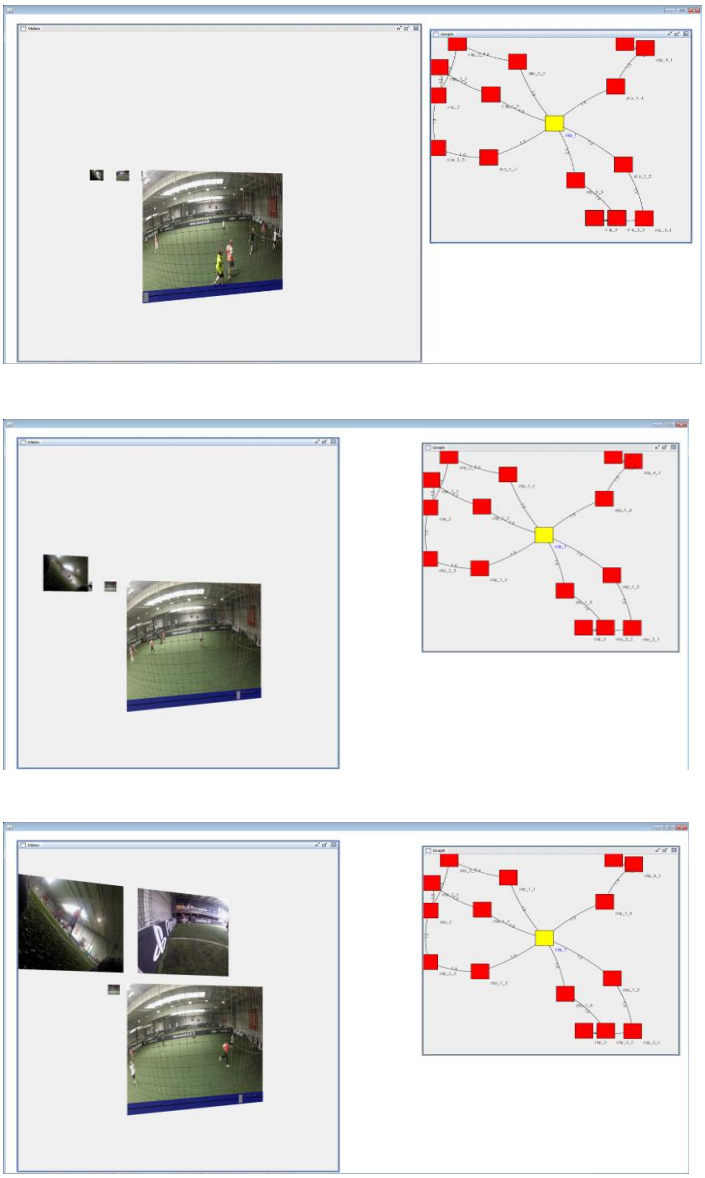


FIGURE 20: VIDEO COLLECTION NAVIGATOR OF VERSION 0

4.3. Conclusions

This first implementation helps the user to visualize his collection since it shows him what video to look at any time. However, we have seen that to select the main video in the Graph draw may not be intuitive, and if instead of 4 videos, we have 60 (for example), this graph might be too big. Thus, we thought that it may be better to let the user to interact with the video window, rather than with the Graph one. It might improve the navigation if the graph is only used to interpret the video connections and not as a tool for navigating through the video collection.

After this implementation, we decided to keep studying more cases of video collection navigation. This time, we would look also the first state of the general scheme: the generation of the graph.

On one hand, if we keep our current temporal connection between video clips, we will have a very huge graph with thousands of edges, and its complexity may be elevate. However, we have to keep in mind that if we want to simplify our graph and to try different things, we may need to use a more complex matching criteria.

5. VIDEO COLLECTION NAVIGATION BY VIEWPOINTS

Our next step was to set up a simple version but a functional one. That means, to build a system that is able to find by itself the relations between videos, to build a graph, to interpret it and to display a result.

Although in the previous version the user was able to visualize several clips related to a main video by its temporal overlapping, one of its problems was that sometimes a given video may have a lot of connections and therefore, the graph might be very big. Thus, the system should display all the video clips connected to the main video in the background and it could be a bit messy.

In order to fix that, we need to reconsider our definition of node/edge.

5.1. Defining our nodes and edges.

In order to define what are going to be our nodes and edges we have discussed some ideas, which are represented in Table 11:

Node	Edge
1. Video Clip	Time
2. Places	Paths
3. Video Clip	Events
4. Points of view	Objects

TABLE 11: DIFFERENT DEFINITIONS OF NODE AND EDGE

They are based on the first ideas we have got at the beginning but in a more edge/node point of view.

Our last implementation has an edge/node interpretation as the first one (Time/Video Clip), but as discussed, it generates a lot of connections between each pair of videos.

The second one, navigating between important places, would be for instance if we have as nodes the Eiffel Tower and the Louvre museum and the edges are paths that connects that two places by, for example, proximity. This may be nice for the user (like strolling in the city with a

map, since graph's edges would be as paths and its vertices destinations) but it may be as well quite complex and not very intuitive (see Figure 21). We would need to show all the places where the user can go while he is watching a video, and when there are thousands of different options the user may get lost in the interface. Moreover, we would need to trust GPS information in order to set its location, what may differ from one device to another one and it may be even wrong.

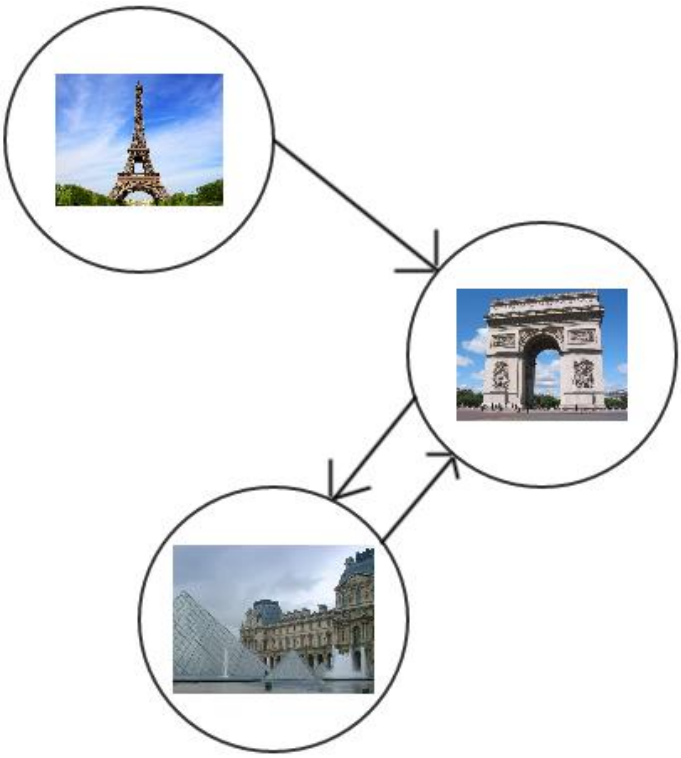


FIGURE 21: EXAMPLE OF GRAPH CONSIDERING A PATH/PLACE REPRESENTATION

Defining the vertex as clips and the edges as events it is an approach similar to V0, where the overlapping is not temporal but an action one: it would consists in detecting some important action, as it would be for example a goal in a match, and linking different video clips for occurrence of these actions.

The last idea in the table, the one we have decided to implement in this new version, is to navigate through a video collection by viewpoints. Basically, it will consists in defining the nodes as the videos from our database and the edges like connections between these videos that relates them by their relative position. The graph will be something similar to Figure 22.

