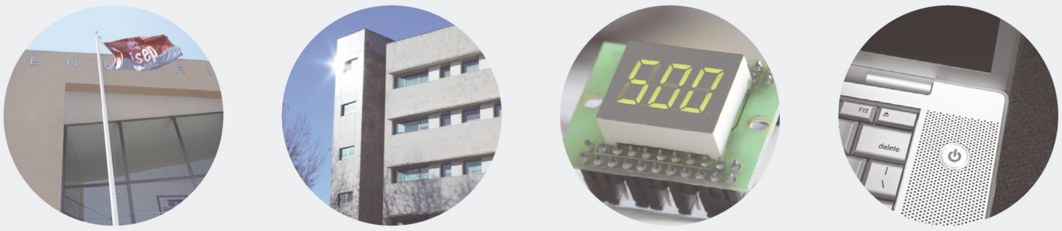
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO

MESTRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES



Cinemetrics: Creative Approaches to Visualize and Analyze Movie Data

ANDRÉ FILIPE CUNHA E SILVA

Novembro de 2021

POLITÉCNICO
DO PORTO

Politécnico do Porto

Instituto Superior de Engenharia do Porto

Cinemetrics: Creative Approaches to Visualize and Analyze Movie Data

André Filipe Cunha e Silva

Master in Electrical and Computer Engineering
Specialization Area of Automation and Systems

Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto

October, 2021

This dissertation partially satisfies the requirements of the Thesis/Dissertation course of the program Master in Electrical and Computer Engineering, Specialization Area of Automation and Systems.

Candidate: André Filipe Cunha e Silva, No. 1150707, `1150707@isep.ipp.pt`

Scientific Guidance: Paula Viana, `pmv@isep.ipp.pt`

Company: INESC TEC

Advisor: Paula Viana, `pmv@isep.ipp.pt`

Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto
Rua Dr. António Bernardino de Almeida, 431, 4200–072 Porto

October, 2021

Acknowledgements

Queria começar por agradecer à engenheira e professora Paula Viana por me ter proporcionado uma proposta de tese cujo tema fosse tão do meu interesse por involver muito conteúdo visual e uma grande componente de programação, e que, como minha orientadora, sempre me ajudou ao longo de todas as fases da realização deste projeto.

Ao longo destes meses onde muitos deles estivemos todos confinados em casa, queria também agradecer-lhe por ter acreditado sempre nas minhas capacidades e por muitas vezes me ter dado motivação nesses tempos mais difíceis da pandemia.

Este agradecimento também serve para todas as pessoas, seja amigos ou familiares, que conviveram comigo (ainda que virtualmente) em momentos nos quais nos ajudamos mutualmente a quebrar o aborrecimento e a abstrair dessa fase complicada.

Devo gratular o Instituto Superior de Engenharia do Porto pelo excelente ensino que me foi proporcionado e pelas pessoas que me fez conhecer e com quem partilhei várias experiências ao longo destes anos todos, tanto os amigos mais próximos como outros colegas de curso, porque todos eles contribuíram de alguma forma para o meu sucesso.

Queria agradecer a toda a minha família desde o meu irmão a primos e tios, por me terem apoiado sempre em todos os aspetos imagináveis, e por terem estado sempre presentes ao longo do meu percurso académico.

Mas, acima de todos, quero agradecer aos meus pais por me terem proporcionado a oportunidade de ingressar no ensino superior e de poder viver esta vida académica que me trouxe tantas alegrias. E pelo esforço, investimento e dedicação dados ao longo de toda a minha vida para que eu pudesse ter as melhores condições, porque nunca me faltou nada do que fosse essencial. E por sempre darem todo o seu apoio incondicional independentemente das minhas escolhas de carreira ou de estudo. Mas principalmente por me terem incutido os valores e me terem criado de forma a que eu me tornasse na pessoa que sou hoje.

Abstract

In the age of visual media that we live in, there is an immensity of content being produced worldwide every day through the cinematographic and TV industries. This, alongside with the widespread availability of Internet access and social media, has democratized the appreciation, discussion and constant analyses of movies and TV shows. Since these are visual media with their own "visual language", where creators use a lot of visual tricks to convey information, emotion, and to show a bit of their signature style, a lot of data can be extracted from them. This represents a great potential for people trying to analyze trends and patterns in video content, for example with colour palettes, length of shots, movement in scenes, audio, etc. But all this can be overwhelming without the aid of technology, and so there has also been a growth in the development of tools based in Artificial Intelligence, and Machine and Deep Learning, to be able to quickly and efficiently analyse this data and produce meaningful and insightful results that might help in the creation of new video-based media or just for anyone interested in video, like cinematographers, video editors, animators or any others. However, with large amounts of data, the results can sometimes be hard to parse by the user, and so there's also been a trend of trying to consolidate data into graphic representations for easier analysis, giving rise to a new field called Data Visualization. With this in mind, we attempted to develop a new tool for video analysis of multimedia files based in scripts that would be able to give information about the composition of shots and scenes, such as the instants of the cuts or estimates of numbers like the total of shots, frames and more, but also to graphically present information using multimedia data visualizations.

Keywords: Cinemetrics, Data Visualization, Big Data, Computer Vision, Shot Transition Detection, Colour Palette, Shot Scale, Clustering, Artificial Intelligence, Deep Learning, Histograms, Frames, Timecode, Average Shot Length.

Resumo

Na era visual em que vivemos, existe uma imensidão de conteúdo a ser produzido por todo o mundo, todos os dias, através das indústrias cinematográfica e televisiva. Isto, em conjunto com a difusão do acesso à Internet e das redes sociais, democratizou a apreciação, discussão e análise de filmes e programas televisivos. Dado que estes são meios visuais com a sua própria “linguagem visual", nos quais os criadores aplicam técnicas visuais para transmitirem informação, emoção, e mesmo um pouco do seu estilo e da própria marca, existe muita informação a ser extraída e avaliada. Isto representa um grande potencial para aqueles que se dedicam a analisar tendências e padrões em conteúdos de vídeo, como por exemplo a palete de cores, duração e movimento das cenas, áudio, etc. Mas pode ser uma tarefa inviável sem a ajuda de tecnologia, e por isso existe um desenvolvimento crescente de ferramentas baseadas em Inteligência Artificial, para que seja possível analisar estes dados de forma rápida e eficiente e produzir resultados significativos e conclusivos que possam ajudar na criação de novos conteúdos de vídeo e apoiar quem esteja envolvido na criação dos mesmos, como cinematógrafos, editores de vídeo, animadores, entre outros. No entanto, esta grande quantidade de dados pode gerar resultados de difícil compreensão e interpretação para o utilizador, pelo que também existe uma tendência crescente para tentar consolidar estes dados através de representações gráficas para uma fácil análise, levando à origem do conceito de “Data Visualization”. Tendo isto em conta, tentamos desenvolver uma nova ferramenta para a análise de vídeo em ficheiros multimédia baseada em "scripts", capaz de recolher informação sobre a composições de “shots” e de cenas, como os instantes dos cortes de cena ou o número estimado de "shots", fotogramas e outros, mas, ao mesmo tempo, apresentando informação através de representações gráficas.

Palavras-Chave: *Cinemetrics*, *Data Visualization*, *Big Data*, Visão Computacional, Deteção de Corte de Cena, Palete de Cores, *Shot Scale*, *Clustering*, Inteligência Artificial, *Deep Learning*, Histogramas, Fotogramas, *Timecode*, *Average Shot Length.*

Contents

List of Figures **ix**

List of Tables **xiii**

Listings **xv**

List of Acronyms **xvii**

1 Introduction **1**

1.1 Contextualization 1

1.2 Problem Description 2

1.3 Objectives 2

1.4 Use Cases 2

1.5 Thesis Structure 3

2 Concepts of Data Visualization **5**

2.1 Big Data 7

2.2 Data Visualization Examples 9

2.2.1 Data Visualization Tools 16

2.2.2 Tableau Examples 17

2.3 Computer Vision 20

2.3.1 History of Computer Vision 20

2.3.2 Computer Vision Applications 21

2.3.3 OpenCV 21

2.3.4 Convoluted Neural Network 22

2.3.5 Colour Theory 24

2.4 IMDb and other Movie Databases 29

3 Cinemetrics **33**

3.1 Concepts of Cinemetrics 34

3.1.1 Shot Transition Detection 35

3.1.2 Fades 36

3.1.3 Shot Scale 37

3.1.4 Camera Movement 38

3.1.5 Shot Length 39
3.1.6 Other Case Studies of Cinemetrics 43
3.2 Frederic Brodbeck's Cinemetrics Project 46
3.2.1 Tool Description 48
3.2.2 Other Examples of Cinemetrics Visualizations 52

4 Code Implementation 57
4.1 Version 1.0 59
4.1.1 Interactive and Automatic Modes 60
4.1.2 Timecode 61
4.1.3 Cadence Variables 62
4.1.4 Numpy Array and Matrices 63
4.1.5 Slit Sequences 64
4.1.6 Colour Clustering 65
4.2 Version 2.0 67
4.2.1 Tkinter GUI 68
4.2.2 Main Shots Feature 70
4.2.3 Fades to White 71
4.2.4 List Creation 73
4.3 Tool Guide 74

5 Results 77
5.1 Informative Outputs 77
5.1.1 Value Comparisons 79
5.1.2 Shots Table 80
5.1.3 Table of Fades to Black/ White 81
5.2 Visualizations 81
5.2.1 Colour Palettes 81
5.2.2 Vertical and Horizontal Prints 85
5.2.3 Slit Sequences 85
5.2.4 Main Shots 89
5.3 Black and White Film 94
5.4 Short Film 94

6 Conclusions and Future work 105

References 109

Appendix A Website Prototype 117
A.1 Django Web Application 117
A.2 Webpage Concept Suggestion 119

Appendix B Interactive Mode **125**

B.1 Flowchart 125

B.2 Shots Table from Lou (Pixar Short) 125

List of Figures

2.1 Flow diagram explaining the typical approach of reading a magazine [4] 6
2.2 Representative and explanatory scheme of the meaning and relation between AI, machine learning and deep learning [7] 8
2.3 "A New Chart of History" [10] 10
2.4 Napoleon march map [10] 11
2.5 Diagram of the causes of mortality in the army in the east from 1854 to 1856 [10] 12
2.6 COVID-19 vaccination tracker in Portugal [11] 13
2.7 General overview of the "Codex Atlanticus" [13] 13
2.8 Detailed view of the page 773 of "Codex Atlanticus" (1499) [13] . . . 13
2.9 Uma figura com duas sub-figuras. 16
2.10 Graph of the top 20 grossing movies VS distributor displayed in the "Movie Money" visualization [25] 19
2.11 Representative scheme of a CNN and its layers [31] 23
2.12 Uma figura com duas sub-figuras. 25
2.13 Uma figura com 3 sub-figuras. 26
2.14 Uma figura com 3 sub-figuras. 27
2.15 Overview of the different and most common resolutions [39] 28
2.16 Example of colour clustering [40] 29

3.1 Descriptive image of the shot scale and its categories as frames [50] . 38
3.2 ASL compared between different genres [56] 41
3.3 ASL compared between two different sequels and their respective movies [56] 42
3.4 Data visualizations based on GT data from movies of six directors – Distributions (top) and chords diagrams (bottom) [64] 44
3.5 System workflow for shot scale classification [65] 45
3.6 Interface of the software "VIAN" [66] 45
3.7 Workflow scheme of *Frederic Brodbeck*'s project [67] 47
3.8 Screenshot of the tool with filtered "fingerprints" of the director Wes Anderson [67] 47
3.9 Anatomy of a movie "fingerprint" [67] 48

3.10 Both examples of a non-moving (a) and a moving (b) "fingerprint" [67] 48
3.11 Screenshots of frames displayed along the "fingerprint" of the movie "The Shining" (1980) [67] 50
3.12 Shot slit-scans extracted from the movie "The Royal Tenenbaums" (2001) [67] 51
3.13 The "Cinema Redux" grids displayed in an exposition [69] 53
3.14 "MOVIEBARCODE" visualizations [70] 54
3.15 "The Colors of Motion" visualizations [71] 55
3.16 The data of the project "Sensing the Shining", as measured by the GSR sensor throughout the whole movie [72] 56
3.17 The method used in the "Sensing the Shining" project to crop the frames [72] 56
3.18 An extract of the "Sensing the Shining" collage [72] 56

4.1 Descriptive shapes displaying how the vertical and horizontal prints are extracted 63
4.2 Cropping and resizing of the frame to increment a slit sequence . . . 65
4.3 A screenshot from the "Cinemetrics" tool showing an instant of a movie (right side) and the "Tkinter" GUI (left side) 70
4.4 Enlarged screenshot of the GUI developed with "Tkinter" 71
4.5 Batman cartoon 72
4.6 5 clusters 72
4.7 8 clusters 72
4.8 Screenshot of the "Cinemetrics" tool in the interactive mode 76

5.1 Distribution of the segments from the movie "There Will Be Blood" 81
5.2 Colour palettes extracted with the "Cinemetrics" tool from six different movies 83
5.3 Colour palette extracted from a scene with multiple cuts of the movie "Taken 3" (2014) 84
5.4 Colour palette extracted from the dancing scene of the movie "Pulp Fiction" (1994) 84
5.5 Colour palette extracted from the pink hologram scene of the movie "Blade Runner 2049" (2017) 84
5.6 Vertical prints 86
5.7 Horizontal prints 87
5.8 Slit sequence from the dancing scene of the movie "Pulp Fiction" (1994) 88
5.9 Slit sequence from a scene of the movie "Enter the Void" (2009) . . . 88
5.10 Two slit sequences from different scenes of the movie "Blade Runner 2049" (2017) 88

5.11 Two slit sequences from different scenes of the movie "Spider-Man: Homecoming" (2017) 89
5.12 First half of the collage made with the slit sequences extracted from the movie "Interstellar" (2014) 90
5.13 Second half of the collage made with the slit sequences extracted from the movie "Interstellar" (2014) 91
5.14 Main shot from the ending scene of the movie "Fight Club" (1999) . 92
5.15 Main shots chosen by the user from the dancing scene of the movie "Pulp Fiction" (1994) 92
5.16 Main shots chosen by the user from the pink hologram scene of the movie "Blade Runner 2049" (2017) 93
5.17 Different visual outputs from the black and white movie "12 Angry Men" (1957) 93
5.18 Five slit sequences extracted from the black and white movie "12 Angry Men" (1957) 95
5.19 Distribution of the segments from the black and white movie "12 Angry Men" 96
5.20 Distribution of the shots from the "Pixar" short "Lou" (2017) 98
5.21 Different visual outputs from the "Pixar" short "Lou" (2017) 98
5.22 Two slit sequences extracted from the "Pixar" short "Lou" (2017) . . 99
5.23 10 main shots chosen by the user from the "Pixar" short "Lou" (2017) 104

A.1 Screenshot of the main page from the unfinished "Django" project . . 119
A.2 Screenshot of the built-in administration page from the unfinished "Django" project 120
A.3 Images of "fingerprints" to indicate movie completion 121
A.4 Prototype of a webpage made to display the outputs generated from a movie with the "Cinemetrics" tool [84] 122

B.1 Flowchart describing the interactive mode of the "Cinemetrics" tool . 126

List of Tables

4.1 A table which compares different play speeds to their processing times (in seconds) 62

5.1 Timecode table from a scene of the movie "Taken 3" (2014) 78
5.2 Table of informative values from movies (12AM: 12 Angry Men; TWBB: There Will Be Blood; EtV: Enter the Void; INCE: Inception; INT: Interstellar; SMH: Spider-man: Homecoming; BR49: Blade Runner 2049) 80
5.3 First half of the table of segments extracted by the "Cinemetrics" tool from the movie "There Will Be Blood" (2007) 100
5.4 Second half of the table of segments extracted by the "Cinemetrics" tool from the movie "There Will Be Blood" (2007) 101
5.5 List of the first 15 "fades to black" detected in the movie "Inception" (2010) 102
5.6 Some informational values annotated with the "Cinemetrics" tool for the "Pixar" short "Lou" (2017) 102
5.7 Other informational values annotated with the "Cinemetrics" tool for the "Pixar" short "Lou" (2017) 102
5.8 The number of false cuts annotated (lower row) and the instants where they occurred (upper row) 102
5.9 List of "fades to black/ white" extracted by the "Cinemetrics" tool from the "Pixar" short film "Luo" (2017) 103

B.1 First third of the table of shots extracted by the "Cinemetrics" tool from the "Pixar" short film "Lou" (2017) 127
B.2 Second third of the table of shots extracted by the "Cinemetrics" tool from the "Pixar" short film "Lou" (2017) 128
B.3 Last third of the table of shots extracted by the "Cinemetrics" tool from the "Pixar" short film "Lou" (2017) 129

Listings

4.1 Code lines that convert the total number of frames into each unit of a timecode stamp 62
4.2 Code lines that represent the first increment of generating the prints 63
4.3 Definition of the first function [75] 66
4.4 Definition of the second function [75] 66
4.5 Extract of code that stitches the colour bars from each shot/ segment 67
4.6 Example of a generated XML file 74

List of Acronyms

3D	three-dimensional
AI	Artificial intelligence
API	Application Programming Interface
ASL	Average Shot Length
BGR	Blue, Green, Red
CEO	Chief Executive Officer
CIE	*Commission Internationale de l'Éclairage*
CMYK	Cyan, Magenta, Yellow, Black
CNN	Convolutional Neural Network
COVID-19	Coronavirus Disease 2019
CPU	Central Processing Unit
CSS	Cascading Style Sheets
CSV	Comma-Separated Values
DVD	Digital Video Disc
EDA	Electrodermal Activity
EOF	End-Of-File
FPS	Frames Per Second
GIF	Graphics Interchange Format
GPS	Global Positioning System
GSR	Galvanic Skin Response
GT	Ground Truth
GUI	Graphical User Interface

HP	Hewlett-Packard
HSL	Hue, Saturation, Lightness
HSV	Hue, Saturation, Value
HTML	HyperText Markup Language
ICR	Intelligent Character Recognition
IMDb	Internet Movie Database
INESC TEC	*Instituto de Engenharia de Sistemas e Computadores, Tecnologia e Ciência*
IoT	Internet of Things
IP	Internet Protocol
ISEP	*Instituto Superior de Engenharia do Porto*
KABK	*Koninklijke Academie van Beeldende Kunsten*
MoMA	Museum of Modern Art
MSL	Median Shot Length
NFTs	Non-Fungible Tokens
OCR	Optical Character Recognition
PhD	*Philosophiae Doctor*
RGB	Red, Green, Blue
RNN	Recurrent Neural Network
SCL	Skin Conductance Level
SQL	Structured Query Language
sRBG	standard RGB
STEM	Science, Technology, Engineering, and Mathematics
SVMs	Support-Vector Machines
Tcl	Tool Command Language
Tk	Toolkit
TV	Television

UK	United Kingdom
USA	United States of America
VFX	Visual Effects
VoIP	Voice over Internet Protocol
WWW	World Wide Web
XML	Extensible Markup Language

Chapter 1

Introduction

This work was developed within the scope of the conclusion of a master's degree in electrical and computer engineering. It serves as a final project and consequent thesis of studies at the *Instituto Superior de Engenharia do Porto* (ISEP) and it was developed in collaboration with the research centre called *Instituto de Engenharia de Sistemas e Computadores, Tecnologia e Ciência* (INESC TEC) and more specifically with its department of multimedia.

The way multimedia content visualization has changed and improved over the years is so extensive and overwhelming that nowadays it has become harder to captivate or surprise users since most approaches have already been used or tested.

1.1 Contextualization

The theme of this project is centred in the analysis and development of approaches for multimedia content visualization with the use of paradigms and algorithms that can be transcribed into programming tools with the help of computer vision.

The decision of opting for this project on top of many other proposals available is based on the personal interest of both areas of multimedia and programming and also from a perspective of wanting to learn and absorb new competences and to be more qualified in distinct areas that can be helpful for the future in the job market.

1.2 Problem Description

Nowadays there are many forms of visualization of multimedia content by which the people are flooded with on their daily basis either on Television (TV), social media or online platforms of multimedia or any other type of digital content that can be accessed through many different technologies like computers, mobile phones, tablets, consoles, etc.

The human mind has been shaped by the immersive and exponential growth of technology that has been happening since the beginning of the 21st century. And because of this, people in the digital era have become impatient due to the fact that they can so easily and quickly access any data on internet databases or any type of digital storage. Also, users are becoming each time more apathetic towards technology and more difficult to impress and appeal to, since they have been subjected to an immense amount of audiovisual stimulus.

1.3 Objectives

The objectives of this project are to:

- study data visualization in the context of multimedia and its state of the art.
- test existing tools and frameworks for video analysis (more specifically for movies) and try to upgrade them or create an original but much improved version (based on these tools) with more functionalities and adapted to today's technologies and programming languages.
- deliberate which characteristics or distinguishing features can be more appealing and interesting to represent movies, and come up with the best methods to extract them.
- produce new forms of content visualization for movies, with the intent of conferring them an identity of their integral content.

1.4 Use Cases

Succinctly, the desired solution of this project is to have a tool that can extract information and different graphical representations from movies.

So this tool would be destined to film experts and devotees, for personal use, or for organizations and corporations which might need data sets and visualizations of movies to be obtained fast and automatically at a professional level.

The generated outputs from this tool can be used as an auxiliary of film study, but their main use case, would be to display them in online movie databases like

"IMDb" and "Letterboxd". Or to develop a website from scratch with an interface that would better suit that content.

In the appendix A there is the description of a prototype of an original website, which was an initial experiment made for this project, but was never concluded. This bonus segment of this thesis with a lot of ideas, concepts and suggestions, should be consulted preferably after reading the whole thesis, in order to better understand the context.

1.5 Thesis Structure

This document is organized with a coherent and clear order of topics, some of them are related to each other, followed by a description of the project development and a final conclusion.

The first chapter of this thesis consists of different introductory aspects such as a brief contextualization of the project, the presentation of the problem in case, a list of the objectives, and examples of use cases.

The second and third chapters together make a display of the state of the art related to this project, where the second demonstrates some concepts of Data Visualization and Computer Vision, and the third one mentions the definition, history and some examples of the "Cinemerics" research field, being this the main focus of this project.

The fourth chapter describes the implementation of the "Cinemetrics" tool, while showing the steps needed to achieve the final product.

The fifth chapter displays a congregation of different informative and/or visual results resulting from the tool.

The last chapter is a short conclusion of the overall project, alongside possible future improvements that can be applied to the developed tool.

Chapter 2

Concepts of Data Visualization

Data visualization, as the name says, consists of the graphical representation of data and information using examples like graphs, diagrams, maps, charts, or animations which provide to the readers an easier and more accessible way of interpreting patterns, trends, and outliers of these data [1]. The true potential of data visualization comes from the interaction of the user with the visualization model [2] as it will be demonstrated in the course of this thesis.

And even though nowadays there is a broad and strong presence of the data visualization thematic, its chronology dates to the prehistory, many years ago where the ancient Babylonians, Egyptians, Greeks, and Chinese wanted to represent information visually with cave paintings which could have been made for many uses such as: to plot the movement of stars, hunting guides, or plans for crop planting and city development. These would normally be drawn on clay and later onto papyrus [3]. As the years went by, other types of visualization started to come up like Egyptian hieroglyphs and maps, which led to the origin of cartography, which were important in the times of discovery and navigation.

It's essential to mention that afterwards, in this chronology, there was the well-known Italian polymath *Leonardo Da Vinci*, who is arguably the most famous and significant figure of the high renaissance era, and to highlight his relevance in the development of the area of visualization, even though he was so keen and expert in so many other fields of science, technology, and art. He became a pioneer and innovator, creating revolutionary methods of conveying scientific or technical information [2].

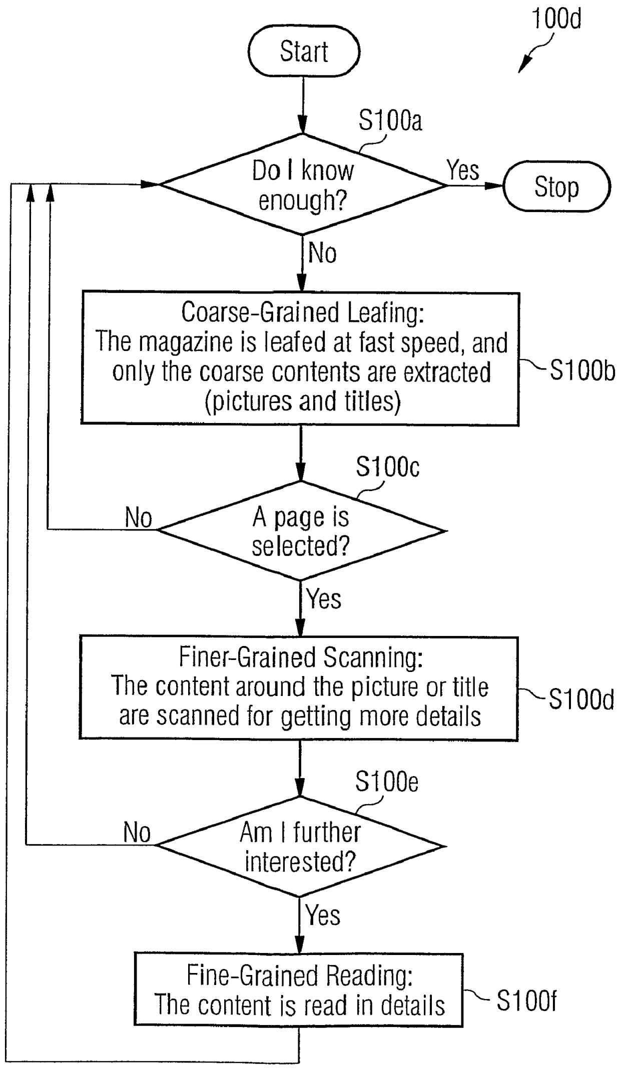

100d
Start
S100a
Do I know enough?
Yes
Stop
No
Coarse-Grained Leafing:
The magazine is leafed at fast speed, and only the coarse contents are extracted (pictures and titles)
S100b
S100c
No
A page is selected?
Yes
Finer-Grained Scanning:
The content around the picture or title are scanned for getting more details
S100d
S100e
No
Am I further interested?
Yes
Fine-Grained Reading:
The content is read in details
S100f

Figure 2.1: Flow diagram explaining the typical approach of reading a magazine [4]

And most recently, to conclude this chronology, there are visual and cinematic arts and multimedia digital content in general which can be seen in social media and in other websites in the shape of photos, videos, or computer-generated images.

As it was mentioned before, the interaction of the user with the visualization model is the key for data visualization and by studying that, it's easier to address this theme in different contexts of visualization.

For example, the magazine is becoming more and more outdated as a way to communicate information just for the fact that it's not digital. But by looking into its essence, we can tell the approach of the typical reader has not changed much until today and it's merely an adaptation for the digital world since it's still based on the reader's interest. This factor varies accordingly with first impressions and with how appealing a title or an image of a section are, for example. The desire of the reader when one reads a magazine, or any other type of informative content, can be measured as a balance between the sense of accomplishment and the need to acquire knowledge and to stay up to date.

As can be seen on the diagram represented in the figure 2.1, the reader starts leafing through a magazine from beginning to end in a quick way just looking at the images and titles and if any of these stand out or seem to be of his own interest then and only then will the reader potentially read all the broad information which is shown in the smaller letters and on a greater detail, because the goal is first to captivate the reader and only then to inform him [4].

2.1 Big Data

The access to data has grown exponentially over the past two decades due to the digitization of physical documents and information, especially at a corporate and business level. But even more due to the proliferation of the Internet, in general, and also of user-generated content sparked by the availability of low-cost commodity digital cameras, and later of smartphones which, in their majority, have built-in cameras [5]. On the positive side, computing power has become more affordable and easily accessible which makes all this accumulated data over the Internet not so pointless.

And it's a fact that each second there is plenty of information being uploaded on the internet all over the world. The numbers state that in 2021 there are around a trillion megabytes of data being created every day and this is because more people are gaining access to the internet daily and they are using more social media and other forms of digital communication like Voice over Internet Protocol (VoIP) calls and texting apps [6].

This is the reason why data visualization software is getting so popular lately. To put in perspective, more than half of the world's data has been generated within the last two years alone so this raises the question of how much data will be created every day in five- or ten-years' time [6].

The concept of big data refers to data that is so large, fast or complex that it's difficult or impossible to process using traditional methods. And it can be studied and examined while being supported by any of these areas such as: data science, data mining and Artificial intelligence (AI), which can include machine learning and consequently deep learning (as it is explained in the figure 2.2).

For all the multimedia content that is uploaded on social networks or anywhere else, there is a unique kind of big data associated with it. Multimedia big data is progressively providing extraordinary opportunities for multimedia services and applications like searches, recommendations, advertisements and substantially for

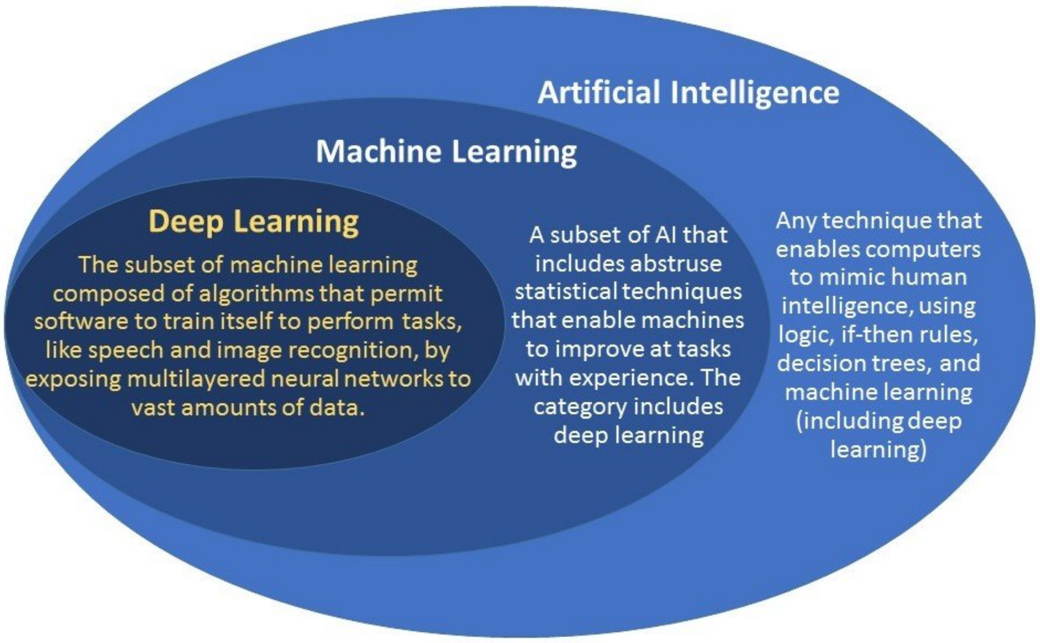

Artificial Intelligence
Machine Learning
Deep Learning
The subset of machine learning composed of algorithms that permit software to train itself to perform tasks, like speech and image recognition, by exposing multilayered neural networks to vast amounts of data.
A subset of AI that includes abstruse statistical techniques that enable machines to improve at tasks with experience. The category includes deep learning
Any technique that enables computers to mimic human intelligence, using logic, if-then rules, decision trees, and machine learning (including deep learning)

Figure 2.2: Representative and explanatory scheme of the meaning and relation between AI, machine learning and deep learning [7]

the Internet of Things (IoT). The presence and popularity of multimedia has been transcendental and overwhelming, for better or worse, in everyday society thanks to platforms like "YouTube", "Facebook", "Twitter" and "Instagram" [5].

