UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

INTERACTIVE MEDIA SYSTEMS Vienna University of Technology

KEYFRAME-BASED VIDEO SUMMARIZATION DESIGNER

A Degree Thesis

Submitted to the Faculty of the

Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona

Universitat Politècnica de Catalunya

by

Carlos Ramos Caballero

In partial fulfilment

of the requirements for the degree in

AUDIOVISUAL SYSTEMS ENGINEERING

Advisors: Horst Eidenberger and Xavier Giró I Nieto

Barcelona, July 2015

Abstract

This Final Degree Work extends two previous projects and consists in carrying out an improvement of the video keyframe extraction module from one of them called Designer Master, by integrating the algorithms that were developed in the other, Object Maps.

Firstly the proposed solution is explained, which consists in a shot detection method, where the input video is sampled uniformly and afterwards, cumulative pixel-to-pixel difference is applied and a classifier decides which frames are keyframes or not.

Last, to validate our approach we conducted a user study in which both applications were compared. Users were asked to complete a survey regarding to different summaries created by means of the original application and with the one developed in this project. The results obtained were analyzed and they showed that the improvement done in the keyframes extraction module improves slightly the application performance and the quality of the generated summaries.

Resum

Aquest Treball Final de Grau és una extensió de dos projectes previs i consisteix en la millora del mòdul d'extracció de keyframes d'un d'ells anomenat Designer Master, mitjançant la integració d'algoritmes desenvolupats en l'altre, Object Maps.

En primer lloc s'explica la solució proposada, la qual consisteix en un mètode basat en la detecció d'escena o shot. Primerament el vídeo és mostrejat uniformement, seguidament s'aplica el mètode de diferència acumulada pixel-to-pixel i finalment un decisor decideix quins frames són o no keyframe.

Per últim, s'analitzen les puntuacions obtingudes per diversos usuaris en el procés d'avaluació, als quals se'ls hi ha presentat diferents resums creats amb l'aplicació original i amb la desenvolupada en aquest projecte. Els resultats mostren que la millora introduïda en el mòdul d'extracció millora lleugerament el rendiment de l'aplicació i la qualitat dels resums que es poden generar.

Resumen

Este Trabajo Final de Grado es una extensión de dos proyectos previos y consiste en la mejora del módulo de extracción de keyframes de uno de ellos, cuyo nombre es Designer Master, mediante la integración de algoritmos desarrollados en el otro, llamado Object Maps.

En primer lugar se explica la solución propuesta, la cual consiste en un método basado en la detección de escena o shot. Primeramente el video es muestreado uniformemente, acto seguido se aplica el método de diferencia acumulada pixel-to-pixel y finalmente un decisor se encarga de decidir qué frames son o no keyframe.

Por último, se analizan las puntuaciones obtenidas por diversos usuarios en el proceso de avaluación, a quién se les ha presentado varios resúmenes creados con la aplicación original y con la desarrollada en este proyecto. Los resultados muestran que la mejora introducida en el módulo de extracción mejora ligeramente el rendimiento de la aplicación y la calidad de los resúmenes que se pueden generar.

This thesis is dedicated to my parents and grandparents. For their endless love, support and constant encouragement I have got over the years.

Acknowledgements

First, I would like to thank my project advisors, Professor Horst Eidenberger, for hosting me at Vienna University of Technology and Professor Xavier Giró i Nieto for all the support and encouragement he gave me during my thesis. Their advices and comments helped me a lot in the development and writing of the thesis.

I would also like to thank Andreas Waltenberger and Manuel Martos for meeting me at the beginning of this project when I was absolutely lost and stuck. All your help and feedback have been absolutely invaluable for me.

To all my friends and colleges, thank you for your understanding and encouragement in my many, many moments of crisis. Your friendship makes my life a wonderful experience. I cannot list all the names here, but you are always on my mind.

To my girlfriend, for all the support, patience and love. Thank you so much for being there every single day throughout my staying in Wien.

Finally, I would also like to thank my family for all of their support and inspiration over the years. Suffice to say, without them, none of this would have been possible.

This thesis is only the beginning of my journey.

Revision history and approval record

Revision	Date	Purpose
0	24/06/2015	Document creation
1	08/07/2015	Document revision
2	10/07/2015	Document approval

DOCUMENT DISTRIBUTION LIST

Name	e-mail
Carlos Ramos Caballero	carlosart_74@hotmail.com
Horst Eidenberger	eidenberger@tuwien.ac.at
Xavier Giró I Nieto	xavier.giro@upc.edu

Written by:		Reviewed and approved by:	
Date	24/06/2015	Date	10/07/2015
Name	Carlos Ramos Caballero	Name	Xavier Giró i Nieto
Position	Project Author	Position	Project Supervisor

Table of contents

List of Figures

List of Tables

1. Introduction

Digital video has become an emerging force in current computer and telecommunication industries for its large mass of data. This fact involves a lot of data that many times users do not have time to review. Therefore, it is important to find a method able to efficiently summarize this large amount of data. For this purpose, the most widespread proposal is the extraction of keyframes. Video keyframes provide a concise access to the video content. It maps an entire video segment to a small collection of representative images. The extraction of keyframes should be automatic and content based so that they could maintain the salient content of the video while avoiding the redundancy.

On the other hand, images are usually represented by thumbnails for a faster image browsing. In addition, new portable devices, such as smartphones or tablets, increase the accessibility and production of videos through social networks and user-generated content sites. The most common way to access video search results is by making use of textual metadata but it is not always the best option to summarize a video. Shared content requires efficient retrieval technologies to access the content properly in a fast and intuitive way.

This thesis addresses the problem of video content summarization using Shot-based methods to extract the keyframes, analyzing the video and helping users to understand a video content item in a fast and visual way.

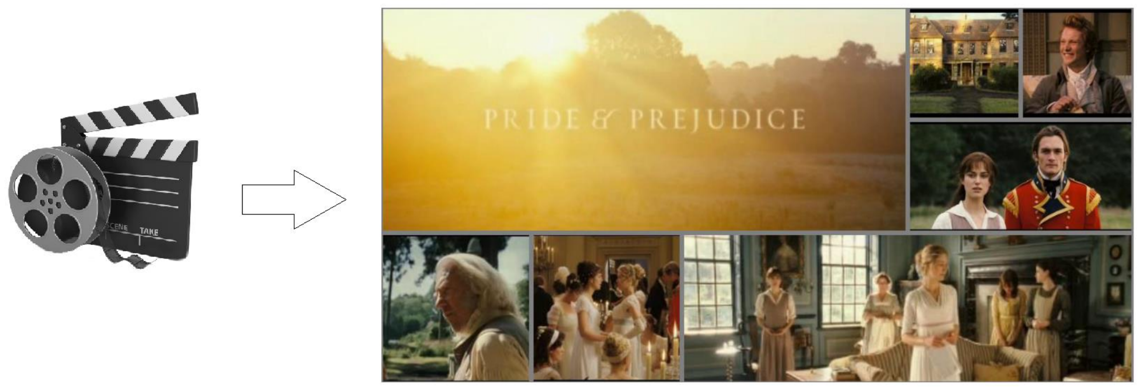


Figure 1. One-image video summary after drag and dropping images in tiles.

1.1. <u>Goals of the thesis</u>

This thesis is an extension and adaptation of the previous work carried out by two students, supervised by the professor H. Eidenberger and performed in the framework of the department research.

The purpose of this project is to improve Designer Master, an existing Java desktop application which was implemented by the student Andreas Waltenberger [1]. This improvement has to be done by using the tools developed by Manuel Martos [2] in his final thesis in order to perform a more accurate summary than the existent one.

The project main goals are:

- Improving the keyframe extraction block in the existing tool.
- Assessing the improvement according to a scientific methodology.

1.2. <u>Motivation</u>

This project starts when Professor Xavier Giró from the Image Processing Group at the UPC (Universitat Politècnica de Catalunya) put me in contact with Professor Horst Eidenberger from the Interactive Media Systems Group at the TUW (Technische Universität Wien) in order to perform my bachelor thesis.

Professor Eidenberger was the advisor of Manuel Martos and Andreas Waltenberger's thesis. Manuel developed a system for automatic video summarization which made use of shot boundary detection algorithms required to reduce time redundancy of the video and different detectors to extract relevant objects from each keyframe. Otherwise, Andreas developed a keyframe-based video summarization interface that allows the user to design a customized one-picture video summary from extracted keyframes by drag and dropping arrangement in tiles. Keyframe extraction is made uniformly and therefore, the images are extracted without taking into account if they represent well an event, shot or a given video sequence.

Hence, professor Eidenberger proposed to carry out an integration of both projects, improving the keyframe extraction block of the Andreas' application.

1.3. Requirements and specifications

Project requirements:

- Development of semantic-aware keyframe extraction algorithms for our application.
- Replacement of the uniform keyframe extraction block of the current UI by integrating Manuel Martos' algorithms.
- The extraction block must work well for videos as generic as possible.
- Comparisons with other state of the art solutions.

Project specifications:

- The current UI, Keyframe-Based Video Summarization Designer requires the following software to be installed:
  - Ant (Version >= 1.9.0)
  - Java Development Kit (Version >= 8)

- It must be developed in Java programming language.
- Taking advantage of the OpenCV library.
- Real-time application (computational cost as low as possible).

The whole project has been developed in Java programming language for two reasons: In the first place, Java is a general-purpose computer programming language that is concurrent and object-oriented specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. In the second place, due to the previous projects to be integrated were written in Java.

Otherwise, we used OpenCV due to it is a powerful open source computer vision library and it was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products, which it is ideal for real-time purposes.

1.3.1. Work packages

WP1: Documentation

WP2: State of the art

WP3: Software

WP4: Datasets

WP5: Results assessment

WP6: Oral defense

In order to analyze more in detail the work packages, we make reference to CRamos_work_plan.pdf. This document has not been modified since it was created, thus initial work plan can be compared with the final plan.

1.3.2. Incidences and deviations

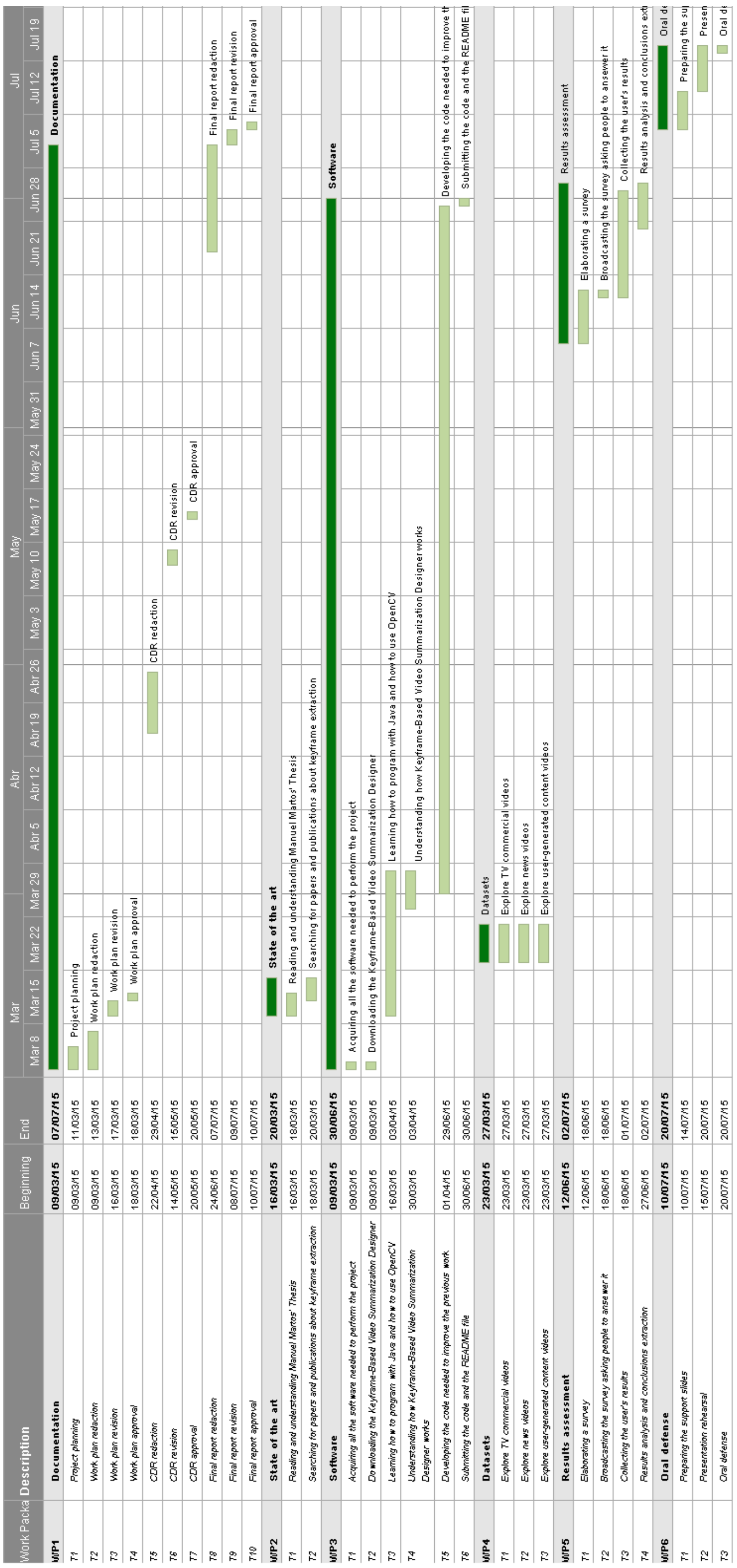
As already mentioned in the critical review document, we had several problems throughout the installation and execution of the state of the art software.

The first problem I had was while studying Designer Master code, I did not be able to run the program due to there were missed libraries when I downloaded the .RAR archive from the Interactive Multimedia Systems (Vienna University of Technology) repository. To solve that, I downloaded another .RAR archive from the GitHub repository with the complete package. Once I loaded the project into the IDE, there were more problems that didn’t let the application work properly such as missed paths to libraries that we had to add manually.

The second problem we had was while studying Manuel Martos’ thesis, this time it was compatibility problems with Java due to the included libraries were compiled using 32bits and objectMaps.jar application can only be executed using a 32bit JRE. Once all the programs were installed and working properly, it was needed a learning process about how to program with Java and how to make use of the OpenCV libraries as well as how Designer Master and Object Maps’ code works. This process took me a bit longer than expected and as a consequence, some tasks were delayed.

Another source of delaying was while acquiring the databases we will use to test the application. It was difficult to download them from Windows, so I had to install a Linux virtual machine in my computer to be able to get them and copy them back to Windows.