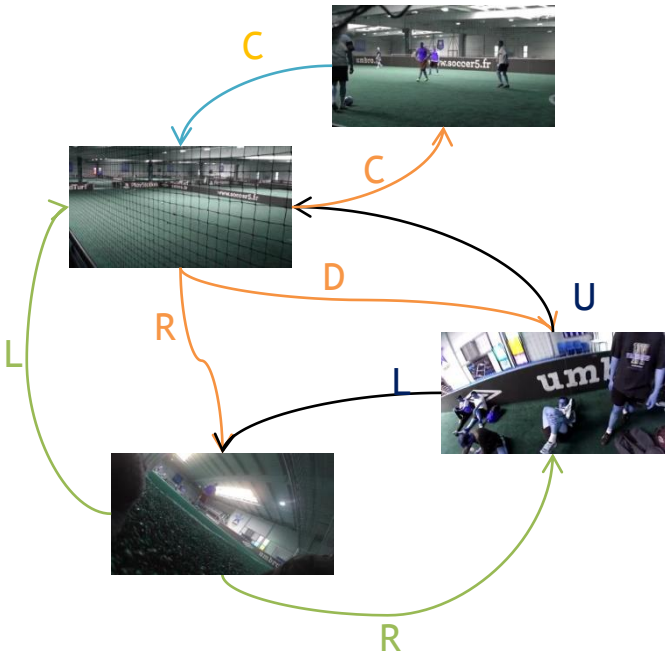


FIGURE 22: GRAPH WHICH NODES ARE VIDEOS AND EDGES ARE THEIR RELATIVE POSITION (LEFT, RIGHT, UP, DOWN AND CENTER).

In the next section we are going to discuss more in detail this mode of navigating.

5.2. System Workflow

In order to understand what means to navigate a video collection by viewpoints, we can take a look at Figure 23. The idea is that when the user is watching a video he is going to be able to watch another one that has been recorded from another point of view (for example, at the right) by clicking on the arrow that point to that direction. Then, the system will show him another video recorded from that direction.

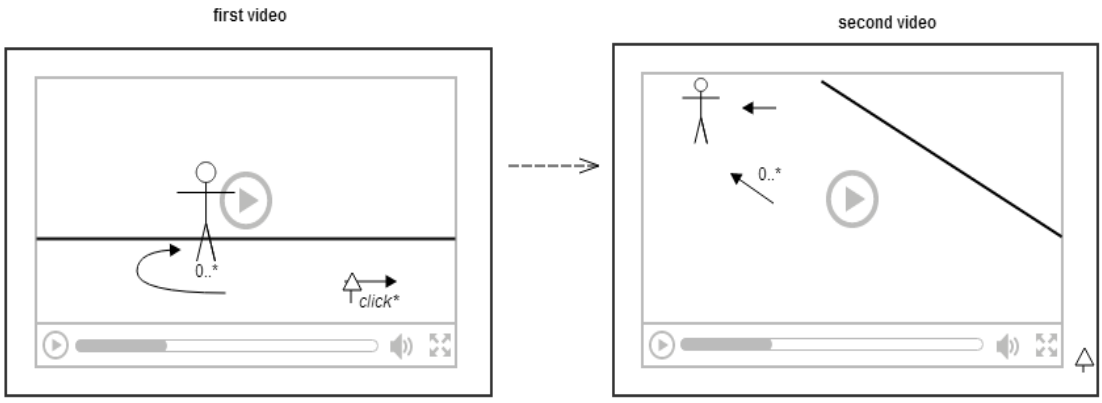
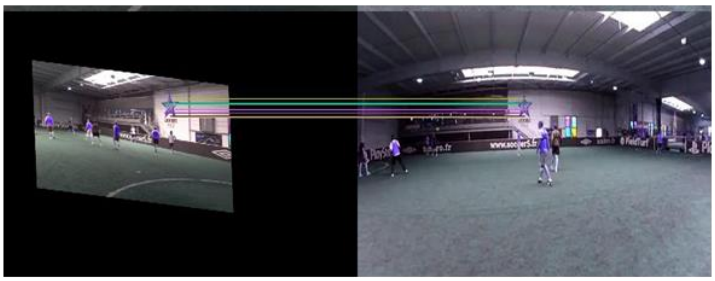


FIGURE 23 EXAMPLE OF CHANGING FROM ONE VIDEO TO ANOTHER ONE WITH THIS APPROACH

Hence, we are going to compare all the videos within our database in order to estimate their relative position. In Figure 24, we can see an example of this process. First, we take a pair of videos in order to compare them and then, we estimate their relative position to finally show to the user one video and its arrow pointing to the other one.

1. Take a pair of videos



2. Find their relative position

3. Show result

FIGURE 24 SET OF PROCESS TO BUILD THAT SYSTEM

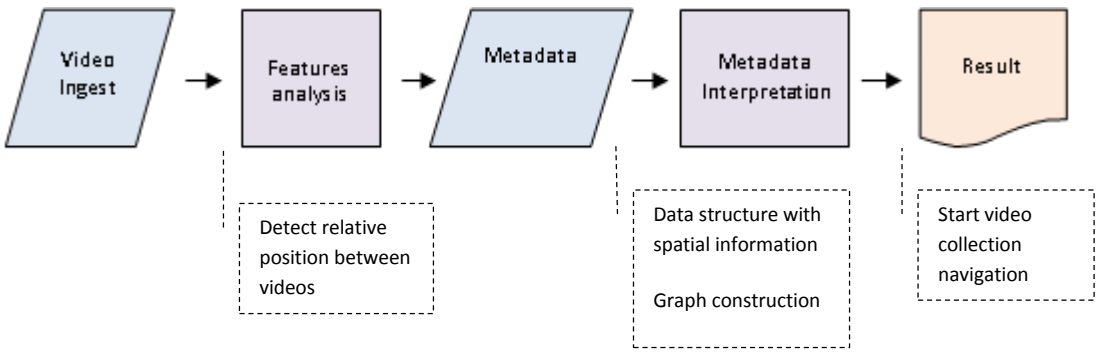


FIGURE 25 SCHEMA OF VIDEO COLLECTION NAVIGATION BY VIEWPOINTS

As we can see at Figure 25, our system will consist in a set of video processing techniques that will allow us to extract metadata for building a graph. This set of techniques will be considered the first stage: the set of procedures to find relations between each pair of videos within our database. Afterwards, second stage will consists in interpreting these relations and building a graph that later will be interpreted in order to create the user interface.

In what follows, we will explain the overflow of our system, focusing first in the first stage which can be quickly summed up in Figure 26.