For this type of big data there is an authentic and endless “oil well” of vast amounts of data that can be processed, mined, learned, filtered, and analysed to achieve game-changing insights. For instance, there is the metadata of multimedia content which, by default, if the device used to capture this content has a Global Positioning System (GPS) tracker, can give you information about the geographic location of where the photo, video or audio were taken.

In this case, with the use of deep learning supported by computer vision, it is possible to teach a neural network to detect patterns in a database categorized by each geographical identity and later recognize a place by associating and comparing elements in common with each identity distinguished and saved in the database. The application of this might be a further search that, through the process of reverse engineering, will be able to identify a place ("Geo-tagging") [8]. The same can be applied to measure the duration of daytime and consequently night time, and to identify seasons, climates or even the sound environment.

One of the drawbacks of big data and of the immensity of information that is available to the public, is that it can easily be obtained usually without the consent of the users.

This is a situation that has raised ethical concerns about the extent of the personal information that can be accessed, as well as the safety and privacy of people using the Internet. There is a portion of these people who are unaware that their personal data is being saved and that some of it can be accessed by anyone[9].

Another question that has been addressed is How much freedom the common individual really has, considering that nowadays, with the implementation of big data analytics, any online platform can have access to their users' activity, likes, interests, needs, thus depriving them of making decisions on their own, consequently questioning their free will[9].

Many people have come to the realization that a big part of their lives lies in the Internet through the type and quantity of content and personal information they choose to post and upload. This conclusion can be drawn also when they see how invasive these platforms can be through the shape of advertisements and personalized recommendations.

Another consideration that needs to be done is about the intellectual property of individuals and corporations. Just like everything that is creative or innovative, data is priceless. But unfortunately, conventional data has not yet been regarded as intellectual property because, according to law agencies, it does not meet the eligibility requirements. And because of this it cannot be protected by laws like copyrights, patents or trademarks, which are used to prevent false claims, theft and infringement, and to give the owners the full rights over their creations[9].

This topic comes down to three types of concern related to big data ethics: privacy, awareness and intellectual property[9].

Any conventional data can be compromised due to the fact that any other entity can freely utilize it, so this is considered also as a matter of social justice in which the various governing bodies need to take more decisive actions to solve all these problems.

While on one side, big data has a great potential, on the other, it generates these ethical concerns which need to be tackled, because if left unaddressed it may become a significant barrier to the fulfillment of expected opportunities and long-term success of big data [9].

2.2 Data Visualization Examples

After a summary of the evolution of data visualization, it's mandatory to reinforce this thesis by showing concrete examples and cases of this area of study that is so vast and diversified. With that being said, this section will show a parallelism between notable and revealing visualizations, starting with some historical ones and ending with some which are more contemporary [10].

Data visualization often conjures thoughts of business intelligence with button-down analysts and for that reason it requires graphic design, great analysis and storytelling skills [10].

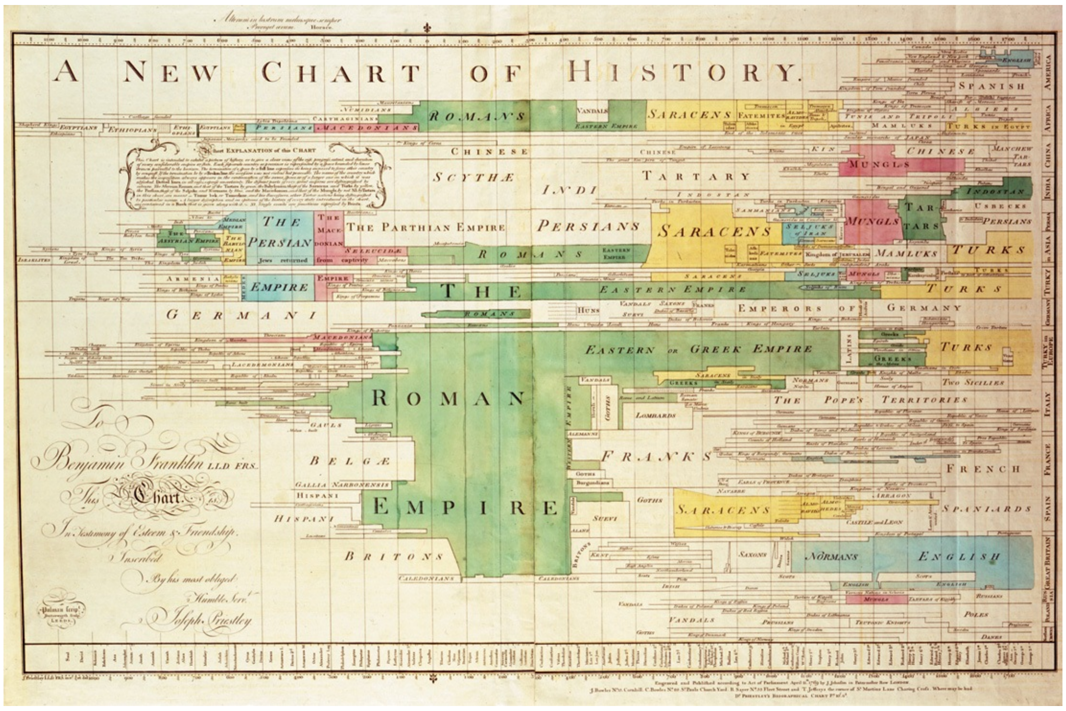


Figure 2.3: "A New Chart of History" [10]

The first and oldest case that is being demonstrated here, as it is visible in the figure 2.3, is called "A New Chart of History". A visualization made in 1769 by the British polymath Joseph Priestley, this chart lists events in 106 separate locations and it illustrates Priestley's conviction that the entire world's history was significant.

This chart of history is described as a prolific and visual way to imagine the massive and complex timeline of recorded human civilization, and it highlights the simultaneous existence and influence of different major empires and cultures through history. Even though it can seem confusing and busy at first glance, this piece is seen as a huge innovation and it excelled at the time due to the use of colours, big size, and a creative y-axis of location [10].

It has a lot of similarities and inspiration in his other known previous work called "A Chart of Biography" that was made in dedication to his friend Benjamin Franklin and provided a 700-year timeline of famous leaders and philosophers who were active in history at the same time [10].

In the figure 2.4, there is the Napoleon march map made by the french civil engineer *Charles Joseph Minard*, who was highly recognized for his significant contribution to the field of information graphics, and also for his representation of numerical data on geographic maps.

This chart has become one of the most famous visualizations of all time, and it tells the story back in 1812 of Napoleon's Russian campaign, when he marched with around 470 000 soldiers in order to conquer the city of Moscow and disastrously

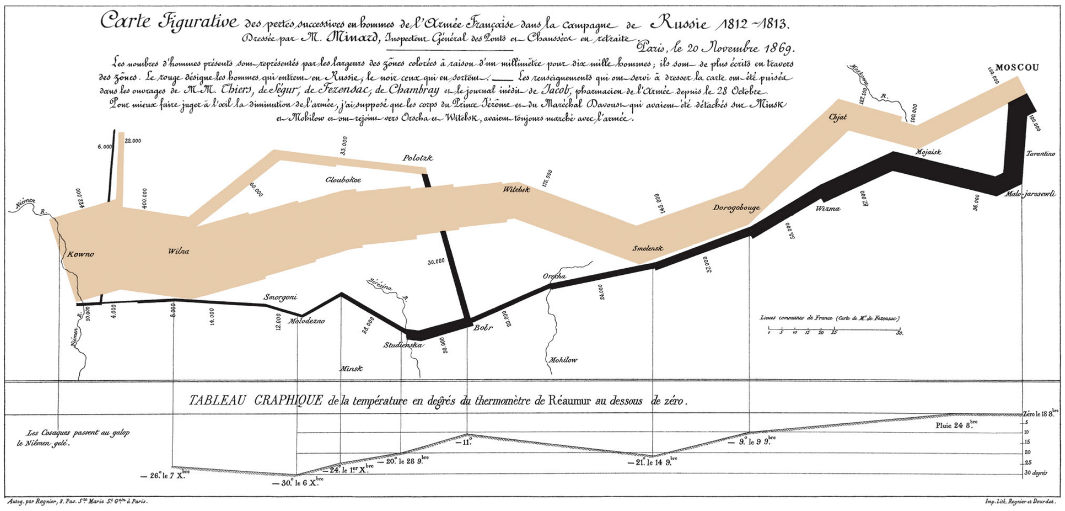

Carte Figurative des pertes successives en hommes de l'Armée Française dans la campagne de Russie 1812–1813.
Dressée par M. Minard, Inspecteur Général des Ponts et Chaussées en retraite.
Paris, le 20 Novembre 1869.
Les nombres d'hommes présents sont représentés par les largeurs des zones colorées à raison d'un millimètre pour dix mille hommes; ils sont de plus écrits en travers des zones. Le rouge désigne les hommes qui entrent en Russie, le noir ceux qui en sortent. —— Les renseignements qui ont servi à dresser la carte ont été puisés dans les ouvrages de M.M. Thiers, de Ségur, de Fezensac, de Chambray et le journal inédit de Jacob, pharmacien de l'Armée depuis le 28 Octobre.
Pour mieux faire juger à l'oeil la diminution de l'armée, j'ai supposé que les corps du Prince Jérôme et du Maréchal Davoust qui avaient été détachés sur Minsk et Mobilow et ont rejoint vers Orscha et Witebsk, avaient toujours marché avec l'armée.
MOSCOU
Kowno
Wilna
Smorgoni
Molodezno
Gloubokoe
Polotzk
Witebsk
Minsk
Studienska
Bobr
Orscha
Mohilow
Smolensk
Dorogobouge
Wizma
Chjat
Mojaisk
Tarantino
Malo-jarosewli
Niemen R.
422.000
400.000
6.000
22.000
60.000
33.000
175.000
145.000
100.000
55.000
37.000
24.000
50.000
28.000
30.000
20.000
14.000
12.000
8.000
4.000
10.000
TABLEAU GRAPHIQUE de la température en degrés du thermomètre de Réaumur au dessous de zéro.
Les Cosaques passent au galop le Niémen gelé.
Zéro le 18 8.bre
Pluie 24 8.bre
— 9° le 9 9.bre
— 21° le 14 9.bre
— 11°
— 20° le 28 9.bre
— 24° le 1.er X.bre
— 30° le 6 X.bre
— 26° le 7 X.bre

Figure 2.4: Napoleon march map [10]

returned with only 10 000 [10].

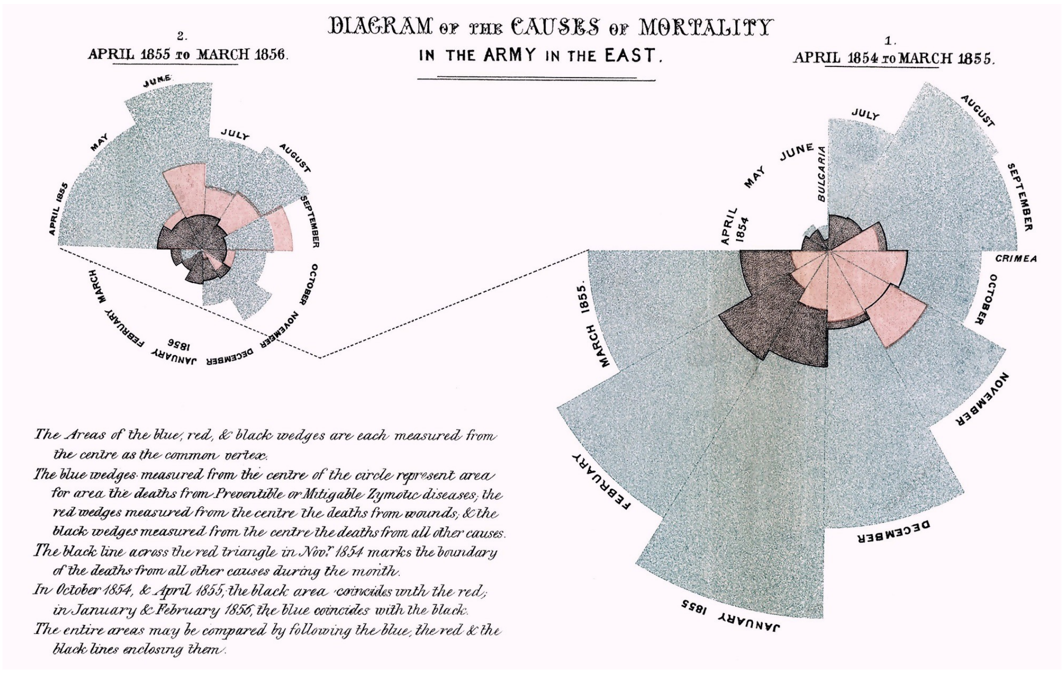
The map portrays the out-and-back journey of Napoleon's troops. The width of the line represents the total number of soldiers and the colour represents the direction (light pink for the path to Moscow and black for the return trip). Below the central visualization there is also a simple temperature line graph illustrating the rapidly dropping winter cold which shows just how devastating the journey was. Because of its fame, there is a lot of critical commentary about this chart and some of it is reasonable, but this work was nonetheless a huge influence and a pioneer for the field that was later denominated data visualization [10].

The next visualization was designed by Florence Nightingale, an English social reformer, statistician and the founder of modern nursing, and it's a diagram that tells the causes of mortality in the Crimean War from April of 1854 to March of 1856.

The main goal of this diagram, as seen in the figure 2.5, was to demonstrate that the soldier mortality rate was high and climbing, and not just because of the battles. It reveals that the majority of deaths were actually caused by poor hospital practices.

As Florence Nightingale visited the hospital many times to get an insight of the situation with her medical expertise, she found, to her surprise, that the cause of so much death was bad medical conditions which could be prevented. The research was part of a Royal Commission looking into the causes of mortality of the soldiers in the Crimean War where she worked with William Farr, a British epidemiologist and statistician who didn't support the idea of using visualizations, but she stood firm and advocated for this now-famed visualization [10].

In these spiral charts, the shaded areas display the quantity of total deaths while the blue wedges represent the deaths from preventable and mitigable zymotic



The Areas of the blue, red, & black wedges are each measured from the centre as the common vertex.
The blue wedges measured from the centre of the circle represent area for area the deaths from Preventible or Mitigable Zymotic diseases; the red wedges measured from the centre the deaths from wounds; & the black wedges measured from the centre the deaths from all other causes.
The black line across the red triangle in Novr. 1854 marks the boundary of the deaths from all other causes during the month.
In October 1854, & April 1855, the black area coincides with the red; in January & February 1856, the blue coincides with the black.
The entire areas may be compared by following the blue, the red & the black lines enclosing them.

Figure 2.5: Diagram of the causes of mortality in the army in the east from 1854 to 1856 [10]

diseases; the red wedges represent the deaths from wounds; and the black wedges represent all the other causes.

On a more contemporary note, with the sudden appearance of a global pandemic in 2019 that was later named Coronavirus Disease 2019 (COVID-19) and which affected the human society on a world wide level, there has been such a vast amount of data to deal with that it lead to the creation of many data visualizations.

One example of that is the COVID-19 vaccination tracker seen in the figure 2.6, that is updated daily and shows both the percentage of people who received at least one dose and the ones who are fully vaccinated relatively to the number of total population in over 80 countries and all the 50 states from the United States of America (USA). It is very intuitive and easy to use, providing an interface which allows the user to scroll or spin the world globe through the different continents on a three-dimensional (3D) perspective with the option of choosing a specific country to look at its own figures.

The data presented in this data visualization is sourced from the "Our World in Data" project at the University of Oxford in the United Kingdom (UK). Uncluttered and simple graphs show the 7-day COVID-19 vaccination rolling average as well. Interactive charts allow the viewer to sort the percentage of population given at least one dose by country or income [12].

First and foremost, since in this thesis it was already mentioned how important *Leonardo Da Vinci* was in the development of this area, it make sense to mention

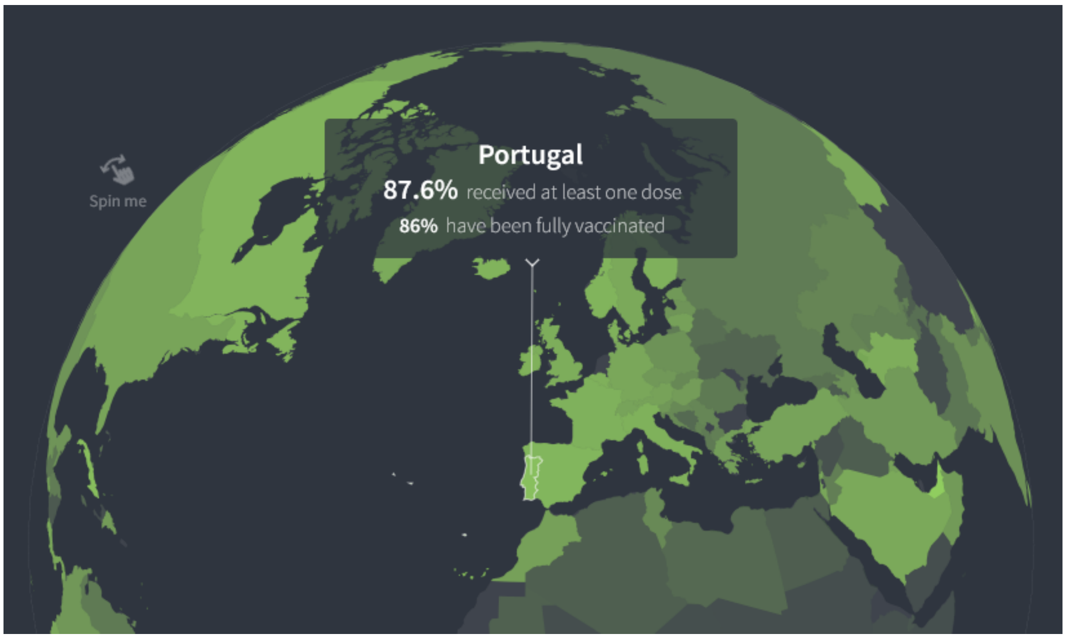

Spin me
Portugal
87.6% received at least one dose
86% have been fully vaccinated

Figure 2.6: COVID-19 vaccination tracker in Portugal [11]

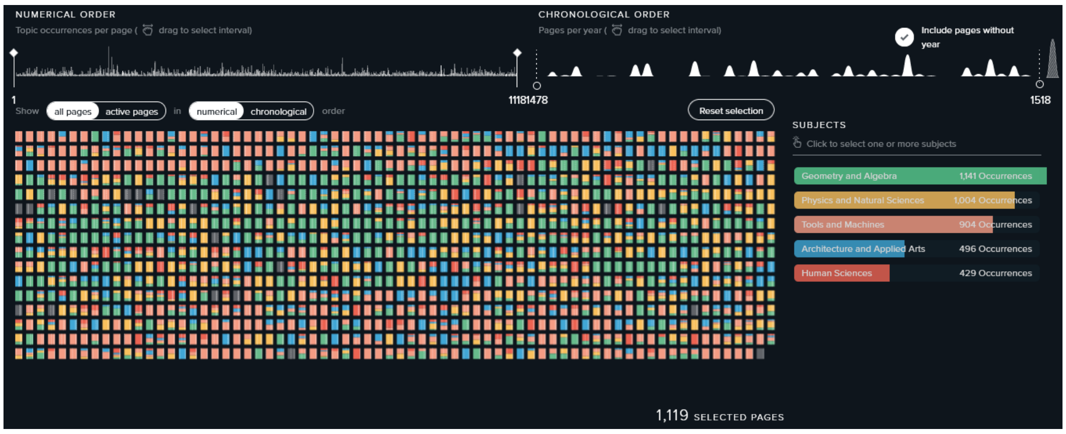

NUMERICAL ORDER
Topic occurrences per page (drag to select interval)
CHRONOLOGICAL ORDER
Pages per year (drag to select interval)
Include pages without year
1
11181478
1518
Show all pages active pages in numerical chronological order
Reset selection
SUBJECTS
Click to select one or more subjects
Geometry and Algebra 1,141 Occurrences
Physics and Natural Sciences 1,004 Occurrences
Tools and Machines 904 Occurrences
Architecture and Applied Arts 496 Occurrences
Human Sciences 429 Occurrences
1,119 SELECTED PAGES

Figure 2.7: General overview of the "Codex Atlanticus" [13]

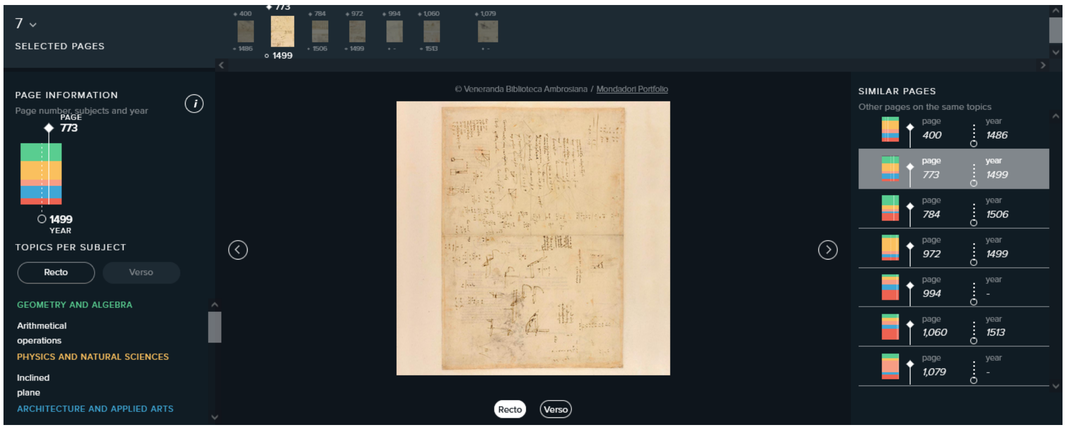


Figure 2.8: Detailed view of the page 773 of "Codex Atlanticus" (1499) [13]

an example where he is featured. But instead of going through any of his real masterpieces, it would be even more interesting to show a more modern example inspired by him.

With that being said, the last and probably the most interesting example of data visualization presented here is the "Codex Atlanticus" which is a part of the list of "The 25 Best Data Visualizations of 2019" published by "Visme" [14].

"Codex Atlanticus" is a digital library developed by an Italian group called "The Visual Agency" and it is comprised of drawings, sketches, journals and notebooks made by *Leonardo Da Vinci* himself, this being considered the biggest digital collection of his works available to date, with a total of 1119 pieces [14].

This project has already won many prizes and awards like the "2019 Gold Kantar Information is Beautiful" award in the Art and Entertainment category, for example. It is seen as a rich resource for education and research, and the original physical collection is preserved at the "*Biblioteca Ambrosiana*" in Milan.

Each work, corresponding to a page, possesses indicators such as the year of writing and the page number and in the main menu view it is possible to reorder the works according to these factors. There are many ways to rearrange or organise these works.

There are also five subjects, represented by different colours, which were identified in these works, considering that every page has a front (*recto*) and back (*verso*), and that on the same side there might be more than one subject.

These subjects or topics are shown on the right side of the main menu, as seen in the figure 2.7, and for each one of them, it indicates the colour and the the number of occurrences by the following descending order:

1. **Geometry and Algebra (green)** – 1141 occurrences
2. **Physics and Natural Sciences (orange)** – 1004 occurrences
3. **Tools and Machines (pink)** – 904 occurrences
4. **Architecture and Applied Arts (blue)** – 496 occurrences
5. **Human Sciences (red)** – 429 occurrences

It's possible to visualize each page individually and observe the different aspects, characteristics, and statistics through the use of functionalities which allow the viewer to compare the different works in detail as it is demonstrated in the figure 2.8 where you can see an example where the work in case refers to every one of the five subjects. And to its right it is possible to view all the other six examples which have the same categories.

Each work has its own "fingerprint" which consists of a rectangle with horizontal sections that vary in size and colour in accordance to the proportion of the topics that it mentions. On the other hand, the indicators previously mentioned are also represented in this rectangle by vertical lines which are arranged along the width of the "fingerprint" depending on it's value:

- **Straight line** – number of the page
- **Dashed line** – year of creation (which can vary between the years 1478 and 1519)

This can be seen in figure 2.8, to the left side of the screenshot.

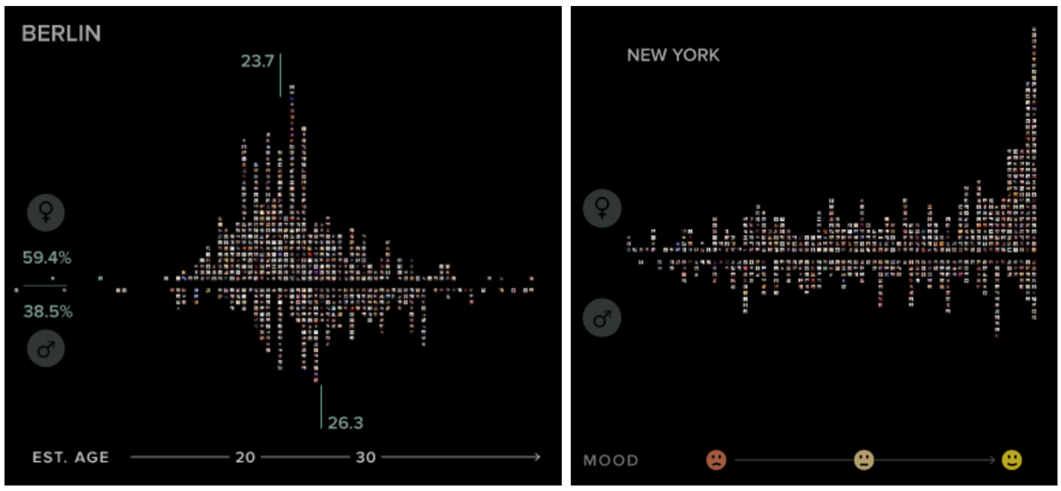
This project is very interesting due to the so-called "fingerprints" which are a good visual representation and example, compared to the work developed in this thesis that will further be demonstrated.

A data visualization should be aesthetically pleasing to enhance its message without the use of jarring colours, imbalanced visual elements, or other features that might distract the viewer in the process of correctly inspecting and interpreting the figure.

In the specific context of data visualization referring to multimedia where some visual elements are integrated and combined, such as video, text, graphics and audio, the idea of representing this type of content has the goal of giving a different and more appealing perception, which can capture the attention of the viewer either by trying to summarize the content or to describe it in greater detail, trying to highlight its characteristics. The most common examples of this could be movie covers, or the so-called "thumbnails" in digital online platforms which provide the service of video visualization such as "Youtube", "Vimeo" or "DailyMotion". These platforms are free and can be accessed through any Internet browser, but there are also streaming services which need to be paid monthly by any user, but can provide a better service and their own dedicated app for multiple devices, like "Netflix", "HBO", "Hulu", "Amazon Prime Video" or "Disney Plus".

Adjacent to these examples, which work as identifiers used to represent and distinguish different multimedia content, there are also other types of data to be visualized and analysed, such as colour histograms, audio frequency graphs or even its structure (the timeline or the chapters).

The project "Selfiecity" [15] is a great case of study for multimedia data visualization because it presents a wide view of selfie data at a transnational level. 120 000 selfies were collected from "Instagram" and were then analyzed to study how people take selfies. These pictures were selected with the help of face analysis software, from all around the world, specifically from five metropolitan cities which are: Bangkok, Berlin, Moscow, New York and Sao Paulo. This software had algorithms which recognized self-portraits, the so-called selfies, and generated an estimate of the age group and gender of the person in each photo.



(a) Berlin (age factor) (b) New York (mood factor)

Figure 2.9: Examples of distributions of miniatures with two different factors and cities from "Selfiecity" [15]

This application can identify many features like the eyes, the mouth, the use of glasses or even the mood, if the person looks calm, angry or happy. And it allows the user to filter the photos by city or by being cropped and/or rotated. One of its most incredible factors is just how comprehensive its study is, and how seriously it slices and dices every aspect of selfies [10]. It's possible to find trends in everything from head tilt angle or pose trends by city, to smile frequency by age group and gender. Unsurprisingly, selfies are much more common among younger people.

In the webpage of "Selfiecity", the two main and most interesting types of visualization are the distributions of age and smile per city, both compared to the gender which is situated on the vertical axis. These two examples can be seen in the figure 2.9 for different cities where the small squares are miniatures of every photo taken in that city, and there is a functionality that allows the user to move the cursor around the squares to see any of the photos amplified.

2.2.1 Data Visualization Tools

This subsection will function as an overview of some examples of data visualization tools which exist to make the creation of these visualizations significantly easier and better for data analysts, scientists or statisticians, because when dealing with data sets that include a large scale of data points, it helps a lot and saves a lot of time to have automated processes that work with these data sets. And depending on how big these data sets are, it can be impossible for a human being alone to study them or conclude anything just by looking through an excel file, for example, that has hundreds or thousands of cells [16]. This is an example of big data, and these tools exist basically to generate simpler and more comprehensible visualizations of this data to highlight or train the most relevant pieces of it [1].

Data visualization tools are so valuable nowadays because they are essential to the world of big data and to the current era, which has the most technologically advanced society to date, where its most valuable resource is no longer oil, but data itself [1].

Another argument for their value is that every STEM field benefits from data visualization alongside many other fields like government, finance, marketing, history, consumer goods, service industries, education, sports, and so on [1].