1.3.3. Time plan (Gantt diagram)



Work Packa	Description	Beginning	End
WP1	**Documentation**	**09/03/15**	**07/07/15**
T1	*Project planning*	09/03/15	11/03/15
T2	*Work plan redaction*	09/03/15	13/03/15
T3	*Work plan revision*	16/03/15	17/03/15
T4	*Work plan approval*	18/03/15	18/03/15
T5	*CDR redaction*	22/04/15	29/04/15
T6	*CDR revision*	14/05/15	15/05/15
T7	*CDR approval*	20/05/15	20/05/15
T8	*Final report redaction*	24/06/15	07/07/15
T9	*Final report revision*	08/07/15	09/07/15
T10	*Final report approval*	10/07/15	10/07/15
WP2	**State of the art**	**16/03/15**	**20/03/15**
T1	*Reading and understanding Manuel Martos' Thesis*	16/03/15	18/03/15
T2	*Searching for papers and publications about keyframe extraction*	18/03/15	20/03/15
WP3	**Software**	**09/03/15**	**30/06/15**
T1	*Acquiring all the software needed to perform the project*	09/03/15	09/03/15
T2	*Downloading the Keyframe-Based Video Summarization Designer*	09/03/15	09/03/15
T3	*Learning how to program with Java and how to use OpenCV*	16/03/15	03/04/15
T4	*Understanding how Keyframe-Based Video Summarization Designer works*	30/03/15	03/04/15
T5	*Developing the code needed to improve the previous work*	01/04/15	29/06/15
T6	*Submitting the code and the README file*	30/06/15	30/06/15
WP4	**Datasets**	**23/03/15**	**27/03/15**
T1	*Explore TV commercial videos*	23/03/15	27/03/15
T2	*Explore news videos*	23/03/15	27/03/15
T3	*Explore user-generated content videos*	23/03/15	27/03/15
WP5	**Results assessment**	**12/06/15**	**02/07/15**
T1	*Elaborating a survey*	12/06/15	18/06/15
T2	*Broadcasting the survey asking people to answer it*	18/06/15	18/06/15
T3	*Collecting the user's results*	18/06/15	01/07/15
T4	*Results analysis and conclusions extraction*	27/06/15	02/07/15
WP6	**Oral defense**	**10/07/15**	**20/07/15**
T1	*Preparing the support slides*	10/07/15	14/07/15
T2	*Presentation rehearsal*	15/07/15	20/07/15
T3	*Oral defense*	20/07/15	20/07/15

Figure 2.- Gantt Diagram

2. State of the art of the technology applied in this thesis

In this chapter, we describe the video summarization techniques employed today to achieve new levels of understanding. The main goal of this project is to improve the keyframe extraction block of the current application, Designer Master. Thus, the first section is an explanation of the existing types of video summarization techniques. In section 2.2 we review some color image models. Finally in section 2.3, we explain the technologies involved in the process and discuss several temporal segmentation methods.

2.1. Definitions

A video summary can basically take two forms: *static image summary* and *moving-image skimming*. Video summarization aims to allow users to access video content easily, providing concise video summaries. This field has received more attention in recent years due to new utilities such as social networks or portable devices.

2.1.1. Moving-image skimming

The *moving-image skimming*, also known as *dynamic video skim*, consists of a collection of video clips, as well as the corresponding audio segments extracted from the original sequence and is thus itself a shorter version of the original video. They can be classified into two types: *Overview* and *Highlight*.

In the classic case of movie trailers, the user is usually unaware of the content and is interested in a much reduced summary of the video content to decide before watching the full versions. We call this kind of video skimming *overview*. For a specific domain like news or sports, users want to see the most important events in the video (goals, news headlines) according to their interests. This type is called *highlight*. Unlike *overviews,* which are presented as single condensed videos, highlight-based summaries are usually presented as an organized list of interesting events along with some associated metadata.

2.1.2. Static summaries

A *static summary*, also known as *storyboard summary*, is a small collection of salient images or a single one extracted or generated from the underlying video source. According to the method used to extract representative images, we can classify static video summaries into sampling-based, shot-based, motion-based, mosaic-based and object mapping methods.

Sampling-based methods select video keyframes by random or by uniform sampling of the input video. For *shot-based* methods, the source video is temporally segmented into shots using shot boundary detection algorithms. *Motion-based* methods refer to the temporal dynamics of the video by motion analysis using image pixel differences or optical flow. When the camera motion can be detected, a *mosaic image* can be constructed to represent the whole content of a dynamic shot. Finally, object mapping aims to extract relevant objects from the source video to create a composite image.

2.2. Color Models

In this section we review the four types of accepted colour models in our shot detection system.

2.2.1. RGB color model

The RGB color model is widely used throughout computer graphics. This model specifies the intensity of the three primary colors: red, green, and blue on a scale of 0 to 255, with 0 (zero) indicating the minimum intensity. The three primary colors and their combination in visible light spectrum are shown in Fig.3. With different weights, (R, G, B), their combination can indicate different colors and they are represented by a three-dimensional, Cartesian coordinate system as shown in Fig.4. The colors on the diagonal line, from the origin to the coordinate (1, 1, 1) of the cube, means the grey-level values [7] [9].

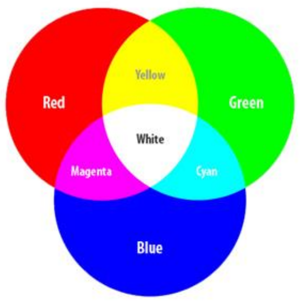


Figure 3.- RGB graph of the primary colors.

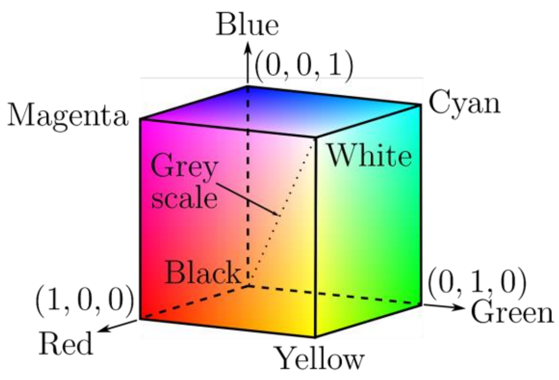


Figure 4.- RGB primary color cube.

The RGB color model is the most prevalent choice for computer graphics due to color displays use red, green and blue to create the desired color. Therefore, the choice of this space simplifies the architecture and design of the system.

2.2.2. CMY color model

The CMY color model is based on complementary colors: cyan, magenta, yellow. This color model can be expressed as:

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

(1)

Fig. 3 shows the relationship of the component color of the CMY color model. The CMY color model is applied to the output devices, such as printers [8].

2.2.3. YIQ color model

The YIQ color model is designed to refer to the characteristics of the human's visual system. In the human's visual system, people are more sensitive to the lightness component than the hue component. So, the YIQ color model is set to separate colors into luminance (Y) and hue (I and Q) [8] [7]. The relationship between YIQ and RGB is expressed as:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

(2)

Where Y is the luminance, I and Q indicate the weights of hue.

The advantage of the YIQ color model is that we can deal with the luminance component independently. The YIQ color model is the standard model applied to the signal transmission of color TV sets.

2.2.4. YUV color model

The YUV color model is also considered to be similar to human eye's retina. The main channel, luminance, denoted as Y channel, describes the intensity of light. Chrominance components, called U and V, carry the color information [7] [9]. The relationship between YUV and RGB is expressed as:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

(3)

2.3. Shot segmentation

Temporal redundancy is a very important issue that needs to be solved when facing video processing. Deleting redundant information is achieved by segmenting the video into shots. A shot is a continuous recording of video content without breaks in a scene. Then, keyframes may be extracted from each shot with different techniques based on pixel-to-pixel comparison, histogram-based comparisons, motion flow vectors, etc. This process is called *Shot Boundary Detection.*

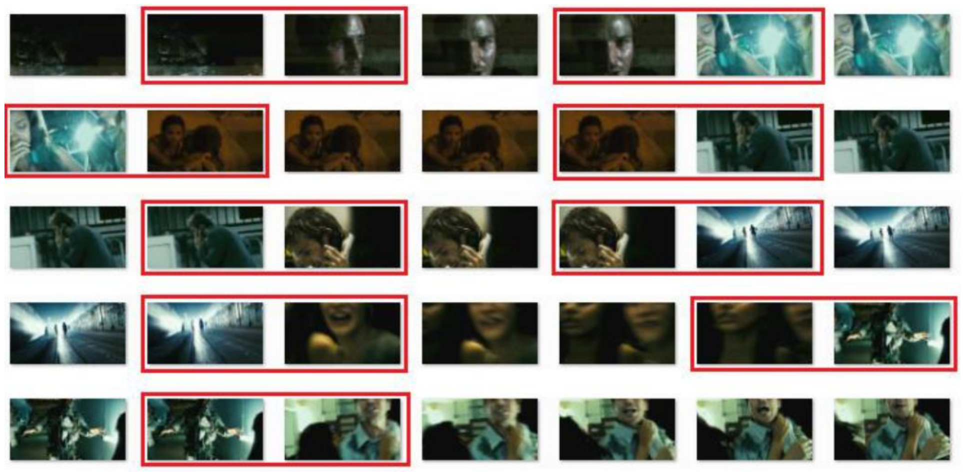


Figure 5.- Shot boundary detection example [2].

Pixel-to-Pixel methods are the core methods and probably the most straightforward ones [3]. Indeed, the first idea that comes to mind when we want to compare two images in terms of similarity is to compare their pixels.

Histogram-based methods get better reflection of global properties of a picture, which is their main advantage [4]. These techniques are significantly more robust against camera and object motion. However, there are drawbacks: a shot boundary occurring in two frames with similar histograms will be missed; also, significant luminance difference between frames will declare false positives in shot boundary detection.

Histograms may be compared in different ways [5]. A first approach would be to calculate the histogram of each color channel that form the image and, then, calculate the difference between the bins in each histogram of the two successive frames. Another technique is to calculate the difference of all channels between the histograms in the two images and take the maximum to the sums in order to detect significant changes in one channel. Finally, a variation of the last technique is to weight the importance of each color channel.

A method that uses Hausdorff approximation to determine the outliers is used in [5]. The Hausdorff method performs an edge detection process of the image and compares the location of the edge points produced by the edge detector. The method checks for each point whether a corresponding edge exists in the successive image. If the sum of non-correlated edges is greater than some threshold, a shot boundary is declared.

[5] also presents a combination of all the commented methods by building an ensemble method, *Neural Network* (NN). The inputs are the outputs of the different methods with a supervised learning process to easily adapt results for different type of videos. Weaknesses of each method are compensated by the others and the NN is adapting to any given threshold by propagating the errors to its weights.

More recent techniques include a higher-level segmentation of videos into scenes. Rasheed and Shah [6] present a method based on a graph partitioning problem that clusters shots into scenes constructing a graph called *shot similarity graph* (SSG). Each node represents a shot and the edges between them are weighted based on their similarity based according to color and motion information. Then, the SSG is split into sub-graphs by applying *normalized cuts* representing individual scenes. They also propose a method to describe the content of each scene by selecting a representative keyframe.

To sum up, there exist several shot segmentation techniques:

- Simple approaches compare pixel intensity and image histogram to decide whether two frames belong to the same shot.
- Later approaches include edge evaluation and comparison between frames using Hausdorff distance.
- Learning processes using NN are also used to adapt the shot detection to the source video regardless of thresholds.
- Recent techniques use clustering methods to group similar frames based on pixel color, motion flow information, etc.

In the next subsections we describe in detail the approaches that were analysed throughout the thesis development.

2.3.1. Software Initiative Studies at UCSD

In this approach, each frame is divided into NxN regions. Then, the pixel change is estimated for each region and pairs of frames. If the pixel change is greater than some threshold and its cumulative sum is greater than the region threshold for the frame, then it triggers the shot boundary detection. This technique also provides a simple frame averaging to avoid luminance changes that could be detected as a shot boundary. This pixel-to-pixel method combines low computational requirements with satisfactory results, but also tends to generate some false detection, which generate an over-segmentation of the video.

2.3.2. Course Project of Binshtok and Greenshpan at BGU

This second software kit was developed by Max Binshtok and Ohad Greenshpan [5], two students at the Ben-Gurion University of the Negev (BGU) in Israel. The proposed software includes four different algorithms for the shot boundary detection: a pixel-to-pixel method, a histogram-based method, a third one based on the Haussdorf distance, and a learning process based on NN.

There are many types of pixel comparisons used in the approach:

- *Global Pixel-to-Pixel*: This method sums the pixels' intensity values over the whole image, and compares it to the sum of the pixels' intensity values in the second image as shown in equation (4).

$$\frac{\left|\sum_{i=1}^{X}\sum_{j=1}^{Y} I(t,i,j) - \sum_{i=1}^{X}\sum_{j=1}^{Y} I(t+1,i,j)\ \right|}{256 \cdot X \cdot Y} > \tau \qquad (4)$$

  $I(t,i,j)$ represents the intensity value of pixel (i,j) at time. If the difference is bigger than some threshold (τ), a shot detection is declared. It is obvious that the local differences between pixels' intensity values are ignored.

- *Cumulative Pixel-to-Pixel*: This method sums the differences between each pixel's intensity value in one image and its intensity value in the successive image. We take into consideration local details in the images as shown in (5).

$$\frac{\sum_{i=1}^{X}\sum_{j=1}^{Y}|I(t,i,j)-I(t-1,i,j)|}{256\cdot X\cdot Y}>\tau \tag{5}$$

The histogram-based methods compare the pixel histograms of neighboring frames to determine the shot boundaries. They introduce robustness against camera and object motion, but they fail at segmenting two shots whose colors are similar. Important methods are:

- *Simple histogram:* This method calculates one histogram per color channel that form the image and compute the difference between the bins in each histogram of the two images using (6).

$$\frac{\sum_{C\epsilon\{channels\}}\sum_{b=0}^{Bins}|H(t,c,b)-H(t-1,c,b)|}{|pixels|\cdot|channels|\cdot 2}>\tau \tag{6}$$

$H(t,c,b)$ represents the histogram value of the bin *b* in the color channel *C* at time *t*.

- *Max histogram*: This method calculates the differences over all channels between histograms in the two images and takes the maximum of the sums. It can be influenced by an intense change in one channel as shown in formula (7).

$$\frac{\max\limits_{C\epsilon\{channels\}}\sum_{b=0}^{Bins}|H(t,c,b)-H(t-1,c,b)|}{|pixels|\cdot 2}>\tau \tag{7}$$

- *Weighted histogram*: It also takes into account the histograms' difference in all channels and gives each one a weight, determined by luminance proportions of the channel, thus giving more weight to the prevalent color channel in the image as shown in (8).

$$\frac{\sum_{C\epsilon\{channels\}}\sum_{b=0}^{Bins}\frac{w_c}{w_{mean}}|H(t,c,b)-H(t-1,c,b)|}{|pixels|\cdot|channels|\cdot 2}>\tau \tag{8}$$

The *Hausdorff method* performs an edge detection process with the Sobel operator on the images and compares the locations of these points between frames. It is a good approximation to get the same face or object twice if there exists any smoothing or view improvements.

Finally, Binshtok and Greenshpan's thesis states that the option that combines the three methods using a neural network provides the best results for the typical keyframe extraction.

3. Methodology / project development:

After considering the state of the art and the requirements that our tool have to fulfill, we specify the approach design. Furthermore, we describe the implementation of the elements.

This chapter is structured as follows: Section 3.1 provides an explanation about the implemented solution. In section 3.2 we explain the development environment and finally, in section 3.3 we explain how to use the application.

3.1. Implemented solution

3.1.1. Overview

Video shot boundary detection is the first and most important step in the video processing framework. We have to remark that our proposed solution is not exactly a shot boundary detector but it is inspired on it. Hence, in order to achieve the specific purpose of our application we have decided to implement a pixel to pixel method to carry out the keyframe extraction task. Concretely, we have integrated the *cumulative pixel to pixel difference method* from the software kit resulting from a course project by Max Binshtok and Ohad Greenshpan, two students at the Ben-Gurion University of the Negev (BGU) in Israel that it was already used in Manuel’s thesis.

The reasons why we have chosen this method are: in the first place, because it is well known and it has been widely studied and despite its relative simplicity, it produces good results [5] [2]. Secondly, because it does not require too much computational effort which it is ideal for the accomplishment of the real-time requirement of the application. Finally, because with this setting Object Maps got its best performance. However, it is easily adjustable in the source code, so another Shot-based method could be applied.