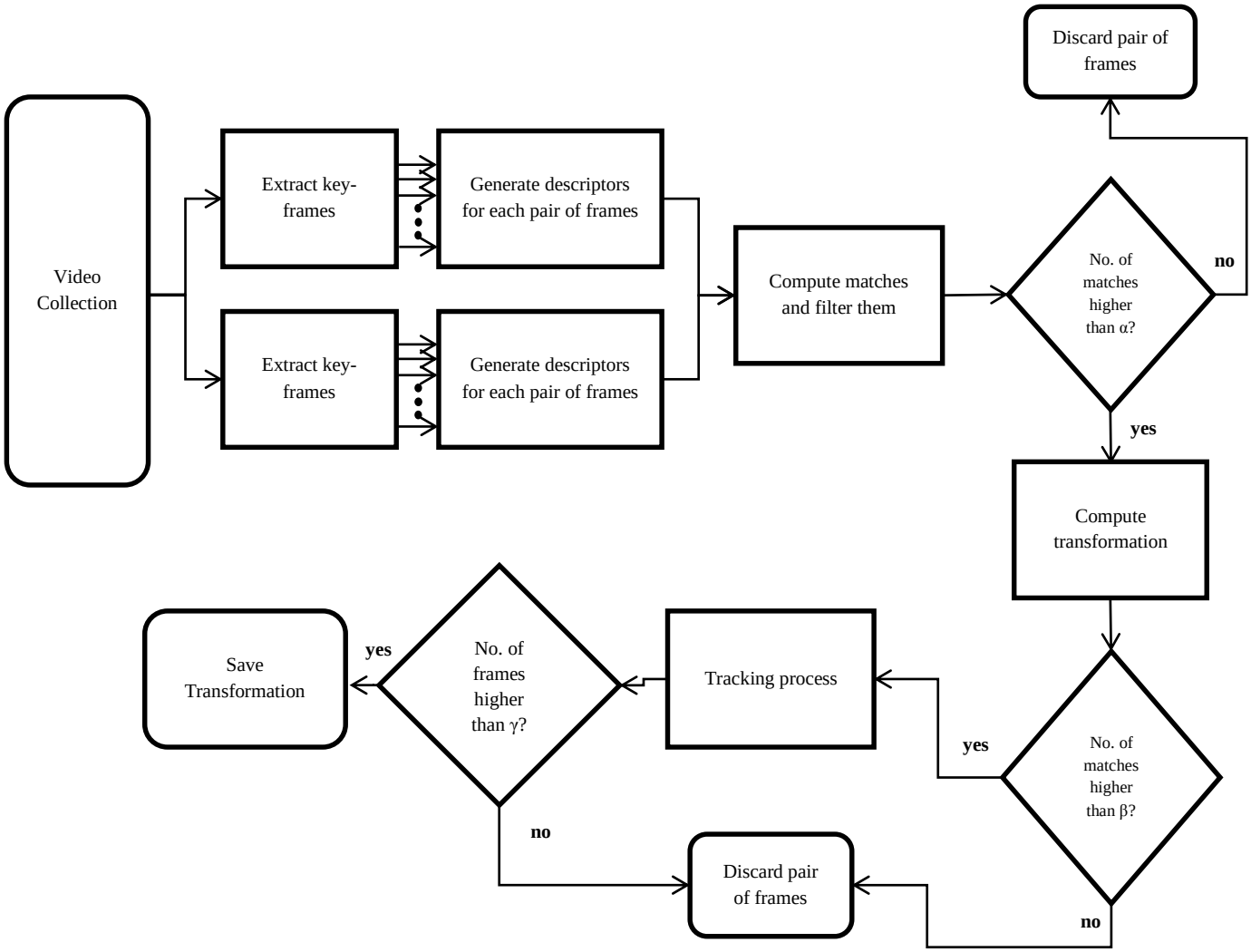


FIGURE 26 WORKFLOW FIRST STAGE OF VIDEO COLLECTION NAVIGATION BY VIEWPOINTS

5.2.1. First stage

5.2.1.1. Finding good matches

For each video within our collection, we have extracted a number of key-frames by an uniform partition of the video. Thus, we can obtain key-frames from different moments of the video.

Afterwards, for each pair of video, we have matched each key-frame between them in order to find relations. This matching process has been done by using SIFT algorithm (an algorithm to detect and describe local features in images, see [7]) in order to detect features in each key-frame and then we have matched them by using FLANN (an algorithm that searches the nearest neighbours for each point in a given set of matches, for further explanation see [8]. Our criteria to use SIFT was that it has been proved to be a good method in different video and image processing works and because we needed to start by one algorithm, but we should have chosen a different one such as SURF. This reason is valid as well for FLANN, thus we decided to use standard techniques in order to start building our system and let the possibility to use different methods for future work. In Figure 27 we can see an example of the result after computing matches.

FIGURE 27 MATCHING PROCESS BY USING SIFT FEATURES DETECTION AND FLANN MATCHING ALGORITHM

As we need these matches to be the best as possible in order to do a good relative position estimation, we decided to filter them in order to discard those that are not right (outliers). We have though in two different filters:

1. **Matching consistence filter**: Given two images, A and B, we consider a match as a good one if it appears in the matching process for A to B and also when doing the same process but invers: for B to A (see Figure 28)
2. **Distance filter:** We consider a good match if the distance of its second nearest neighbour found by FLANN algorithm (the distance to the K-nearest neighbours) is bigger than the one of the first nearest neighbour.

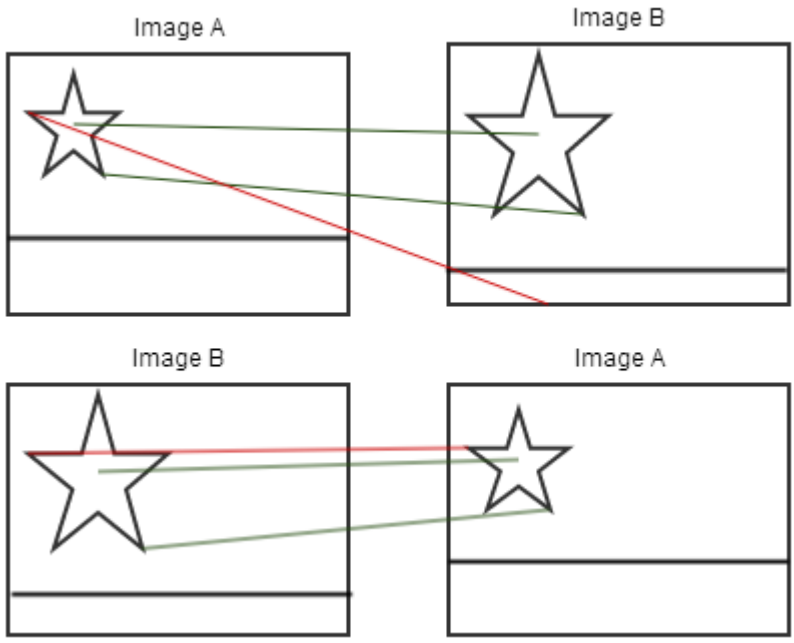
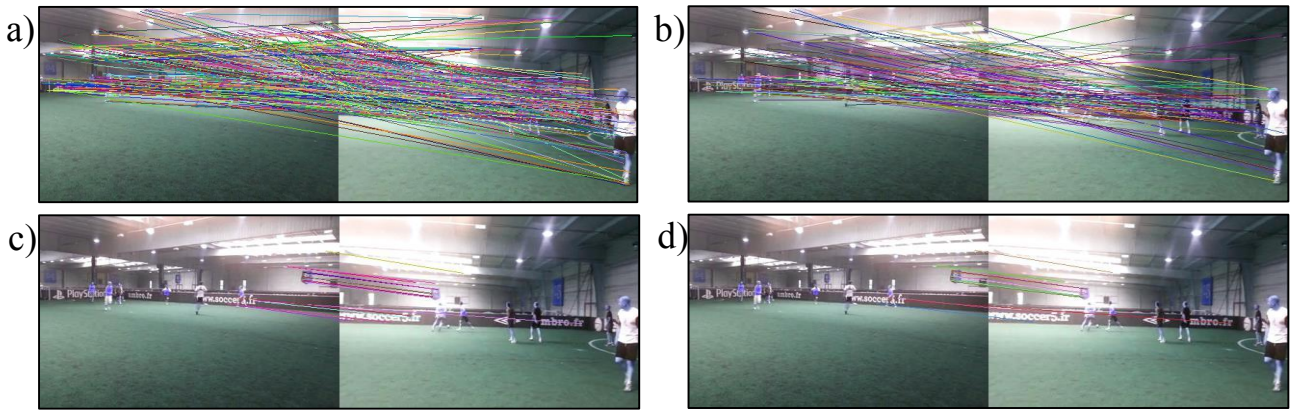


FIGURE 28: FRAME MATCHING CONSISTENCE: MATCHES IN GREEN ARE CONSIDERED GOOD MATCHES BECAUSE THEY ARE IN BOTH MATCHING PROCESSES. ON THE OTHER HAND, RED ONES ARE DISCARDED

After testing this two filters, we have realize that for some images the distance filter worked better than the correspondence one and for some other images was the opposite. Thus, we decided to implement both filters.