There are a lot of purposes for designing and developing data visualizations such as: dashboards, annual reports, sales and marketing materials, investor slide decks or any other information where there is a need to shape the data for a better interpretation [16].

These are the best and most mentioned examples of data visualization tools which were gathered from different sources [16, 17, 18, 19]:

- **Tableau**
- Infogram
- Sisense
- ChartBlocks
- Datawrapper
- Grafana
- Google Charts
- IBM Cognos Analytics
- Microsoft Power BI
- Chart.js

2.2.2 Tableau Examples

From the list of tools, the tool that stands out the most for its features and functionalities is “Tableau”. For this reason, in this subsection there will be a greater focus on it. "Tableau" offers such a wide range of output formats like maps or charts, and hundreds of data import options such as Comma-Separated Values (CSV) files, "Google Ads", "Salesforce" data and more [16]. Its most notable disadvantage is the price point (70 dollars per month per user for the "Tableau Creator" software) but, on the flip side, it has a public version which is free.

By being public it means that every data analysis developed by this version can’t be private, and it will consequently be part of their database and shown on the website “Tableau Public”, that consists of a platform to publicly share and explore visualizations online. In this platform there are over 3 million interactive data visualizations (nicknamed “Vizzes”) created by more than 1 million people from around the world, and the themes are endless, varying from sports, to politics, to music, cinema and much more. It is possible to browse through it on every imaginable topic [20].

Even though in the last section already covered examples of data visualization, the platform "Tableau" has such a variety of those that it is imperative to still mention some examples made here even if they are referred to in less detail.

And before jumping to the next topics of this thesis which are more centered in the cinematography industry, this part of this subsection will cover some "Tableau" examples related to this theme.

The first visualization is named "What are you watching on Netflix?". It analyzes the popularity of "Netflix" and of binge watching while comparing age groups, gender, diversity of content and viewership statistics [21].

The study concludes that "Netflix" is more popular among the age group of Millennials (the generation of people born in the 1980s or 1990s, by definition) and among women. It also says that the new "Netflix" viewership metrics are based on a 2-minute watch time of any series or movie and it shows a round graphic which indicates the content with more views in a relative amount of days [21].

Another interesting visualization is "The movies of the decade", a chart that uncovers the best and worst movies released between 2010 and 2019 based on the ratings made by users on the famous website named Internet Movie Database (IMDb) with a minimum of 1000 votes [22, 23]. The size of each circle is proportional to the number of votes, while the position on the horizontal axis tells the average rating from 0 to 10 (from left to right).

The next data visualization is called "Blockbuster" and it analyzes the 10 highest grossing actors of all time. In the center there is an axis of two dimensions (X and Y) where the audience and critics score are compared and each dot represents a role of one of the actors in a movie and it's sized by gross [24]. Being gross, by definition, the total quantity of money, as the amount of sales, salary or profit, without any deductions like taxes or expenses.

Adjacent to the highest grossing actors displayed in the example previously mentioned, there is also a similar visualization but for movies made in the year 2019. This one is titled "Movie Money" and it evaluates the 2019 movies releases and their domestic gross revenue grouped by each week of the year. This visualization shows many different graphics, including the one represented in the figure 2.10 which indicates the top 20 grossing movies and the overall gross of its distributors (production companies) [25].

Now, as the final example, there is the visualization "Disney Movies Income (1937-2016)", which is actually inspired by the previously mentioned "Movie Money", considering the fact that in its graph represented in the figure 2.10 it is easy to recognize that "Walt Disney Studios" is by far the highest grossing company. So, in this case, this other author decided to create a visualization specific to the "Disney" company where it is possible to look deeper into its numbers [26].

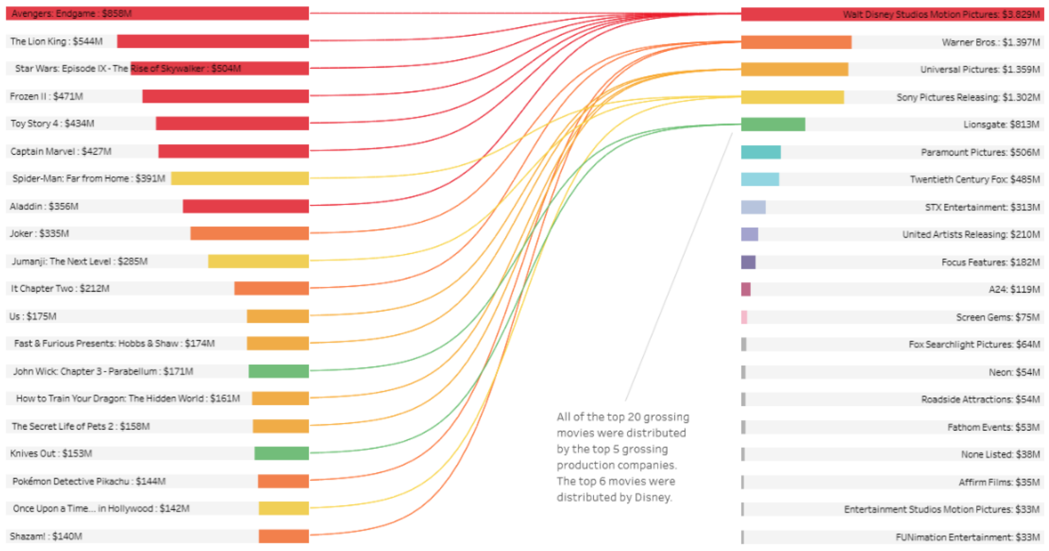


Figure 2.10: Graph of the top 20 grossing movies VS distributor displayed in the "Movie Money" visualization [25]

This example accounts for a total of 579 movies that have been produced by the "Disney" company and it shows many statistics like the fact that the 1990s were the era with highest number of movie releases so far, or that by comparing all the week days, 80 percent of the movies are released on a Friday [26].

Since the debut of its first musical movie in 1937, which was "Snow White and the Seven Dwarfs", the "Disney" company has generated a total gross revenue of 68,76 billion dollars, up until 2016 [26].

To conclude this topic, it's fair to say that there are many data visualization tools available but that only some of them standout for having a better interface, ease of use, capability of handling big sized data, collection of options or processing power. But even between these ones, there is not an absolute answer for which one is the best because in the end, since they all have different characteristics and singular balance of pros and cons, it's going to depend on the project in question.

In this situation, the solution is to look for the features that work better and point directly at the needs and goals of that specific project. Besides, in these applications, there is a trade-off between its simplicity and complexity, which means that while on one hand an application might be too basic or inappropriate for the project, in the other hand there might be others that can have too steep of a learning curve because they offer more features and resources than might be needed. Having too much complexity is not always the solution because if the project has a professional purpose then normally there are deadlines involved and even if it is on a personal level it might not be of the person's interest to waste that much time learning it.

2.3 Computer Vision

Computer vision is a field of AI that trains computers, systems or machines to derive meaningful information from any visual input like digital images or videos while using deep learning models in order to make those machines identify and classify objects accurately [27, 28].

To understand the relation between the human mind and a computer, there is this parallelism which explains that AI is addressed to thinking, the same way that the specific branch of computer vision is addressed to seeing, observing and understanding [27].

The purpose of computer vision is to replicate human vision. The difference is that the human sight has the advantage of lifetimes of context that teach humans how to tell objects apart or to perceive distance. On the other hand, a well trained system with functional algorithms and optimal performance easily surpasses human capabilities because it has no imperfections [27].

2.3.1 History of Computer Vision

Early experiments in computer vision took place in the 1950s when some of the first neural networks were used to detect the edges of an object and to sort simple objects into categories of the most common geometric shapes like circles and squares. At about the same time, the first computer image scanning technology was developed, enabling computers to digitize and acquire images [28, 27].

In 1974 there was the introduction of the Optical Character Recognition (OCR) technology which served to interpret printed text in any type of font. An advancement which was also designed to help and assist blind people. In addition, there was the appearance of Intelligent Character Recognition (ICR) which could decipher hand-written text using neural networks [27].

As the Internet matured in the beginning of the 2000s, large sets of images were being uploaded online. This growing available content started to be used for analysis which gave machines the possibility to identify specific people in photos and videos, as the digital facial recognition techniques started to flourish.

The effects of these advances on the computer vision field have been astonishing. The accuracy rates for object identification and classification have gone from 50% to 99% in less than a decade, which makes today's systems more accurate than humans at quickly detecting visual inputs [28].

By 2022, it is expected that the computer vision and hardware market is going to reach the 42 billion euros mark. It is such a substantial part of the quotidian to the extent that many people experience computer vision at such a high recurrence without even noticing when and where that technology is deployed [29].

2.3.2 Computer Vision Applications

Nowadays there is a lot of visual information that is being captured daily from people's smartphones, security systems, traffic cameras and other visually instrumented devices. Real-world applications demonstrate how important computer vision is to endeavors in business, entertainment, transportation, healthcare and everyday life [27].

The applications for computer vision are innumerable. They can go from recognizing faces, to processing the live action and broadcasting of a sports game, or to giving support for autonomous driving (self-driving vehicles) [28].

This last example refers to self-driving vehicles that have built-in cameras and sensors used for object identification and tracking which help to prevent accidents or collisions. It's essential for these vehicles and their intelligent systems to check their periphery in real-time in order to take notice of other cars, traffic signs, lane markers, pedestrians, bicycles and all the other visual and informative details encountered on the road [27].

The most obvious and common application of computer vision is image classification and there is a limitless range of examples: people's distinguishing features, animal footprints, notable landmarks and monuments, or content-based image retrieval for browsing, search and navigation in large data stores such as online platforms like browsers, websites for multimedia content visualization or social networks.

One concrete case that is worthy to mention is "Google Lens", which comes in the form of a feature in the mobile application of "Google Translator" or, by default, in most of today's smartphones, somewhere next to the capture button of the camera depending on the operating system. "Google Lens" is a tool that allows the user to identify text, labels, barcodes, QR codes and more, with the analysis of images present in the gallery of the phone or of real-time camera recording. It can also redirect users to webpages or to relevant and related search results.

2.3.3 OpenCV

There has been a massive increase in the adoption of computer vision tools and software by a variety of areas such as in healthcare, agriculture, smart cities, security, automotive industry, and more. In recent years, with the development of the tech industry, it is impossible to overlook the improvement of numerous toolkits, frameworks, and software libraries of computer vision [30]. The following list names the most notorious examples:

- **OpenCV**
- TensorFlow
- MATLAB
- CUDA
- SimpleCV
- Keras

- BoofCV
- YOLO
- Theano
- GPUImage

This subsection will have a special focus since it is destined essentially to the tool "OpenCV" for the reason that it's the one used in this project as it will be demonstrated in the next chapter. "OpenCV" is an open-source software library mainly aimed at computer vision which was created by "Intel" and originally released in the year 2000. It is the most used and well-known library for this purpose nowadays, and consequently it has a large community support.

The main advantages of it are being simple to use and the fact that it is multi-platform since it possesses multiple interfaces (Python, C++, Java and MATLAB) and it supports most operating systems (Windows, Linux, Mac OS and Android) [30].

"OpenCV" was built with the mission of providing a common infrastructure for computer vision applications, and it gives access to more than 2500 classic, fundamental and state-of-the-art algorithms able to play out some image and video processing tasks at a very high and competent level [30]. These tasks might be: object identification, facial detection and recognition, monitoring moving objects, tracking camera movements, changing the colour spaces of images, tracking eye movements, extracting 3D models of objects, creating an augmented reality overlay with a scenery, recognizing similar images in an image database, etc.

2.3.4 Convoluted Neural Network

There are two technologies which are essential to train a computer how to recognize anything: a Convolutional Neural Network (CNN) and deep learning. This type of learning uses algorithmic models that enable the computer to teach itself about the context of visual data. If enough data is fed through the model, the computer will visually inspect through the data and teach itself how to tell images apart. The point is to create algorithms which enable the machine to learn alone, rather than someone programming it to recognize an image [27].

A CNN provides vision to a machine learning or deep learning model by breaking images down into pixels that are given labels or tags. The neural network makes predictions after performing convolutions and then checks their accuracy in a series of iterations until the predictions achieve the true value the same way that a human brain, in a matter of milliseconds, can recognize and interpret at a distance what is seeing by comparing and merging features. A CNN does this process by first discerning hard edges and simple shapes, then fills in information as it runs the iterations. A CNN is used to understand single images while a Recurrent Neural Network (RNN) is used in a similar way for videos to help computers understand

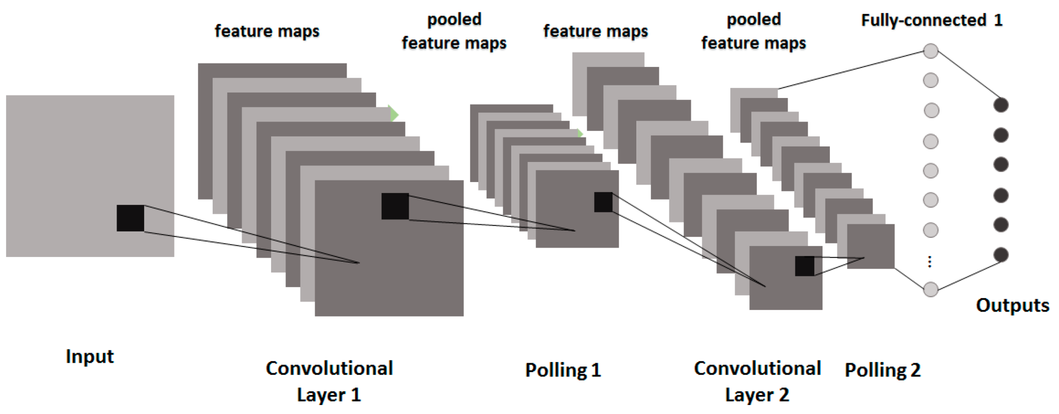


Figure 2.11: Representative scheme of a CNN and its layers [31]

how in a series of frames, each picture is related to one another [27]. This type of neural network might be interesting to use in multimedia content like movies.

In a CNN there are many types of layers: the input layer, the hidden layers, and the output layer (final layer) that is made of classes which identify the various objects or concrete aspects of the picture.

Any input is seen by the computer as a matrix of numeric values. In the specific case of an image there are, for example, 3 channels in the RGB mode which means you will have 3 different matrices overlapped.

The hidden layers can either be convolutional layers or pooling layers as are illustrated on the figure 2.11.

Inside the convolutional layers there are filters which can also be called feature detectors, feature maps or kernels. These filters are of a smaller magnitude than the input, and because of this, the process of convolution occurs in a specific way. Considering an example of a filter which can be denominated as an edge detector, in a hypothetical situation it can be a block (a mini matrix) with the dimensions of 3 by 3 where each value varies between 0 and 1. What happens in this specific case is that in this convolutional layer, the neural network is going to compare this filter to every existent 3 by 3 block in the picture (input) and, as a result, there will be a value that measures in numbers how similar each block is to the filter. Imagining that the picture also has a value between 0 and 1 for each pixel, if the computer calculates the product between these two equally dimensioned matrices the result will be a singular value which can be assigned and saved to another matrix that will consequently have the same dimension of the input.

The convolutional operator in layman terms, is the mathematical combination of two inputs, in other words, the dot product of two functions which produces a third function.

The more the filters the CNN has the more features it will be able to extract and consequently have better classifications as the end product.

The goal of the first layer of filters is to detect the simplest shapes like edges, corners, horizontal or vertical lines, circles, squares, rectangles or any other geometric shape.

The next convolutional layer will detect more detailed and concrete features like eyes, nose, beaks, hair, scales, fur, feathers or other patterns like fire, water, sand, the sky, etc.

It's easy to understand and realize that each layer is a combination of the elements detected in its previous layer and that the deeper the network goes, the more sophisticated the filters become. And that's why the final layer of the CNN can be capable of recognizing high detailed faces, full living beings like animals, plants, trees or even any other object that stands out in the picture.

The pooling layers are also very important since they are used to downsample the filters, keeping the most relevant parts and discarding the rest. This happens in order to reduce the overfitting rate. So that calculations can be sped up in further layers due to the reduction of the image spatial size, of the "noise", background, or emptiness in the image. These layers must be studied and developed optimally to prevent overfitting and underfitting. Overfitting is a concept in data science, which occurs when a statistical model fits exactly against its training data, while underfitting represents its opposite [32].

2.3.5 Colour Theory

Colour is a concept which represents the result of visual perception obtained through photo-receptor cells of the human eye and provoked by the electromagnetic radiation (of the visible spectrum) that takes place in nature. An object appears coloured because of the way it interacts with light [33].

Colour theory, on the other hand, is the collection of guidelines and rules utilized by designers in order to communicate with users through appealing colour schemes in any type of visual interface [34].

In 1666, the English physicist Isaac Newton established colour theory when he invented the colour wheel due to his prism experiments [33, 34].

And even though he recognized that the spectrum was continuous, Newton used the seven colour names: red, orange, yellow, green, blue, indigo, and violet to represent the segments of the spectrum by analogy with the seven notes of the musical scale [33]. And at the same time, to categorize colours, he defined the following three groups:

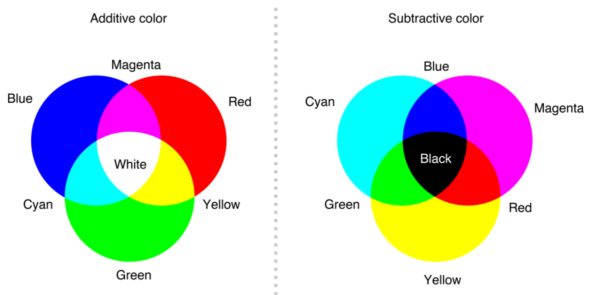
1. **Primary** – red, blue, yellow
2. **Secondary** – mixes of primary colours
3. **Tertiary (or intermediate)** – mixes of primary and secondary colours

The spectrum of colours, which can be seen entirely by refracting light through a prism, varies between approximately 430 and 750 terahertz (THz) of frequency and between 400 and 700 nanometers (nm) of wavelength of colours that extend from the red to the violet, this last being the one with the smaller wavelength [33].

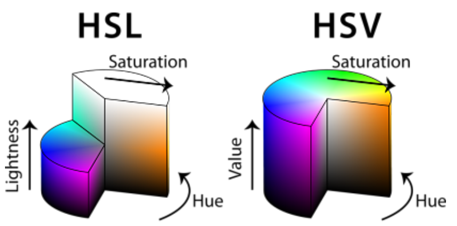
However, what this spectrum does not show is the variation of intensity of each colour, which if taken into consideration expands the quantity of colours to a much bigger scale where other colours take place such as the black, white or grey.

With this being said, each existing colour carries three properties [33, 34]:

- **Hue** – the correspondent segment of the visible spectrum.
- **Chroma (also known as saturation)** – which defines how pure is the colour compared to the original hue, because it can have shades (black added), tints (white added) or tones (grey added).
- **Brightness (also called intensity or value)** – which depends on the total amount of light energy present. A combination of hue and chroma.



(a) RGB and CMYK [35]



(b) HSL and HSV [36]

Figure 2.12: Graphical representation of four different types of colour models

For colour representation there are many different models or colour spaces developed such as:

- **Red, Green, Blue (RGB)** — where the colours are demonstrated and measured by the intensity level of each one of these three categories present in its name.
- **Cyan, Magenta, Yellow, Black (CMYK)** –- frequently used in the process of printing and it works as the opposite of RGB, which is of an additive nature because in this procedure the categories present in the name are subtracted as it is demonstrated in the subfigure 2.12a.
- **Blue, Green, Red (BGR)** -– similar to RGB but with a different order of categories.

- **Hue, Saturation, Value (HSV)** -– a scheme which is represented as a 3D shape, normally a cylinder or a cone where the parameter hue corresponds to the colour of visible spectrum, while the indicators saturation and value just represent respectively the intensity of the colour in question and its proximity to the colour black.

- **Hue, Saturation, Lightness (HSL)** -– it's represented the same way as the HSV but with the exception that instead of the indicator of value it uses the indicator of luminosity, which indicates the proximity to the colour white, in contrast. This is portrayed as in the subfigure 2.12b.

- **CIE 1931 XYZ** -– this model refers to the first attempt of creating a colour space based on the measure of colours by the human perception. It was created in the year referred to in its name by an international authority called *Commission Internationale de l'Éclairage* (CIE) and it would become the pioneer and base for the remaining colour models and spaces. It's a graphic 3D model that possesses the X, Y and Z axis.

- **CIE, Lightness*a*b (CIELAB)** -– a model also defined by the CIE (which was responsible for the creation of many other norms and models related to colour study). In this graphic model, also 3D, the three axes correlate to the parameters A, B and luminosity.

It is important to mention that for all of these models the range usually goes from 0 to values which are powers of two, mathematically speaking (2^n - e.g. 0-255), because at a computing level it's more convenient due to the number of bits.

In 1996, the concept of standard RGB (sRBG) was formalized collaboratively by the technology companies "Hewlett-Packard (HP)" and "Microsoft" for Internet use.

Also the term "Web Colors" was created to represent webpages in the RGB through a six-digit number formed by 3 bytes in hexadecimal notation and named as "Hex Triplet". It is based in RGB for the reason that each byte corresponds to each one of its categories red, green and blue. It can be used in HyperText Markup Language (HTML) or Cascading Style Sheets (CSS), for example.

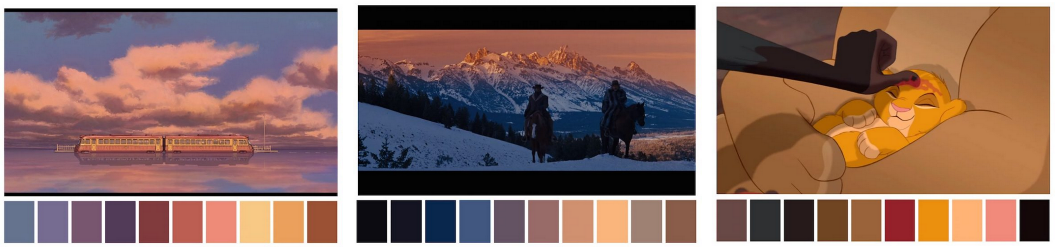
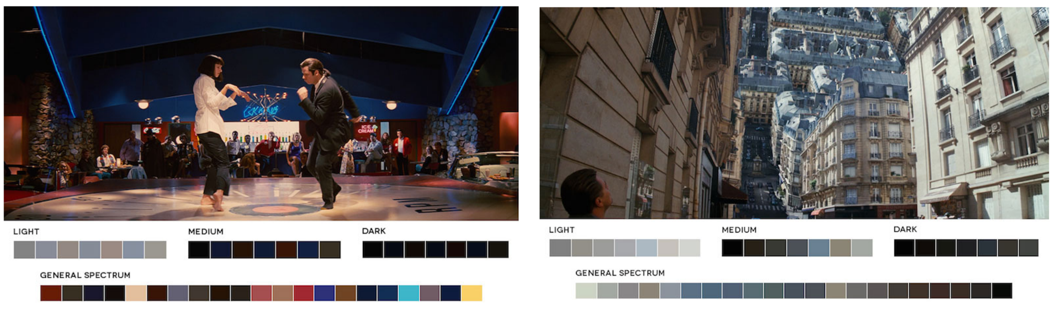

(a) "Spirited Away" (2001) (b) "Django Unchained" (2012) (c) "The Lion King" (1994)

Figure 2.13: Similar colour Palettes from famous movies [37]

The idea of generating colour palettes for frames of movies is very interesting and useful to graphically represent these movies, as it is visible in the figure 2.13. The colours used in a movie are very elucidating of the mood in which the directors try to project every scene. For example, in the movie "The Matrix" (1999) it is often seen the colour green mixed alongside with black to suggest a theme of the early monochrome computer monitors used back in the days. Also, in the movie "Kill Bill" (2003), the colour yellow is portrayed a lot, symbolizing Uma Thurman's character's madness and instability. Even with movie genres, it is possible to detect some patterns like the use of pastel shades (beige, pink or lilac) in romantic comedies or shades of blue, green, black and white in Sci-fi movies [37].

A good case for an intriguing project involving colour palettes is "Movies In Color" [38] created by a Los Angeles based designer named *Roxy Radulescu*. She derives three different palettes from each movie still (with a total of 388 movie stills) for different levels of lightness and then merges them into a general spectrum with 21 colours, as it is illustrated in the figure 2.14. These visualizations demonstrate just how much the tone or intensity of the colours present in a frame can describe the intensity of the movie scene itself.



(a) "Pulp Fiction" (1994)

(b) "Inception" (2010)

Figure 2.14: Colour Palettes of two movies, that are categorized by three levels of lightness: light, medium and dark [38]

To replicate this idea of colour extraction of images, it's important to think how it would be possible to generate these colours through reverse engineering. And the first thought would be to count the occurrences of each pixel and its colour present in the image and order them from the one with the most occurrences to the one with least. In other words, every pixel of the image would be analysed and it would be seen how many of them are duplicates while grouping and counting them if they had the same value of colour. And to put in perspective a 720p frame which is one of the most common resolutions used in videos nowadays is the equivalent of 720 vertical pixels times 1280 horizontal pixels, as seen on the figure 2.15, which makes a total of 921 600 pixels.

The first problem encountered in this reasoning is that the number of possible colours are not being taken into account because the RGB model is the one being

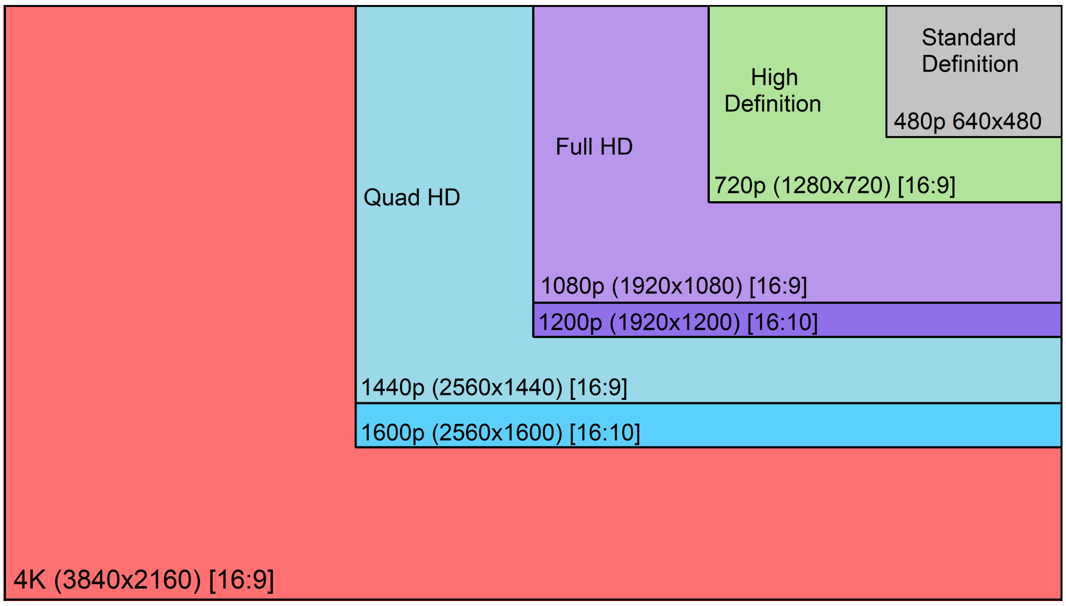


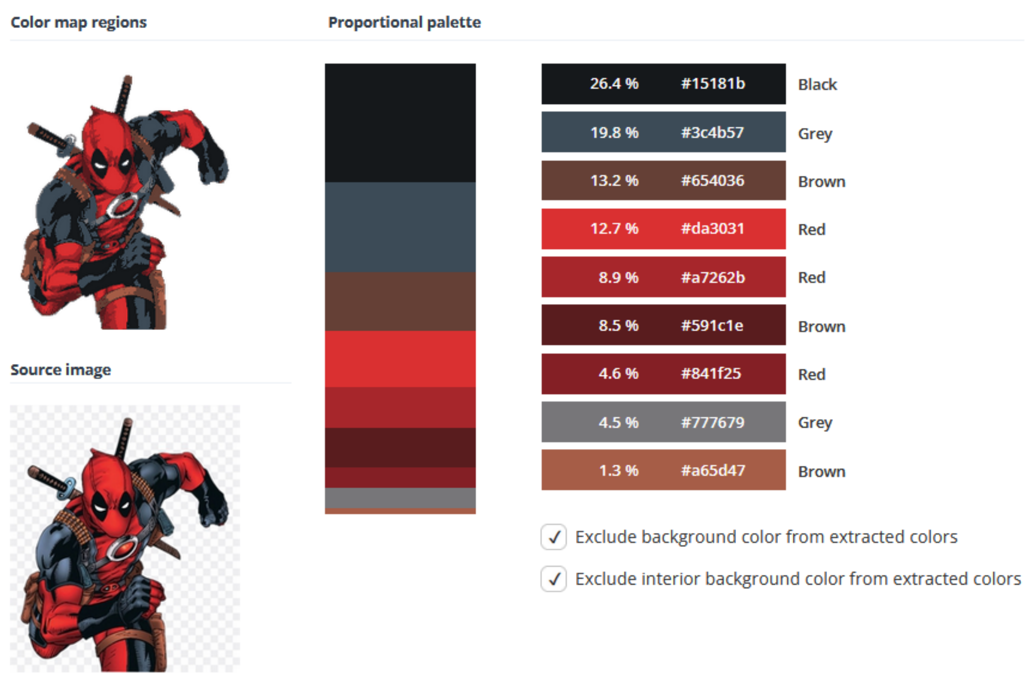
Figure 2.15: Overview of the different and most common resolutions [39]

used, where we have a total of 16 777 216 colours due to the combination of the three categories red, blue and green considering they have 256 values each.

With this being said, it's easy to conclude that even in such a large image with so many pixels like one with a 720p resolution, there are still going to be few or even no colours repeated especially in real action movies where the scenes are recorded by cameras which capture many natural fades or gradients of colour like shadows, or any other details that give the viewer the perception of depth in the characters, the objects or in the environment. Also, even if it's considered the smoothest gradients imaginable in a frame there would still be a difference between the pixels even if they weren't recognizable or distinguishable by the human eye.

After taking all this in consideration, the next and most obvious step of reasoning would be to somehow shorten the number of colours into a smaller scale. Ideally it's possible to achieve this goal with the use of an algorithm which will process an image in a way of extracting a final product like a copy of the original but with a smaller number of colours and consequently with less detail. In brief, an interval of colours would be defined where most of the pixels would be substituted for a more generic colour that would represent the interval where that pixel (colour) would be inserted. This process already exists, called clustering, and has been used in the area of computer vision.