Cumulative pixel to pixel method is sensitive to object motion and other local changes in the scene which means that naturally generates over segmentations of the videos due to changes in luminosity or points of view. Thus, it might provide different views of the same object which it is desirable to build the final one-picture video summary by selecting the best view of each keyframe.

Fig. 6 shows the architecture of the proposed solution. The first block consists in a uniform sampling of the source video. The sampled frames are scaled to grey due to luminance information is by far the most important to distinguish visual features. Straightaway the cumulative pixel to pixel difference is applied every two successive sampled frames and normalized in order to apply a classifier with a defined threshold (τ). Finally if the difference value is bigger than (τ) the current frame will be considered as a keyframe.

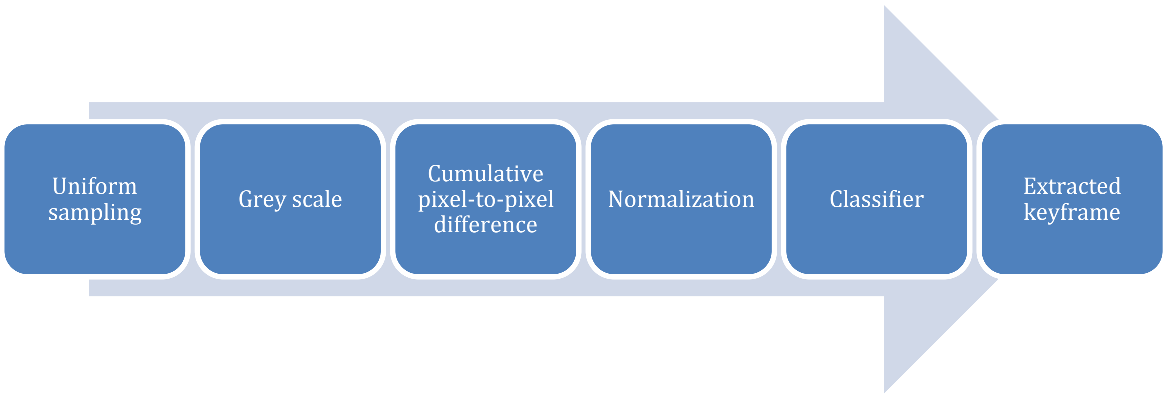


Figure 6.- Implemented solution architecture.

3.1.2. Uniform Sampling

In this block, a uniform sampling extraction of the video frames is performed using the *FrameGrabber* and *FfmpegFrameGrabber* classes from the ffmpeg library, wrapped by JavaCV, a Java Interface from OpenCV. This extraction of frames is performed with a fixed sampling rate and aims at reducing the processing time, as shown in equation (9):

$$Sampling\ Rate = max(round\ (fps_i), round\ \left({}^{L_i}/_{N_0} \right)) \tag{9}$$

Where fps_i is the frame rate of the input video obtained by using the method *getFrameRate( )* from the class *FfmpegFrameGrabber*. L_i is the total number of frames of the input video obtained by using the method *getLengthInFrames( )* also from the class *FfmpegFrameGrabber*[1] and N_0 , the total number of frames we want to keep in order to be processed by posterior algorithms.

[1] https://code.google.com/p/javacv/source/browse/src/main/java/com/googlecode/javacv/FFmpegFrameGrabber.java

Our application is going to process a maximum of one hundred frames, which means that N_0 is set to 100. These frames are the ones which will be processed by the subsequent blocks. Firstly, we have considered this as a good value due to the user does not need to choose amongst more than 100 keyframes to perform a good one-picture video summary. Secondly, due to this value was already used in the previous work [1] [2].

3.1.3. Gray scale domain

At this stage, once the system has extracted all the frames to be processed, it is applied a pre-processing step which converts RGB images to grayscale. This step is commonly used in many applications of image processing due to color information does not help us to identify important edges or other features. Concretely, our algorithm converts the RGB images to YIQ color space (see section 2.2.3) and separates the luminance component (Y) from I and Q components, in order to get the Y component. This is because, in YIQ color space, the luminance is the *intensity* of the image, that is, the *grayscale* signal.

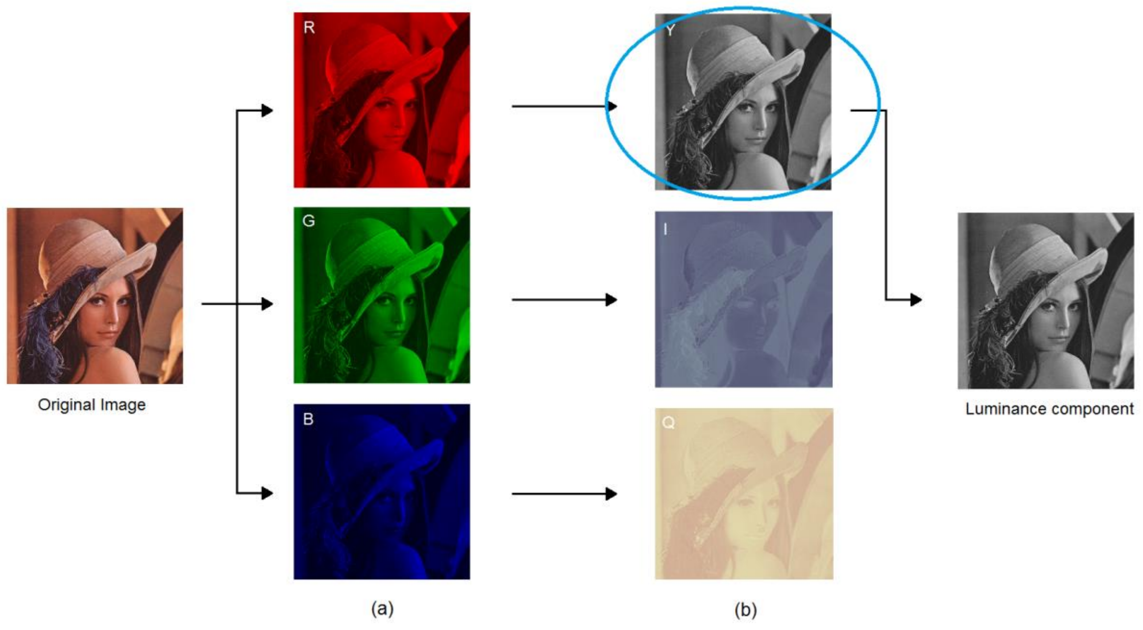


Figure 7.- Example of color model transformation from RGB to YIQ done in this block. (a) RGB components of the Lena standard image in RGB color model. (b) YIQ components.

3.1.4. Difference computation

The next block to be addressed is the one that computes the cumulative pixel to pixel difference to the grayscale frames which come from previous stages. This method sums the difference between each pixel's intensity value in one image and its intensity value in its successive image. This method takes into consideration local details in the images. It is using the following formula:

$$d = \sum_{i=1}^{X} \sum_{j=1}^{Y} |I(t,i,j) - I(t-1,i,j)| \tag{10}$$

The sum is running on all the pixels in the image and $I(t,i,j)$ represents the intensity value at time frame *t* in pixel(i,j). *X* and *Y* are the width and height of the video frames, respectively.

3.1.5. Normalization

The scores obtained from computing the cumulative pixel to pixel difference in the previous block may have very different magnitudes depending on pixel values of the images used in the computation. For this reason, a post-processing stage is required to make all the values comparable; in this case, it is based on normalization. This step consists in dividing each score by the following normalization factor, shown in (11):

$$Normalization\ factor = 256 \cdot X \cdot Y \qquad \hat{d} = \frac{\sum_{i=1}^{X} \sum_{j=1}^{Y} |I(t,i,j) - I(t-1,i,j)|}{Normalization\ factor} \tag{11}$$

Where $\hat{d}$ is the normalized value, 256 is the number of grey levels, *X* and *Y* are the width and height of the video frames, respectively. Therefore, this stage is used to make all the difference values comparable in order to be able to classify them efficiently afterwards.

3.1.6. Decision making

At this stage, the value obtained after computing the normalized difference between successive frames, goes through the decision block, which decides if the analysed frame will be selected as a candidate image to be a part of the final summary or not. That is, the analysed frame will be considered as keyframe if only the difference is bigger than some threshold value, otherwise it will be discarded (as shown in Fig. 8). The threshold value used in our application is $\tau = 0.1$, the same value used by Manuel in ObjectMaps [2].

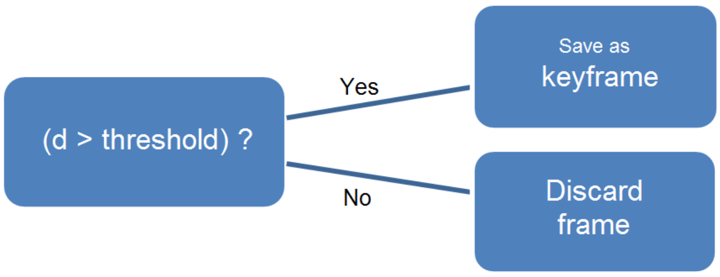


Figure 8.- Decision model consisting of a one-level decision tree.

3.2. Environment

This section describes the technologies and programming tools used for the application development.

Eclipse:

Eclipse[2] is an open source development platform, tools and runtimes for building, deploying and managing software. It was created by IBM in 2001. It allows developing projects in many languages as Java, C, C++, Python, etc. For the development of this thesis, it is been used Eclipse Luna IDE.

Java:

Java[3] is a programming language developed by Sun Microsystems which is now subsidiary of Oracle Corporation. Java is a general-purpose, concurrent, class-based, object-oriented language. One of the advantages of using Java is that its applications are

[2] https://eclipse.org/
[3] https://www.java.com/

compiled to a class file (byte code) that can run on any Java Virtual Machine (JVM) regardless of the computer operating system.

OpenCV:

OpenCV[4] is a powerful Open source Computer Vision library of programming functions mainly, developed by Intel, and now supported by Willow Garage. It was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. It has Java, C++, C and Python interfaces and supports Windows, Linux, Mac OS, iOS and Android. It is written in optimized C/C++ and the library can take advantage of multi-core processing. OpenCV was designed for computational efficiency and with a strong focus on real-time applications. It is free for both academic and commercial use.

JavaCV:

JavaCV provides wrappers to commonly used libraries by researchers in the field of computer vision (OpenCV, FFmpeg, OpenKinect, etc.). Furthermore, although it is not always required, some functionalities of JavaCV used in the project rely on FFmpeg.

FFmpeg:

FFmpeg[5] is a complete, cross-platform solution to record, convert and stream audio and video. It tries to provide the best technically possible solution for developers of applications and end users alike. It also is a free software project that contains libavcodec, libavutil, libavformat, libavfilter, libavdevice, libswscale and libswresample which are the most notable libraries that can be used by applications.

3.3. The application: Designer Master

In this section, we explain how Designer Master works and how to use it, describing the different functionalities of every option and pop-up windows. Concretely, we focus on the performance of the application, not on the code that implements it. However, all the Java classes and libraries are joined in the code folder where you can see how it is implemented with helpful comments.

[4] http://opencv.org/
[5] http://ffmpeg.org/

As already explained in previous sections, Designer Master is a desktop java application which consists basically in an interface that allows the user to choose a template that will be the one-picture video summary after drag and drop images on it. Once the template is chosen, a video file must be opened and the extraction of keyframes implemented in this thesis, starts automatically.

To start running the application just click on the executable jar file, Designer Master v2. The user does not need to install OpenCV and JavaCV libraries because they are also released with the code. As you can see, the main window of the application has appeared, and it looks as shown in Fig. 9:

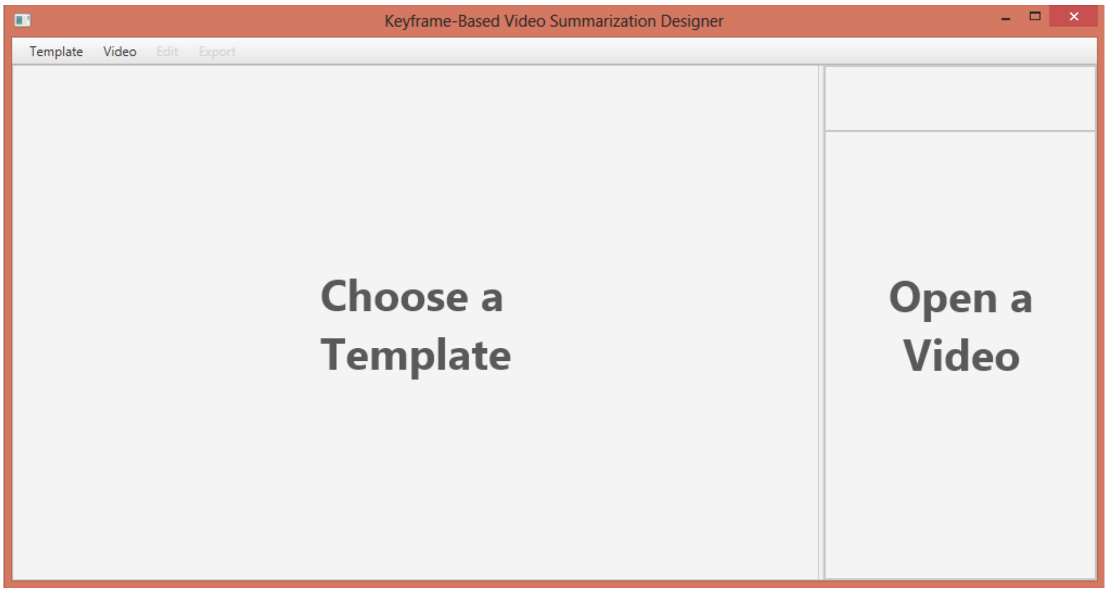


Figure 9.- Main window of the user interface.

Once the application is executed, it can be observed that the main program window appears, as shown in Fig.9. There are four tabs: Template, Video, Edit and Export which allow the user to browse through them and carry out different actions. As it is noticed, the application asks about opening a video "Open a video" and choosing a template "Choose a Template" in order to the one-picture video summary can be created by the user.

Therefore, the first thing to be done consists in choosing a template, just by clicking on the ***"Template"*** button. This tab allows the user to choose or import a template which the keyframes are going to be dragged and dropped in. If the the *"Choose..."* option has been chosen, the application let the user choose amongst two templates that are already built in. Otherwise, if the *"Import..."* option has been chosen, the user can import its own templates from the directory where they are located.

The customized templates are defined via XML-Files. Each template has an overall width and consists of multiple rows with tiles and possible sub-tiles which can contain rows as well. (see apéndice about how to define templates).