In Figure 29 we can see some results of this process.



	a) Nearest Neighbors	b) Correspondence filter	c) Distance filter	d) Both filters
Nb. of matches	384	138	16	12

FIGURE 29 PROCESSESS TO GET GOOD MATCHES BETWEEN A PAIR OF KEY-FRAMES

Notice that the number of matches from the beginning until applying both filters has decreased more than 90%. Still, this is not a concern as long as we have good matches and enough number of them. Thereby, after matching key-frames we check if we have more than a threshold and if not, we discard that pair of key-frames and we take the next ones.

5.2.1.2. Finding good homographies

Once we have found matches between key-frames we were interested in relating these key-points. Thus, we decided to use a transformation matrix in order to relate these pairs of key-frames. With this idea in mind, we computed homography transformations by using RANSAC algorithm (see [9]).

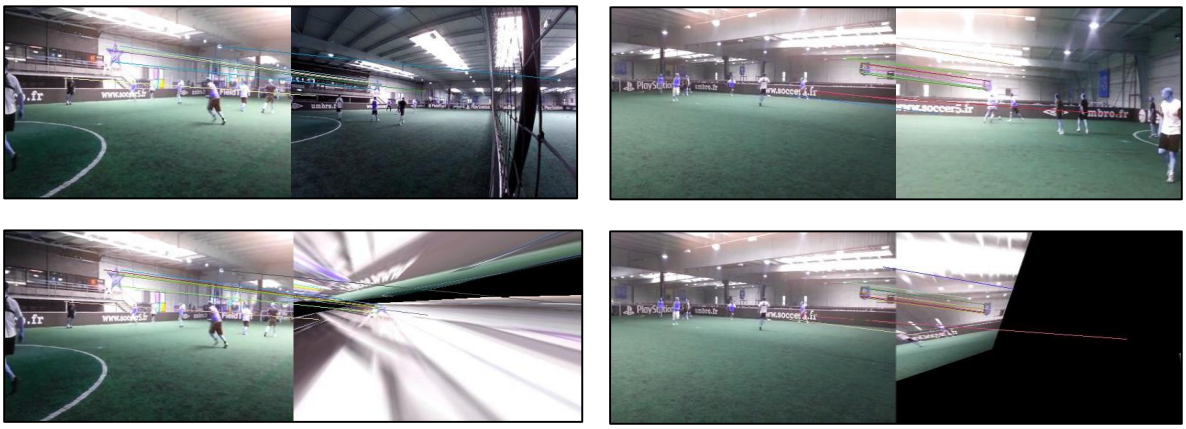


FIGURE 30 HOMOGRAPHY TRANSFORMATION AND COMPARATION WITH THE ORIGINAL KEY-FRAMES

However, as it can be shown in Figure 30, results are difficult to interpret. In these pictures, we can see in the images at the top two key-frames from two different videos, and in the bottom the key-frame from the first video and this same key-frame transformed by using the homography found. Technically, this last image should look similar to the key-frame of the second video. However, sometimes the results are not straightforward to understand.

Since we are looking for finding the relative position for each pair of video (for example, saying that one video is more at the left from another one) and we do not aim to reconstruct a 3D scene or neither locate the cameras in the space, we thought about using a simpler transformation such as Affine Transformation.

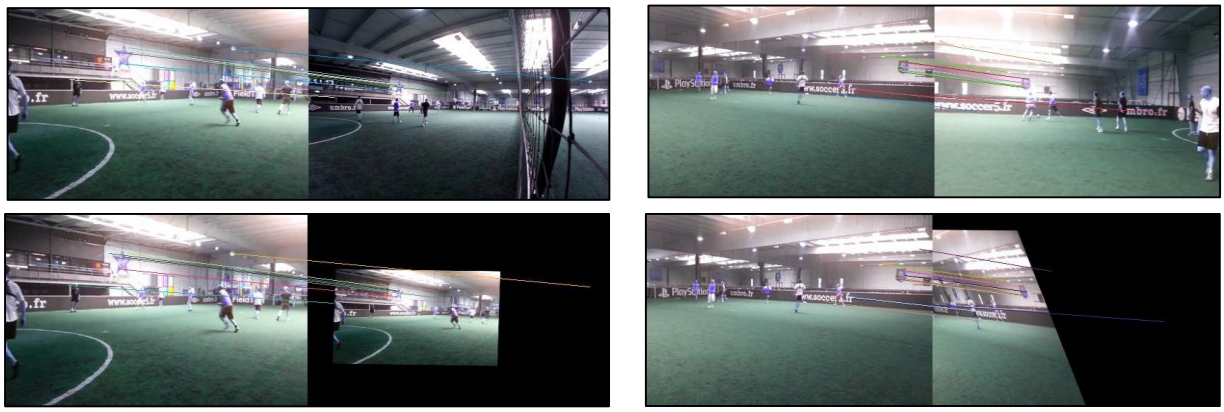


FIGURE 31 AFFINE TRANSFORMATION AND COMPARATION WITH THE ORIGINALS KEY-FRAMES

As we can see in Figure 31, results were easier to interpret, thus we decided to use this type of transformation.

In order to check if the transformation obtained was robust, we checked if it had enough matches that justify that transformation. Otherwise, we would also discard that pair of frames.

5.2.1.3. Tracking process

Finally, in order to see if the transformation found was representative for the video (that means, that we have chosen a key-frame that represents the overall point of view from that video) we decided to see if the key-points detected on that key-frames do not disappear in the future and in the past frame.

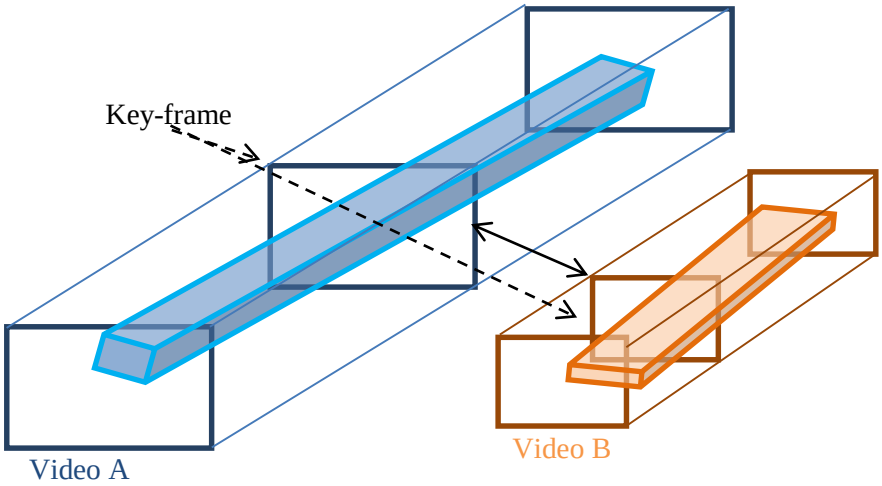


FIGURE 32 TRACKING PROCESS OF TWO VIDEOS, STARTING FOR ITS KEY-FRAMES AND LOOKING AT THEIR PAST AND FUTURE FRAMES.

In order to do that we are going to look at the optical flow of these key-points until we detect a lot of movement or that they are no longer there.
We look for a number of frames forward and backward the key-frame and if we lose the tracking then we discard that pair of key-frames.

After all this set of processes, if the key-frames have succeeded, we stored that transformation in order to interpret it later, in the second stage.

But before continuing, we decided to study how many connections we had. In order to do that, we built a graph to show the connections found between frames, as we can see in Figure 33.