The figure 2.16 reveals an example of the process of clustering where the original image is situated down and in the upper left corner it is displayed the final result, which is effectively less detailed in order to extract a smaller batch of colours.



26.4 %	#15181b	Black
19.8 %	#3c4b57	Grey
13.2 %	#654036	Brown
12.7 %	#da3031	Red
8.9 %	#a7262b	Red
8.5 %	#591c1e	Brown
4.6 %	#841f25	Red
4.5 %	#777679	Grey
1.3 %	#a65d47	Brown

✓ Exclude background color from extracted colors

✓ Exclude interior background color from extracted colors

Figure 2.16: Example of colour clustering [40]

All the content visible on the figure 2.16 was generated with a tool provided by the "TinEye Labs" [40].

"TinEye" is a tech company based in Toronto, Canada, that delivers image search and recognition solutions supported by the use of computer vision, neural networks and machine learning, and with their mission being to make images searchable [41].

Besides giving the information about the proportion of each extracted colour, this tool also allows the user to navigate through similar images by clicking on the colours. By similar it means that if the user selects a specific colour from the list, the webpage will redirect to a gallery of pictures of that colour. The user can also click on the full palette and the same will happen but for comparative images that have the same colour pattern all around. All these images are sourced from the database of the website "Flickr".

2.4 IMDb and other Movie Databases

Between the themes of big data and data visualization for multimedia, one of the most common subjects is movie data.

The concept of film has been around since the 19th century through the shape of the art of motion picture photography, or as it is called cinematography. The term "Cinema" is also used as a short for it.

The film industry and its popularity has come a long way and it is immersed in Hollywood, a neighbourhood in the central region of Los Angeles, state of California.

Nowadays, there have been forms of gathering and saving film content and its information digitally. The main source of movie data is the, as it has always been the acclaimed and distinguished website Internet Movie Database (IMDb) [23]. "IMDb" provides information about thousands of movies and TV series.

The origin of "IMDb" dates to 1980 when Col Needham, an English software engineer and movie fanatic from Bristol (England), began to create a list with the movies he had seen until then. As the years went by, as he kept watching more movies, he was making the list bigger and bigger, until the day October 17th of 1990, when Needham posted his movie listing software in a "USENET" film discussion group [42].

The site was cooperatively improved and expanded, and later became an early migrant to the World Wide Web (WWW). Then in 1998, the company "Amazon" bought the rights to "IMDb". And even though, since then, the website has been owned by "Amazon" as a subsidiary, it never lost its true identity and autonomy, with the founder, Col Needham, remaining in the position of Chief Executive Officer (CEO) [42].

Since then, the "IMDb" site has grown beyond Needham's expectations as what he saw in the beginning as a recreational project. The website holds all kinds of imaginable information about any movie from cast and plot description to awards and gross values.

It is an interactive website where users can rate and review movies or look through lists of the top rated movies by gender, year, director and other factors. And with the growth of smartphones, eventually the company also developed a mobile app.

In 2008, "IMDb" made an important acquisition by buying the rights of "Box Office Mojo", a website founded in 1999 that parses Hollywood box-office grosses in great detail [42].

Nowadays, the IMDb's headquarters are based in Seattle, USA. The website is managed by a team of contributors who work towards maintaining and improving what is seen as the best platform and service available for movie data.

Other honorable mentions go to "Rotten Tomatoes" [43], "Metacritic" [44], and "Letterboxd" [45], with this last one being the example which better fits the definition of social network. The edge that "Letterboxd" has over other platforms is that it has a very clean, simple and easy to use interface. "Letterboxd" is a social network where users can discuss and rate movies and share opinions and reviews which can be seen by other users. It allows the user to keep track of every movie that he has watched, and at the same time, adding other movies to the watchlist, to see them in the future.

The website is named this way, because the concept of "Letterboxing" is defined by the practice of broadcasting movies in a widescreen with a different aspect ratio compared to the standard width of the typical movie format (by adding black bars above and below the movie frames) [45].

Chapter 3

Cinemetrics

In this thesis there will be a greater focus in the cinematography industry, more concretely with some of the most popular movies that have been produced over the years. And for the niche of people and organizations who dedicate themselves to observing and interpreting cinematographic pieces, either on a professional or recreational level, there have been created tools to visualize data with different functionalities, for the purpose of facilitating and going deeper in the process of movie representation and post analysis.

The modernized meaning of "Cinemetrics" is about creating a graphical and appealing representation of a movie and of its content, partial or total, revealing different features such as: the colour palette, the audio, its structure, lines, motion and among others.

The concept of "Cinemetrics" was born from an eponymous website [46] founded in 2005 by *Yuri Tsivian*, a professor from the University of Chicago who, until this date, still runs the website while working in the Departments of Art History, Cinema and Media Studies of that institute. For this reason, it's easy to conclude that the professor *Yuri Tsivian* is the biggest pioneer and responsible for the majority of the impact and rise of this concrete area of studies that is "Cinemetrics". In 1984, he earned his *Philosophiae Doctor* (PhD) in film studies from the Institute of Theater, Music and Cinema in Saint Petersburg (named Leningrad at that time), Russia [47].

The central purpose of this website [46] is to study films in an easier and more practical way. Therefore, it has a data collection tool that facilitates the annotation of movies and the analysis of their content like the dialogue scenes or the action.

It functions as a repository that exhibits the database of the quantified data from a vast volume of films of many different genres across history. Since it's a repository, every user can contribute with their examples in order to expand the database [48].

The website also has a section that mentions published articles which have used the website's database. And for all of these reasons, it is considered as a notable source which can be inspiring and accessed by scholars and researchers who are interested in taking a quantitative approach to film studies [48].

Back then, the "Cinemetrics" was thus understood as a subject of statistical analysis of quantitative data which describes the content, structure and style of films.

3.1 Concepts of Cinemetrics

The following section works as an introduction for basic concepts and terms applied to "Cinemetrics", which are frequently used in quantitative analysis of movies or any other type of digital videos.

The term that will probably be the most discussed and mentioned in this project is the shot. A shot is the interval between one cutting point to another, of a camera recording sequence that can be edited in the digital production of a movie and consequently positioned next to other shots, which are assembled to make the structure and constitution of the movie. And the instant breaks which separate the shots are called cuts.

Another commonly referred term is the frame rate, which corresponds to the number of frames that are displayed or projected for each period of time, usually per second, being the Frames Per Second (FPS) the most universal measuring unit.

The standard frame rate for movies and TV shows is 24 FPS, because it was determined to be the most optimal speed needed to capture video while still maintaining realistic motion. The human eye sees about 30 to 60 FPS, this means that some people might not have the perception or the capability to distinguish between 30 and 60 FPS, for example. A video with a higher frame rate tends to have a higher quality, and also a bigger file size for the obvious reason that it comprises more frames [49].

Not all devices and multimedia platforms support all frame rates. The higher the frame rate, the more detail the video will have. The frame rate of 30 FPS is normally used to portray videos with higher motion than usual. Any frame rate of 60 FPS or over, is commonly used in video games and their recordings, because a lot of things are happening at the same time on a computer or console screen during a video game. It is also used in sports broadcasting because, besides some of them happening at a very fast speed, it is important sometimes to slow down moments to show replays in at a high quality [49].

Shot length, as the term implies, is the number that indicates how long a shot is. For each movie, there can also be the terms: Average Shot Length (ASL) and Median Shot Length (MSL), which might be of interest to compare between movies to characterize them. The ASL can be calculated with the division of the duration of the movie in seconds by the total number of shots. The numerical inverse of ASL is the number of shots (or cuts) per second which is normally converted to cuts per minute for a more convenient analysis [50].

In the 1970s, an Australian film historian named Barry Salt first suggested that studying the ASL of films would help characterize them while discerning different aspects between directors' styles and between different times depending on the approaches that were more trendy and prevailing in the year each movie was made. This would later help narrate the evolution of the film industry [51].

During the pre-production of a movie, the director, alongside with the cinematographer, create a shot list which is a list that describes explicitly how each shot should be filmed to best illustrate the story visually and indicates all the logistic details for the set such as the type of camera, equipment, recording techniques, actor and lines, sound and the description of the action. The best movies made until this date, the big Hollywood blockbusters, were not filmed sequentially since that would be inefficient and it would slow down the production. The purpose of creating a shot list is to establish the most optimized shooting schedule possible [52].

3.1.1 Shot Transition Detection

Shot transition detection is a field of research of video processing, and it can also be called shot boundary detection, cut detection or simply, shot detection.

It can be done through the use of algorithms based on [53]:

- Histograms (HSV or RGB)
- Deep learning (with CNNs)
- Genetic algorithms
- Fuzzy Logic
- SVMs
- Theory information
- SURF matching score

While processing a video, the simplest way to come up with a threshold is by a principle with two phases: scoring and decision. In the scoring phase, the idea is to assign a value (a score) to each pair of consecutive frames which will quantify the resemblance (or the difference) between them. Then, in the decision phase, there is the need to evaluate and study all the obtained scores and assign a value to the designation of shot boundary detection. A value which is high or low enough, that overrides a certain limit, the so-called threshold.

The difficulty is essentially in detecting soft cuts, because the hard cuts are easy to detect since the values of the pixels change abruptly from one frame to another. For a human, the task of identifying any type of cut is simple. However, it is much more complex to detect gradual cuts (like fades) through the use of algorithms.

3.1.2 Fades

One way to mitigate this problem is by using a third method which consists of detecting a series of near to black frames as a possible indicator of the cinematographic technique called "fade to black". As the name implies, this happens when a shot fades into total darkness and another shot comes into play.

It is important to mention that inside the category of fades there is also: the "fade to white", which in contrast to "fade to black", and as the term implies, it goes into whiteness instead of darkness; the fade in, which happens when the scene gradually appears on screen; and the fade out, which is the opposite of the fade in, and it occurs when the shot gradually turns into a single monochrome frame [53].

The transition effects that are more complicated to detect are: the dissolve, which consists of overlapping two shots, i.e. when a shot starts to disappear gradually, being replaced by the next shot which is appearing gradually simultaneously; and the wipe, a moving effect which can be done in any direction, when a shot "pushes" the other one off the screen [53].

The task of detecting these transitions can become really complex if they are done smoothly. And while there are a lot more transition effects, the positive side is that they are rarely used in movies, and more in presentations, photo collections and other types of videos.

The "fades to white" deserve an emphasis on this subsection for the fact that they are not as common as the "fades to black", but can be as much interesting as those. The traditional "fade to black" can be used in a movie ending normally to give a sense of closure, a certain irrefutable finality. In contrast, the "fade to white", when used by filmmakers as an inversion of the norm, can raise the hypothesis of an enigmatic scenario for what would follow, creating a certain suspense or mysterious and ambiguous sense, like a "cliffhanger" [54].

The usages of this type of fade can be: to portray death, or the ascension to higher abstract or imaginary planes like heaven or a dream, for example; to transition into a scenery of snow; to represent a temporal or physical teleportation effect; or to illustrate an explosion, or a flash of light, where the camera often focuses on a character's face, while the screen turns white [55].

There are many famous movies where "fades to white" take place, such as [55]:

- "Star Trek V: The Final Frontier" (1989)
- "Looper" (2012)
- "The Hunger Games" (2012)
- "Eraserhead" (1977)
- "Requiem for a Dream" (2000)
- "Black Swan" (2010)
- "Total Recall" (1990)
- "The Thing" (1982)
- "Leon: The Professional" (1994)
- "Solaris" (1972),
- The series of "The Lord of the Rings"

It can also be found in video games, comic books, and music videos [55].

3.1.3 Shot Scale

Then there is the so-called scale of shot that was the standard one used in the 1940s by the film industry and describes the perspective and proximity in which shots are recorded. This scale is categorized as seen in the figure 3.1 and described in the following list [50]:

1. **Big Close-up (BCU)** – shows head only
2. **Close-up (CU)** – shows head and shoulders
3. **Medium close-up (MCU)** – includes the body from the waist up
4. **Medium Shot (MS)** – includes from just below the hip to above the head of actors in an upright posture (vertical position of being sitting or standing with the back straight)
5. **Medium Long Shot (MLS)** – shows the body from the knee upwards
6. **Long Shot (LS)** – shows at least the full height of the body
7. **Very Long Shot (VLS)** – shows the actor distant in the frame

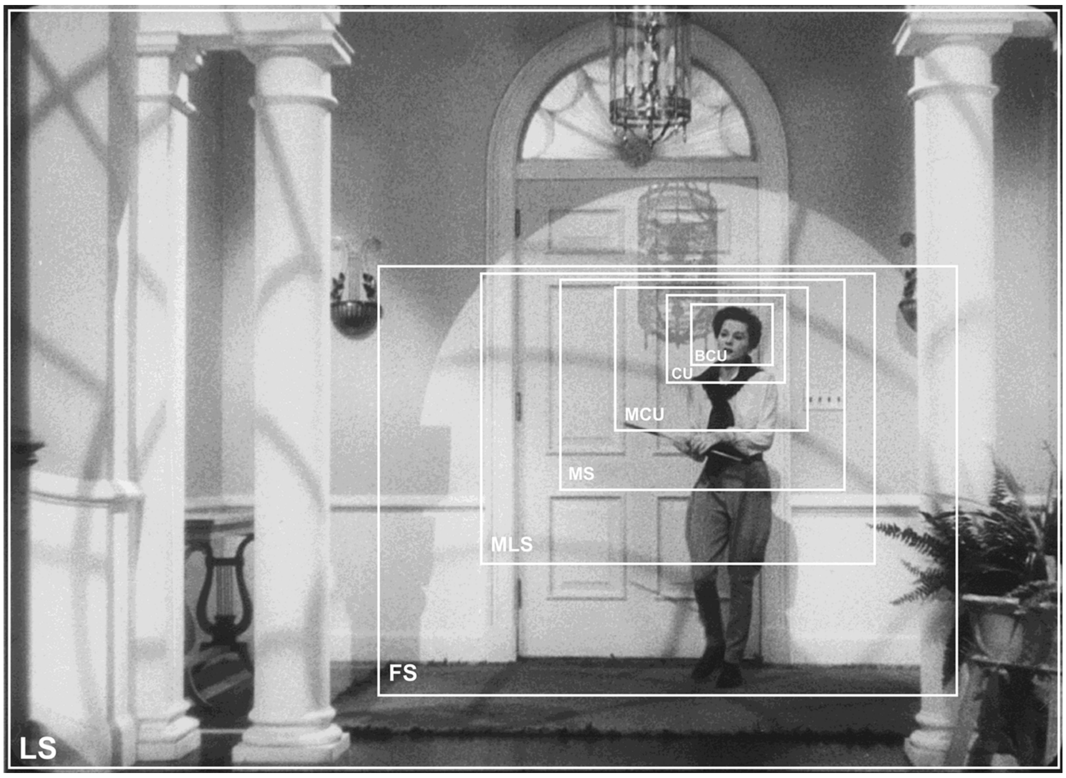


Figure 3.1: Descriptive image of the shot scale and its categories as frames [50]

In figure 3.1, it's easy to realize that the acronyms are not exactly the same. This is due to the fact that there is not a definite and universal standard since there are so many different models. And also because of the directors who play such an important role in deciding angles and viewpoints of recording that most of them have their own style which makes it pointless to have so many denominations if in the end they might not use them.

In recent decades in the cinematographic industry, the term "Wide Shot" has been introduced in replacement of the different types of long shot [50]. But for the nature of these projects, it is essential and more convenient to detail and distinguish the shots as much as possible.

3.1.4 Camera Movement

There is another form of categorizing shots which is related to the camera movement and it's crucial to acknowledge it because it can make the previous categorization a bit ambiguous considering that the same shot can have more than one perspective or have variation in distance. The most typical techniques of moving the camera while recording are:

- **Panning** – swivelling the camera to the sides (horizontally) from a fixed position. The term has its origin in the word "panorama" and equivalently, it shows a wide angle that couldn't be visualized through a normal picture or with the eyes in a still position due to the limitations of the human spatial and peripheral vision. This technique has the aim of gradually revealing and incorporating off-screen space into the shot.
- **Tilting** – analogically to panning, tilting is the same except that the camera rotates vertically (up and down). This type of motion is similar to the point of view of a person raising or lowering their head to look up or down. This can be the real intention of the shot, to look through the eyes of a character, or to express an effect of surprise or attention.
- **Zooming** – As the word suggests, this is the act of performing a zoom with the camera to get closer or farther (zoom in or zoom out) from what is being recorded. For this, each camera with this functionality can possess a parfocal and a varifocal lense in which the first one keeps the focus while zooming while the other loses it.
- **Tracking** (or "travelling") – physically moving the camera by pushing or pulling it in any direction (forward, backwards or to any side) through some rails (placed on a railroad track). It can give the impression of walking/moving or to follow a character. This technique also works with a handheld camera or a gimbal to make the recording steady because, in the end, it depends on what the director is trying to portray making the shot more natural and raw.

To describe shots, it's possible to go even further, and combine any of these techniques where any combination between these four methods is actually conceivable.

Besides these techniques, any recording can be done with a crane or a drone instead of with a normal camera directly and physically handled by a professional.

3.1.5 Shot Length

In the "Cinemetrics" website of the professor *Yuri Tsivian*, there is a section called "Measurement database". Inside it there are two different databases: the original one, which gathers some of these concepts like ASL and MSL for many different movies, with more than two thousand submissions made by users; and the second one which was supplied by the historian Barry Salt and gives a bit more of details than the original, including the camera movement techniques that were just discussed.

This data provided by the historian Barry Salt was a major step in what would later become the concept of "Cinemetrics" for the fact that he single-handedly annotated all this information by hand and without the help of any digital computerized tool. And because of this, it is fair to presume that Barry Salt's pioneering work was limited by the laborious, time-consuming exercise of measuring films in the pre-digital era in which he was at his peak [51].

The theme of "Cinemetrics" is seen somewhat as an inexact science due to the fact that counting the exact number of shots can sometimes be difficult. If counted manually or even with an assistant digital tool, it is always prone to some error also because the definition of cut is still a bit ambiguous. And in these specific databases [46] there is no second step verification of the data that is submitted [56].

By interpreting many of these data sets which compare the ASL or the cuts per minute of different movies in time, the general idea that is given is that over the years, more concretely in the last decades, the number of shots per movie as drastically increased. A suitable reference that supports this theory is the visualization called "Cut" [57], made and exhibited with "Tableau" by an American data visualization artist and analyst named Bridget Winds Cogley. This "Viz" shows different graphs that demonstrate how the ASL varies over the years or comparing it between different directors. This study, like many others, is based upon the data from the "Cinemetrics" original website [46] and from "IMDb" [23].

According to an article [56] in which movies released between 1997 and 2016 were reviewed and sourced also from *Yuri Tsivian*'s databases [46], the average number of shots in a movie is 1045. The most notable cases from this list are: "Doomsday" (2008), being the one with more shots at an exceeding 4052 shots and categorized as an action movie as expected; and "Russian Ark" (2002), a movie with a duration of 99 minutes filmed in one continuous shot.

And most recently, in 2019, there has been another fascinating case study of a movie with only one shot. This movie named "1917", is a drama set during World War I and its story is about a pair of young soldiers on a mission to deliver a message to stop an attack [58].

The idea of the director Sam Mendes was to give the perception of real and natural time along the whole movie. He wanted to make the audience feel the rush and anxiety that the characters felt as time was passing and to accompany every step they made, while giving the awareness of the difficulties they had to go through like the physical distance they had to travel. Essentially to create affinity with the characters as the viewers would sense all their emotions and live the story through them during the whole 2 hours of the movie [59].

Most of these movies are not actually recorded in one shot, they are instead a unification of a smaller sample of shots which are recorded in a way that the first and last frame coincide with one another. The better a production achieves this goal, the easier it will be to edit and conjugate those shots in post-production to make the sequence look as more fluid as possible giving it an appearance and illusion of a singular continuous shot.

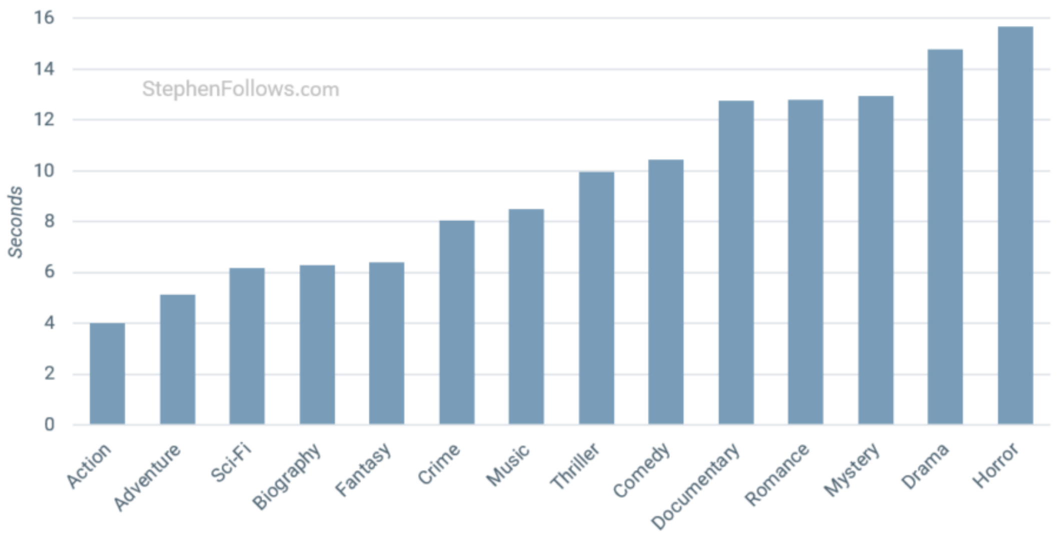


Figure 3.2: ASL compared between different genres [56]

Another example of a movie where this effect occurs is "Birdman" (2014) which was directed by *Alejandro G. Iñárritu*. "Birdman" tells the story of an actor who began to lose his popularity and then decided to revive his career by performing in a Broadway play.

A Canadian film editor named Harrison Edgecombe posted a 4-minute video on "Vimeo" in which he gathered a total of 28 hidden cuts present in the movie "Birdman" (2014) [60].

And even though the author states that his work does not serve as a debunking of the seamless look of "Birdman", but rather appreciates it much for the use of this technique, it can still be said that 1917 (2019) is clearly the best ever movie that was made to look like a single take, in other words, the one movie in which this technique was better applied [60, 58].

The best metric to compare movie shots is to use the ASL instead of the number of shots because it takes note of the duration of each movie in relation.

In the figure 3.2 it is demonstrated how the value of ASL fluctuates between the different genres of the film industry. And unsurprisingly, the first three genres that are displayed on the graph as having a faster pace are: action, adventure and science fiction. While, on the other hand, horror, drama and mystery movies are the slow paced ones presumably due to the long duration of some shots which are infused within action of the movie to create an environment of suspense.

Especially for action movies which have an ASL of 4 seconds according to the graph represented on the figure 3.2, it can be oftentimes overwhelming for the viewer to be constantly exposed to so much movement and shot transitions. At the same time, for other viewers it might be their preference and an ideal formula to break their monotony of their daily routine as their organism is releasing adrenaline.

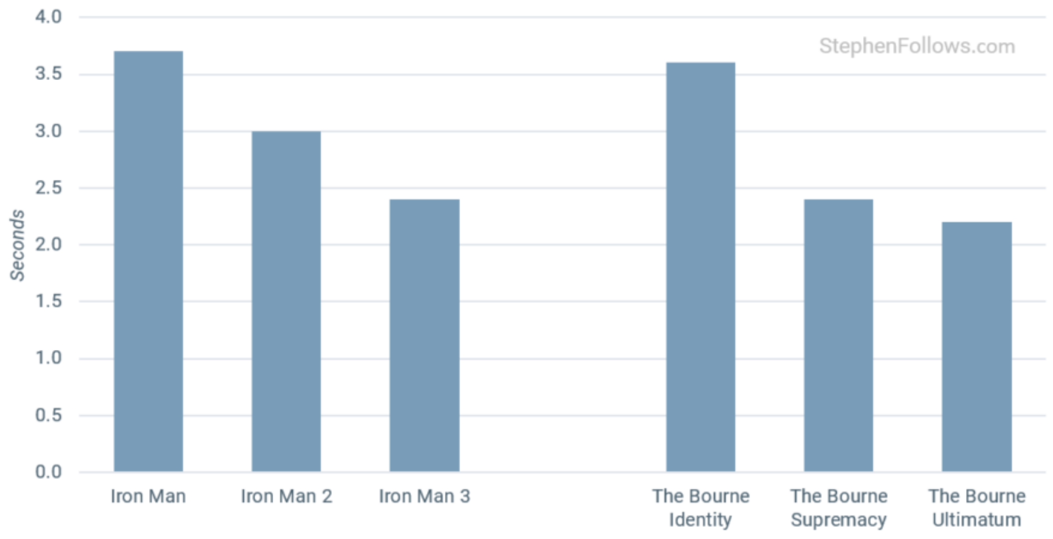


Figure 3.3: ASL compared between two different sequels and their respective movies [56]

Lately there seems to be somewhat of a carefree attitude in movie productions, especially with series in which the first movie had a great success, and there is a rush to try to capitalize on its success by making sequels, sometimes with diminished gains, quality-wise.

In the same article mentioned before [56], the author tries to prove how the ASL has changed over time (how it has shortened), but he didn't have enough data to support this. However, he concluded, with a graph (represented in figure 3.3), that both the "Iron Man" and "Bourne" series had a considerable reduction in the value of ASL over the course of their three movies.

The same can be said for the movie "Taken 3" (2014), where there is an infamous scene of 6 seconds that needed 14 camera shots, only to show actor Liam Neeson jumping over a fence. The pattern displayed here is the same since this movie was also the last one produced of a series of 3 movies [61].

This is a good example that demonstrates how the practice of using too many shots in a scene can be labelled as derogatory to the work, but this doesn't necessarily mean that movies with a bigger ASL are better than fast paced movies.

However, the movie "There Will Be Blood" (2007) directed by Paul Thomas Andersen, is an admirable case study for having a high ASL in a long course of action of 158 minutes. The ASL is approximately 14 seconds, which is a lot compared to the 3 or 4 second average among Hollywood movies today [62].

This movie is an example where the cuts have, in fact, more impact, not just because they occur less often, but also for the cinematographic techniques used by the director such as close ups or movement tracking. This can be explained by a logic chain in which longer takes mean more impactful cuts, which consequently mean a more devoted attention to framing and composition. The shot length decreases at such a slow speed over movie's action that it is imperceptible for the viewer, but at the same time, in a paradoxical way, noticeable enough to unconsciously give the viewer a disconcerting effect as the drama builds up. By deconstructing all the 678 shots of this movie, it is possible to see how each one ends up serving a greater dramatic purpose [62].

As an added curious comparison, according to the "Cinemetrics" database [46], the longest shot in the movie "Taken 3" is 11.4 seconds, which is even lower than the ASL of "There Will Be Blood".

3.1.6 Other Case Studies of Cinemetrics

An example of an inspired project that followed the work of professor *Yuri Tsivian*, which established the concept of "Cinemetrics" as the annotation of movie data, is the "Shot Logger" [63].

"Shot Logger" is an open-source project that consists of a database with more than a thousand of films and TV programs logged, that uses the "VLC Media Player" to capture images from video. It was born in 2007, on August 8, when a professor from the University of Alabama named Jeremy G. Butler, uploaded shot data of an episode from the TV show "Friends" (1994) into the system database of a server [63].

Until 2020, all the data from "Shot Logger" was only allocated in that server, which belongs to the Telecommunication and Film Department of the College of Communication and Information Sciences from that university [63]. Meanwhile, the data was moved to a new server, but the professor, nowadays, still uses this service to lecture on the subject of Creative Media and to show it to his students.

There is another interesting project, made by an Italian investigation group, related to the study of shots, which associates movies to their directors. It is a recognition based on the statistical analysis of two simple shot features: the duration of each camera take (shot), and the shot scale [64].

The authors of this project produced four different data visualizations, represented in figure 3.4, which were based on annotated Ground Truth (GT) data of movies from six different directors: *Michelangelo Antonioni*, *Ingmar Bergman*, *Federico Fellini*, *Jean-Luc Godard*, Martin Scorsese, Quentin Tarantino, and *Bèla Tarr*. More precisely, 77 movies for shot duration GT data and 120 for shot scale GT data. GT is a term used to refer to information that is known to be true or real, determined by direct observation and measurement (i.e. empirical evidence).

The two graphs displayed on the top of figure 3.4, represent distributions of both, shot duration, on the left, and shot scale, on the right.

On the lower part of the figure 3.4, there are two chord diagrams where the left one represents the transitions between shot duration classes, and the right one represents the information related to transitions between shot scales.

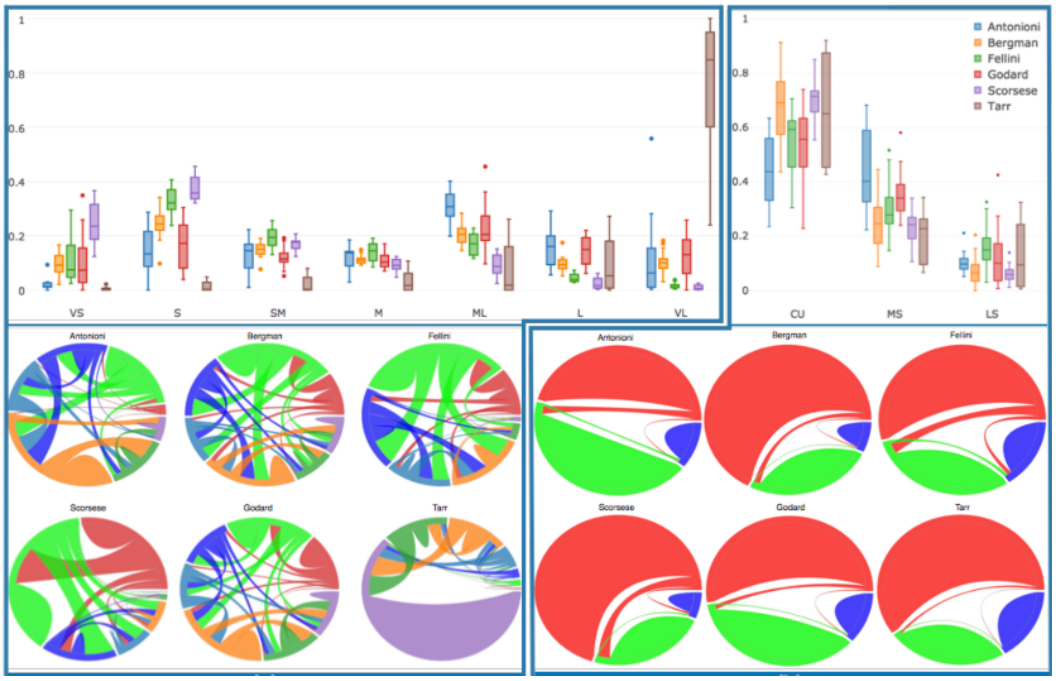


Figure 3.4: Data visualizations based on GT data from movies of six directors – Distributions (top) and chords diagrams (bottom) [64]

The inferences that stand out the most from these graphs are: that *Bèla Tarr*'s preference for very long shots (in duration) is evident, the *Michelangelo Antonioni*'s preference for medium shots (in distance) or Scorsese's low standard deviation on the use of shot scale.