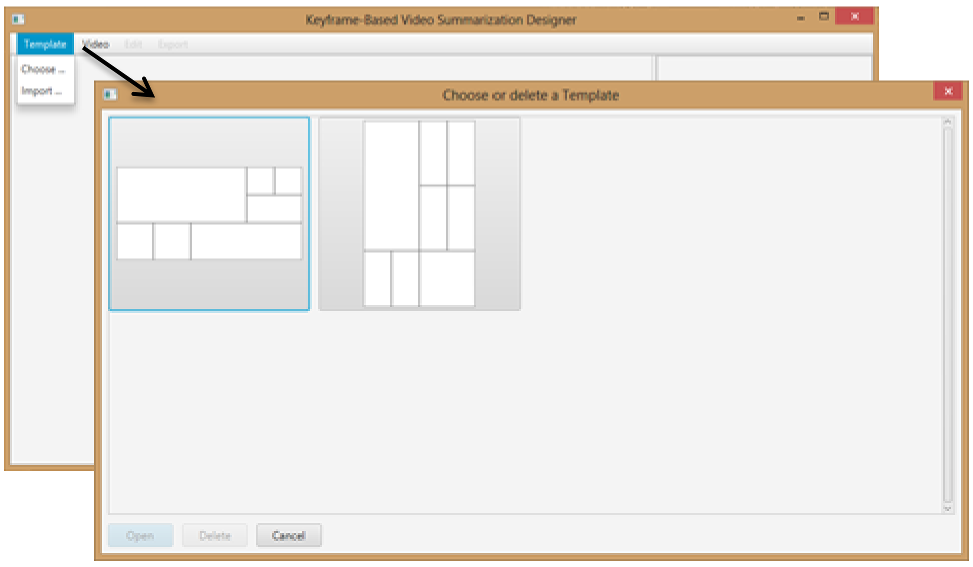

Keyframe-Based Video Summarization Designer
Template
Video
Edit
Export
Choose ...
Import ...
Choose or delete a Template
Open
Delete
Cancel

Figure 10.- Pop up window with the available templates

Straightaway, once the template to be used has been selected, the video file must be opened in order to extract its keyframes and carry out the visual summary. To do it, just click on the ***"Video"*** button. This tab allows the user to *"Open…"* a video from the directory where it is located and, once it is selected, the keyframe extraction starts automatically. The supported input video formats are the same than supported by the FFmpeg library, due to the application uses the FfmpegFrameGrabber class to read the video file. Thus, .avi, .mkv, .mp4, .mpg, .wmv, .mov are some well known video formats accepted by our program, amongst many others.

One of the main functionalities of the application, as it can be observed throughout the extraction, is that keyframes appear sequentially on the right side of the interface as soon as each one is extracted. It was discussed with professor Eidenberger that real-time applications have to show results to the user as soon as possible. Initially, the application showed all the keyframes once the extraction task was done, which means that the user had to wait for results an indefinite period of time, without knowing if the application was working or not. That fact could lead to an undesirable result: to make the user does not use the application again.

Hence, adding this functionality, the user can start making the one-picture summary without the need of waiting for the extraction process to end. In addition, the application has a progress bar which keeps refreshing its status at same time that the algorithm carries out the extraction. In this way, the user knows the remaining time to complete the whole keyframe extraction.

Another functionality could be used while the extraction of keyframes is going on; the user can cancel the extraction process any time, just by clicking on the cancel button “X”, as shown in Fig.12.

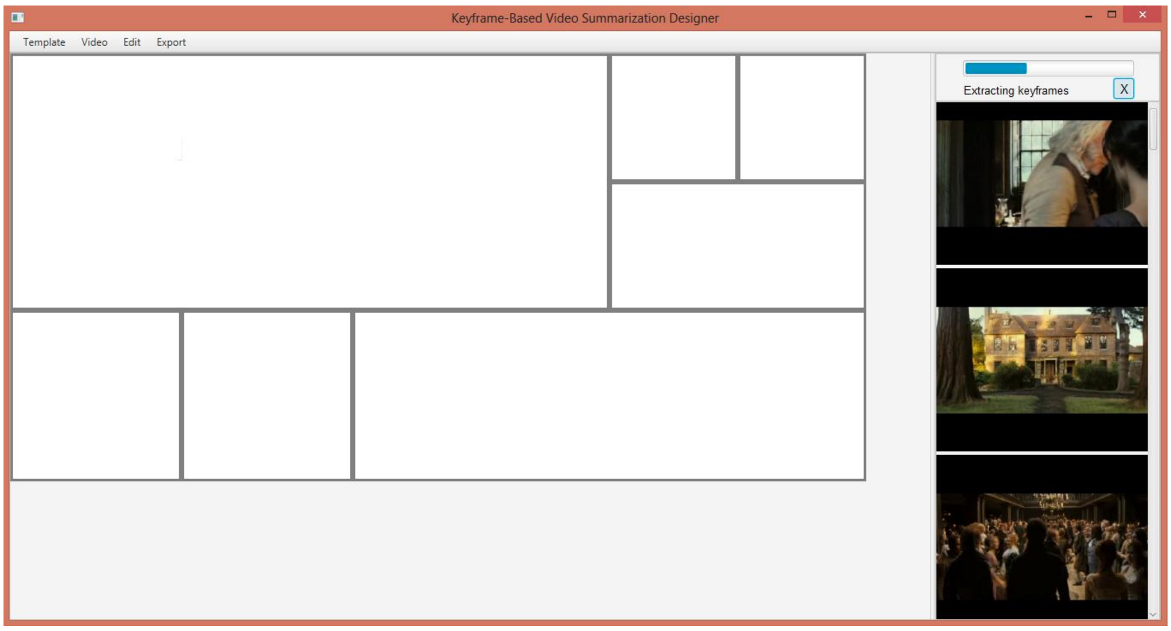


Figure 11.- Automatic keyframe extraction after opening the video.

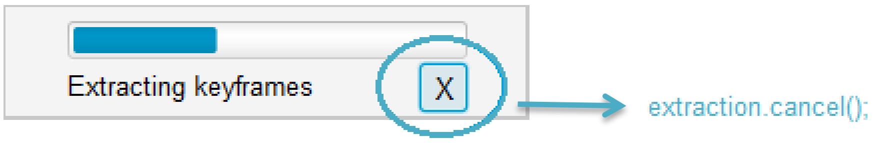


Figure 12.- Keyframe extraction progress bar and cancellation button “X”.

Once the keyframe extraction is completed, it can be observed that a new slider has appeared at the same place where the progress bar used to be. This slider allows the user to reduce the number of selected keyframes which are shown on the right side of the interface. Reducing the number of keyframes could be useful to do the summary faster due to the user have less candidates to look through. The application extracts as default value, a maximum of 100 keyframes that corresponds to the top right position of the slider. Thus, the user is able to carry out the one-picture video summary by dragging the desired image and dropping it in one of the tiles on the template.

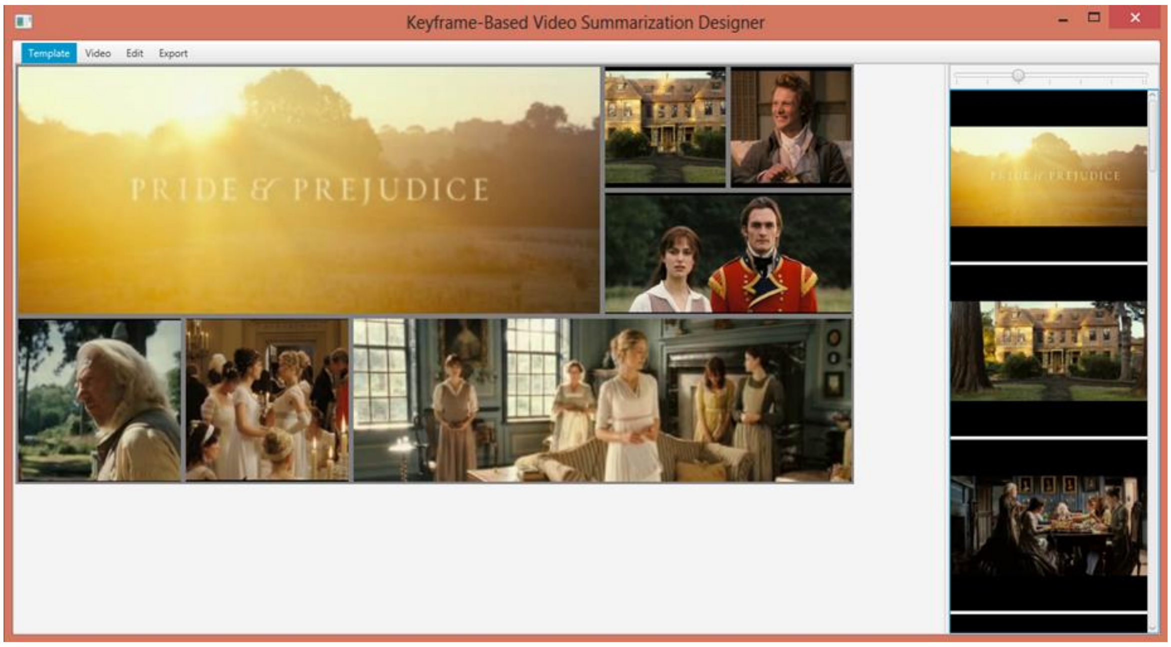


Figure 13.- One-picture video summary after drag and drop the keyframes manually.

In addition, when the selected keyframes are already in the template, the user can replace any image at any time just by drag and dropping a new one. We observe that if the mouse cursor is located above the images in tiles, these can also be enlarged (zoom in) and inverted horizontally, as shown in Fig.14 (b) and Fig.14 (c) respectively.

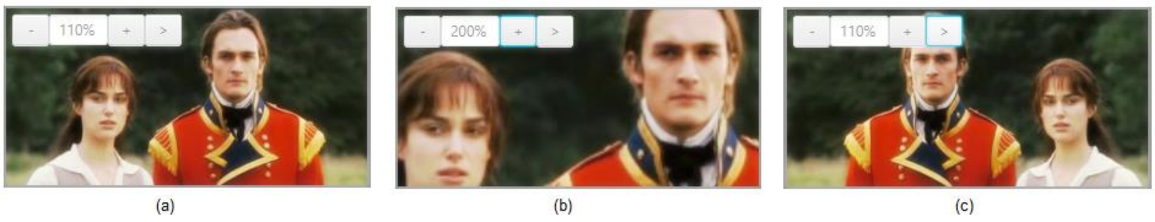


Figure 14.- Example of available actions to the images in tiles. (a) Original image in tile. (b) Zoom In action to the original image. (c) Horizontally inversion to the original image.

While creating the one-picture summary, the application let the user edit the template. To do it, just click on the ***"Edit"*** button and a pull-down menu appears, as shown in Fig.15. The user can enlarge the main window's template in order to visualize more in detail a specific tile just by clicking the Zoom In button. A color palette can be also observed, that allows to change the color of the templates in order to make the summary more attractive.

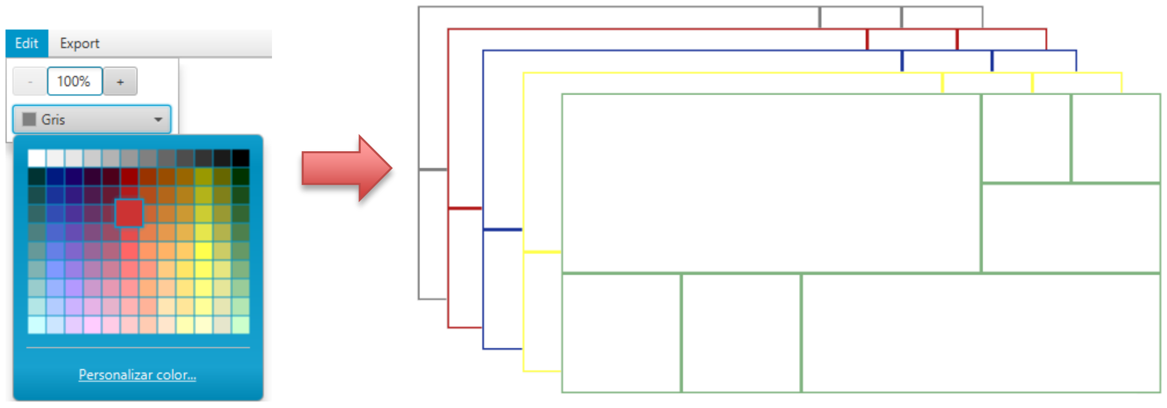


Figure 15.- Edit tab allows the user to change templates' color and enlarge them (zoom in).

Finally, to save the created one-picture summary, just click on ***"Export"*** → *"Image"*. As observed, it appears a pop-up window which contains one slider, the image summary as well as other buttons such as "Save" and "Cancel" and a Check box called "Save scaled image" in the bottom left of the window.

Therefore, by pushing "Save", the image will be saved as a *.png* file format with its original size, that is 1000x500 pixels. Otherwise, the application allows the user to save a scaled image just by making able the option *"Save scaled image"* and clicking on the "Save" button afterwards. By moving the slider, the user can resize the image and see its preview right away (see Fig. 16).

Independently if the user chooses to save the original or the scaled image, after clicking "Save", another emerging window appears asking for the name and location of the image to be saved.

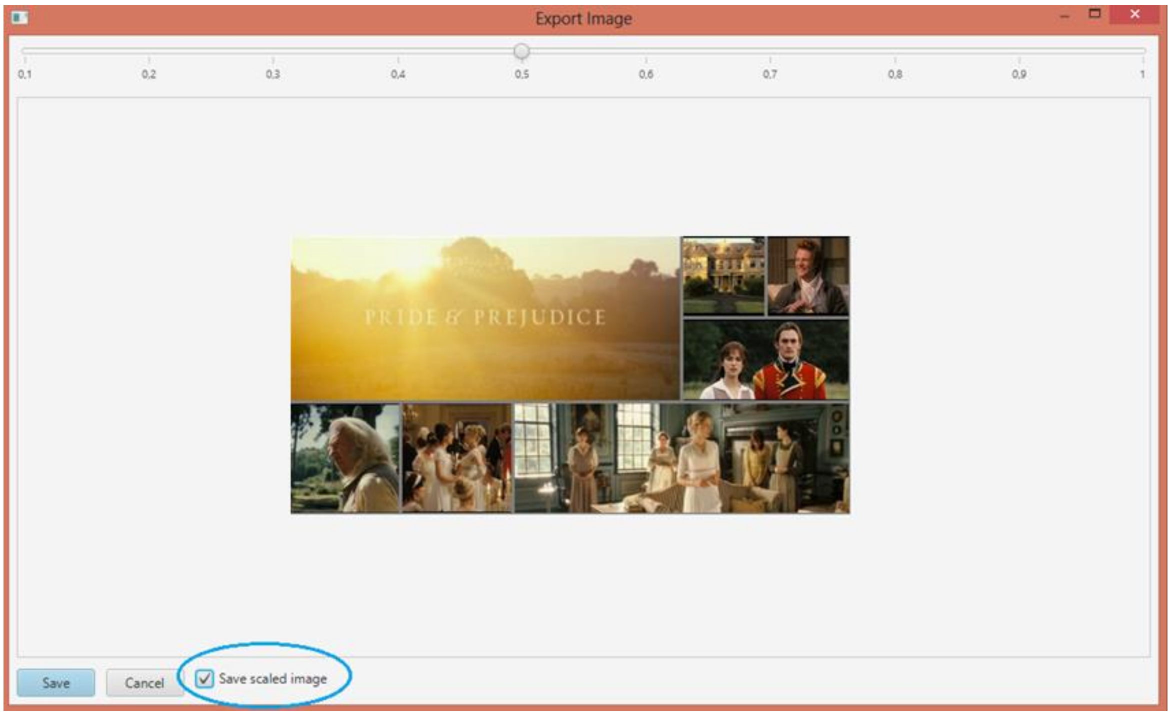


Figure 16.- Example of scaled image to half of its original size (slider to 0.5), i.e. 500x250 pixels ."Save scaled image" button is enabled in order to save the scaled video summary.