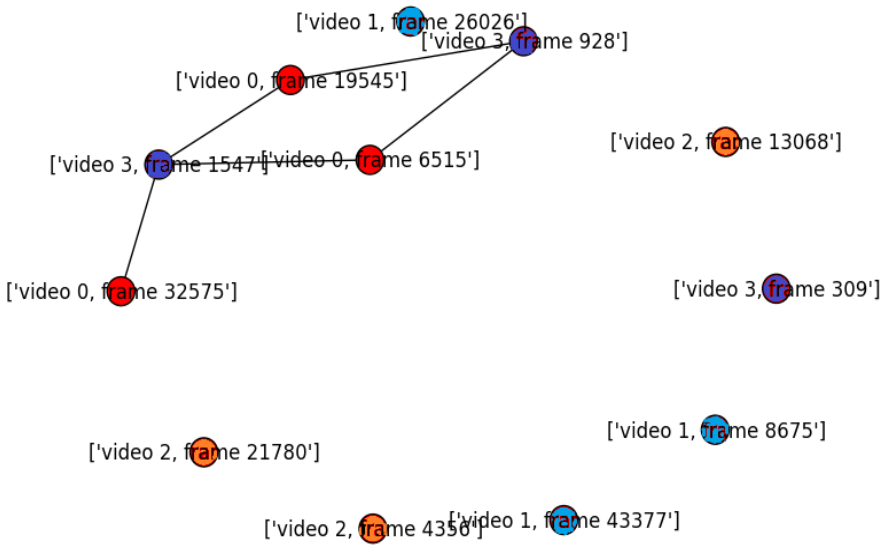


FIGURE 33: GRAPH STRUCTURE TO SHOW THE TRANSFORMATIONS FOUNDS BETWEEN 4 VIDEOS

Notice that we had only four connections, and even there were some videos not connected at all with the rest of the database. Furthermore, we realized about the problematic to choose the "right" transformation. For example, video 3 and video 0 are connected by two different transformations. Which one should we take to estimate their relative position? At this point we realized of two important things. First, videos are not static; they change over time. Thus, they may have more than one relative position over the time. Second, we could find more relations between videos by using the transitivity of our database.

5.2.1.4. Transitivity condition

As said before, videos change over the time. As we can see in Figure 34, the viewpoint for the pair of videos in the first key-frame is not the same than in the last one. In the image there is also represented the transitivity condition.

The transitivity condition states that we can go from one key-frame to any other one of any other video even if that pair of key-frames are not directly connected but by using the transformations of their neighbourhoods as bridges. For example, in the image, there is a way to go from key-frame 339 from the first video to the 1640 of the second video. By using this information we are able to build a graph richer.

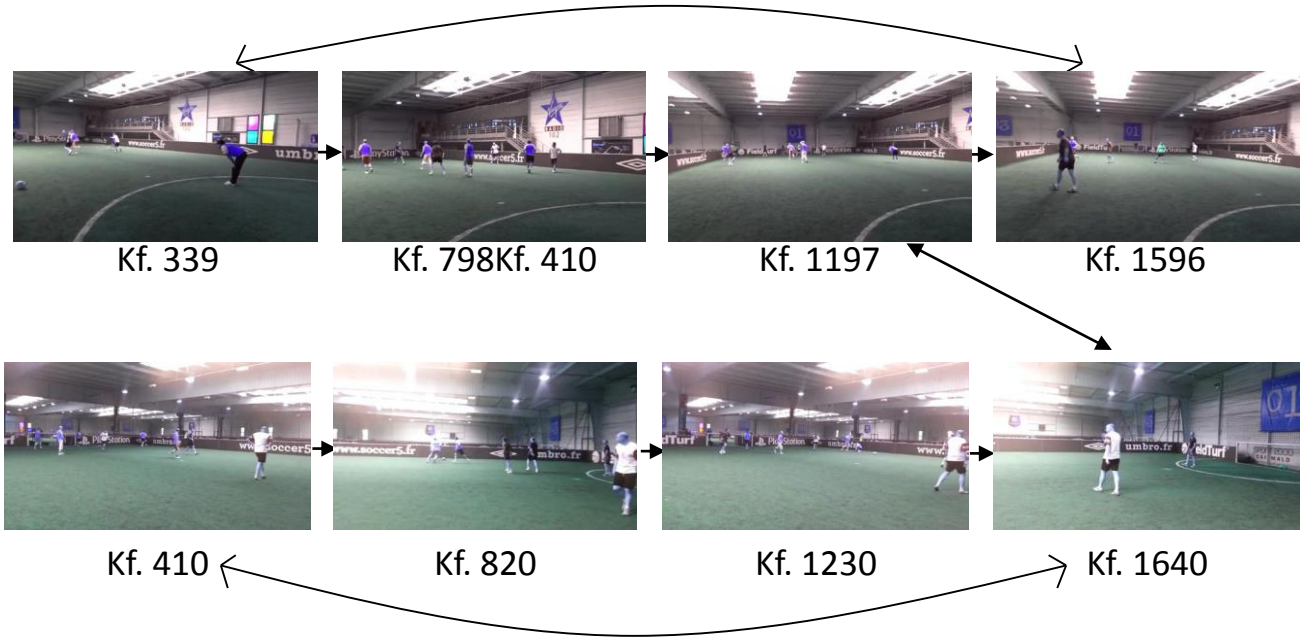


FIGURE 34: EXAMPLE OF VIDEOS WHICH CHANGE THEIR VIEWPOINT OVER THE TIME AND OF THEIR BRIDGES TO GO TO ALL THE OTHER POSSIBLES FRAMES.

5.2.2. Second stage

5.2.2.1. Estimating relative positions

Since to interpret the matrix transformation found was rather complex, we propose a different approach.

As said before, we are not interested in determining the exact position of a video but a general relative position with respect to another one (left, right, up, down, etc.). In order to do that, we propose the following analysis for each link:

We divided each frame to consider into 9 directions, where one of them corresponds to the identity transform (the centre of the space). Hence the image is divided into 9 rectangles (see Figure 35 (left)). The principle is to take two masks with the same dimensions as the two images under consideration. Then, transform the first one with the estimated transformation and to place it over the second frame. By quickly computing the overlapping pixels (those with 1 and 1) we can get for each of the nine rectangles the percentage of overlapping pixels, and thus take a decision on the direction in which we must navigate from the first image to go towards the second one (see Figure 35 (right)) by looking at the rectangle with bigger percentage.

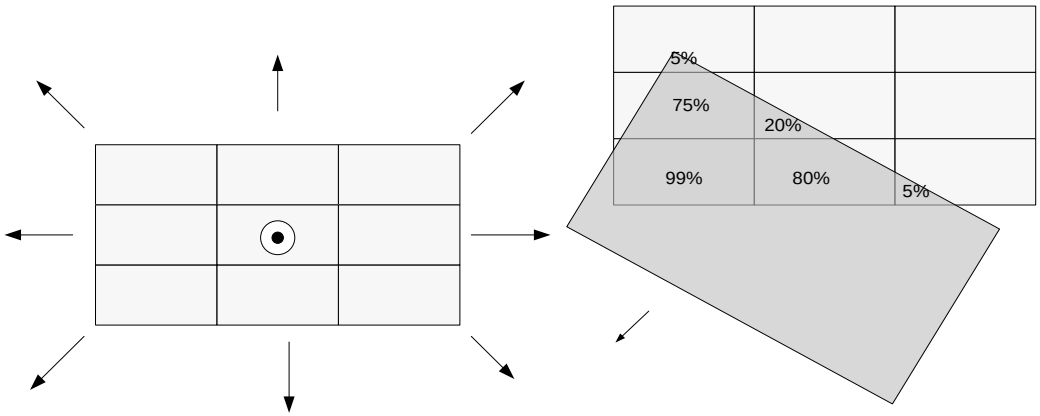


FIGURE 35: COARSE QUANTIZATION OF THE SPACE AND THE IMAGE (LEFT). INTERPRETATION OF THE TRANSFORM (RIGHT) TO GET A RALTIVE POSISIOTNING OF THE IMAGES

When we will be navigating through our collection and suddenly we decide to change our point of view, the system may have different options of videos recorded in that point of view and it will have to choose one of them. In order to estimate the best option, we needed a quality measure to rate our estimations.

This quality measure consists in looking at the sum of distances of the matching process done for each key-frame with the other one transformed. Technically, when we are transforming one of the key-frame with his transformation found, the result should look similar to the other key-frame. Thus, the distances of the matching process should be small.