Another similar article explains how Support-Vector Machines (SVMs) can classify and distinguish the variants of shot scale [65]. For example, by detecting features like faces or body shapes, it is possible to tell how far a subject is from the camera thus concluding which the type of shot it is in the shot scale. Other domains that can help in the classification are the motion, the colour intensity or the perspective, as seen in the workflow that is represented in figure 3.5.

To finish this section it is important to mention one of the most complete tools ever made in the context of "Cinemetrics", which is "VIAN" (VIsual ANnotator) [66]. This software is described as a visual annotation tool for film analysis with an interactive interface that places emphasis on visual aspects of film style and colour aesthetics. This project was developed in collaboration with an extensive film studies project that has more than 17 000 segments from around 400 movies, which were analyzed and manually labelled with different keywords.

The article of this project describes the state of the art of film visualization, and compares the "VIAN" software with other analysis tools in this field of study. The conclusion made is that "VIAN" has such a wide range of functionalities, in comparison, and features which stand out like: semantic annotation, colour feature

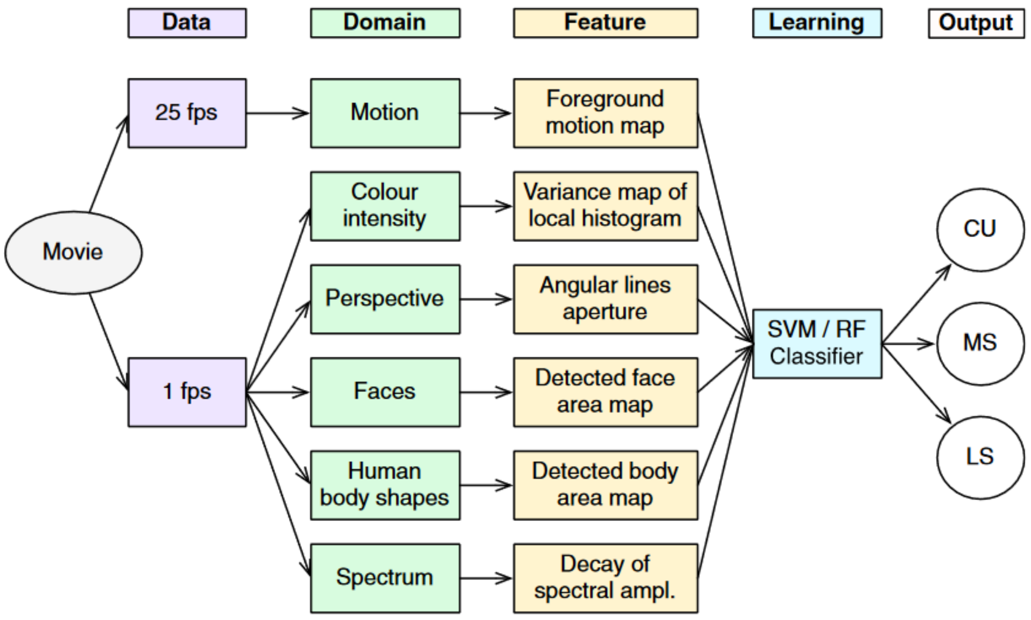


Figure 3.5: System workflow for shot scale classification [65]

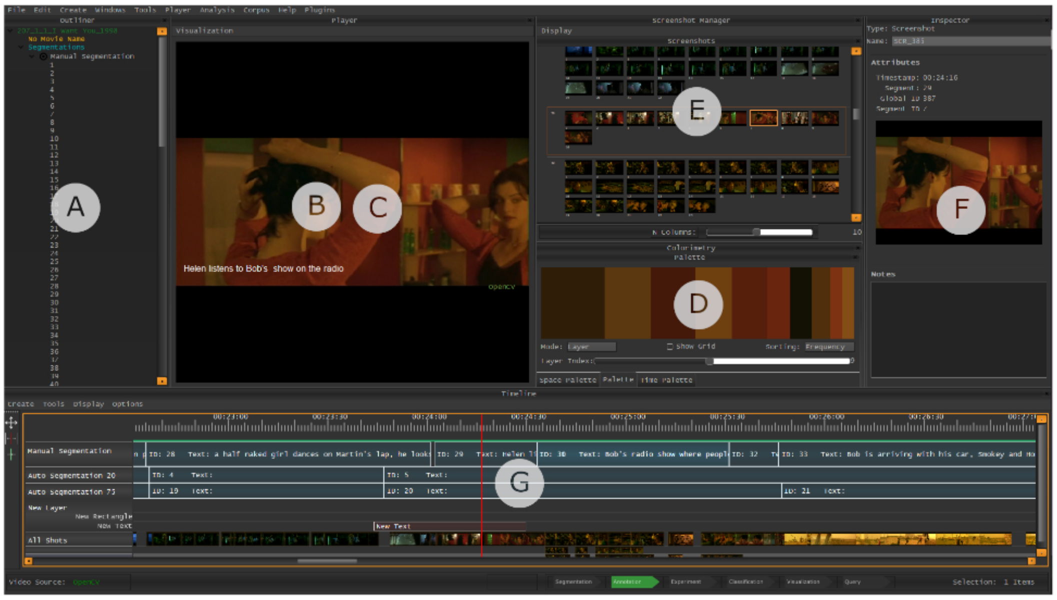


Figure 3.6: Interface of the software "VIAN" [66]

extraction, colour visualization, foreground detection and corpus visualization [66].

"VIAN" is implemented in Python with the use of the "OpenCV" for image manipulation and its Graphical User Interface (GUI) was built with the "PyQt" frame-work. It uses "VLC" as an embedded media player to display the content, and "Keras" with a "TensorFlow" backend for semantic segmentation. The data which it generates comes in the form of two type of files: "HDF5" for numeric features; and "JSON" for everything else. It uses algorithms of clustering to obtain colour palettes [66].

The figure 3.6 consists of a screenshot from the interface of "VIAN". The letters indicate different widgets such as: the outliner [A], the media player [B], the scene's description [C], the colour palette of that instant [D], the screenshot manager [E], a section that gives details about the current selected entity [F], and the timeline with tiers of temporal segments and screenshots [G].

3.2 Frederic Brodbeck's Cinemetrics Project

The project described along this thesis is based and inspired by another, more recent "Cinemetrics" project which was developed by a German student, based in Berlin, named *Frederic Brodbeck* [67]. It was developed as a bachelor graduation project in graphic design at the Royal Academy of Art (*Koninklijke Academie van Beeldende Kunsten* (KABK)) located in the Hague, Netherlands.

Frederic Brodbeck's visualizations are arguably the most insightful and aesthetically pleasing example made until today in the context of "Cinemetrics", for their circular shaped "fingerprints" and for having so many features.

Brodbeck developed scripts to automatically disassemble films and tally certain key aspects of movies. His system workflow or architecture is described in the scheme illustrated in the figure 3.7.

An impressive point is that this project is still considered prevailing and modern considering that it was developed in 2011, and until today, it hasn't been much refined and studied, at least not as much as it should have been. There is still a lack of search results regarding this topic on the Internet. In other words, there are not yet many projects trying to follow-up this one.

In the description of this project, the author points out the importance of programming in graphic design with a new and different approach, in this case, referring to generative design which is based on rules and algorithms normally developed with computer programs.

The author also stresses the fact that there already exists so much available information relating to movies, like budgets, awards and others, and for that reason he wanted to discover a new form of obtaining information from the movie itself, using its audiovisual file as source. However, his application also comprises all of

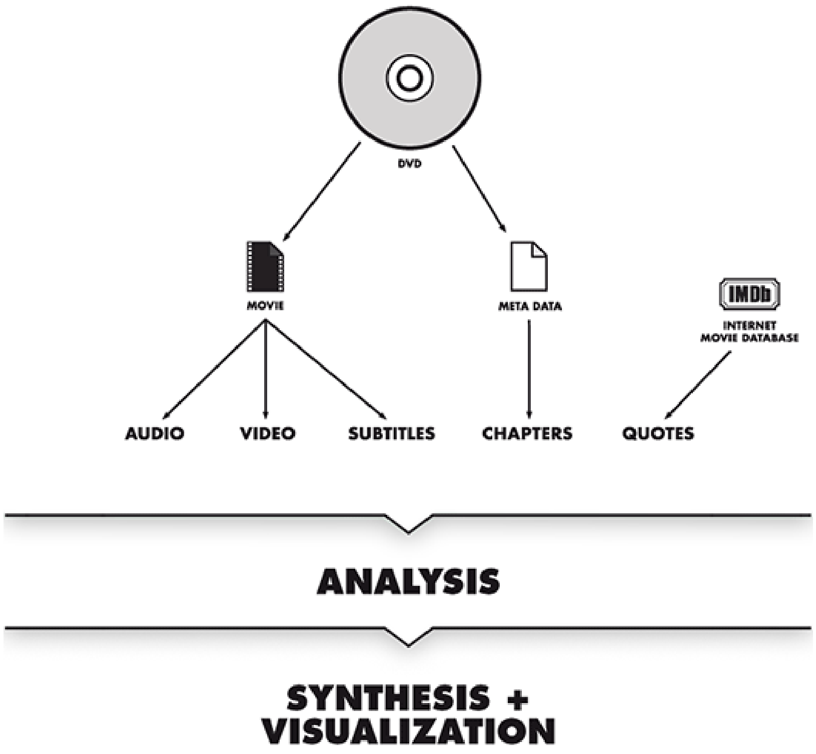


Figure 3.7: Workflow scheme of *Frederic Brodbeck*'s project [67]

this information together, because, beyond the audiovisual content, it also gathers metadata from the file like chapters, subtitles or file details, and the other types of information already mentioned that can be found in online databases and websites like "IMDb".

The biggest advantage of this application is being able to interpret and compare content from different movies which can differ in type of footage, director, genre, and possibly identify common patterns. In figure 3.8 a screenshot of the application is displayed in which the user opted to filter the "fingerprints", using the ones specific to a director and where there are visible similarities.

Another plus is that the "Cinemetrics" visualizations can give a feedback to a

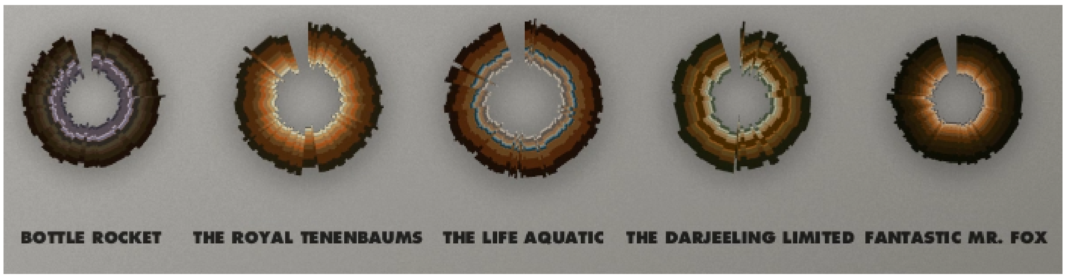
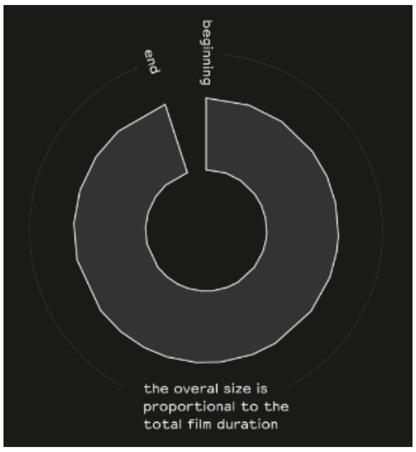


Figure 3.8: Screenshot of the tool with filtered "fingerprints" of the director Wes Anderson [67]

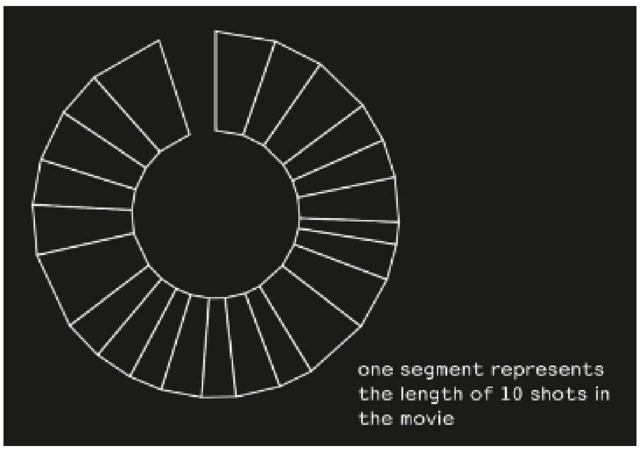
viewer who is pondering to watch a certain content, helping him decide if the movie in case interests them or if they intend to watch it integrally.

3.2.1 Tool Description

To explain how these visualizations work, *Frederic Brodbeck* made a video which can be summarized through the figure 3.9 that explains the anatomy of a "fingerprint", and the figure 3.10 which shows the types of colour palettes and the motion that is associated with the movie.

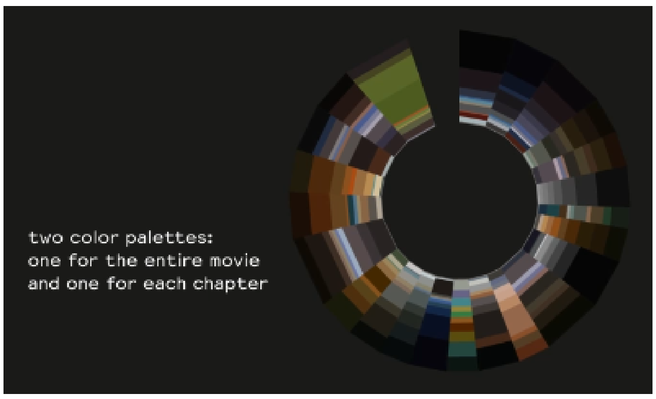


(a)

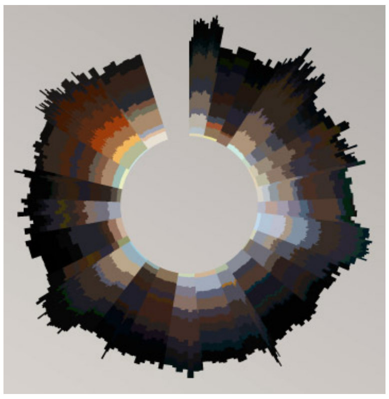


(b)

Figure 3.9: Anatomy of a movie "fingerprint" [67]



(a) Type of colour palette for each chapter



(b) "Fingerprint" of the movie "Quantum of Solace" (2008)

Figure 3.10: Both examples of a non-moving (a) and a moving (b) "fingerprint" [67]

Brodbeck chose to use segments which are sets of 10 shots, because considering that a typical movie can have around 1000 shots, if he chose to portray them all, they wouldn't look clear to the naked eye.

In the world of cinematography, the hierarchy of the units that constitute an audiovisual product is:

1. **Frame**
2. **Shot**
3. **Scene**
4. **Sequence**

For better convenience, the model used in this project to describe the structure of the different parts of a movie, is subdivided as the following ascending order:

1. **Frame**
2. **Shot**
3. **Segment** as defined by *Frederic Brodbeck*
4. **Chapter**

The colours of a "fingerprint" can be represented in two ways: with a colour palette for each segment; or with a colour palette for the whole movie (which looks cleaner).

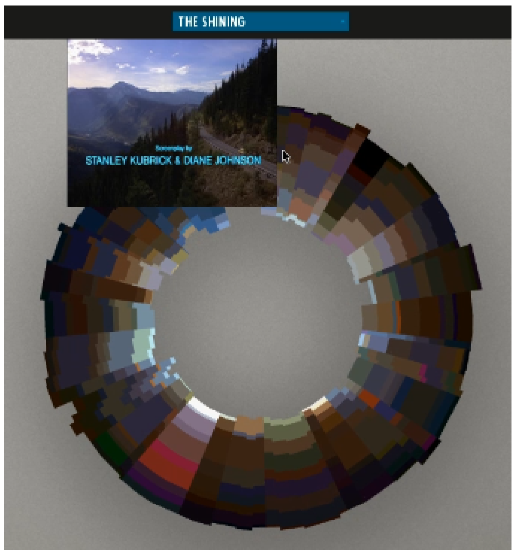
All the fingerprints have the same size so they can be easily compared, while the proportion of each segment varies to give a notion of the length of the movie.

A very interesting feature about this application is that it can show the motion of a "fingerprint". It does this by moving each segment to the inside and outside of the circle at a proportional speed or cadence depending on its value of motion, a value who calculates the movement in every shot by comparing frame to frame.

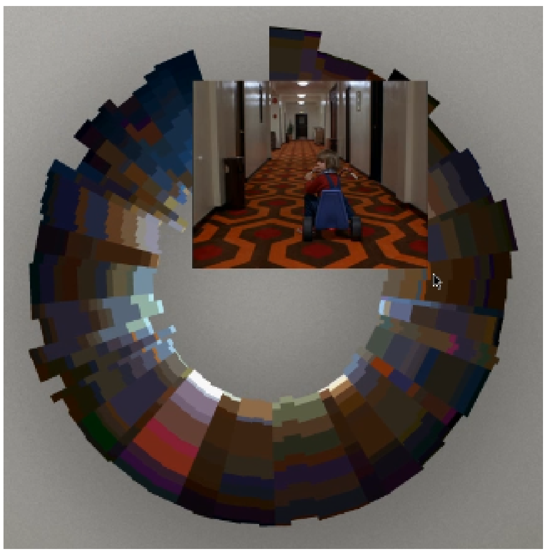
Similarly to "Selfiecity", an application already characterized in the chapter 2, this interface allows the user to see an amplified frame of each segment if the user moves the mouse through the "fingerprint". This phenomenon can be seen in the figure 3.11.

The features that this tool extracts are endless and because of this, only the most relevant for the context of this thesis, are going to be described. The shot slit-scans are another incredible feature which lay out movies as a whole. They consist of a visual and sequential representation of all the frames of a movie that are shortened and cropped in a way of fitting all of them in a poster like the one seen in the left side of figure 3.12 while the right side corresponds to an amplified section of it.

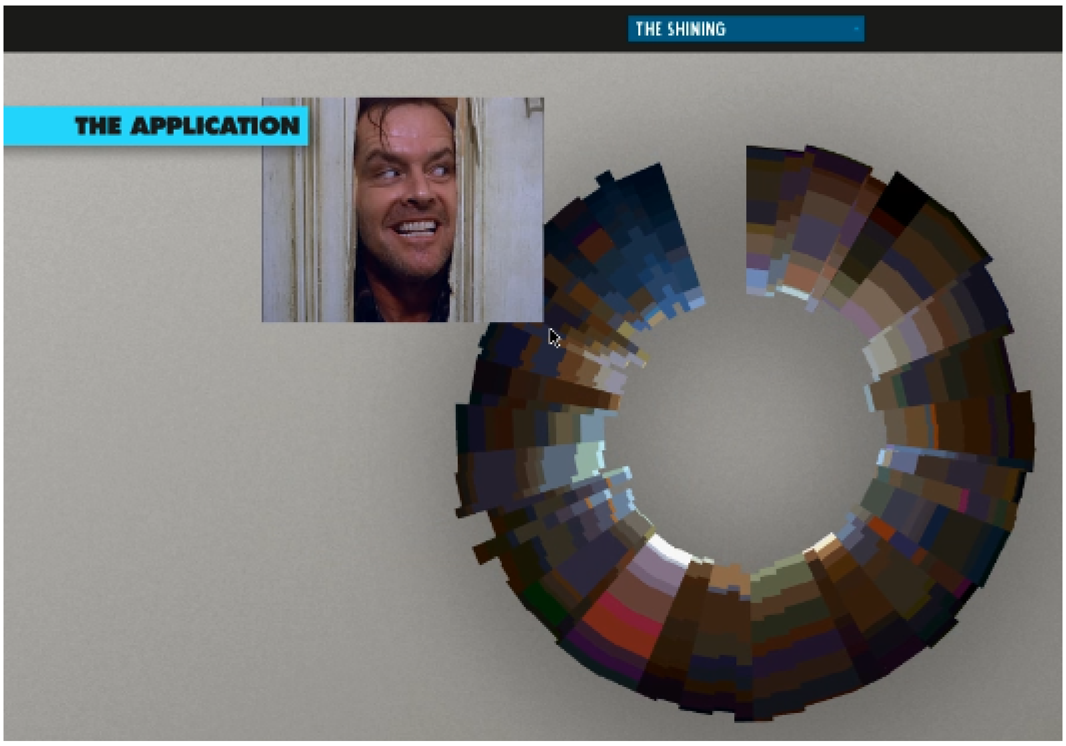
Some of the application source code has been allocated and made available in "GitHub" for the public [68]. It is constituted by many different scripts in which most of them are written in the programming language of Python with the use of the "OpenCV" library, while others are executed with the tool "FFmpeg".



(a) Frame of the movie's beginning (credits)



(b) Simple frame present in the middle of the movie



(c) Frame seen towards the end of the movie

Figure 3.11: Screenshots of frames displayed along the "fingerprint" of the movie "The Shining" (1980) [67]

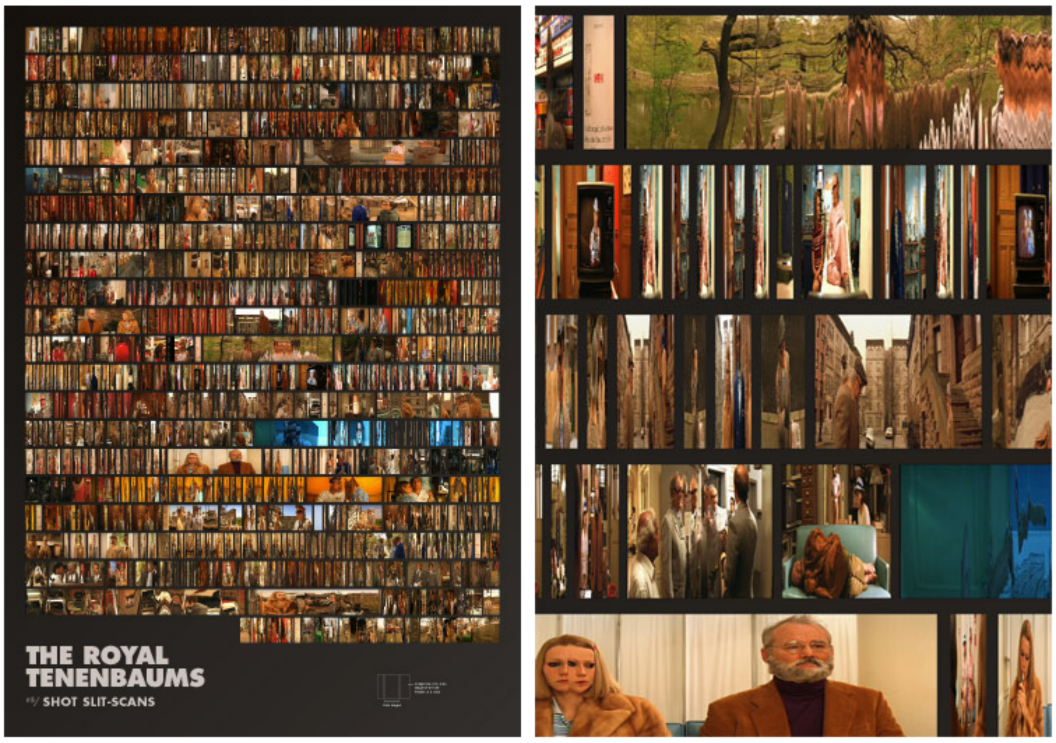


Figure 3.12: Shot slit-scans extracted from the movie "The Royal Tenenbaums" (2001) [67]

The tool "FFmpeg" is a free, open-source and cross-platform software developed in the Linux platform, that consists of a suite of programs and libraries for handling video, audio, and other multimedia files and streams. By handling, it means that it can record, convert, edit, crop, and extract content or information from these files. Alternately to "FFmpeg" there are many other tools that have similar functionalities like: "HandBrake", "FormatFactory", "VLC", and "Gstreamer".

These scripts of *Frederic Brodbeck*'s project were made for different aspects such as colour, shot boundary detection, motion, and audio. To have a notion of how varied these aspects can be, the scripts consist of a total of 22 files and the following list names some of them [68]:

- `01_1_new-project.py`
- `02_1_shot-detection.py`
- `02_2_save-shots.py`
- `02_3_shot-slitscan.py`
- `02_4_final-cut.py`
- `02_5_100-stills.py`
- `03_1_shot-colors.py`
- `03_2_shot-colors_avg.py`
- `03_3_movie-colors.py`
- `03_4_adjust-chapters.py`
- `03_5_chapter-colors.py`
- `04_1_motion.py`

- `04_2_sort_motion_spectrum.py`
- `05_1_trim-audio.py`
- `05_2_audio.py`
- `06_1_plot_statistics.py`
- `06_2_subtitle_sentiment.py`
- `lib.py`

All of these listed files are written in Python, therefore they are ".py" type of files.

There are also 4 files of the type ".bat" which stands for "batch" archives, in other words, text archives that contain command lines that can be executed sequentially. Inside these files, there are lines from "FFmpeg" which work with ".vob" files, a type of multimedia files (like MP4, MOV, WMV or AVI) which stands for "Video Object File" and can normally be found in a Digital Video Disc (DVD).

3.2.2 Other Examples of Cinemetrics Visualizations

This subsection brings up some notorious and acclaimed examples of "Cinemetrics" projects of visualizations, such as the one just mentioned.

The first and oldest example was created in 2004 by Brendan Dawes, an artist based in the UK who works in the field of data visualization. His project is named "Cinema Redux" [69], and it is a registered trademark which was acquired in 2008 for the Museum of Modern Art (MoMA) permanent collection, in New York.

"Cinema Redux" consists of unique visual distillations of entire movies, where each row is comprised of 60 frames, and it represents one minute of film time. The results are fingerprints, or grids, as seen in figure 3.13, which can be used to formulate new theories about the language of film by the people who study them, from film students to experts and professionals who work in the cinematographic industry [69].

In figure 3.14 there are four visualizations provided from a page displayed on "Tumblr" named "MOVIEBARCODE" [70]. These illustrations make it easier to compare the ambiance in which movies are developed. In the movie "The Hateful Eight" (2015), the first minutes take place in the snow while the rest of the movie happens in a single, poorly-lit cabin. All of these details are coherent with its visual representation, the movie "barcode" seen in the subfigure 3.14b.

An example of an award winning project is "The Colors of Motion". This ongoing project was created, designed and developed by a designer named Charlie Clark. It started out as a data visualization made to explore colours in movies where each "fingerprint" consists of a timeline made of lines stacked from top to bottom which portray the average colour of each frame used throughout a film [71].

In 2015, this project would end up winning a "Webby", an award dedicated to credit Internet content. So, later on, he started selling physical prints of his visualizations on "Etsy", an American e-commerce platform, because of the great demand he was having due to the increasing popularity the project was getting.

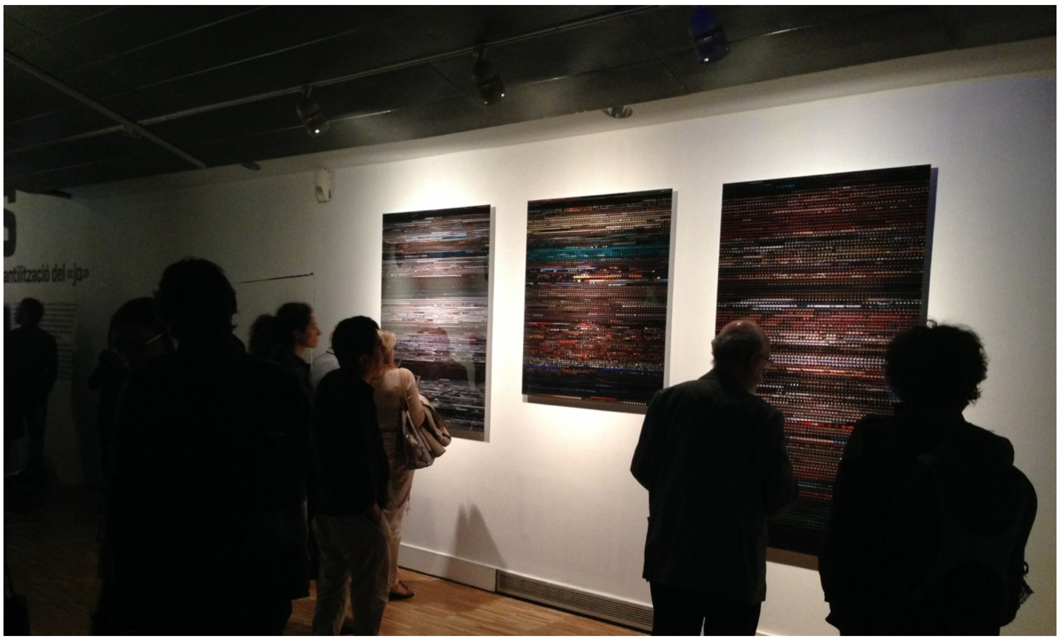


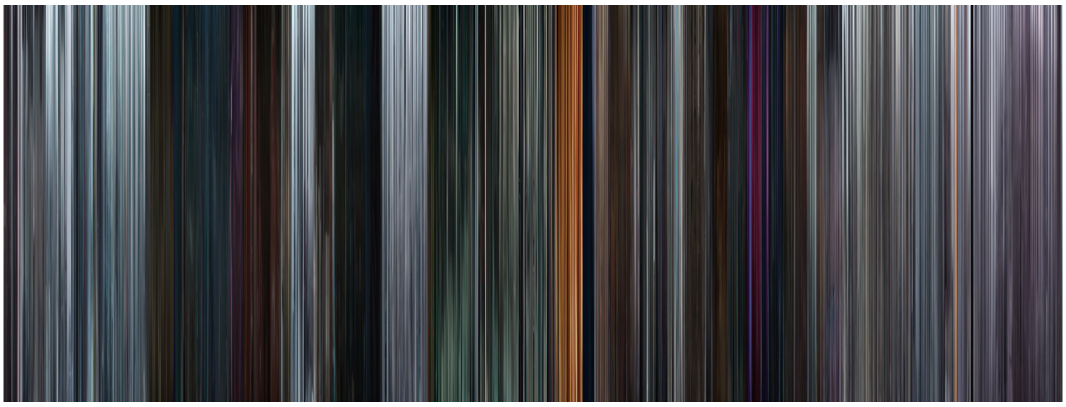
Figure 3.13: The "Cinema Redux" grids displayed in an exposition
[69]

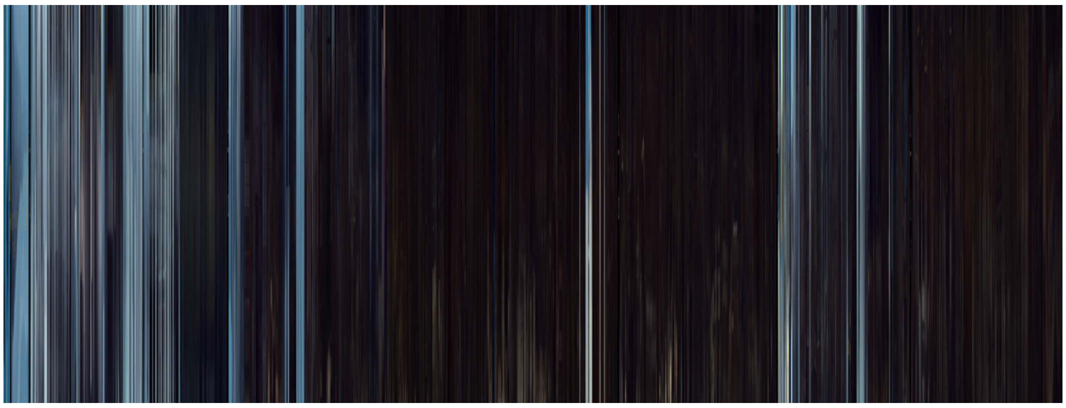
Most recently, with this colour data, he decided to start creating and selling Non-Fungible Tokens (NFTs) [71]. The figure 3.15 represents eight visualizations from the vast collection that is "The Colors of Motion".