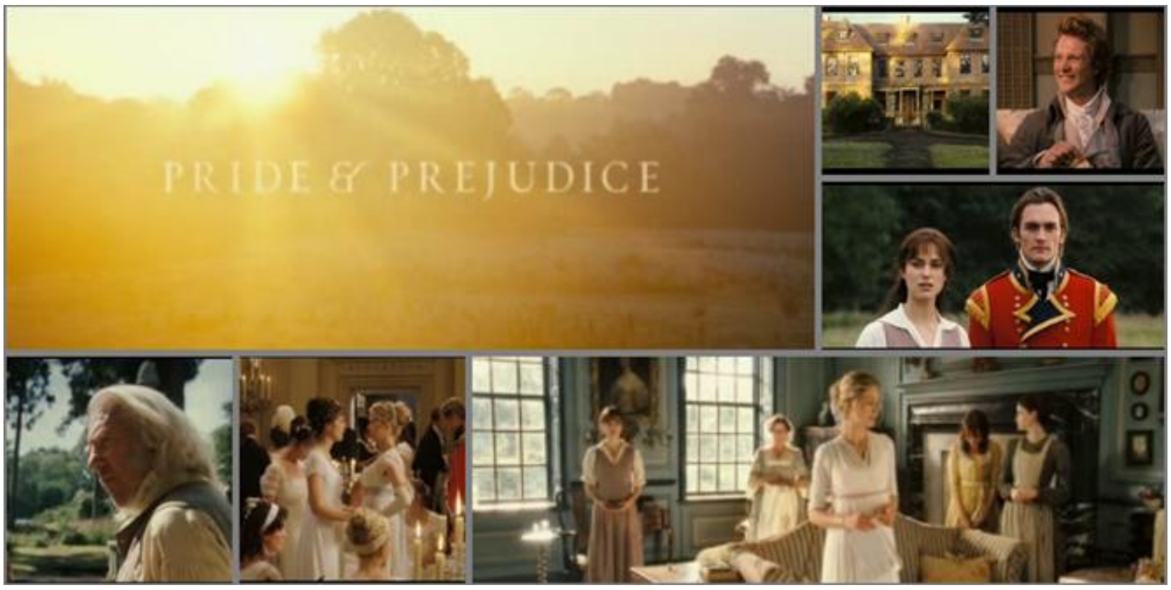


Figure 17.- Final summary created with Designer Master.

4. Results

The results of this project are the one-image summaries that each user creates after using the application. Thus, these could be different according to the aesthetic taste of each user. However, with this evaluation we try to verify if the integrated solution contributes to improve the performance of the application thanks to its better representation of the extracted keyframes.

This chapter is structured as follows: In section 4.1, the adopted method to evaluate the application is commented. Section 4.2 and 4.3 describe the participants and the test data used in the study, respectively. Section 4.4 describes the procedure it has been followed to obtain the results. Section 4.5 shows different tables with time measurements. In section 4.6, the results of the evaluation are discussed. Finally in section 4.7, the findings throughout the assessment are commented.

4.1. Method

To evaluate our tool in terms of performance and quality of the created images, we decided to apply the same method proposed by Manuel Martos in his thesis [2]. We chose an integer score ranging from 1 (Unacceptable) to 5 (Excellent) which was used by The TRECVID Summarization Evaluation Campaign to rate all the summaries [9].

We have compared both applications; the original version, which extracts keyframes uniformly and the version we have developed, which makes use of shot detection techniques in order to extract the keyframes. The evaluation process of our work consists of two parts and for this reason, two tests were designed.

In the first test, the participants are asked to test both applications and create a summary with each one. Next, they are asked to complete a survey regarding to their created images. The purpose of this first test is to collect several pairs of images that are going to be used in the second test, afterwards. In addition, this test gave us information about the application performance and quality of the generated images from the point of view of the user. The disadvantage of this part is that it was difficult to get a high percentage of participation due to this experiment had to be done one by one ‘in situ’ and it took quite a long time to complete each one.

However, the second test was designed to evaluate the application performance and the quality of the generated images as a web-based survey in order to get as much participation as possible.

4.2. Participants

In the first test, a total of 11 participants were recruited in order to create their summaries by using the applications and complete the survey they were asked to answer.

In the second test, a total of 43 participants answered the web-based survey which was shared on Facebook social network.

4.3. Test data

Table 1 reports the video used in the test. We have only tested the applications with one video due to the length of the experiment. In our case, the first test conditioned the second part of the assessment.

Commercial movie trailers have been used due to they are one of the main advertising tools of the movie industry and are chosen among the popular genres and well-known films. The source of each video trailer is the iTunes Movie Trailers[6] and different summaries were created by the users making use of both applications:

Title	Genre	Format	fps	Duration	Resolution	Size
The Intouchables	Biography, comedy, drama	.mp4	23fps	00:02:18	1280x688	29.1MB

Table 1.- Video used in the user study.

It was decided to do the test with movie trailers due to several reasons: in the first place, because to improve Designer Master, we have been working with ObjectMaps [2], which focused at processing movie trailers and thus, we had previous well-known results. In the second place, we wanted that the time each participant spent doing the test was between 5 and 10 minutes. For this reason, we considered that carrying out the test with complete films meant much more time and therefore, it was inviable in order to get a minimum of

[6] http://trailers.apple.com/

participation. Finally, due to pixel-to-pixel methods worked quite well in the keyframe extraction task of movies.

4.4. Execution time

Table 2 shows a comparison between Designer Master v1 and Designer Master v2 in terms of processing time. What we mean with processing time is, how much time each application takes to extract the keyframes and show them all in the interface. As can be observed, different input videos have been tested in order to have an idea of how long it could take to process similar videos using each version of Designer Master.

Title	Format	fps	Duration (h:m:s)	Resolution	Size	Designer Master v1 (min:sec)	Designer Master v2 (min:sec)
Big Hero 6	.mkv	60fps	01:41:52	1920x1080p	3.04GB	07:47	08:18
Monsters INC	.mp4	23fps	01:32:15	852x456	0.99GB	01:48	01:51
Pride prejudice	.mpg	25fps	02:01:26	352x288	774MB	01:05	01:13
The lake house	.avi	25fps	01:34:30	576x240	702MB	00:39	00.37
Mirror Mirror trailer	.mp4	25fps	00:02:30	1280x688	30MB	00:08	00:38
The Intouchables trailer	.mp4	23fps	00:02:18	1280x688	29.1MB	00:07	00:36

Table 2.- Designer Master v1 & Designer Master v2 processing time comparison.

As can be seen in table 2, Designer Master v2 processing time is higher than the time that Designer Master v1 takes to process the video. These results are as we expected due to computing the cumulative pixel-to-pixel difference is computationally more expensive than doing only a uniform sampling. However, there is not much difference between processing times for a given video, which means that Designer Master v2 performs better than version one with approximately the same time.

4.5. Procedure

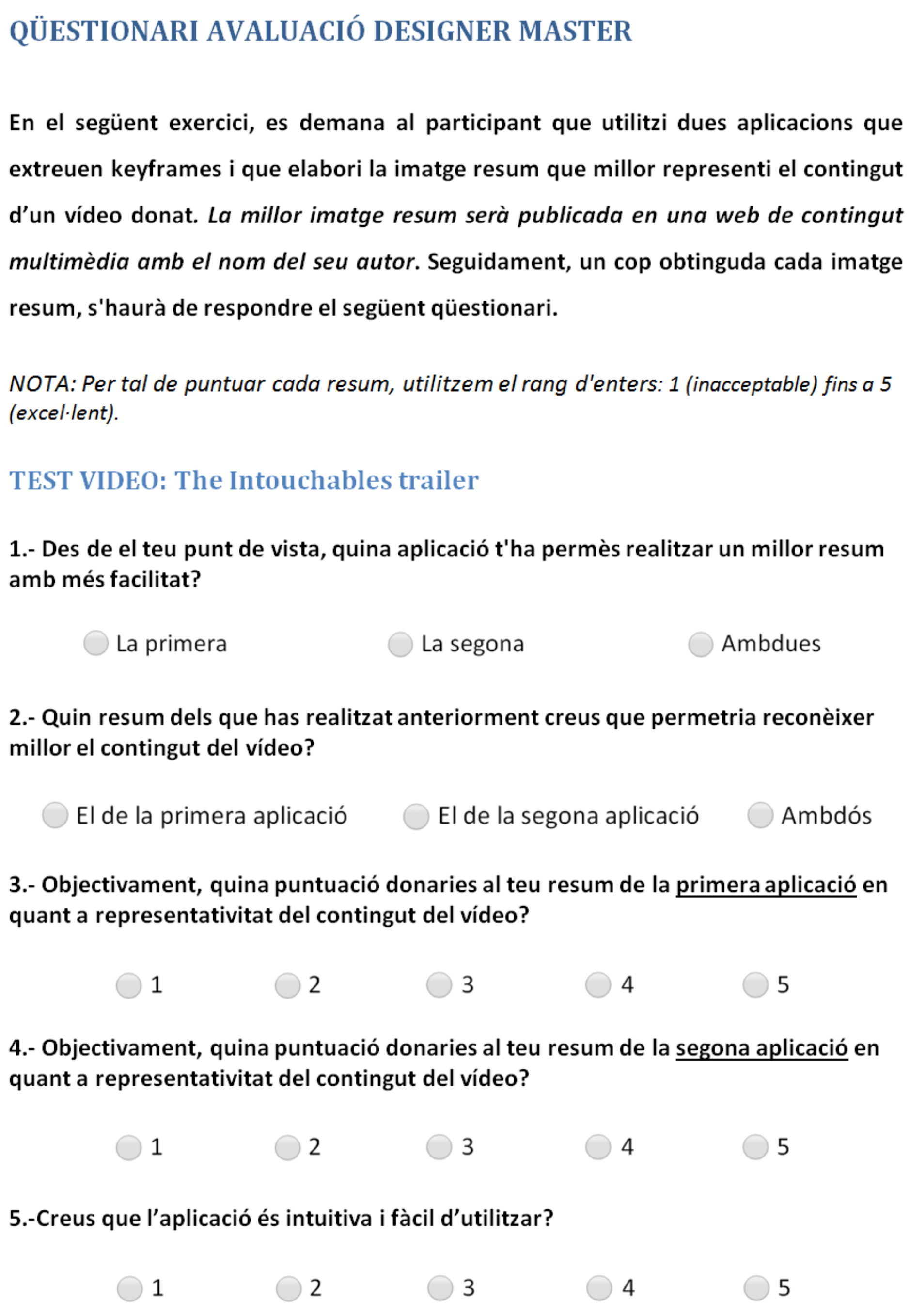
To obtain the results, a survey was created for each test. The first survey was answered by every participant locally, after testing the applications. The second survey was created in Google forms[7] and the link was given to the participants through social networks in order to perform the experiment. At the beginning of each survey, a short introduction about the evaluation procedure was explained.

In the first experiment, each participant was asked to test Designer Master with and without the improvement that it has been implemented. Let’s call Designer Master v1 to the version that extracts keyframes uniformly and Designer Master v2 to the version that extracts keyframes making use of shot-based methods.

Hence, before starting to use the application all participants had to watch the movie trailer they had to summarize in order to guarantee the same knowledge conditions about it. People usually remember elements of the video items they see and use them in their summaries during the test. The order in which each application was used could therefore influence the outcome of the test. For this reason, we also took into account the order that participants were testing the applications trying to minimize this influence. Each user tested the applications without knowing if they started using Designer Master v1 or v2.

Next, as soon as the participants obtained the video summaries, they were asked to complete a survey as the one shown in Fig.17. Participants were asked to select which application allowed them to create a better summary easily and which representation let them better recognize the video content. They were also asked to rate each representation of their own summary with and integer ranging from 1 (Unacceptable) to 5 (Excellent). Finally, they were asked to rate the ease-of-use of the application.

[7] http://www.google.com/forms/about/



QÜESTIONARI AVALUACIÓ DESIGNER MASTER

En el següent exercici, es demana al participant que utilitzi dues aplicacions que extreuen keyframes i que elabori la imatge resum que millor representi el contingut d'un vídeo donat. *La millor imatge resum serà publicada en una web de contingut multimèdia amb el nom del seu autor*. Seguidament, un cop obtinguda cada imatge resum, s'haurà de respondre el següent qüestionari.

NOTA: Per tal de puntuar cada resum, utilitzem el rang d'enters: 1 (inacceptable) fins a 5 (excel·lent).

TEST VIDEO: The Intouchables trailer

1.- Des de el teu punt de vista, quina aplicació t'ha permès realitzar un millor resum amb més facilitat?

○ La primera ○ La segona ○ Ambdues

2.- Quin resum dels que has realitzat anteriorment creus que permetria reconèixer millor el contingut del vídeo?

○ El de la primera aplicació ○ El de la segona aplicació ○ Ambdós

3.- Objectivament, quina puntuació donaries al teu resum de la <u>primera aplicació</u> en quant a representativitat del contingut del vídeo?

○ 1 ○ 2 ○ 3 ○ 4 ○ 5

4.- Objectivament, quina puntuació donaries al teu resum de la <u>segona aplicació</u> en quant a representativitat del contingut del vídeo?

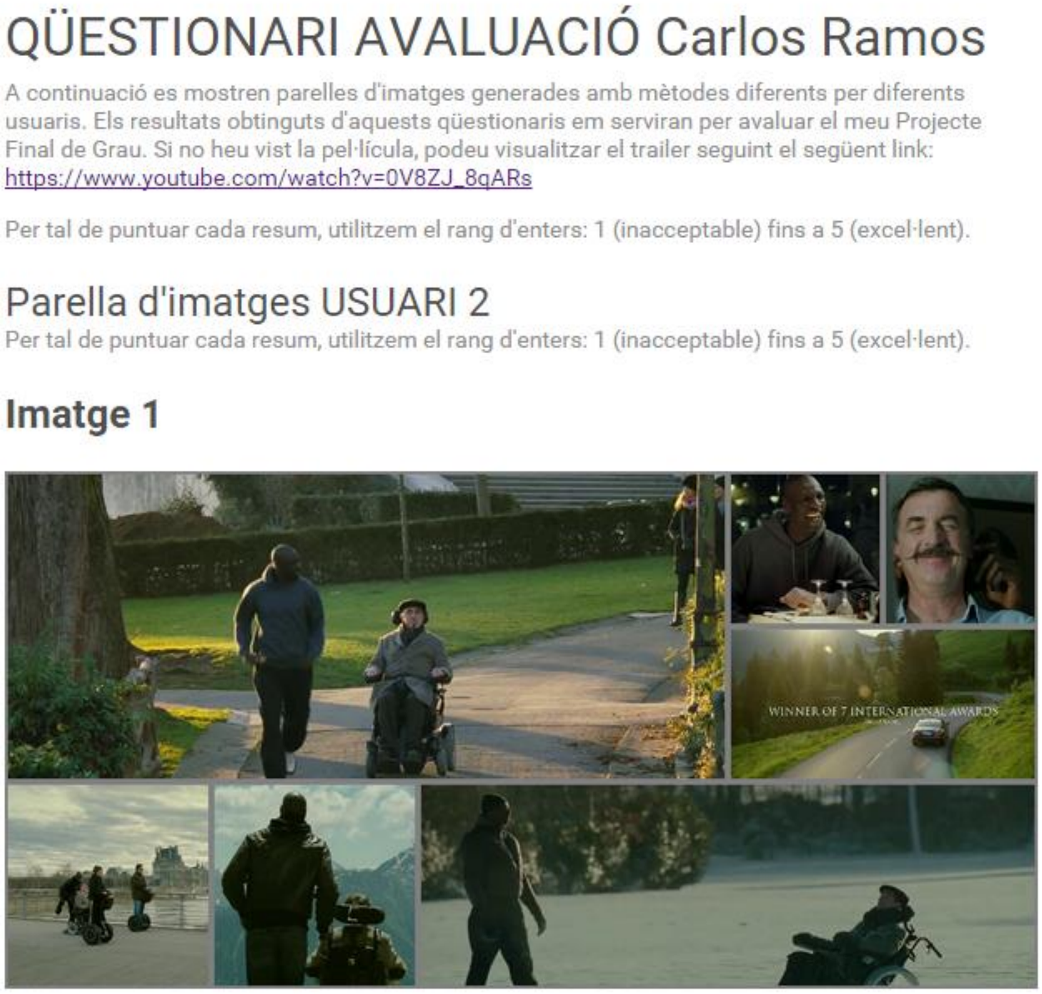
○ 1 ○ 2 ○ 3 ○ 4 ○ 5

5.-Creus que l'aplicació és intuitiva i fàcil d'utilitzar?