We have defined a score of quality for each transformation in a way that the lower is the sum of distances of the matching process, the better score it has.

The previous process is iterated for all the pairs of key-frames, hence all the pairs of video segments. A complete graph can then be constructed.

5.2.2.2. Building the graph

Once we have done all these processes for each pair of videos within our database, we can create a graph where the nodes are the videos and the edges are their relative position.

Afterwards, once we have built and tested our algorithm, we computed the graph of the whole database. Results are shown in Figure 36.

Finally, we decided to test with a different event in order to see how our current algorithm performs in different contexts. We decided to use a collection called Roller, which it is about an event in Rennes (France) in the street. The results are shown in next section.

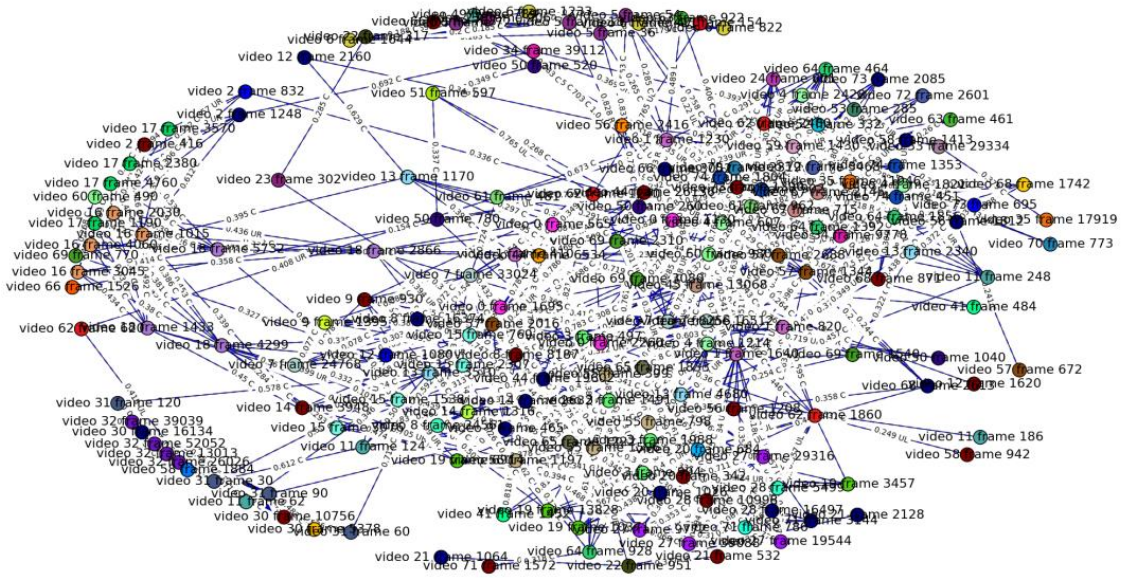


FIGURE 36 GRAPH SOCCER COLLECTION WITH 75 VIDEOCLIPS

5.2.3. Results

Collection	Nb. files	Time to build the graph	Nb. of Edges	Nb. of Nodes	Nb. Key-frames
Soccer	4	440 sec	12	5	3
Soccer	75	39456 sec	2196	151	3
Roller	65	6794 sec	88	56	3

As we can see, results are richer in Soccer collection. Furthermore, when trying with the big database, we find plenty of connections even if we have reduced the number of key-frames in order to speed up the computation. However, Roller collection has much less connections. That is due to the fact that our algorithm is completely based on the Soccer collection, which is recorded in a square and not very big space, with objects repeated over the video clips and videos rather static. On the other hand, Roller sequence is in the street, so the space is much bigger and videos have a lot of movement (Since the most of them are videos recorded with Go-Pro devices). Hence, the system has more problems in finding matches and in the tracking process.