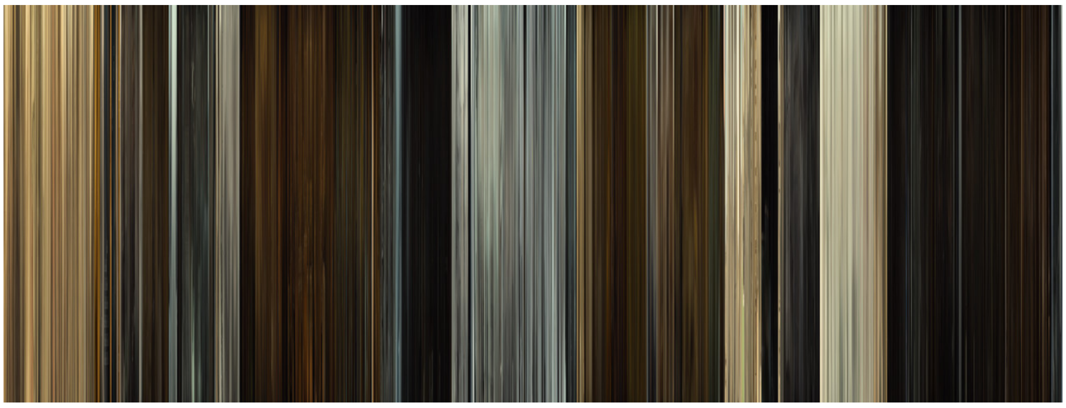
And the last, but not least, example was developed by a North American research and design studio named "OFFC", and it consists of an experiment with the goal of creating a new kind of feedback interface for the movie "The Shining" (1980) directed by Stanley Kubrick, which highlights the more emotional and impactful parts of movie content [72].

With the advances in the areas of: biofeedback, sensor technology, and algorithms, there has been an increasingly deep analysis of the human's emotional responses to visual content. Movies are traditionally made to cause these emotional responses such as surprise, delight or shock [72].

This project named "Sensing the Shining" [72], is about the construction of a poster for this film where the stills (frames) are extracted with a specific dimension proportional to the emotional reaction that each moment of the movie causes to the viewer. This concept is conveyed with the help of the Galvanic Skin Response (GSR) technology, which measures the continuous variation in the electrical characteristics of the skin. The way that it does this is by measuring the speed of an electromagnetic pulse between two sensors attached to two different fingers, which increases depending on the sweat level that is triggered by emotional reactions. This term is also referred to as Electrodermal Activity (EDA), Skin Conductance Level (SCL), and many others.


(a) "Deadpool" (2016)


(b) "The Hateful Eight" (2015)


(c) "James Bond - Spectre" (2015)

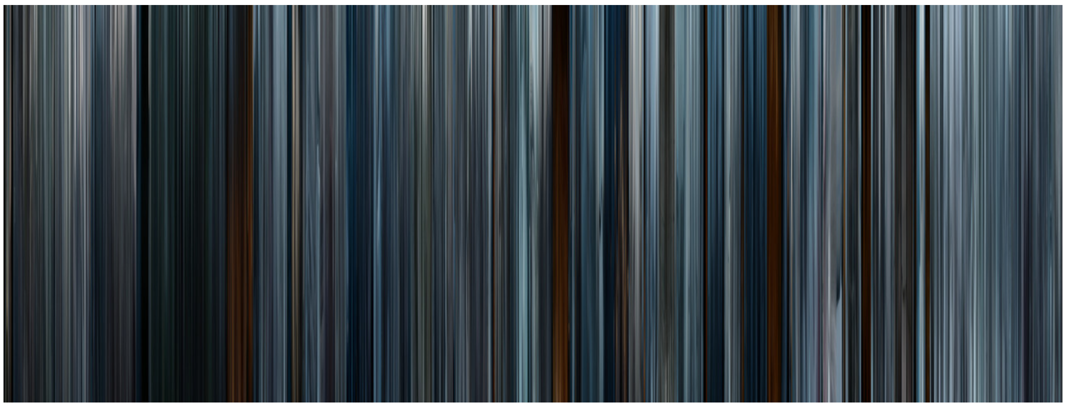
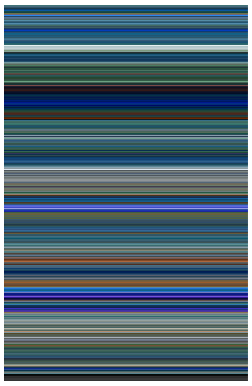
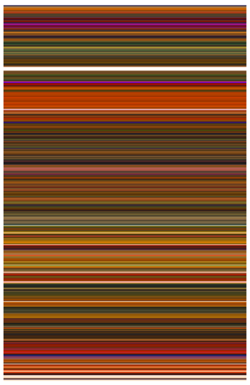
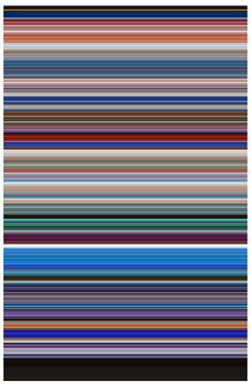
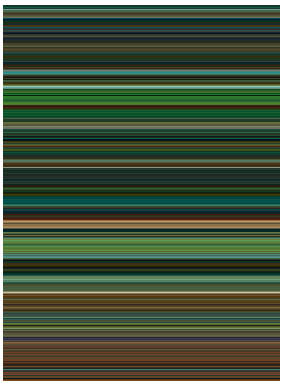

(d) "The Revenant" (2015)

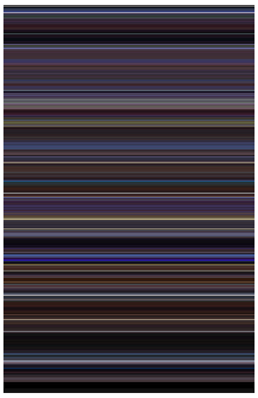
Figure 3.14: "MOVIEBARCODE" visualizations [70]

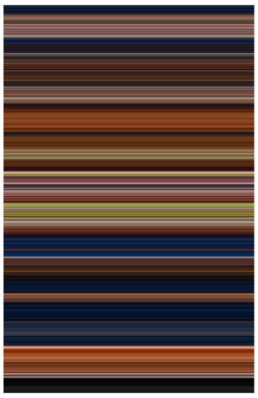

(a) Avatar (2009)

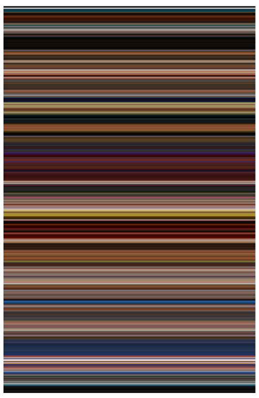

(b) Enter the Void (2009)


(c) Monsters, Inc. (2001)


(d) Joker (2019)


(e) Donnie Darko (2001)


(f) Harry Potter and the Sorcerer's Stone (2001)


(g) Big Fish (2003)

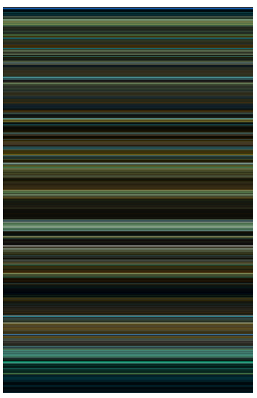

(h) Fight Club (1999)

Figure 3.15: "The Colors of Motion" visualizations [71]

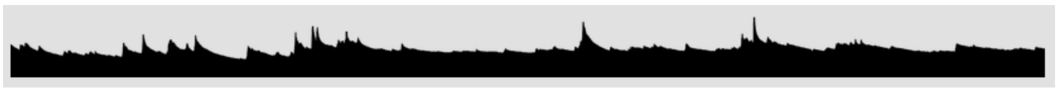


Figure 3.16: The data of the project "Sensing the Shining", as measured by the GSR sensor throughout the whole movie [72]

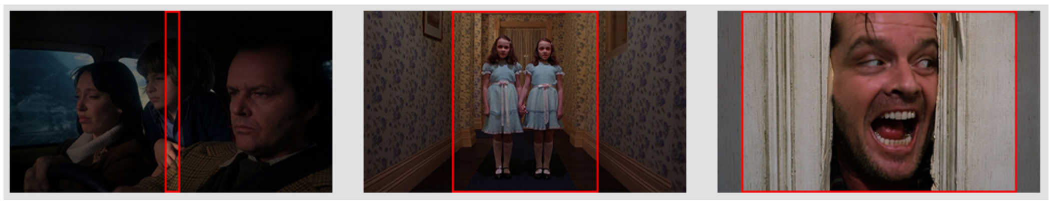


Figure 3.17: The method used in the "Sensing the Shining" project to crop the frames [72]

The figure 3.16 illustrates the raw data obtained with the GSR sensor which measures the emotional reactions and peaks of the viewer in 0.2 second intervals.

In this project, only the frames corresponding to the sensor measurement intervals are used. Each one of these frames is cropped with a specific width according to the sensor values. This is described in figure 3.17 where each type of moment has a different size: the quiet moments have the least size; the tension moments have medium size; and the sudden moments have the biggest size.

This approach is used to emphasize the most emotional and meaningful scenes of the movie, while the less important parts of the movie become almost visual noise.

The cropped frames are then stacked together in sequence to create the movie poster collage. A portion with a sequence of intense moments of this collage, is displayed in figure 3.18.

This type of visualization is identical to the concept of "Shot Slit-Scans" from *Frederic Brodbeck*'s project.

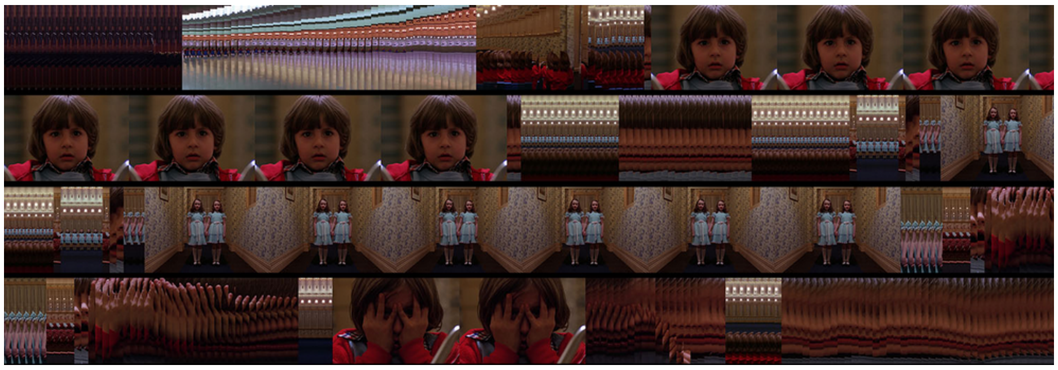


Figure 3.18: An extract of the "Sensing the Shining" collage [72]

Chapter 4

Code Implementation

The main approach of this tool's development was to, step by step, gather many different features into a single script based on the shot detection file from *Frederic Brodbeck*'s project. This chapter will describe the whole progress of the practical component of this project, where the base script went from less than 200 lines of code to over 500.

One thing that was very helpful to test and learn about different functions of Python and "OpenCV" was the "Jupyter Notebook", which is a document in the "JSON" format, containing an ordered list of input and output cells used to experiment pieces of code.

The first problem encountered was that the "OpenCV" version was outdated. One of the most draining and longstanding processes of this project was to adapt the "OpenCV" library from the old version named "cv" to the new improved version named "cv2".

The first method used in this tool to detect cuts, consists of finding the pixels that changed significantly from one frame to the next one. The way that it does this is by creating a image which is the absolute difference of the current frame (in the HSV model) and the previous one. And then, with the threshold function of the "OpenCV" library in the binary mode, it creates another image where all the pixels are either 0 and 1. The pixels of value 1 are the pixels which the difference was bigger than 10, the value chosen for the threshold. After that, a calculation is made to count the proportion (percentage) of non-zero pixels compared to the total number of pixels of the image. This value obtained corresponds to the difference of colour.

The second method calculates the alterations in the histograms of these frames. To do this, the "OpenCV" library has some dedicated functions that help in this process, which are:

- cv.CreateHist()
- cv.CalcHist()
- cv.NormalizeHist()
- cv.CompareHist()

All these functions would be later adapted to the "cv2" version.

A histogram is a graph that represents the distribution of intensity of a picture or frame [73, 74]. For example, in a gray scale it is possible to show the number of pixels that exist for each value from 0 to 255 (in the horizontal axis). By interpreting this type of histogram, it is possible to study many factors like the brightness or the contrast of an image.

After getting the value of difference of the histograms, both methods are combined into other value to make a decision with another threshold. This value is calculated with 40 percent of the difference of colour and 60 percent of the difference of the histograms.

The condition that triggers a cut detection is when this value is equal or bigger than the threshold, which is 0.48 for coloured movies, and 0.40 for black and white movies. These values would not be changed throughout the improvement and development of the tool.

To detect and save the instant of a cut, the condition just mentioned will trigger a set of tasks that will annotate the information of the previous shot and of the time of that cut.

One of the most interesting points about the shot detection script is that it has a variable called `"DEBUG"` which works like a flag, in a way that if its value is "True", the tool will display the video (movie) with the function `"cv.ShowImage()"` (which was later adapted to `"cv2.imshow()"`), while it is being processed.

Besides that variable, there is another called `"DEBUG_INTERACTIVE"` which if it is "True", the application will interact with the user every time it detects a cut, asking them to confirm if that instant actually corresponds to a cut. This interaction is made through two different libraries: "win32api" and "win32con".

The first one generates a message box with a typical interface of the Windows operating system, while the second provides two buttons with the options `"YES"` and `"NO"`, and a condition in which the integer value of 6 corresponds to `"YES"`.

The general way that this script works is with the code lines:

- `"cv.CreateFileCapture()"`, the function that takes the multimedia file as an argument (adapted to `"cv2.VideoCapture()"`);
- `"cv.QueryFrame("` which queries all the frames of the file from the first to the last. In the adaptation, the function `"cap.read()"` is used, in which `"cap"` is the variable associated with the first function.

And the "while" cycle, from this point on, would also break with the condition if the code reaches the last frame, besides already breaking by pressing the "Esc" key.

The script gives the number of frames and the frame rate with the use of pre-defined functions from "OpenCV". And with the library "time", there is a function named `"time.time()"` which stores the value of the instant in which the tool starts running. At the end of the script, that function is called again in order to subtract both values and obtain the time in which the tool was processing the video.

If the user passes the string "bw" as a third argument, considering that the file itself is the first (argument number 0), when executing the script in the command prompt, the tool will change to its black and white mode. This only happens to adapt the threshold to another value, which is 0.4, while the normal threshold is 0.48, and to make the script use the channel "value", instead of the channel "hue", to compare the difference in every pair of consecutive frames.

In the original file, every frame of the movie is reduced to a fourth of its size. While the adapted version only reduces to half because even though it makes sense to reduce it, so then, every time the tool extracts visual outputs they can be smaller in file size, the point here is to also display the movie to the user while they interacts with the tool.

The default colour model used in "OpenCV" is the BGR, so the image is always converted to HSV to make it easier to compare channels and histograms.

Before the full description of the best and last version of this tool, it is important to explain which advances were made halfway through the development of it. For this purpose, this midway version will be named as version 1.0, while the final product will be the version 2.0.

4.1 Version 1.0

The version 1.0 is the first version and adaptation of the shot detection script from *Frederic Brodbeck*'s project, which is fully transcribed from "cv" to "cv2" and has some alterations of Python functions which also changed over time.

4.1.1 Interactive and Automatic Modes

The "sys" module is imported to read the arguments that the user types while executing the file. This feature was retouched in order to create two main modes of video processing: the interactive mode and the automatic mode.

The interactive mode corresponds to the variable `"DEBUG"` being "True", which was aggregated to the `"DEBUG_INTERACTIVE"`. This means that, from this point on, this mode would both display the movie and allow the user to interact with it. The problem with this mode is that, even though it is more efficient for having human intervention in the process of decision, it can also be very time-consuming and tiresome considering that it can last twice the duration of the movie (as an approximate estimate).

To compensate for this fact, there is the automatic mode which can run the script and process the video in the background, while the user can keep doing other things on their computer.

To generate the automatic mode the idea is to take and duplicate the extract of code inside the “if” cycle of the message box function of the “win32api” library which is triggered by the condition of the “win32con” library in case the user presses the button `"YES"`. The solutions for this would be just to copy the extract and paste it on another condition or to create a function for it which would have to receive a lot of parameters like local variables which doesn’t seem to be the best option. And the first solution as easier as it may sound it will overload the script with repeated code lines and make it visually confusing.

So, the most efficient way to solve this problem is just to include another condition next to the one that already exists and unite them as another condition with a logic operator `"OR"`. As simpler as this idea looks, the first problem that was encountered with it is that it shouldn’t generate the window of the message box if it is in the automatic mode. And to solve that there is what is called “short-circuiting” in programming terms. This concept explains that if in the predicate `"A OR B"`, `"B"` will be evaluated only of `"A"` is false. Likewise, in the predicate `"A AND B"`, `"B"` will be evaluated only of `"A"` is true. So, if in the condition, the expression that checks the status of the variable `"DEBUG"` goes first, the script will not evaluate the second expression and therefore will not generate the message box window.

To choose a specific mode, the user has to write it in the command line and pass it as the third argument (`"sys.argv[2]"`). If the user opts to not pass any argument for the mode, the tool will run its script in interactive mode and with the black and white mode turned off, by default.

The problem with this is that the user wouldn’t be able to run the tool in the automatic mode for a gray scale movie (black and white mode). This would be later corrected in version 2.0, where the black and white mode would move to the fourth argument (`"sys.argv[3]"`).

In the interactive mode, since some cuts can be wrongly detected, the variable of the number of shots only increments when the user confirms the existence of a cut, while there is another variable (`"nshots_det"`) that increments every time a cut is detected by the script. This is to compare both values and see how efficient the tool can be for each case.

4.1.2 Timecode

In this script the duration of the movie in seconds, is calculated with the following equation:

$$DURATION = FRAMES * FPS \tag{4.1}$$

So later, a condition was implemented which would reveal if the movie was a short film or a feature-length film. A short film is defined as a movie with less than 30 minutes, while a feature film is a main movie with a regular and "full" length which can be around one to three hours long.

The variable `"length"` corresponds to the total number of frames the movie has. So, since this project is related to cinematography, it is rather convenient that it uses a timecode to represent any time variable.

A timecode is a sequence of numeric codes generated at regular intervals by a timing synchronization system. It is used in video production, broadcasting, and in other areas which require temporal coordination or logging of time stamps. Timecode is represented as in the following model:

$$HOURS : MINUTES : SECONDS : FRAMES \tag{4.2}$$

Each one of these four sections is made of two integer digits. That means that even if a number is lower than 10, it has to have a zero to its left. The reason behind this is that the frame rate value is never over 100 (the most common values being 24, 30, and sometimes 60 FPS), while every other value is a multiple of 60 of the precedent unit. And the value of hours never reaches a third digit, because even the longest movies are not remotely close to the 100 hours mark of duration.

To convert the variable `"length"` into timecode, the script repeats this fragment of code many times for different purposes:

```
hrs = int(length/(3600*fps))
mins = int(length/(60*fps))%60
secs = int(length/fps)%60
frs = length%fps
```

Listing 4.1: Code lines that convert the total number of frames into each unit of a timecode stamp

4.1.3 Cadence Variables

Also in the interactive mode, there is the variable `"play_cadence"`, which as the name implies, was created to define the playback speed of the interactive mode. What was quickly understood is that this speed cannot be changed like this, because even though it is possible to choose the frequency in which the frames of the movie are displayed, this will not affect the time that the script takes to go through every frame. Because of all the conditions, processes and code lines that the `"while cap.isOpened()"` cycle has, the processing time of the videos is always longer than their duration.

All of this was concluded by processing a video clip of 21 seconds (a scene with proper cuts from a episode of a TV series) with different value for the variable `"play_cadence"`, as a test. This test had the purpose to see if different values for this variable would affect the processing time, and by looking at the table 4.1 it is easy to tell that they didn't. This is because the standard deviation of the values is 3 seconds, a low value which corresponds to the human error of reacting to each cut and having to choose and click on the buttons (`"YES"` or `"NO"` options) of the message box.

This variable was never deleted, but changed permanently to the value 1 so every frame could be displayed to the user for the best viewing quality, while using the interactive mode.

Table 4.1: A table which compares different play speeds to their processing times (in seconds)

Variable `"play_cadence"` (frames)	1	5	10	20	30	60
Processing time of the tool (in seconds)	80	83	80	81	82	82

Another similar variable is the `"print_cadence"` which defines the rhythm at which the prints are going to be obtained. This variable is assigned to the value that is returned by the function and calculation: `"math.ceil(length/1000)"`, so that each increment of the prints are done in order to have images of roughly 1000 pixels of width as the final products. The function `"math.ceil()"` makes a round up,

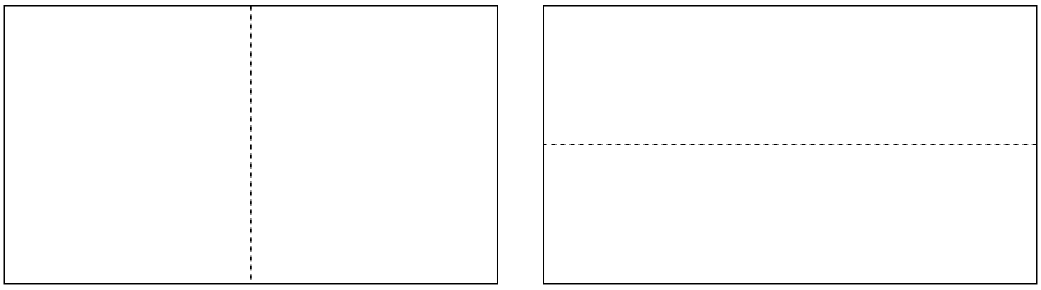


Figure 4.1: Descriptive shapes displaying how the vertical and horizontal prints are extracted

and this is to have an integer value above zero (1) in case the value of the variable "`length`" is under 1000.

These prints that were mentioned are two graphical representations of the integrity of a movie in which one is extracted vertically and the other horizontally, as it is described in figure 4.1.

The listing 4.2 shows the extract of code of the first iteration that create the prints. Firstly, the images are cropped to obtain the vertical and horizontal central lines. Then, since the final product is suppose to be a rectangle with more width than height, the horizontal line needs to be reduced to half and rotated 90 degrees to be vertical.

```
vmid = int(width/2)
hmid = int(height/2)
vert_print = img[0:height, vmid:vmid+1]
horiz_print = img[hmid:hmid+1, 0:width]
half_redu = (vmid, horiz_print.shape[0])
horiz_print = cv2.resize(horiz_print, half_redu, interpolation =
    cv2.INTER_AREA)
horiz_print = np.rot90(horiz_print, 3)
```

Listing 4.2: Code lines that represent the first increment of generating the prints

From X to X frames, being X the value of the variable "`print_cadence`", the prints are incremented with the function "`np.hstack()`" which stacks horizontally the product from that frame to the total product put together until that point.

4.1.4 Numpy Array and Matrices

This script also calculates the amount of motion in each shot. For that there is the variable "`motion_sum`" which annotates the difference between each pair of consecutive frames (one of the methods that is used for shot detection) in percentage, and sums them (increments) so that in the end of that shot, it calculates an average by dividing it for difference of number of frames between the initial and last cut of that shot.

Before the scripts starts processing the frames, there are three "numpy" arrays created:

- An information array of size 11, in which the first six values correspond respectively to: the length (number of frames); each unit of the duration of the movie in timecode (hours; minutes; seconds; frames); and the frame rate (in frames per second). The last 5 values are filled in the end of the script with: the number of real shots; the number of detected shots (which will be the same if the mode is automatic); the number of segments (in case the movie is longer than 30 minutes, if it is a short film, this value will be 0); the number of "fades to black"; and the processing time in seconds.
- An array of size 8 filled with zeros, for the shots table, which will later turn into a matrix. The values correspond to: the number of initial frame of each shot; each separated unit of the instant of that cut in timecode (hours; minutes; seconds; frames); the duration of that shot in timecode, but only with the seconds and frames; and the average percentage value of motion. The reason why this array becomes a matrix is that when each shot, or segment (for feature-length films to not make the table too extensive) is detected, the script will fill the duration and motion of the previous shot (the last three positions of the array), and it will create a new row (array of the same size) with the first five values and with three zeros in the end.
- An array similar to the one of the shots table (the previous one), but for the "fades to black" and of size 7, because it has one less value (the one of motion).

At the end of the video, the tool will generate a file, with the ".xlsx" file extension, where these three "numpy" variables will be transcribed to three pages: the shots table, the information values, and lastly, the table of "fades to black". The way this is done is with the use of the "pandas" library, which converts the "numpy" variables to data frame format, and later writes the data frames into a file. This file can be opened with "Microsoft Excel" and it can be passed onto any database, hence the fact that the values of the timecodes are separated, so they can be more conveniently handled.

4.1.5 Slit Sequences

Another outputs that this tool generates are the slit sequences, which correlate to the "Shot Slit-Scans" of the *Frederic Brodbeck*'s project. Similarly to the prints, this

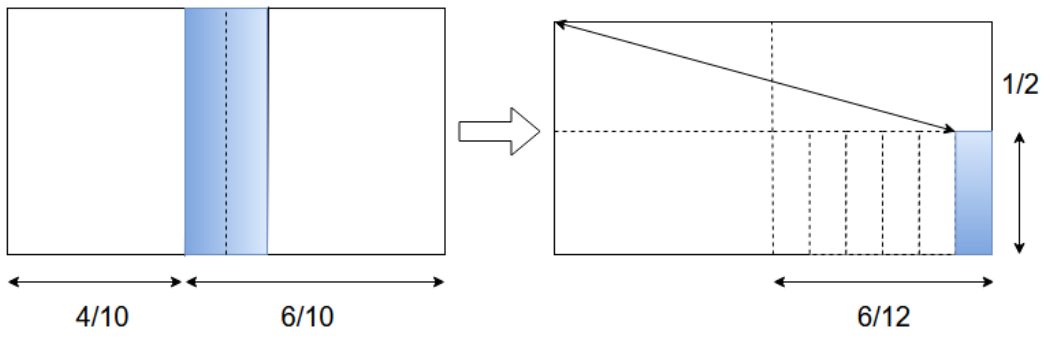


Figure 4.2: Cropping and resizing of the frame to increment a slit sequence

is an output in which consecutive portions of the frames are stacked horizontally. The extraction process can be seen in figure 4.2. Firstly, the frame is cropped with the extraction of a slit, more precisely the vertical center section which corresponds to one fifth of the frame. And secondly, that slit is resized with the "cv2" function: `"resize(image, narrow_factor, interpolation)"`. This variable called `"narrow_factor"` is passed as an argument of the function that indicates the desired size of the resized image. These dimensions are: the width divided by 12, and half of the height, both from the original size of the frame.

Like with the other variables of cadence, this extraction is made from 30 to 30 frames. In the other cases, the condition is when the rest of the division of the number of the frame by that variable, equals 0. In this case the rest of the division is by 30 and when it equals 1 to make sure the first frame goes through this process.

When a slit sequence reaches the width of 1000 pixels, that sequence is saved in the outputs' folder, the number of slit sequences increments, and a new one begins. This is done so the users can take closer look to each part of the movie, but all the slit sequences can be merged in a singe picture that shows the movie as a whole.

4.1.6 Colour Clustering

Colour clustering, which is a method already described in this thesis, has the objective of extracting the more dominant colours from an image.

To generate a colour palette for a movie processed with this tool, the script extracts the dominant colours from each shot (or segment) with a library of Python and "OpenCV" named "K-Means", which consists of a clustering algorithm.

In general, what the library "K-Means" does is that it yields a smaller sample of clusters from the original sample of data points. The mean of each cluster is named "centroid" or "center". The data points are separated and grouped inside the particular cluster that has the nearest mean [75].