○ 1 ○ 2 ○ 3 ○ 4 ○ 5

Figure 18.- Evaluation survey of the application test.

In the web-based survey, participants were asked to complete 5 polls as shown in Fig. 18. In each of them, the two summaries created by the participants of the first test were presented. Participants were asked to select the representation which let them better recognize the video content. They were also asked to rate each representation of the summary with and integer ranging from 1 (Unacceptable) to 5 (Excellent).



QÜESTIONARI AVALUACIÓ Carlos Ramos

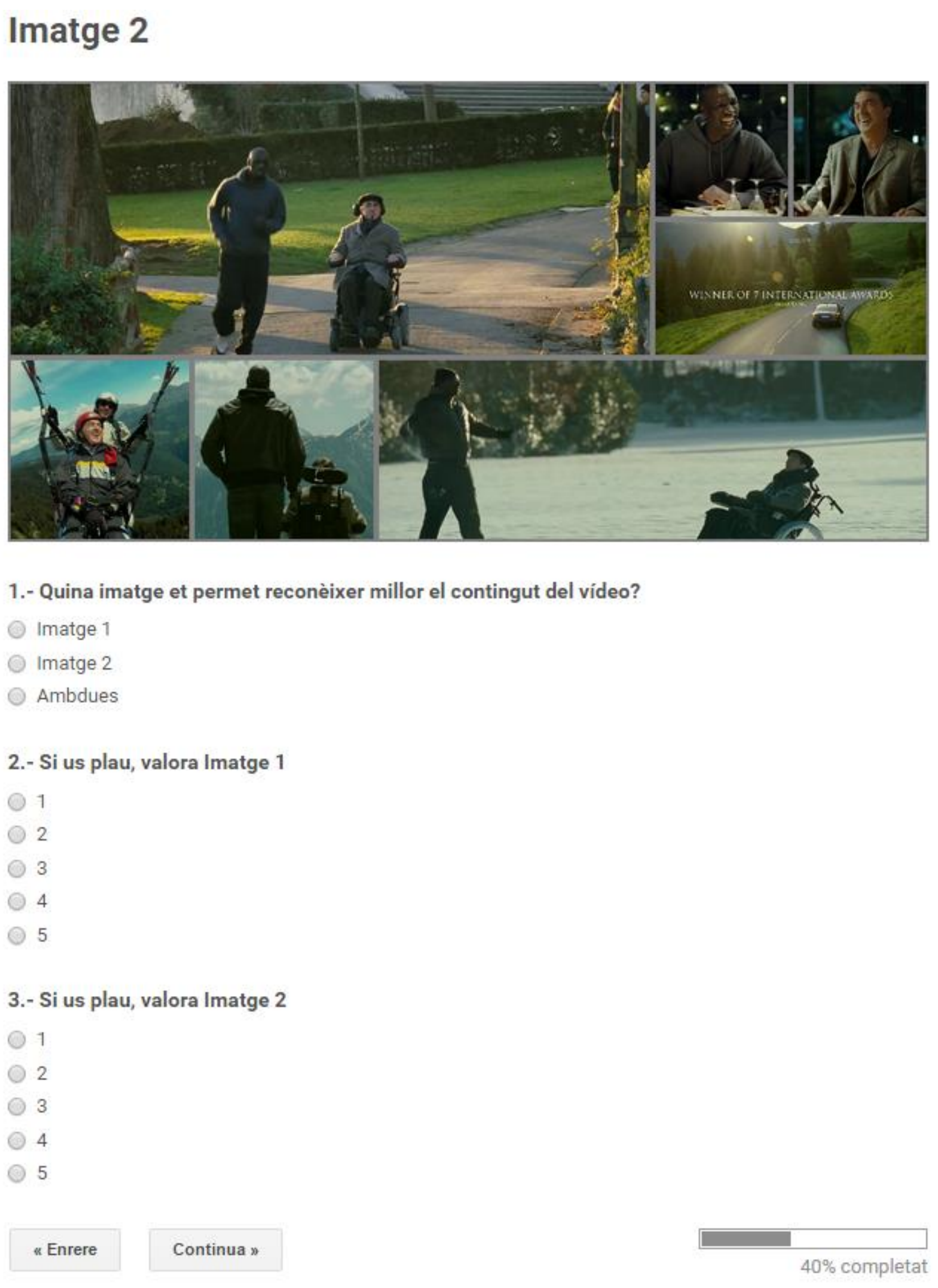
A continuació es mostren parelles d'imatges generades amb mètodes diferents per diferents usuaris. Els resultats obtinguts d'aquests qüestionaris em serviran per avaluar el meu Projecte Final de Grau. Si no heu vist la pel·lícula, podeu visualitzar el trailer seguint el següent link: https://www.youtube.com/watch?v=0V8ZJ_8qARs

Per tal de puntuar cada resum, utilitzem el rang d'enters: 1 (inacceptable) fins a 5 (excel·lent).

Parella d'imatges USUARI 2

Per tal de puntuar cada resum, utilitzem el rang d'enters: 1 (inacceptable) fins a 5 (excel·lent).

Imatge 1



Imatge 2

1.- Quina imatge et permet reconèixer millor el contingut del vídeo?

- Imatge 1
- Imatge 2
- Ambdues

2.- Si us plau, valora Imatge 1

- 1
- 2
- 3
- 4
- 5

3.- Si us plau, valora Imatge 2

- 1
- 2
- 3
- 4
- 5

« Enrere   Continua »

40% completat

Figure 19.- Web-based evaluation survey shared on Facebook social network.

4.6. Data analysis

After collecting all the results, the time each participant spent carrying out the first test (using the applications + 'in situ' survey) was about 7 minutes on average. Otherwise, the web-based survey for the second test was designed to spend less than 3 minutes. The evaluation process considers the Mean Opinion Score (MOS) test, which is a widely used measure of the system quality by averaging the ratings given by the users. All the results have been obtained using only one input video: The Intouchables trailer.

Fig. 19 shows the global analysis of the ratings obtained from the question: "*Please, rate the summary 1"* and *"Please, rate summary 2"*. As can be observed, Designer Master v2 gives the best performance in terms of quality of the created images (MOS = 3.81). Nevertheless, Designer Master with uniform extraction method achieves almost the same result (MOS = 3.76).

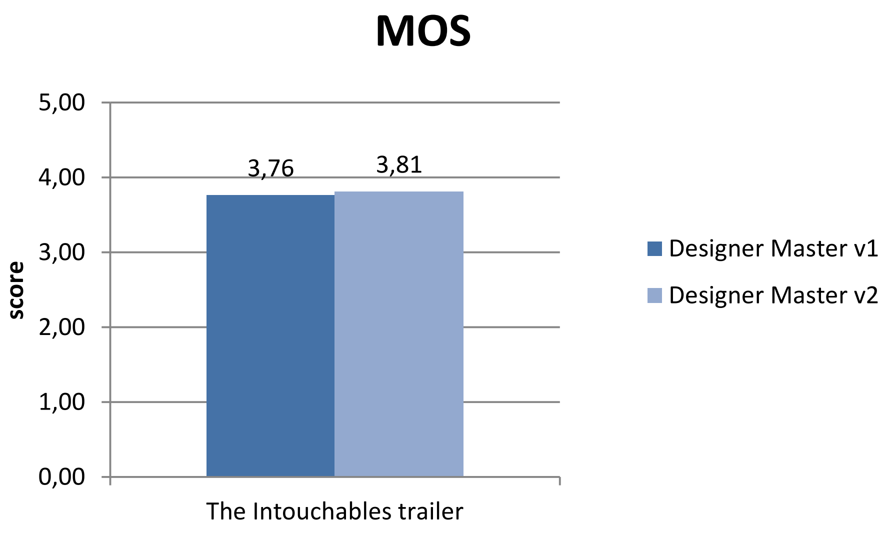


Figure 20.- Content quality of the created images rating.

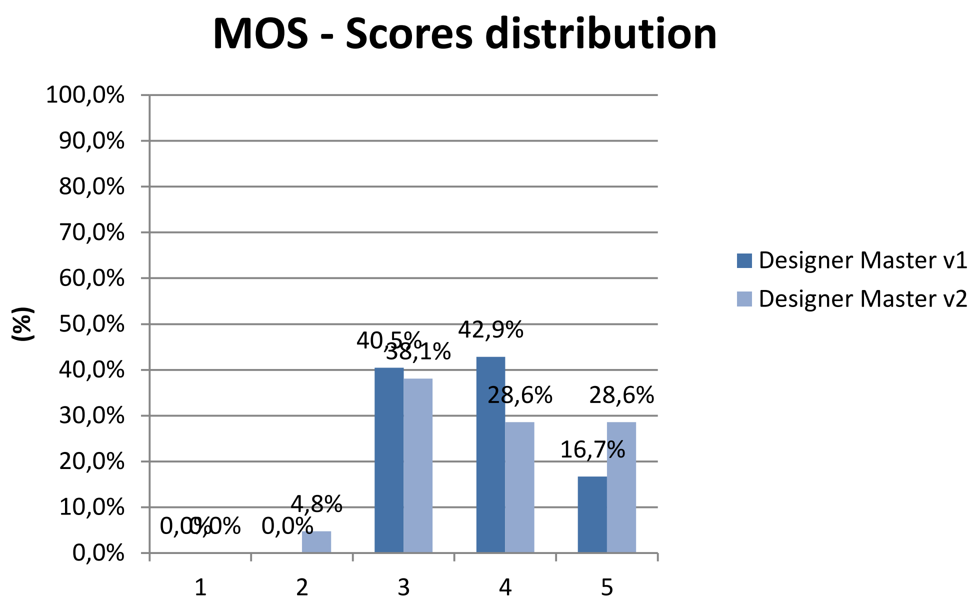


Figure 21.- Results of the summary rates used to compute the MOS.

In addition to the MOS analysis, the representativeness of the summaries is assessed through a user recognition rate of the related movie, obtained from the question: "*Which summary let you better recognize the video content?*"

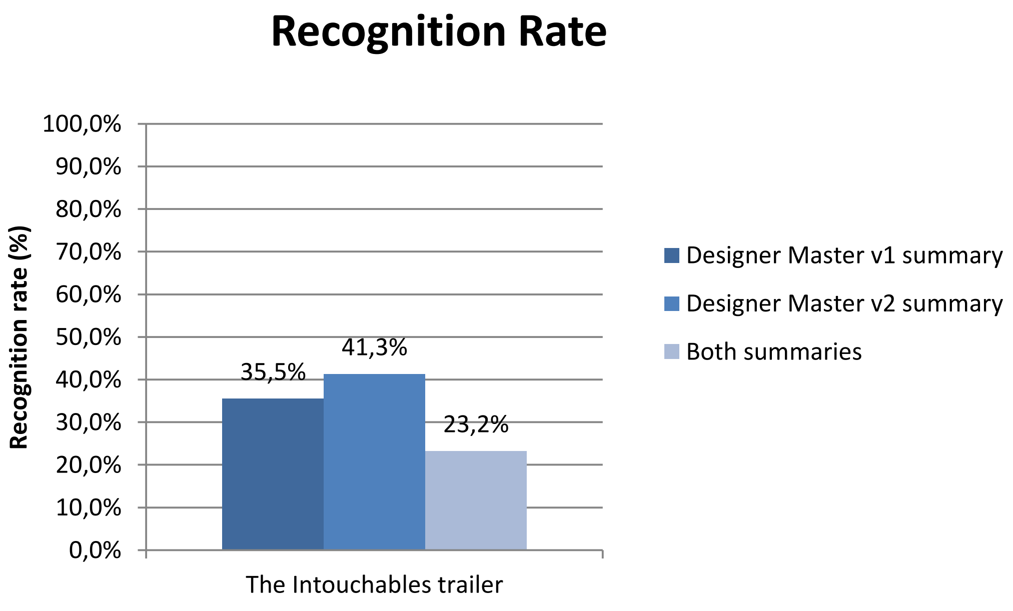


Figure 22.- The Intouchables trailer recognition rate.

In Fig. 21 shows how the 41.3% of participants affirmed that they better recognized the movie with the summaries obtained by using Designer Master v2. The 35.5% of participants affirmed they could better recognize the movie with the summaries obtained by using Designer Master v1. Finally, 23.2% of participants affirmed that both summaries are equal in terms of video recognition.

Finally, a measure of the ease-of-use of Designer Master was also asked with an integer ranging from 1 (Unacceptable) to 5 (Excellent). The ratings were obtained from the question: *“Do you think the application is intuitive and easy to use?”*

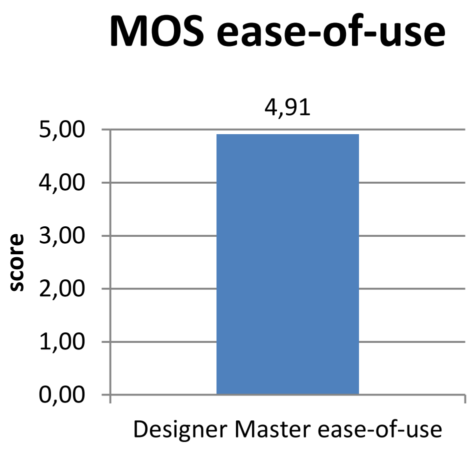


Figure 23.- Ease-of-use rating.

Both applications were tested by different participants with ages between 20 and 55 years. A significant fact that we extracted was that users considered Designer Master an intuitive and easy to use application, as can be observed in Fig. 22. In terms of ease-of-use, Designer Master was valued by the users between “very good” and “Excellent” (MOS = 4.91), concretely 9.1% scored with a 4 and 90.9% scored the maximum value 5.

5. Budget

This project has been carried out with Java, Eclipse as IDE and making use of OpenCV. The main reasons are that everything about Java is open source: from the language, to the standards, to the core libraries, to the virtual machine and the development tools. In addition, OpenCV is released under a BSD license. The BSD license is a permissive free software and therefore, we do not have a license cost. An important aspect of Eclipse is that it is an open source IDE which focuses on enabling the use of open source technology in commercial software products and services under the Eclipse Public License (EPL), a commercial OSI approved licensed. Hence, due to all the software we used is open source under free licenses, we only have the junior engineer cost.

	Personnel	Cost / hour	Dedication	Salary	# Months	TOTAL
Junior Engineer	1	8.00€ / hour	80h / month	640€	5	3200€

Table 3.- Personnel cost.

Used software	Units	Cost / unit	TOTAL
Java	1	Open source	0€
OpenCV	1	Open source	0€
Eclipse	1	Open source	0€

Table 4.- Open source means no cost.

TOTAL COST	3200€

Table 5.- Total cost of the project.