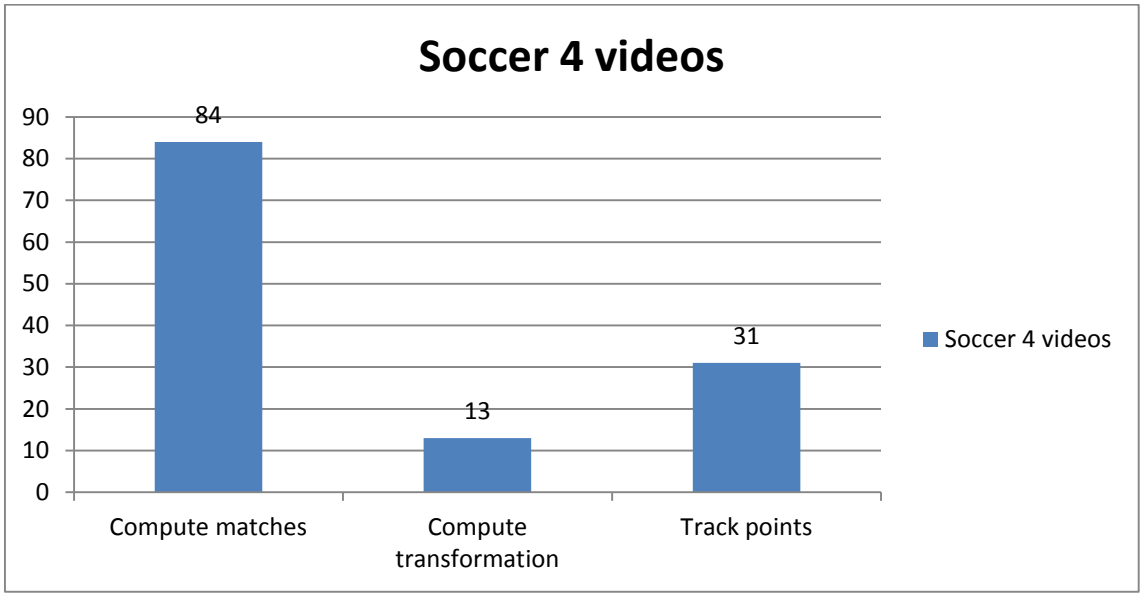


FIGURE 37 NUMBER OF FRAMES FROM SOCCER SMALL COLLECTION DISCARDED IN EACH PROCESS

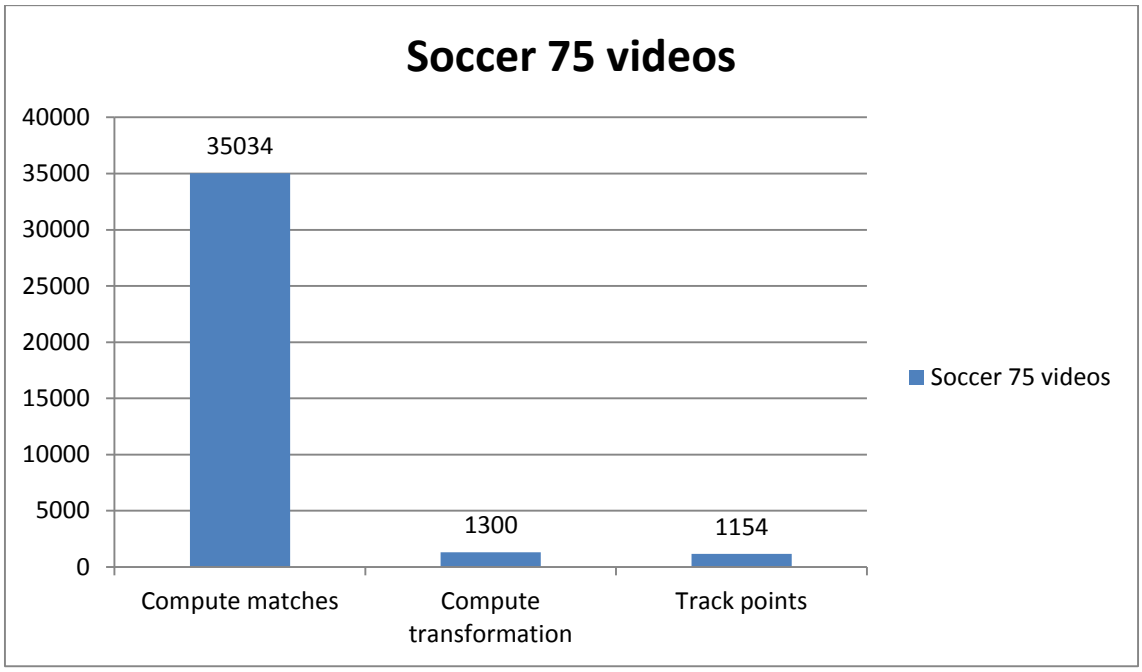


FIGURE 38 NUMBER OF FRAMES FROM SOCCER BIG COLLECTION DISCARDED IN EACH PROCESS

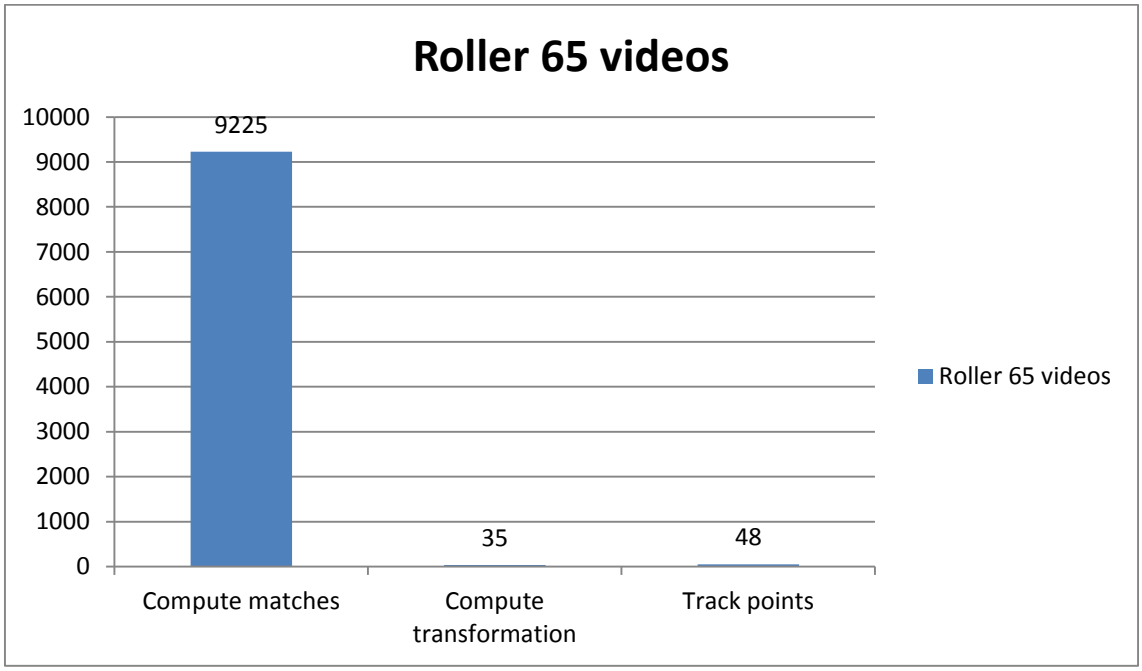


FIGURE 39 NUMBER OF FRAMES FROM ROLLER COLLECTION DISCARDED IN EACH PROCESS

5.2.4. Second state: Setting up our interface

Our final interface is shown in Figure 40. The first thing the user is going to see is a window at the right showing the graph with all the connections. This graph has represented each video in a different row and their key-frames in a different column. The videos are connected by arrows with different colours representing their relative positions (as it is shown in the legend at right).

The user will be able to pick one node and visualize that video in that key-frame in a different window that will show up at the left. Then, he will start navigating through the collection by clicking on the arrows, which will change dynamically in time.

The main idea was not to show the graph to the user, since we think is not obvious and intuitive to him. The idea was that he will only have the window with the video player and with that he will be able to navigate through all the collection even if the nodes are not directly connected (by looking at the transitivity condition). However, since right now we had not enough time to implement this condition, some nodes are not connected between them. Thus, the user may not be able to go to all the videos within the collection. Hence, the graph may be useful when he is visualizing the same clips over and over and he wants to change to a different one.

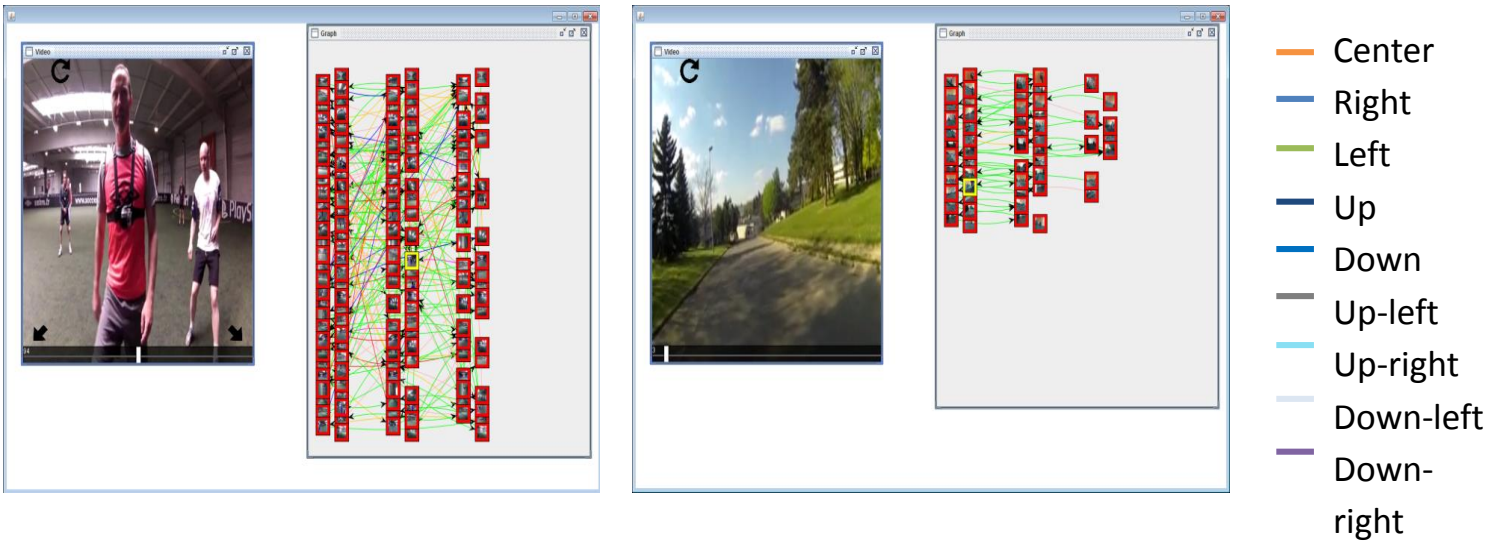


FIGURE 40 VIDEO COLLECTION NAVIGATION BY VIEWPOINTS INTERFACE

6. CONCLUSIONS

In summary, in this project we have proposed to develop a novel system for navigating video collections uploaded from a common social event by multiple users, with the goal of providing richer ways of experiencing crowd-generated content. We have analysed, designed and implemented novel forms of interactive navigation. Finally, we have set up two versions of functional navigation interfaces that are intuitive, ready-to-use and end-user oriented.

Interface. In this work, different kinds of interfaces have been analysed. It has been proposed different solutions to the problem of what the user may be interested to see when he wants to navigate through any video collection. We have discarded those ideas that did not improve the human-machine interaction (since we want the user to decide what he wants to play at any time) and those that offered more disadvantages than advantages.