In these example, since this clustering will apply to images, the data points are the pixels of RGB images. Because of this, the pixels that belong to a given cluster will be more similar in colour than pixels belonging to a separate cluster. For this tool, the number of clusters is defined as 5, by default. This corresponds to the number of colours that will be extracted.

"K-Means" is a part of the "scikit-learn" library, and it before any of its functions are called, there are two functions that need to be defined.

The function "`centroid_histogram()`" is sourced from an article [75], and it returns a histogram based on the number of pixels assigned to each cluster that is created by grabbing the number of different clusters. The histogram is normalized so that it sums to one.

```
def centroid_histogram(clt):
  numLabels = np.arange(0, len(np.unique(clt.labels_)) + 1)
  (histk, _) = np.histogram(clt.labels_, bins = numLabels)
  histk = histk.astype("float")
  histk /= histk.sum()
  return histk
```

Listing 4.3: Definition of the first function [75]

The second function "`plot_colors()`", contrary to the first one, it has some adaptations in order to plot the bar vertically, instead of horizontally, to later stack the other bars which represent all the other shots/ segments.

What it does is that it initializes a bar chart representing the relative frequency of each of the colours. Then it loops over the relative percentage of each cluster and the colour of each cluster to plot them. In the end, it returns the bar chart.

```
def plot_colors(histk, centroids, barszx):
  bar = np.zeros((200, barszx, 3), dtype = "uint8")
  startY = 0
  for (percent, color) in zip(histk, centroids):
    endY = startY + (percent * 200)
    cv2.rectangle(bar, (0, int(startY)), (barszx, int(endY)),
        color.astype("uint8").tolist(), -1)
    startY = endY
  return bar
```

Listing 4.4: Definition of the second function [75]

When a shot, or segment, are detected, the tool will run the extract of code indicated in the listing 4.5 to stitch together the colour bars from each shot/ segment to form the colour palette of the movie in case.

The first step is to reshape the image to be a list of pixels, then cluster the pixel intensities, represent the number of pixels labeled to each colour, and lastly to stack horizontally the colour bar extracted to the total of bars that were merged until that point.

In this code extract, the "K-Means" clustering object is created, consequently called as the only parameter for the "`centroid_histogram()`" function, and the returned histogram is called by the function "`plot_colors()`", alongside the clusters' "centroids" and the bar size correspondent to the proportion of that shot compared to the length of the movie, which is calculated by a rule of three in order to obtain a final colour palette of roughly 1000 pixels of width.

```
imgr = prev_shot_img.reshape((prev_shot_img.shape[0] *
    prev_shot_img.shape[1], 3))
clt = KMeans(n_clusters = NCLUSTERS)
clt.fit(imgr)
histk = centroid_histogram(clt)
barsize = round(frame_diff*1000/length)
bar = plot_colors(histk, clt.cluster_centers_, barsize)
# adding/stacking all the bars
barf = np.hstack((barf, bar))
```

Listing 4.5: Extract of code that stitches the colour bars from each shot/ segment

4.2 Version 2.0

In some experiments with the version 1.0, there was one problem in which right at the end of processing a movie, the script would break without generating the final outputs. This happened due to an error related to the function "`cv2.resize()`", in which there would be no frame to process.

The way to fix this problem was to change the approach to choose a better End-Of-File (EOF), in order to properly terminate the "while" cycle initiated by the function "`cv2.VideoCapture()`". In computing, the EOF is the condition where no more data can be read from a data source, in this case from the movie file.

Until this point, the cycle was defined to break with a condition that checks if the variable "`frame_counter`", which indicates the number of the current frame and increments after each repetition of the cycle, is equal to the variable "`length`", which corresponds to the total number of frames that the movie has. The problem here is that the variable "`length`" is obtained through the invocation of an attribute from the "OpenCV" video capture, named "`cv2.CAP_PROP_FRAME_COUNT`", and this attribute extraction is not totally reliable, because it gives an approximate estimate.

Therefore, to reassure that the cycle breaks before reaching the last frame, there was created a variable called "`credits`" that, as the name says, is a calculated estimate time of credits (in the unit of frames) depending on the type of movie:

1. **Short film** – 30 seconds:

$$CREDITS = 5 * 60 * fps \tag{4.3}$$

2. **Feature film** – 5 minutes:

$$CREDITS = 30 * fps \tag{4.4}$$

In the condition of the EOF, this variable `"credits"` is added to the variable `"frame_counter"` in order to break the cycle sooner.

This addition has also the advantage of removing residual frames that might adulterate or tamper with the results like the number of cuts detected, the average shot length, the visualizations, or even the processing time.

In this version, a string variable is created to start using the timecode all together in its original format (`"HH:MM:SS:FF"`), instead of separating each unit.

One of its uses is in the creation of a new type of "Microsoft Excel" file, a CSV file which will also have a list the initial instants of every shot, but with the default timecode format because it is compatible with most popular video tools. It might be another viable option to directly copy and paste the shots list into another tool or database.

The information array from the version 1.0 has changed to a size of 14 instead of 11. The first position of the array (`"info[0]"`) now corresponds to the variable `"movie code"`, which will later be explained. The other two informative values added are the ASL in seconds, and the number of cuts per minute.

One thing that was concluded is that, in the version 1.0, a feature-length film (with the most common or average length) would have around 500 slit sequences which is too much. To improve that, the cadence of frames to extract the slits was changed to 300 to 300, instead of 30 to 30. This implies that in the improved version, a movie would now generate around 50 slit sequences which is the perfect sample size considering that later the user can make a collage with all the sequences of a dimension of 5 by 10. This doesn't mean that every movie will have a collage of these dimensions, because the number of slit sequences is proportional to the length of the movie.

4.2.1 Tkinter GUI

The main difference between the two versions is that in the version 2.0, a GUI was implemented with the "Tkinter" package, the standard Python interface to the Tool Command Language (Tcl) / Toolkit (Tk).

The widgets that stand out from the figure 4.4, which represents the GUI, are: the progress bar; the two checkboxes; the dropdown menu; and the skip button.

The two checkboxes are so distinct from each other, because the first one toggles between the colours red and green, and it asks the user to confirm the presence of the cut.

The function that creates the checkbutton has a parameter named "indicatoron" that is responsible for creating the check mark (indicator) of the box, and it is "True" by default. If this variable is turned to "False", the check mark disappears, and it is possible to personalize the box in any way by changing its size and choosing the colours for each boolean value [76].

The second checkbox is the one that allows the user to select an initial frame of a shot or segment, that they want to save and highlight. This checkbox is always deselected after each frame cycle, for the user to not select frames by mistake since the idea is to have, in the end, only a few that stand out.

The skip button is a regular button that is pressed when the user wants to proceed with the processing tool after choosing (or not) the options each widget. When a cut is detected, the script stops and, with the use of the "Tkinter" function `"wait_variable()"`, it waits for this button to be pressed.

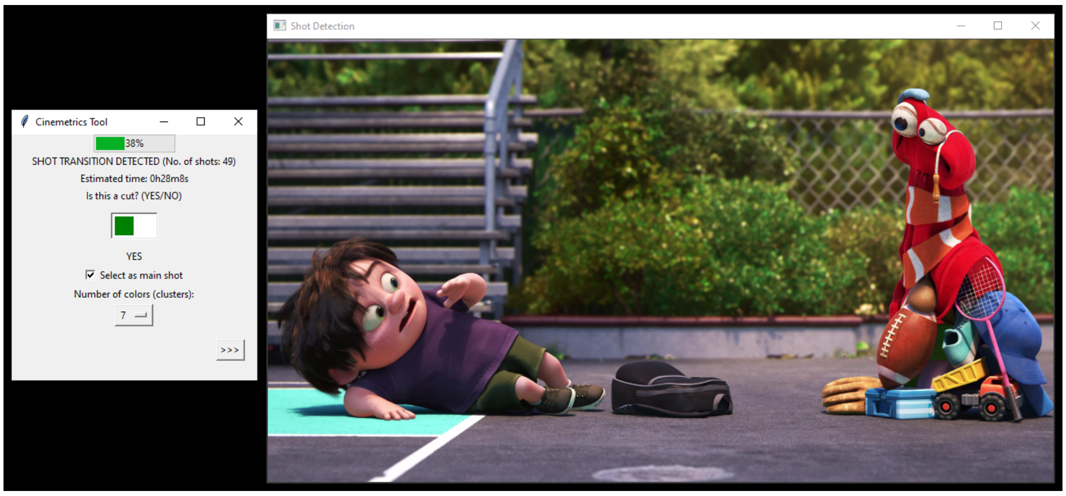
Some tests were made with the same video clip used in the experiments from table 4.1, to see if the presence of GUI would slow down the process of the tool in the automatic mode. The reason behind this is that the GUI is nonessential for this mode, but it would useful to at least give a feedback to the user about the progression of the process.

So even though, there are ways of showing a percentage bar in the command prompt, the first test was done by running the automatic mode only with the progression bar in the GUI. Surprisingly, the processing times of the tool with and without this GUI were the same. What was no surprise, on the other hand, is that the processing times had a value of 69 seconds which is significantly lower than the average value from the table 4.1, which was 81,(3) seconds, since this experiments were made in the interactive mode.

The way that the percentage from the progress bar and the estimated time are calculated is very similar, because it is done by a rule of three where the current frame and total number of frames are compared. But the only difference between the two is that the estimated time can only be calculated after a few moments so there can be a value of time used for reference, and in the progression bar, the value only changes in the GUI if the integer value of the percentage has gone up.

This means that the estimated time has to be calculated and updated in every repetition of the "while" cycle. Therefore, it is going to be displayed differently through every frame.

Because of this, the second test was done comparing the speed of the tool with and without the estimated time updates, and the values were also of 69 seconds.



38%

SHOT TRANSITION DETECTED (No. of shots: 49)

Estimated time: 0h28m8s

Is this a cut? (YES/NO)

YES

☑ Select as main shot

Number of colors (clusters):

7

Figure 4.3: A screenshot from the "Cinemetrics" tool showing an instant of a movie (right side) and the "Tkinter" GUI (left side)

A fact that can't be ignored is that the processing time also varies depending on the computer used and on the power of its Central Processing Unit (CPU).

4.2.2 Main Shots Feature

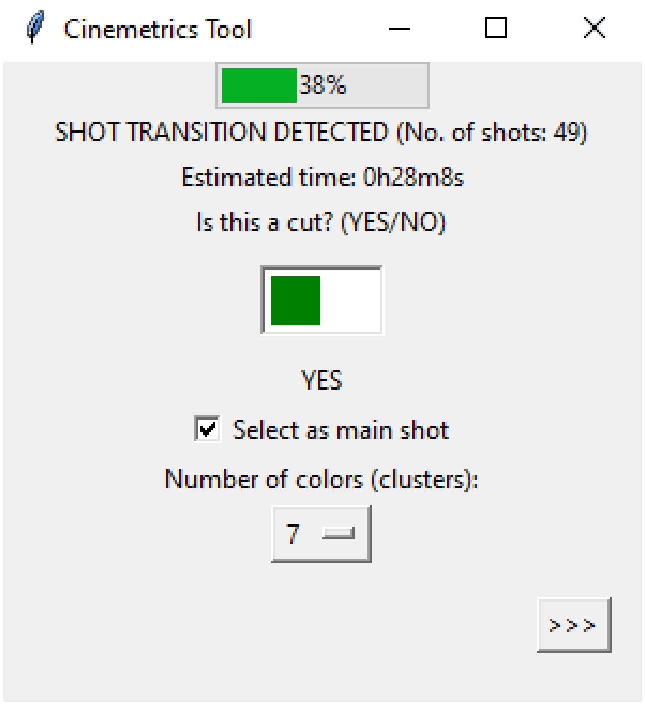
The main reason for developing a GUI for this tool was to be able to generate main shots. The purpose of this concept is to highlight frames of the movie that are seen as important moments, famous scenes, shots, or colourful and aesthetically pleasing pictures. Or just something that catches the interest of the user.

These main shots consist of frames that represent the beginning of shots, and that are merged with a vertical colour bar on its right side which portrays the most predominant colours. They also have a timecode stamp that indicates the instant of the movie where they occur. This stamp is located on the bottom right corner of the picture and it is either black or white to be more visible by creating a contrast depending if the pixels of that area are darker or lighter.

The interface allows the user to choose the number of colours they want to extract from the frame. The idea of this is to have the leverage that the human interaction has in visually picking most dominant colours in a picture in a better way.

For example, in figure 4.5, it is very clear for a human to tell that the most dominant colours present in this image are blue, black, grey, yellow, the white from the background, and maybe the colour of the skin even if it is just a small fraction of the picture. With this considered, total of colours should be 6.

To the figure 4.5, it was applied the "K-Means" clustering algorithm with 5 clusters, the default number of clusters defined in this script. In figure 4.6 it is illustrated the colour bar obtained from this algorithm, and it is easy to observe that this number is not enough to show, for example, the colour yellow or gold, a



SHOT TRANSITION DETECTED (No. of shots: 49)
Estimated time: 0h28m8s
Is this a cut? (YES/NO)
YES
☑ Select as main shot
Number of colors (clusters):
7

Figure 4.4: Enlarged screenshot of the GUI developed with "Tkinter"

colour that is evident in the symbol and the belt from the picture to the human eye perception.

By using 8 clusters which is the maximum number available in the options of the dropdown menu, instead of 5, the results are much different as it is seen in figure 4.7, where there is a bigger variety of colours. Now there are two different shades of both blue and grey, and besides having the yellow, there is also a very small fragment of beige which is the colour from the skin of the cartoon which is not abundant in the picture.

In the appendix B there is a flowchart which describes the way in which the interactive mode of this last improved version of code is processed.

4.2.3 Fades to White

In the original shot detection file of *Frederic Brodbeck*'s project, the threshold to detect "fades to black" is 0.6. This means that after dividing the channels of the frame in the HSV model, the average of the channel "value" (0-255) has to be equal or lower than 0.6.

After running the tool with different movies, this threshold was adapted to the value of 5.5. And to spare some processing time of the tool, the method to detect fades to black, which was considered the third and last method until this point, was moved in the script to become the first method. This is because a "fade to black" is detected by consecutive almost black frames which satisfy the condition. And

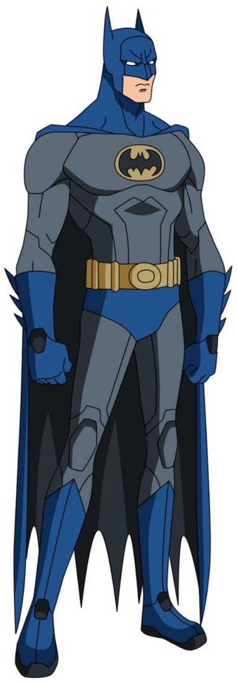


Figure 4.5: Batman cartoon

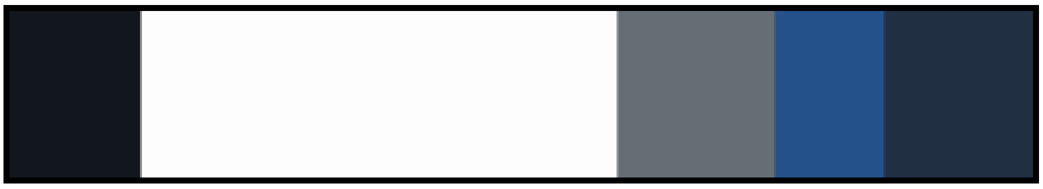


Figure 4.6: 5 clusters

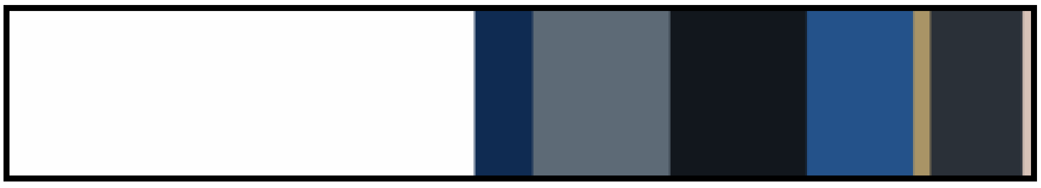


Figure 4.7: 8 clusters

it only ends when a non-almost black frame emerges. Therefore, a condition was created to skip the other methods in case a "fade to black" is occurring.

Using the same paradigm, the detection of "fades to white" was also implemented at this point, because in theory the condition just needed to be changed to also detect frames with an average "value" closer to 255. After running some tests, it was decided that this value would have to be bigger than 200. But to reinforce this condition, for the "fades to white", it was also added a maximum average value of 100 for saturation.

The variable `"f2w"` functions like a flag, because its value is zero by default, but when the first frame of a "fade to black/ white" is closer to white than black, it changes to 1, and the scripts resets it again to 0 when that fade ends. When adding the row to the matrix of "fades to black" with the information of that fade, there is also that value that is either 0 and 1, where 0 corresponds to a "fade to black" (absence of colour), and 1 to a "fade to white". This is described in the index of that last column of the table in the third page of the "Excel" file.

4.2.4 List Creation

In the *Frederic Brodbeck*'s application source code available on "GitHub" [68], there is a file named `"01_1_new-project.py"`. This is the file that initiates the application by creating a project directory for the specific movie file, and by writing its path in a Extensible Markup Language (XML) file so that all the other scripts can fetch this path using the "ElementTree" Application Programming Interface (API).

In the "Cinemetrics" tool developed in this thesis, the idea is to have all these functionalities working in the same script and to call the movie file as an argument of the command line that will run the Python file. Some adjustments were made so that the files `"01_1_new-project.py"` and `"02_1_shot-detection.py"` could be merged into one.

It is not imperative to use the "xml.etree.ElementTree" library for this case since the path of the movie film would not be needed anymore. But for a matter of organization, another file named `"create_list.py"` was created to be executed in the projects' directory chosen by the user.

What this file does is that it creates a list where the first sub-element tells the path of the directory that the user chose to place the files and the outputs generated from the movies. The following sub-elements will have the name of the movie (name of the file), the path where the movie file is located, and it assigns a code depending on the order in which that movie will be processed, in other words, if there are already 2 movies processed with the "Cinemetrics" tool and consequently listed on this file, this next film will have a `"movie code"` correspondent to `"0003"`.

This idea was implemented to better distinguish the output files, by naming them with the code instead of using a long string correspondent to the movie's name. The full name and code of the movie are also displayed in the CSV file as the first cell, but the user can always consult the list (XML file) to be up to date about all the movie files, and their paths, processed until that point.

```xml
<?xml version='1.0' encoding='UTF-8'?>
<movies>
  <movie>
    <code>0000</code>
    <name>CINEMETRICS LIST (XML File)</name>
    <path>C:/User/Documents/Cinemetrics</path>
  </movie>
  <movie>
    <code>0001</code>
    <name>There Will Be Blood</name>
    <path>D:/Films/There.Will.Be.Blood.2007.1080p</path>
  </movie>
  <movie>
    <code>0002</code>
    <name>Inception</name>
    <path>D:/Films/Inception.2010.1080p.BluRay.x265</path>
  </movie>
</movies>
```

Listing 4.6: Example of a generated XML file

4.3 Tool Guide

This section functions as a guide which demonstrates how any user should use this tool in order to a have a better understanding and extract all the potentialities of its features.

1. The user needs to install or check of his Python installation has all the libraries imported in the first lines of the script which are:

   - `sys`
   - `cv2`
   - `numpy`
   - `pandas`
   - `time`
   - `math`
   - `winsound`
   - `os.path`

- `xml.etree.ElementTree`
- `tkinter`
- `KMeans`

2. The user needs to make sure that the movie file is of a common multimedia type, like MP4 (MPEG), MOV, WMV, MKV or AVI.

3. In case the movie file has any black borders, the user has to remove or crop them. And, if needed, change the name of the file to the original title of the movie with spaces and uppercase in the first letters, to later pass the name to all the directories and outputs in the correct way.

4. Before executing the "Cinemetrics" tool, the user needs to choose a folder/ directory where they want the outputs, generated from movies, to be saved. The Python files `"cinemetrics_tool.py"` and `"create_list.py"` also need to be there. After that, the user will need to run the file `"create_list.py"` in order to create a list, and to do this, they need to write the following command with the path of that file:

   - `py path/create_list.py"`

5. The way to execute this tool is to type both paths of the python file and of the movie file (an easy way to do this to drag the files into the command prompt), with the last two arguments being optional (default mode: interactive with the black and white mode turned off):

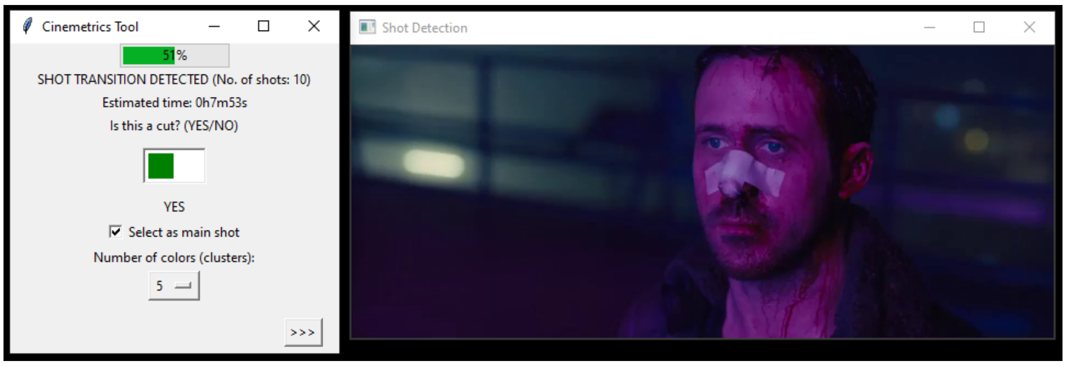
   (a) `py path/cinemetrics_tool.py path/file.mp4 interactive bw`

   (b) `py path/cinemetrics_tool.py path/file.mp4 automatic bw`

6. If the user chooses the automatic mode, they can just let the tool run in the background until they hear a "beep". To stay updated about the progress, the GUI shows the estimated time that will take to finish and he can also look at the percentage bar. If he chooses the interactive mode, the different widgets will come up in the GUI, as they are illustrated in the screenshot of the figure 4.8, and each time the tool detects a cut it will "beep" and stop until the user clicks on the button to skip:

   (a) The first label of the GUI gives an alert when a shot transition is detected, and it tells the number of shots validated until that instant.

   (b) To switch the button from `"YES"` to `"NO"` or vice versa, the user can click on the button to see it shifting between the colour green and red (red by default).



51%
SHOT TRANSITION DETECTED (No. of shots: 10)
Estimated time: 0h7m53s
Is this a cut? (YES/NO)
YES
☑ Select as main shot
Number of colors (clusters):
5

Figure 4.8: Screenshot of the "Cinemetrics" tool in the interactive mode

(c) If for some reason, the user likes the frame in case, for example, if they find it pleasing, aesthetic, with good colours, or if they just consider it as an important frame of the action of the movie, they can mark the checkbox which will make the tool generate a special frame with the image of that instant, a colour palette of it, and a timecode stamp of the instant, in bottom right corner (the box is unchecked by default).

(d) In the dropdown menu, the user can select how many colours (clusters) they want to extract with the clustering algorithm. These colours will be represented in the colour palette with each proportion.

(e) The last widget is the skip button that is placed in the right bottom corner of the GUI, and it has the simple function of proceeding with the analysis of frames until the tool detects another shot boundary.

7. When the script ends, the GUI will close automatically, and from that moment on, the user can open the created folder of that movie and look through all the outputs generated: the informative tables (excel files) and the visualizations.

Chapter 5

Results

The results of the implementation of this project consist of every output generated with the final version of "Cinemetrics" tool from a list of 8 different and diversified movies:

- **12 Angry Men** – directed by Sidney Lumet (1957)
- **There Will Be Blood** – directed by Paul Thomas Anderson (2007)
- **Enter the Void** – directed by *Gaspar Noé* (2009)
- **Inception** – directed by Christopher Nolan (2010)
- **Interstellar** – directed by Christopher Nolan (2014)
- **Spider-Man: Homecoming** – directed by Jon Watts (2017)
- **Blade Runner 2049** – directed by Denis Villeneuve (2017)
- **Lou** – a short film produced by "Pixar Animation Studios" (2017)

Apart from these full movies, there also some results which correspond to specific scenes and clips from certain movies.

5.1 Informative Outputs

To initiate this section, in table 5.1, there is a demonstration of one example of a timecode table that is produced in the shape of a CSV file. This type of generated file only has the the number and time of the instant of each shot, but for informational purposes, the duration and and value of motion were also added.

The index of this table shows the number or order of each shot, the initial instant in timecode format, in which the shot appears, the duration of each shot in number of frames, considering they are all less than one second, and the value of motion in percentage.

In this example, instead of a full movie, a short 6-second scene from the movie "Taken 3" (2014) [61] was processed (the one already mentioned in the chapter 3).

Table 5.1: Timecode table from a scene of the movie "Taken 3" (2014)

Shot no. (#)	**Instant (timecode)**	**Duration (frames)**	**Motion (%)**
0	00:00:00:00	16	33
1	00:00:00:16	19	48
2	00:00:01:10	10	49
3	00:00:01:20	11	51
4	00:00:02:06	10	37
5	00:00:02:16	10	44
6	00:00:03:01	11	29
7	00:00:03:12	13	46
8	00:00:04:00	11	30
9	00:00:04:11	7	34
10	00:00:04:18	9	35
11	00:00:05:02	8	46
12	00:00:05:10	10	31
13	00:00:05:20	9	43
14	00:00:06:04	13	34

Three of the average values of motion are equal or bigger than 48 percent (threshold = 0.48), which just proves that the tool detected some false cuts; also, since these values are an average, there could be even more false cuts for the fact that all the values are very high and close enough to the threshold.

The script detected a total of 22 cuts, which can be explained due to the fast paced action and to the rapid camera movements that take place in this scene. However, the real number of cuts actually corresponds to 14, because this video clip was processed in the interactive mode. So, in the end, this value is based on the eye test, and it is exactly the right number that was mentioned before in a previous chapter [61].

So the efficiency ratio (number of real shots/ number of shots detected) of this process in the automatic mode would be 0,(63). And the processing time of the interactive mode (77 seconds) was roughly 12 times bigger than the duration of the clip (6 seconds and 17 frames).

5.1.1 Value Comparisons

The values presented in table 5.2 were collected with the "Cinemetrics" tool and transcribed to the second page of the `"results_table_XXXX.xlsx"` file, `"XXXX"` being the movie code. This data is part of the informational "numpy" array. Both the duration and the processing time values are in minutes. The values of the movie duration were actually referenced from the "IMDb" database [23], instead of using the timecode duration format of the file. And the values of processing time were converted from seconds to minutes for an easier comparison.

To see how efficient the tool is for each case, some comparisons were made between the values of number of shots to calculate the efficiency ratio. To do so, the divisor is always the higher value so the division value is under 1, this is because the idea is get a value that indicates the closeness of both values. The number of shots detected is the value obtained through the use of "Cinemetrics" tool, while the GT value is indicated by a specific reference or by the submission in the "Cinemetrics" database [46] that looks more trustworthy.

To be more specific, the GT data of the movie "12 Angry Men" came from a submission [77], in an advanced mode, out of a total of 8 submissions uploaded in the "Cinemetrics" database.

The GT data of "There Will Be Blood" came from an article which talks about the importance of each shot in the movie [62]. This source provides arguably the most accurate value of the total number of shots, because these were counted and annotated rigorously and manually.

The data of the movie "Enter the Void" is from a singular submission in "Cinemetrics" [78], and the number of shots is an estimate calculated based on a rule of three, considering that this data provided only applies to a portion of 40 minutes of the movie, because of this it is not very reliable. The processing time of this movie was surprisingly fast comparing to the duration of the movie, taking into account that it had a very low efficiency from the tool in detecting the cuts precisely. This can be explained by the fact that the film "Enter the Void" has a lot of colour changes, blinks, psychedelic effects, and flashes of light, which describe the hallucinating moments of the characters while on drugs, or the night lights from the city of Tokyo, which are so characteristic.

The data from "Inception" [79] and "Interstellar" [80] also come from submissions of "Cinemetrics".

Similarly to "Enter the Void", the source from "Spider-Man: Homecoming" is not accurate, because the data corresponds to an excerpt from the movie of half an hour, so the number of shots was calculated by proportion [81].

For the movie "Blade Runner 2049", there are 1190 VFX shots that occur for a total of about an hour and 40 minutes (100 minutes) [82]. Considering that the full duration of the film is 164 minutes, that leads to an estimate of around 1952 shots.

Table 5.2: Table of informative values from movies (12AM: 12 Angry Men; TWBB: There Will Be Blood; EtV: Enter the Void; INCE: Inception; INT: Interstellar; SMH: Spider-man: Homecoming; BR49: Blade Runner 2049)

Index / Film	**12AM**	**TWBB**	**EtV**	**INCE**	**INT**	**SMH**	**BR49**
Frame rate (FPS)	24	24	25	24	24	24	24
Fades to B / W	2	75	279	51	152	101	162
Cuts per minute	3	4	112	18	17	25	11
ASL (in seconds)	18	13	<1	3	3	2	5
G-truth ASL (s)	16	14	6	3	4	3	5
No. of segments	33	71	1740	273	301	347	190
Number of shots	327	707	17393	2730	3003	3463	1897
GT No. of shots	359	678	1555	2757	2317	2341	1952
Efficiency (%)	91	96	9	99	77	68	97
Proc. time (min.)	188	329	234	341	335	304	309
Duration (min.)	96	158	161	148	169	133	164
Processing ratio	1.96	2.08	1.45	2.30	1.98	2.29	1.88

The main conclusion from table 5.2 is that films with a slow pace have a lower chance of getting false cuts detected by the tool. The bigger exception in this case is "Inception", which is a fast paced movie, and it was, surprisingly the one with more efficiency (99%).

It is important to highlight again that most of this GT data is not totally reliable, which makes it ambiguous to call it Ground Truth (GT) data, but this denomination is merely used to distinguish the obtained value from the reference value, for comparison.

5.1.2 Shots Table

The tables 5.3 and 5.4 are, in this case, tables of segments, and display the data about each one of them from the movie "There Will Be Blood". This is the coloured movie with the least number of segments (71), and these tables show the number (or order) of each segment, its starting frame, the instant of that frame (in timecode), and the duration (in seconds and frames) and motion (in percentage) of each segment.

The graph in figure 5.1 represents the distribution of the segments, where the X-axis corresponds to the order of the frame, and the Y-axis corresponds to the duration in seconds (left side) and to the motion in percentage (right side).