6. Conclusions

This thesis has been developed in compliance with the requirements stated by the Technische Universität Wien (TUW) and Universitat Politècnica de Catalunya (UPC). The main goal aims at improving Designer Master making use of the implemented algorithms by Manuel Martos in his thesis. Designer Master is a Java desktop application that extracts keyframes uniformly and was implemented by Andreas Waltenberger, a computer science student at the TUW. Our proposed solution consists in implementing a shot-based method using *cumulative pixel to pixel difference* as engine to carry out the keyframe extraction task.

Straightaway, we assess the improvement according to a scientific methodology by means of a comparison between the state of the art solution, Designer Master v1 and the improved version, Designer Master v2. This assessment aims to verify if our work provides a better performance to the current application, that is, we want to confirm whether Designer Master v2 lets the user create better summaries and more easily thanks to the better quality of its extracted keyframes.

Regarding to the first goal, we can affirm that it has been accomplished successfully due to we have been able to solve the problem initially proposed by Professor Eidenberger in time and as he required it. A proof of this is that Designer Master v2 works properly and Professor Eidenberger as been very satisfied with the work done. An indicator of quality is that they may develop this work into a product for the Austrian Broadcasting station ORF.

Furthermore, after obtaining and analyzing all the results, it has been demonstrated that Designer Master v2 performance has been slightly better than Designer Master v1 even though both versions were valued by the users between “good” and “very good” with MOSv1 = 3.76 and MOSv2 = 3.81, as shown in Fig. 19

In terms of quality of the summaries, the participants have been able to create good summaries with both versions, independently of which keyframe extraction method was implemented in each one. In this case, Designer Master v2 is still slightly better than version one, the 41.3% of participants affirmed that created summaries by using version two allow a better understanding and recognition of the video. While the other 35.5% and 23.2% belong to those participants who affirmed that summaries obtained by using

Designer Master v1 let them better recognize the movie and who affirmed that both summaries are equal in terms of video recognition, respectively. Thus, we can affirm that our improvement, Designer Master v2 allows the user to create better video summaries and easily due to the better quality of the extracted keyframes. These results make sense because in version one keyframes are extracted uniformly and therefore, the images are selected without taking into account if they represent well an event, shot or a given video sequence.

To conclude, we can affirm that with our work, although we have not obtained very high results, Designer Master has been slightly improved regarding the original version. Nevertheless, from my point of view, I think that the results we obtained in our test were so similar due to both applications were tested with one movie trailer. If we had tested with several complete movies, we would probably have appreciated, in terms of performance, a major difference between the uniform extraction and our proposed method.

Finally, I would like to say that carrying out this project has been a challenge for me. I have learnt a lot about how to be organized and how to solve the different engineering problems that I had to face. I have been alone doing my thesis most of the time and this fact has helped me a lot to improve the autonomous learning skill as well as to contact different people and colleges from different degrees in order to comment and discuss the problems and trying to find out a solution.

Bibliography

[1] Waltenberger, A. “Keyframe-Based Video Summarization Designer”, Vienna University of Technology, Austria (2014).

[2] Martos, M. “Content-based Video Summarization to Object Maps”, Vienna University of Technology, Austria (2013).

[3] Zhao, W., Wang, J., Bhat, D., Sakiewicz, K., Nandhakumar, N., & Chang, W. (1999). Improving color based video shot detection. *Proceedings IEEE International Conference on Multimedia Computing and Systems* 752–756. IEEE Comput. Soc. doi:10.1109/MMCS. 1999.778579

[4] Kasturi, R., Strayer, S. H., Gargi, U., & Antani, S. (1996). An Evaluation of Color Histogram Based Methods in Video Indexing.

[5] Binshtok, M., & Greenshpan, O. (2006). Segmentation of Video Incorporating Supervised Learning.

[6] Rasheed, Z., & Shah, M. (2005). Detection and representation of scenes in videos. 7 *IEEE Transactions on Multimedia* 299–1105. IEEE. doi:10.1109/TMM.2005.858392

[7] Podpora, M., Paweł Korbas, G., & Kawala-Janik, A. (2014). YUV vs RGB - Choosing a Color Space for Human-Machine Interaction. *Federated Conference on Computer Science and Information Systems* 29–34. doi:10.15439/2014F206

[8] Che-Yen Wen, & Chun-Ming Chou (2004). Color Image Models and its Applications to Document Examination. *Department of Forensic Science, Central Police University, 56 Shu Jen Road, Ta Kang Chun, Kwei Shan, Taoyuan, Taiwan*, R.O.C. (333).

[9] Over, P., Smeaton, A. F., & Awad, G. (2008). The trecvid 2008 BBC rushes summarization evaluation. Proceeding of the 2nd ACM workshop on Video summarization TVS 08 (pp. 1–20). ACM Press. doi:10.1145/1463563.1463564

[10] http://www.compression.ru/download/articles/color_space/ch03.pdf
[11] https://code.google.com/p/javacv/wiki/OpenCV2_Cookbook_Examples
[12] http://blackhole1.stanford.edu/vidsearch/dataset/stanfordi2v.html
[13] https://github.com/bytedeco
[14] https://code.google.com/p/javacv/source/browse/src/main/java/com/googlecode/ /javacv

[15] http://docs.opencv.org/java/

[16] http://www.mon-club-elec.fr/mes_docs/my_javacv_javadoc/

[17] http://www.rubydoc.info/gems/ruby-opencv/OpenCV/

[18] http://sourceforge.net/projects/opencvlibrary/?source=typ_redirect

[19] https://docs.oracle.com/javase/8/docs/api/

[20] https://docs.oracle.com/javase/8/docs/api/

Appendices

I. How to create customized Templates

Templates are defined via XML-Files. Each template has an overall width and consists of multiple rows with tiles and possible sub-tiles which can contain rows as well. Let's look at an example:

```
…
<template width="1000">
        <row height="300">
                <tile width="700" />
                <tile width="300">
                        <row height="150">
                                <tile width="150" />
                                <tile width="150" />
                        </row>
                        <row height="150">
                                <tile width="300" />
                        </row>
                </tile>
        </row>
        <row height="200">
                <tile width="200" />
                <tile width="200" />
                <tile width="600" />
        </row>
</template>
…
```

The first line defines the overall width of the template (something around 1000px is generally a good idea). After that the main rows follows, there has to be at least 1 row in a template up to as many as you want (each row has a defined height which has to be at least 50px) - so this example gives us a Template which is 1000px wide and 500px high.

Every row has to contain at least 1 Tile up to as many as you want (each tile has a defined width which has to be at least 150px) - this gives us a Template with 2 rows, whereas in the first row there are 2 Tiles and 3 Tiles in the second row (the sum of the tile widths has to match the overall width of the Template).

In addition, every Tile in the Template could be a new Template, that is every Tile can contain rows which contain Tiles. This recursion works infinitely and is constrained only by the minimal width of the tiles.

II. Test data

This appendix shows the pair of images every user created using Designer Master v1 and Designer Master v2 used for the web-based survey described in Chapter 4.

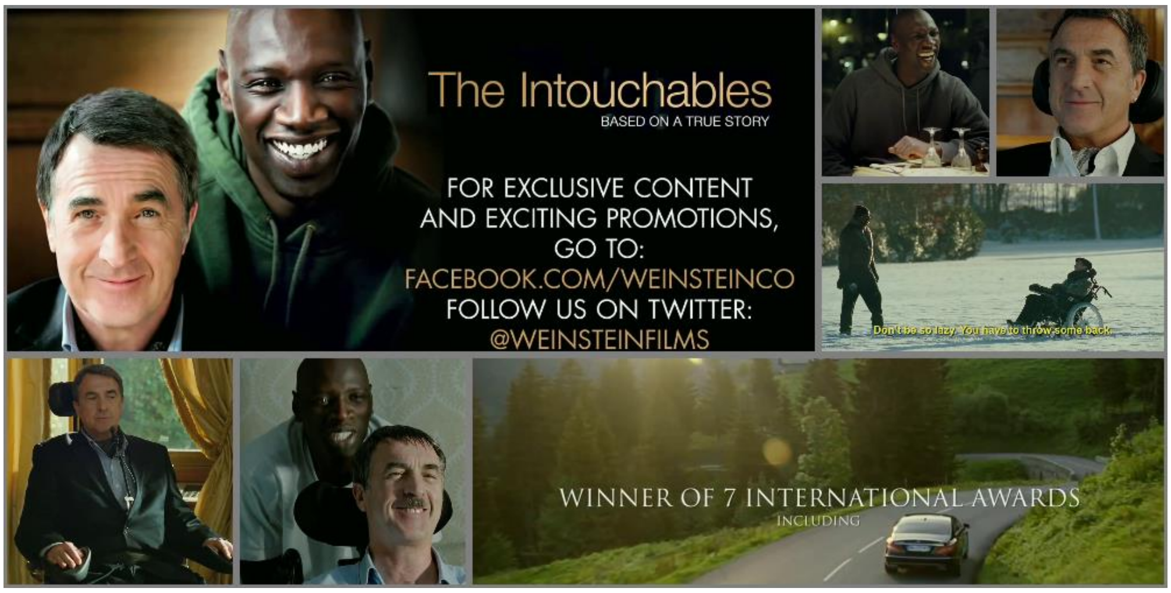

The Intouchables
BASED ON A TRUE STORY
FOR EXCLUSIVE CONTENT AND EXCITING PROMOTIONS, GO TO:
FACEBOOK.COM/WEINSTEINCO
FOLLOW US ON TWITTER:
@WEINSTEINFILMS
Don't be so lazy. You have to throw some back.
WINNER OF 7 INTERNATIONAL AWARDS
INCLUDING

Figure 24.-Image created by participant 1 with Designer Master v1.

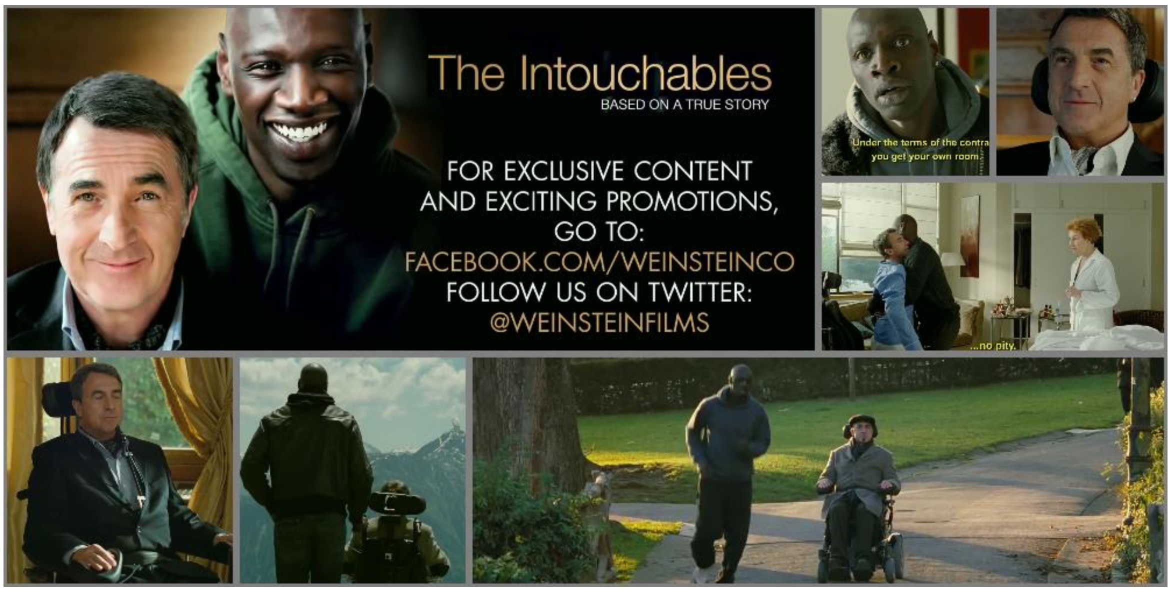
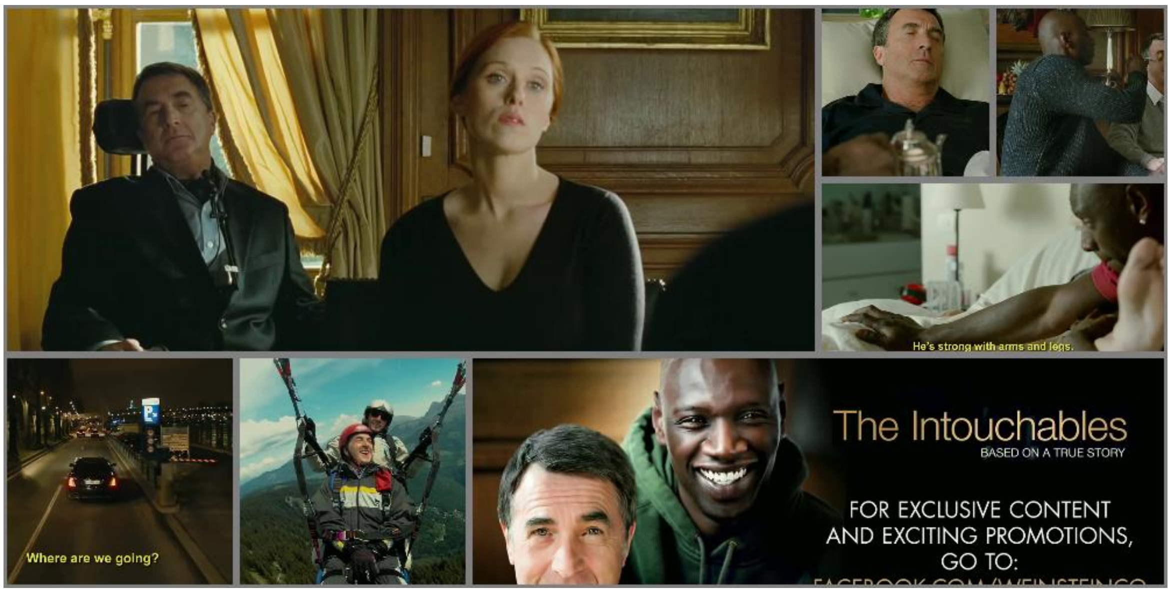


Figure 25.- Image created by participant 1 with Designer Master v2.



He's strong with arms and legs.
Where are we going?
The Intouchables
BASED ON A TRUE STORY
FOR EXCLUSIVE CONTENT AND EXCITING PROMOTIONS, GO TO:

Figure 26.- Image created by participant 2 with Designer Master v1.