Graph structure. We decided to use Graphs analysis techniques because they are a well-known method for studying data connectivity. As we have seen, graphs are used in various image clustering and labelling applications. They are also good to capture different structure in the collection (as in [2]). Thanks to them we have been able to compute the best transformation to change from one video to another one in our last version and also to structure video overlap connectivity and hence use it as part of the interface to let the user navigate his video collection.

User experience. After looking advances in state of the art we wanted to set up a navigator that helps the user through his experience of navigating a video collection. We think that if the system plays multiple videos at the same time, it is necessary to indicate to the user at what video he should look at in order to not miss any detail.

Methods and algorithms. In Video Collection Navigation by viewpoints, we have used different video analysis techniques such as nearest neighbour match, point tracking and spatial transformations in order to estimate relative position between video-clips within a video collection. We have used methods as SIFT, RANSAC and FLANN because they are well-known in video/image processing techniques as methods that give convenient results in a wide range of cases and because their simplicity. Affine transformations were proved to be more convenient for us rather than Homography ones, even if they are less accurate.

We have also talked about our criteria to consider videos as a dynamic thing rather than static and thus, consider that each pair of videos may have more than one relative position.

Finally, we have seen that to evaluate our results is more complicate than expected since sometimes the results are quite subjective. We have seen that even if the system finds the same objects and therefore it finds their relative positions this does not mean that it is right, because we can understand the space and see that it was not the correct match. It was also subjective the

way to interpret the transformation matrices found, thus we decided to use an alternative to compute their relative positions.

7. FUTURE WORK

There is a number of further works that were not implemented in this project due to time constraints, and they include the following ones:

Transitivity condition. As stated before, we can build a richer graph if we look at the information of the neighbourhood of each node in the graph. Thus, we should be able to go to any direction at any time of the video. Right now, our interface only allows the user to go to those nodes directly connected to it.

Navigation through all the collection. One of our goals is that the user should be able to watch all his database. Right now, that may not happen since the system will always display the same videos because it will always look at the option with better score. Hence, it may do always the same path of videos in the graph while other nodes will never be explored.

Test on real users. Another important aspect is to test our final system with different users in order to see if they are able to locate themselves in the interface without problems, if they find interesting this approach to navigate their collections and to detect issues.

Graph analysis techniques. There are a lot of analytical methods that can be used on graphs and we are sure that our graph contains a lot of information about our collection. It would be interesting to find out why some nodes are only connected to a small set and not to other ones, why do we have sets of nodes connected and so on. This information may help us to improve the user-experience.

Test different databases. As we have seen, results have changed a lot from Soccer collection to Roller collection. It is important to test different contents in order to adjust our system in a way that it can work for any type of collections.

Interface improvements. There are some things we would also like to do in our interface. For example, it would be interesting to implement a backward option. Right now, the user may not be able to go to the previous node when clicking on the arrow on the opposite direction of before. For example, if he clicks on the right and then it clicks on the left, it may finish in a different video than the one in the beginning, since the system may have another video with a better score on the left. However, the user may want to go backwards, so this option is important.

Finally, another thing we thought it could be interesting is to implement a Summary mode. Thus, the user would be able to have an overall idea of all his content by watching, for instance, two second of each node by walking through the graph by looking, for example, its optimal path.

8. Bibliography

[1] James Tompkin, Kwang In Kim, Jan Kautz, Christian Theobalt (2012). *Videoscapes: exploring sparse, unstructured video collections*. SIGGRAPH.

[2] Heath, K.; Gelfand, N.; Ovsjanikov, M.; Aanjaneya, M.; Guibas L.J. (2010). *Image webs: Computing and exploiting connectivity in image collections*. CVPR.

[3] Jörg Walter, Daniel Wessling, Kai Essig, and Helge Ritter (2006). *Interactive hyperbolic image browsing - towards an integrated multimedia navigator*. ACM MDM/KDD.

[4] Lyndon Kennedy, Mor Naaman (2006). *Less Talk, More Rock: Automated Organization of Community-Contributed Collections of Concert Videos*. WWW.

[5] Chong-Wah Ngo; Yu-Fei Ma; Hong-Jiang Zhang (2005). *Video summarization and scene detection by graph modelling*. Circuits and Systems for Video Technology

[6] Tron, R.; Vidal, R (2009*). Distributed image-based 3-D localization of camera sensor networks*. Chinese Control Conference,.

[7] David G. Lowe "Distinctive image features from scale-invariant keypoints." International Journal of Computer Vision, 2004.

[8] Marius Muja, David G. Lowe (2009). *Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration*. VISAPP.

[9] M. A. Fischler, R. C. Bolles (1981). *Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography.* Comm. of the ACM.

[10] MacQueen, J. (1967). *Some methods for classification and analysis of multivariate observations*. Fifth Berkeley Symposium on Mathematical Statistics and Probability.

[11] J. Shi and J. Malik (2000). *Normalized cuts and image segmentation*. IEEE Trans. Pattern Anal. Machine Intell.

[12] Tompkin, James and Pece, Fabrizio and Shah, Rajvi and Izadi, Shahram and Kautz, Jan and Theobalt, Christian (2013). *Video Collections in Panoramic Contexts*. UIST.

Appendix

a. Software

In our project we have worked with Java libraries for setting up V0, such as JUNG[2] and Processing[3], a programming language ideal for interactive programs and open source, in order to implement the GUI, since Processing has a lot of tools for displaying videos and letting human-machine iteration in an easy way and it is possible to use it inside a Java project.

In V1, we have done the first stage by using Python and its libraries from openCV. Its second stage was again with Java and Processing.

When we need to read or to write our information in a graph structure, we use the graphml format[4]. It is a comprehensive format for storing and reading information in a graph structure and it is understood for GraphML[5] library in Java, or for NetworkX[6] in Python, making it very appropriate for us. In Figure 41 we can see an example of GraphML file (it is a small part of the one used in V0).

b. Hardware

All our tests and implementations have been done in a Windows 7 machine, with an Intel(R) Xeon(R) CPU processor (of 2.13GHz) and a RAM of 12GB. It is a 64-bit system.

---

[2] http://jung.sourceforge.net/

[3] http://www.processing.org/

[4] http://graphml.graphdrawing.org/

[5] https://networkx.github.io/documentation/latest/_modules/networkx/readwrite/graphml.html

[6] https://networkx.github.io/documentation/latest/_modules/networkx/readwrite/graphml.html

```
<?xml version="1.0" encoding="UTF-8"?><graphml xmlns="http://graphml.graphdrawing.org/xmlns">
<key attr.name="label" attr.type="string" for="node" id="label"/>
<key attr.name="Edge Label" attr.type="string" for="edge" id="edgelabel"/>
<key attr.name="weight" attr.type="double" for="edge" id="weight"/>
<key attr.name="Edge Id" attr.type="string" for="edge" id="edgeid"/>
<key attr.name="x" attr.type="float" for="node" id="x"/>
<key attr.name="y" attr.type="float" for="node" id="y"/>
<key attr.name="size" attr.type="float" for="node" id="size"/>
<key attr.name="Path" attr.type="string" for="node" id="Path"/>
<key attr.name="FirstFrame" attr.type="biginteger" for="node" id="FirstFrame"/>
<key attr.name="LastFrame" attr.type="biginteger" for="node" id="LastFrame"/>
<graph edgedefault="undirected">
<node id="121">
<data key="label">clip_1</data>
<data key="Path">\\RENNDW7RDL3016\StageTemp\Data\Soccer\10362-100000004-
00003_LR.mp4</data>
<data key="FirstFrame">-1</data>
<data key="LastFrame">-1</data>
<data key="size">10.0</data>
<data key="x">340.22806</data>
<data key="y">-468.91437</data>
…..
<edge source="109" target="121">
<data key="weight">1.0</data>
</edge>
```

FIGURE 41: SLICE OF GRAPML FILE USED IN V0