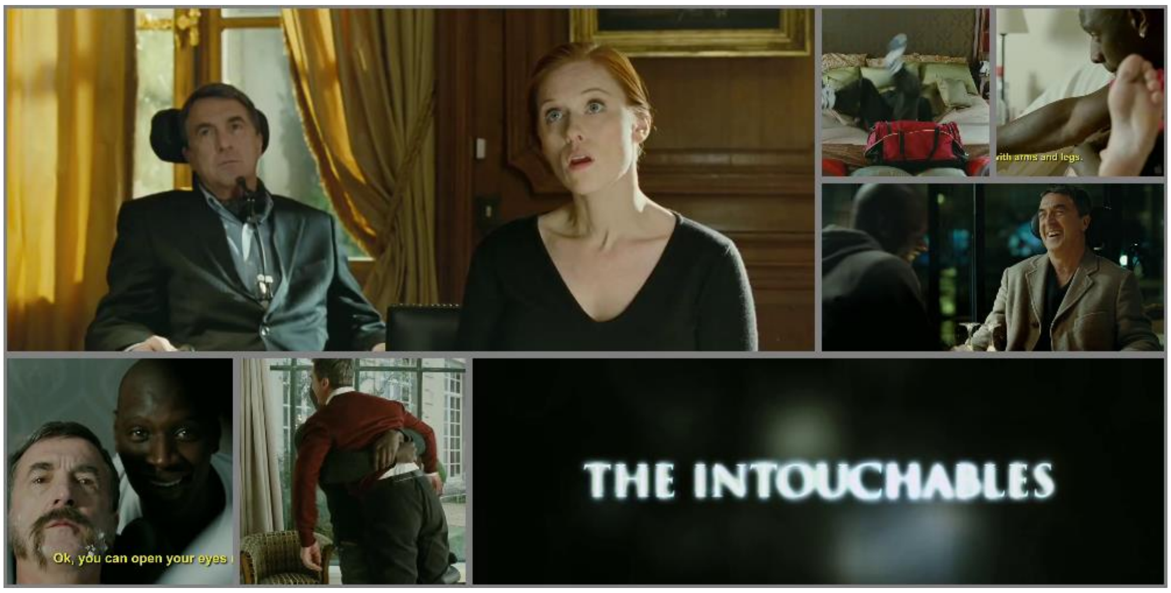


Figure 27.- Image created by participant 2 with Designer Master v2.

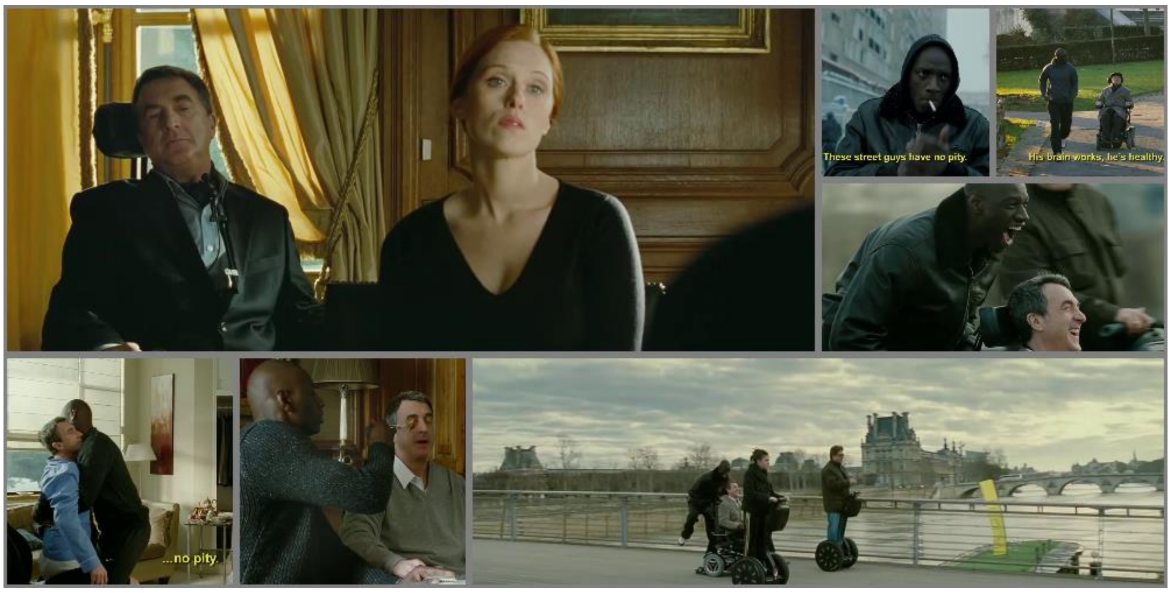

These street guys have no pity.
His brain works, he's healthy.
...no pity.

Figure 28.- Image created by participant 3 with Designer Master v1.

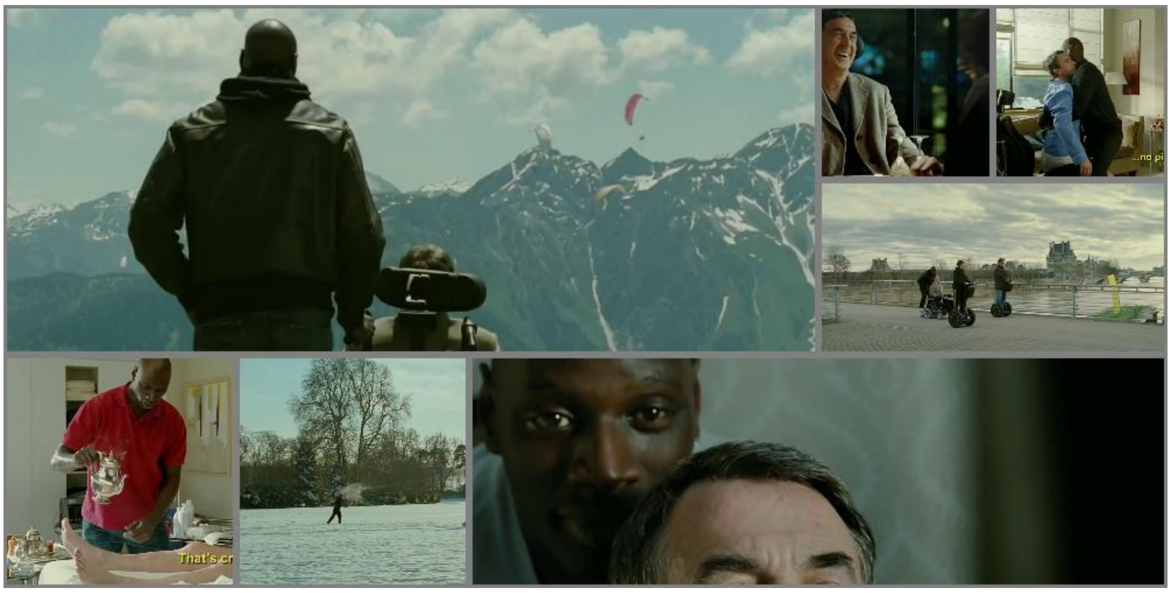


Figure 29.- Image created by participant 3 with Designer Master v2.

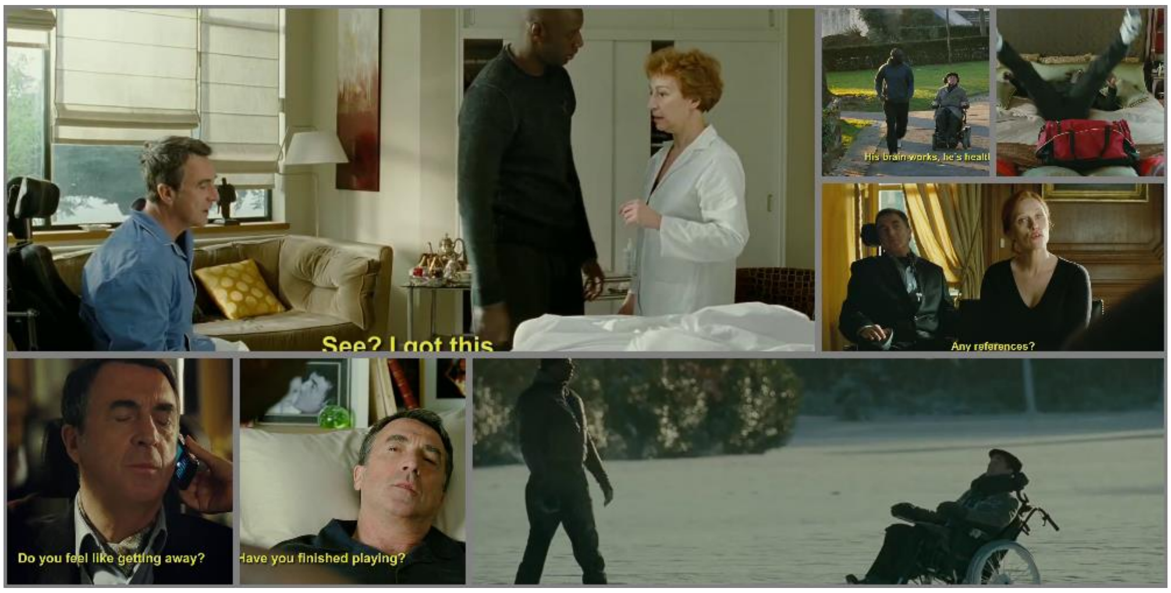

See? I got this
His brain works, he's healt
Any references?
Do you feel like getting away?
Have you finished playing?

Figure 30.- Image created by participant 4 with Designer Master v1.

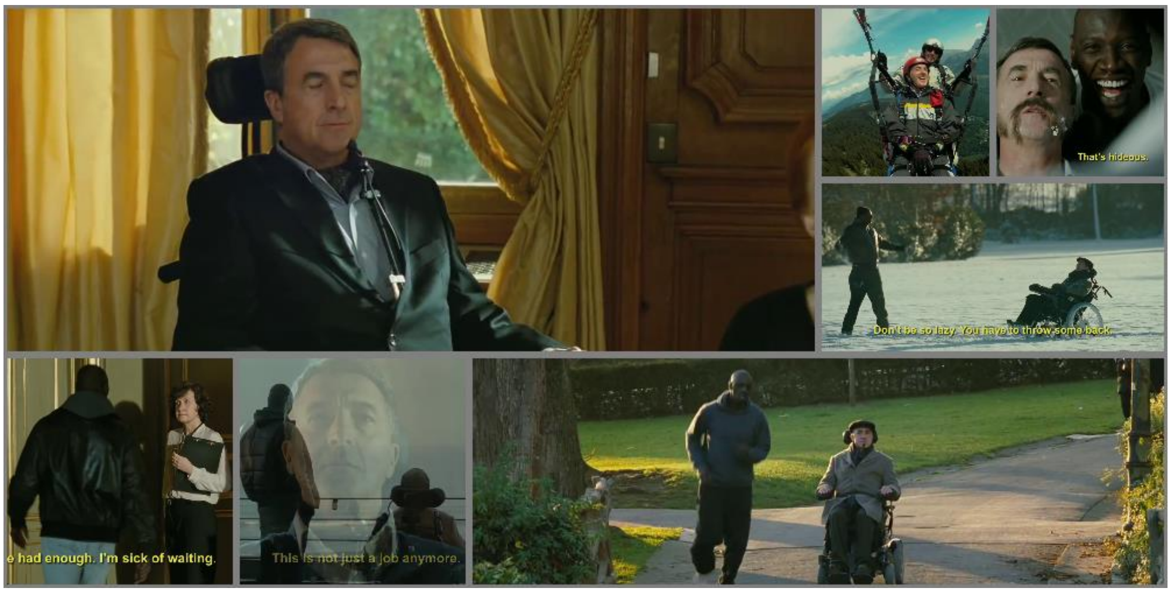

That's hideous.
Don't be so lazy. You have to throw some back.
had enough. I'm sick of waiting.
This is not just a job anymore.

Figure 31.- Image created by participant 4 with Designer Master v2.

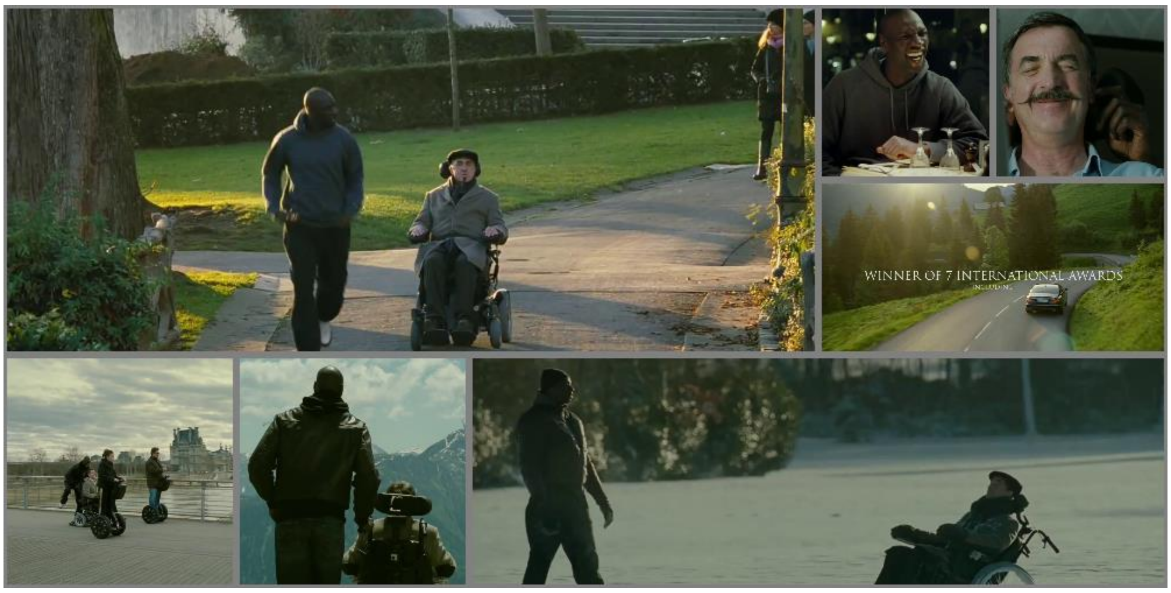


Figure 32.- Image created by participant 5 with Designer Master v1.

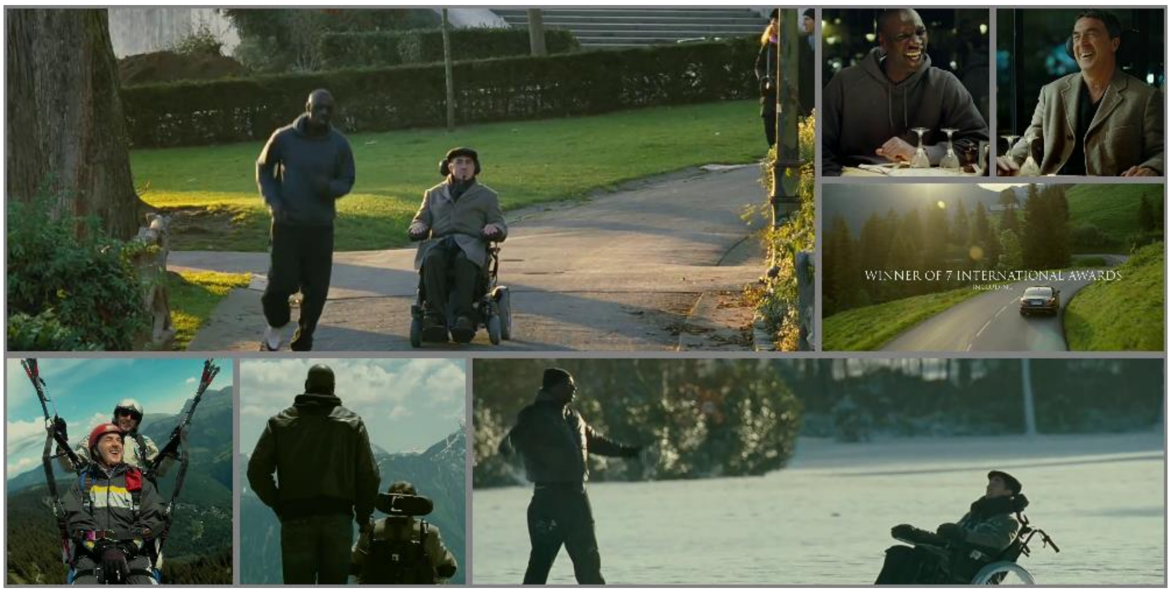


Figure 33.- Image created by participant 5 with Designer Master v2.