The observations that stand out the most are the high peak of motion over 40%, between the 80 000 and the 90 000 frames, and the two long segments with over than 350 seconds, which is almost 6 minutes.

In the appendix B there is another example of a table of shots.

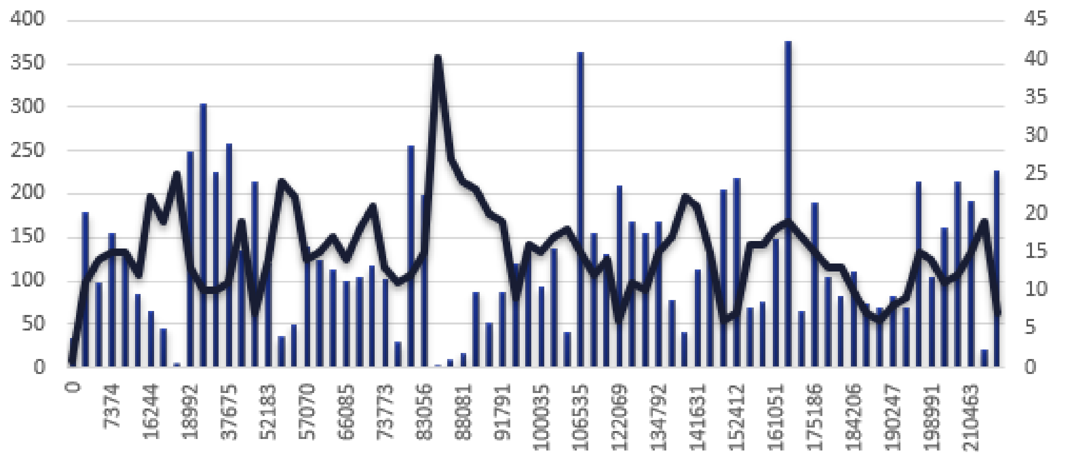


Figure 5.1: Distribution of the segments from the movie "There Will Be Blood"

5.1.3 Table of Fades to Black/ White

The list of "fades to black/ white" are transcribed to the third page of the ".xlsx" file.

In the table 5.5 it is represented the first 15 "fades to black" from the movie "Inception". Even though, the "Cinemetrics" tool also detects "fades to white" and it indicates them in the file, these type of fades is very rare, and because of that, all the fades from this table are exclusively "fades to black".

5.2 Visualizations

This section will be mainly graphic, because it is comprised of the visual outputs generated from the tool. However, some brief comments will be added to discuss certain results.

5.2.1 Colour Palettes

In figure 5.2 there are 6 colour palettes generated from the totality of different movies. An observation that is common among most of these movies is that they have a lot of pixels of low value and saturation in their colours (HSV), and also a lot of black or dark (near to black) pixels. These dark pixels can represent, for example, dark caves or holes, in "There Will Be Blood", the outer space, in "Interstellar", or even clothing like the dark suits of the characters, in "Inception".

In the subfigure 5.2b, the colour palette of "Enter the Void" looks very homogeneous in its saturation, in the same way as when a fading filter is applied to a photo. In this case, the production decided apply this filter to the whole movie, which is not undoubtedly the natural colour scheme that the typical cameras capture. Either on post-production and editing of the movie, or by using a special type of camera for all shots, this technique applied gives it a more interesting look.

The best highlight of these colour palettes belongs to the movie "Blade Runner 2049" illustrated in subfigure 5.2f, that has a great variety of colours throughout the palette, where the ones that stand out more are the most vivid and saturated colours, which coincide with the longest segments. The reason behind this is that the director Denis Villeneuve chose to record longer shots for the most colourful and appealing settings to assign them a greater need for attention, reflection and observation. In other words, to give them a greater appreciation that goes beyond the action of the movie by portraying how the sets can be so impactful, sometimes even more than the plot itself, in any cinematographic content.

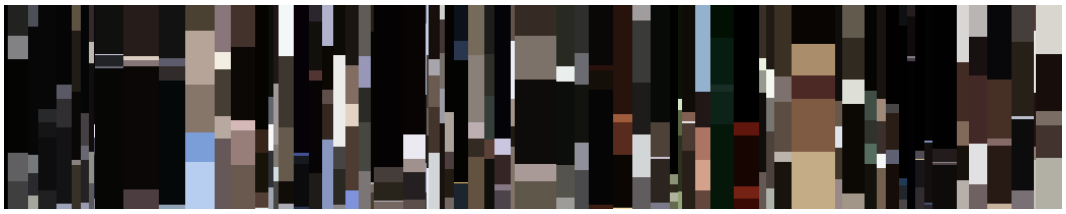
The colour palette from the figure 5.3 corresponds to the "Taken 3" scene represented in the table 5.1. Besides black, grey and white, this palette is mainly constituted of blue shades from the sky, green shades from vegetation (grass, trees and plants), and brown shades from the floor (dirt) and structures.

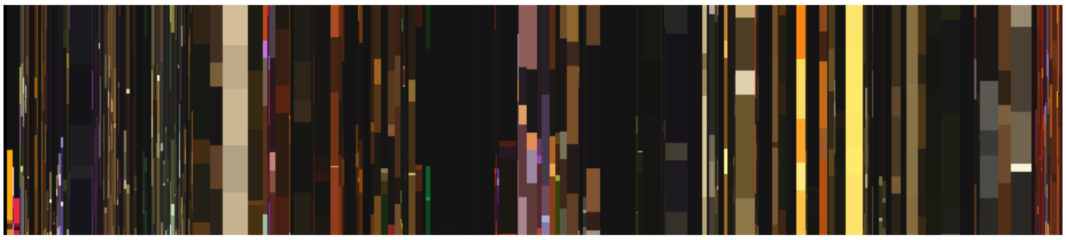
The palette displayed in figure 5.4 comes from the movie "Pulp Fiction" (1994), in the classical scene where the characters played by Uma Thurman and John Travolta dance with each other. This palette is entirely made of black and many different shades of brown, this is due to the elements of the set like the stage or the walls in the longer shots (far camera perspective) according to the shot scale. And in the close-ups of the first shots, the characters are still seated and having a conversation, and since their faces occupy the majority of those frames, the colours that stand out are the hair colour which is black, and the skin colour which is a lighter shade of brown. All these elements are the most predominant in their respective frames and they are mainly brown. And even though there are many more colours in smaller elements of these shots, there are no more colours being extracted to the palette, because the default number of clusters of the script is 5. In figure 2.14a of chapter 2 there is a frame of this scene, and it is possible to observe a much bigger number of colours in the general spectrum.

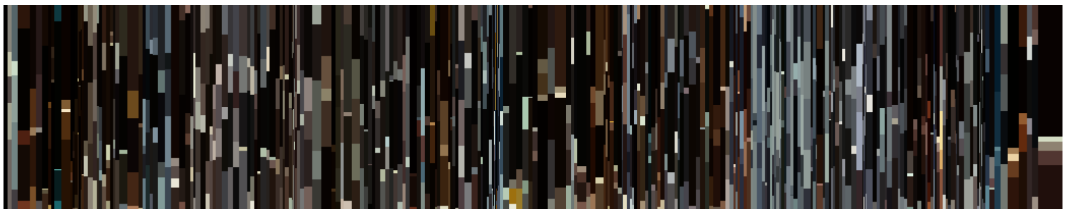
The big middle section of the palette coincides with the actual shot where the characters start dancing, which is 68 seconds long, more than one third of the whole clip.

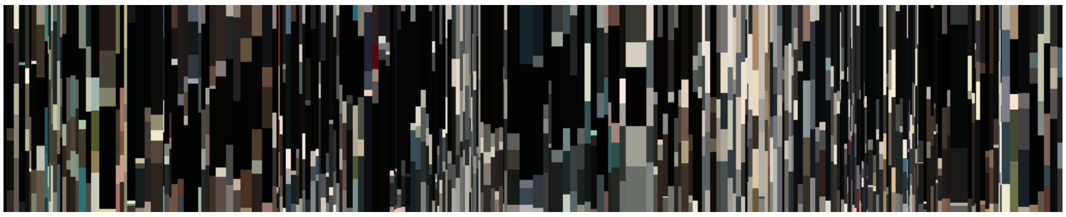
It is much easier to detect a pattern in these type of clips, because besides them having multiple shots, they all belong to a concrete and singular scene. In figure 5.5, the most noticeable patterns consists of a mixture of colours between blue, pink and purple, and it represents a notorious scene from the movie "Blade Runner 2049" (2017), where the main character, played by the actor Ryan Gosling, comes across an enormous pink hologram of a girl. The intensity of the light during nighttime gives the viewers a very powerful and visually appealing contrast.

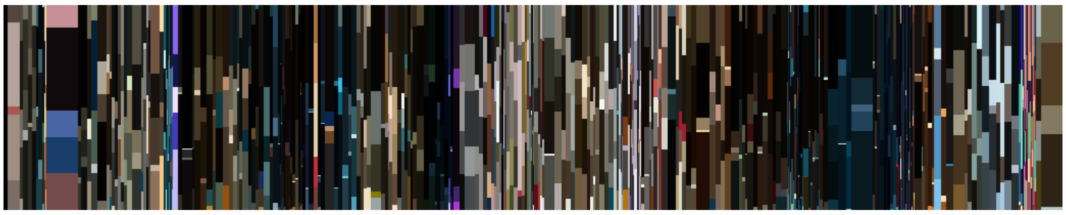
What all of these three last palettes have in common is that since they are all


(a) "There Will Be Blood" (2007)


(b) "Enter the Void" (2009)


(c) "Inception" (2010)


(d) "Interstellar" (2014)


(e) "Spider-Man: Homecoming" (2017)

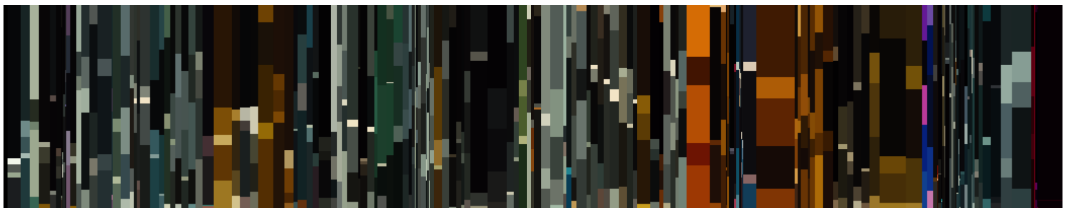

(f) "Blade Runner 2049" (2017)

Figure 5.2: Colour palettes extracted with the "Cinemetrics" tool from six different movies

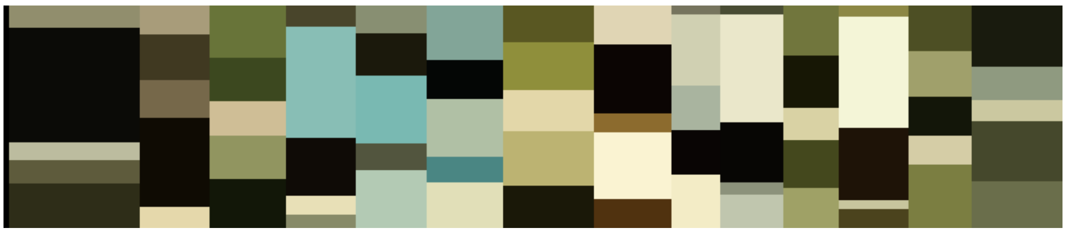


Figure 5.3: Colour palette extracted from a scene with multiple cuts of the movie "Taken 3" (2014)

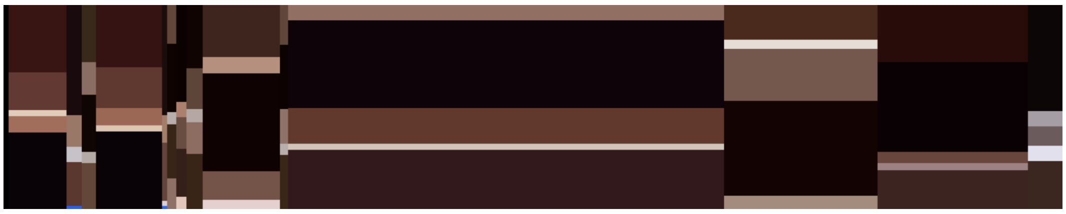


Figure 5.4: Colour palette extracted from the dancing scene of the movie "Pulp Fiction" (1994)

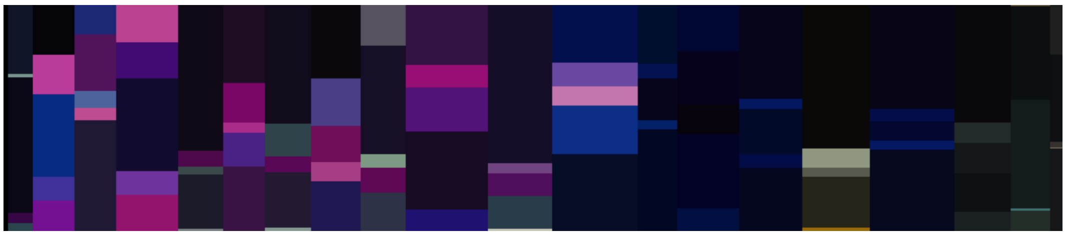


Figure 5.5: Colour palette extracted from the pink hologram scene of the movie "Blade Runner 2049" (2017)

small video clips instead of full movies, they have less compression of colours which makes them have a cleaner look that is easier to observe.

5.2.2 Vertical and Horizontal Prints

In figures 5.6 and 5.7, which illustrate the vertical and the horizontal prints respectively, besides the colours of "Blade Runner 2049" looking very appealing again, the only prints that deserve a special focus are both the vertical and horizontal prints from the movie "Enter the Void". These prints are very characteristic, because, in addition of being also very colourful, they resemble a lot the print made in "The Colors of Motion" for the same movie, which proves just how meticulous this tool can be. For comparison, this print is represented in the figure 3.15b already displayed in the chapter 3. The most prevailing colours of this movie are orange and purple.

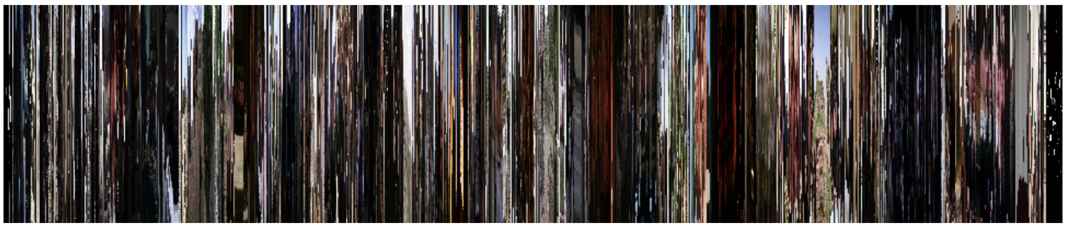
The other interesting point related to these prints is that in the vertical prints in figure 5.6, the one extracted from "Interstellar" seems to have an intermittent use of the "letterboxing" technique. The user guide that explains how to properly use the "Cinemetrics" tool underlined the importance of removing black borders, but this movie functions as an exception, and the nature and reason behind this are a part of a vast list of distinguishing characteristics that makes this movie so unique.

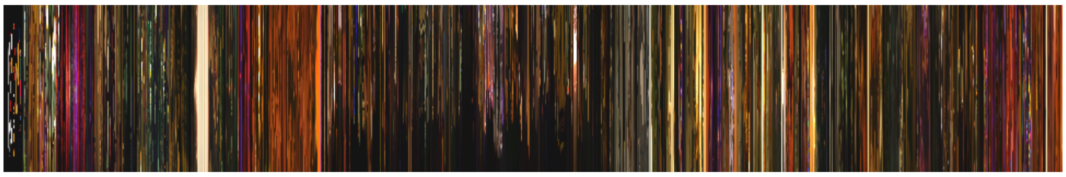
In the production of this film, the director Christopher Nolan uses two types of format for different shots and recordings. This causes the movie to be screened in theaters with periodic changes of the aspect ratio. One of the format is a classic and ordinary one, while the other is "IMAX". "IMAX" is a high-resolution format that should be displayed at specialized "IMAX" movie theaters to portray its maximum capabilities. These theaters are known for having very big screens, bigger than usual, and greater quality [83].

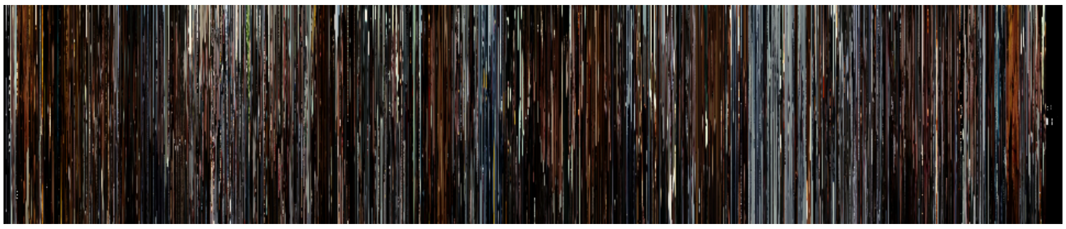
The "IMAX" format is known for having a "tall" aspect ratio, 1:43:1 in this movie, to be more specific. And it is used in key dramatic moments. This means that by observing the figure 5.6d in detail, it is possible to recognize this moments, because they correspond to the sections of the print that don't have the effect of "letterboxing". And it is very logical and coherent that those moments show up more often in the last third of the movie's course considering that most movies grow in intensity towards the end.

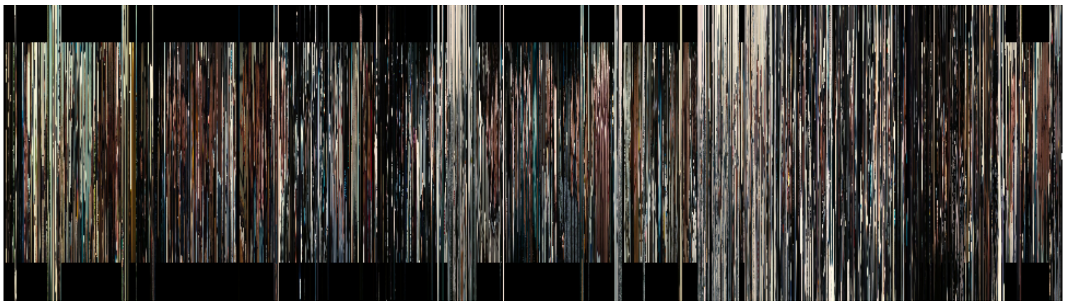
5.2.3 Slit Sequences

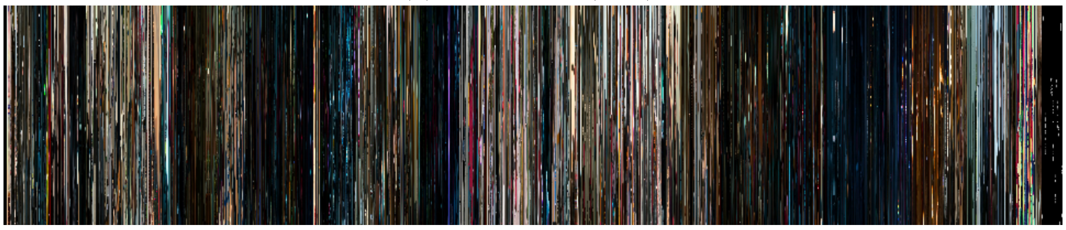
The concept of slit sequences serves to depict all the scenes from a movie. All of them are generated separately in case the viewer wants to see each scene individually or to take a closer look.


(a) "There Will Be Blood" (2007)


(b) "Enter the Void" (2009)


(c) "Inception" (2010)


(d) "Interstellar" (2014)


(e) "Spider-Man: Homecoming" (2017)

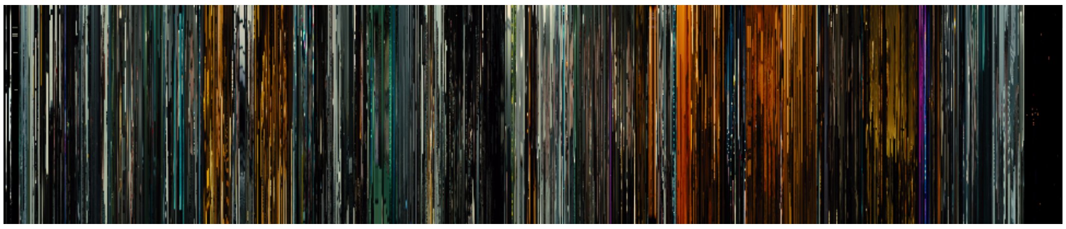
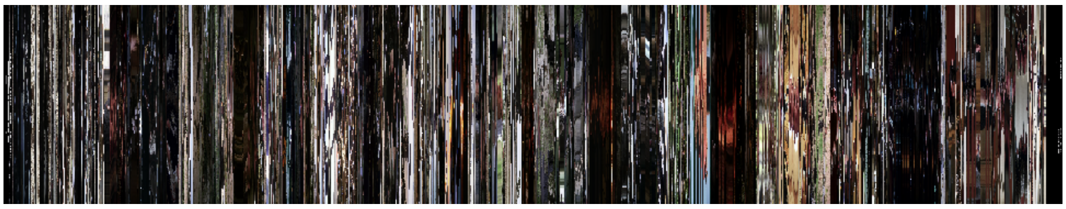
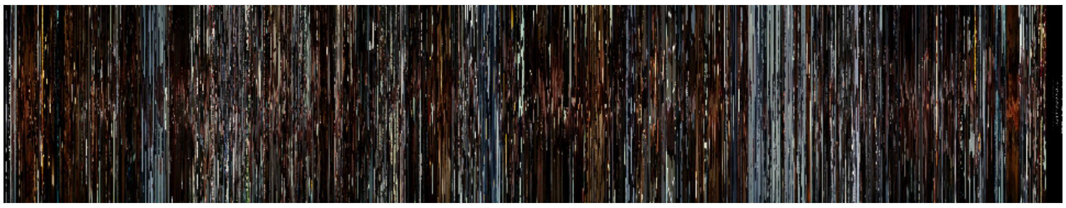

(f) "Blade Runner 2049" (2017)

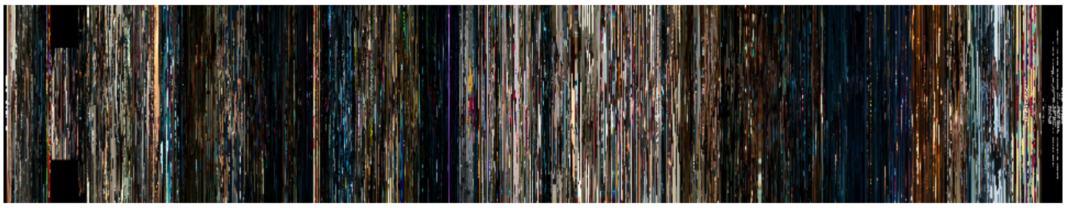
Figure 5.6: Vertical prints


(a) "There Will Be Blood" (2007)

(b) "Enter the Void" (2009)


(c) "Inception" (2010)

(d) "Interstellar" (2014)


(e) "Spider-Man: Homecoming" (2017)

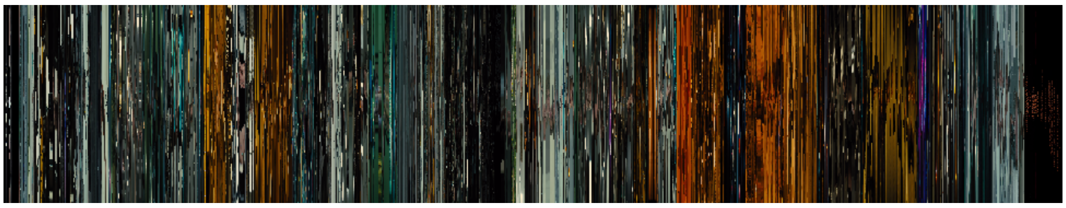

(f) "Blade Runner 2049" (2017)

Figure 5.7: Horizontal prints

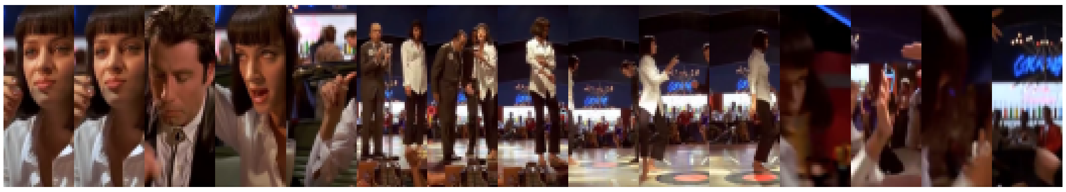


Figure 5.8: Slit sequence from the dancing scene of the movie "Pulp Fiction" (1994)

It is important to mention that these examples of slit sequences were generated with different versions of the tool, therefore they vary in frequency of frames to show diversity.

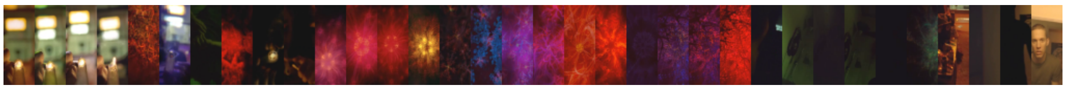
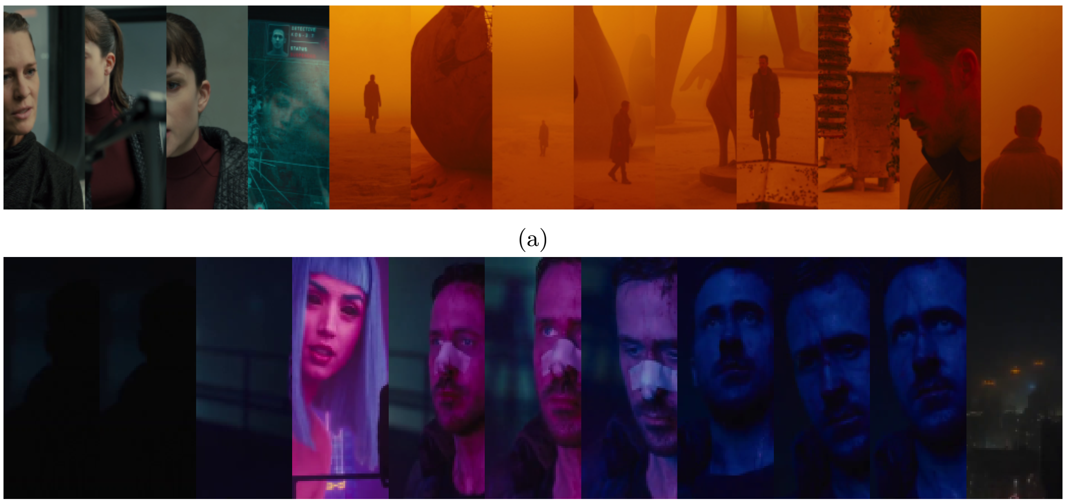


Figure 5.9: Slit sequence from a scene of the movie "Enter the Void" (2009)

The figure 5.9 portrays the hallucinating effects that the character is feeling induced by the consume of drugs.



(a)

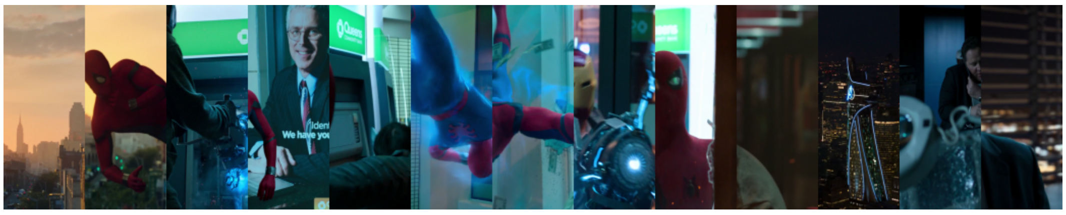
(b)

Figure 5.10: Two slit sequences from different scenes of the movie "Blade Runner 2049" (2017)

The slit sequences in figure 5.10 describe two very characteristic scenes from the movie "Blade Runner 2049" (2017), where there is a majority of vivid and bright colours.

The figure 5.11 shows two highlighted slit sequences where the silhouette or the face (mask) of the main character "Spider-Man" appears in the main plan.

Another way to visualize slit sequences is by merging them, making a collage. The figures 5.12 and 5.13 consist of two halves of a collage done with all the slit


(a)

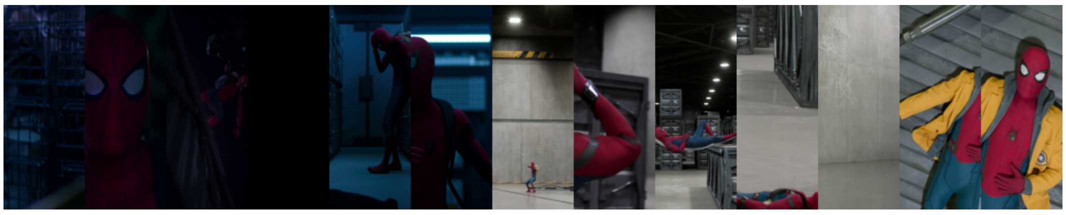

(b)

Figure 5.11: Two slit sequences from different scenes of the movie "Spider-Man: Homecoming" (2017)

sequences generated from the movie "Interstellar" (2014). This type of visualization is very useful to get a feedback from the whole constitution of any movie without making it tiresome. The collage was done using the "Picasa 3" photo editing software. It is made of a total of 60 slit sequences that are ordered from left to right and then down, like reading of a book in the western standards.

"Interstellar" is the perfect case for describing a collage of slit sequences the best way possible, for the reason that it is a very rich movie with a lot of plot twists and turns, and the alternation of aspect ratio and "letterboxing" (black borders), gives the collage a sense of separation between the rows. This conveys the collage an interesting complexity and a type of organization that it wouldn't exist on a regular movie.

5.2.4 Main Shots

The figure 5.14 represents a main frame from the iconic ending scene from the movie "Fight Club" (1999).

The timecode stamps in the main shots of this subsection can be ignored, because all these frames were extracted, with the "Cinemetrics" tool in the interactive mode, from small clips of video, instead of whole movies to be less tedious, since the goal here is just to illustrate some examples of this type of visual output. However, they are still displayed chronologically.

Not all of the colour bars have the default number of 5 colours, since it that came down to the decision of the user.

There are some special frames in every movie which define it for being important to the plot or for being engaging due to its colours and shapes.

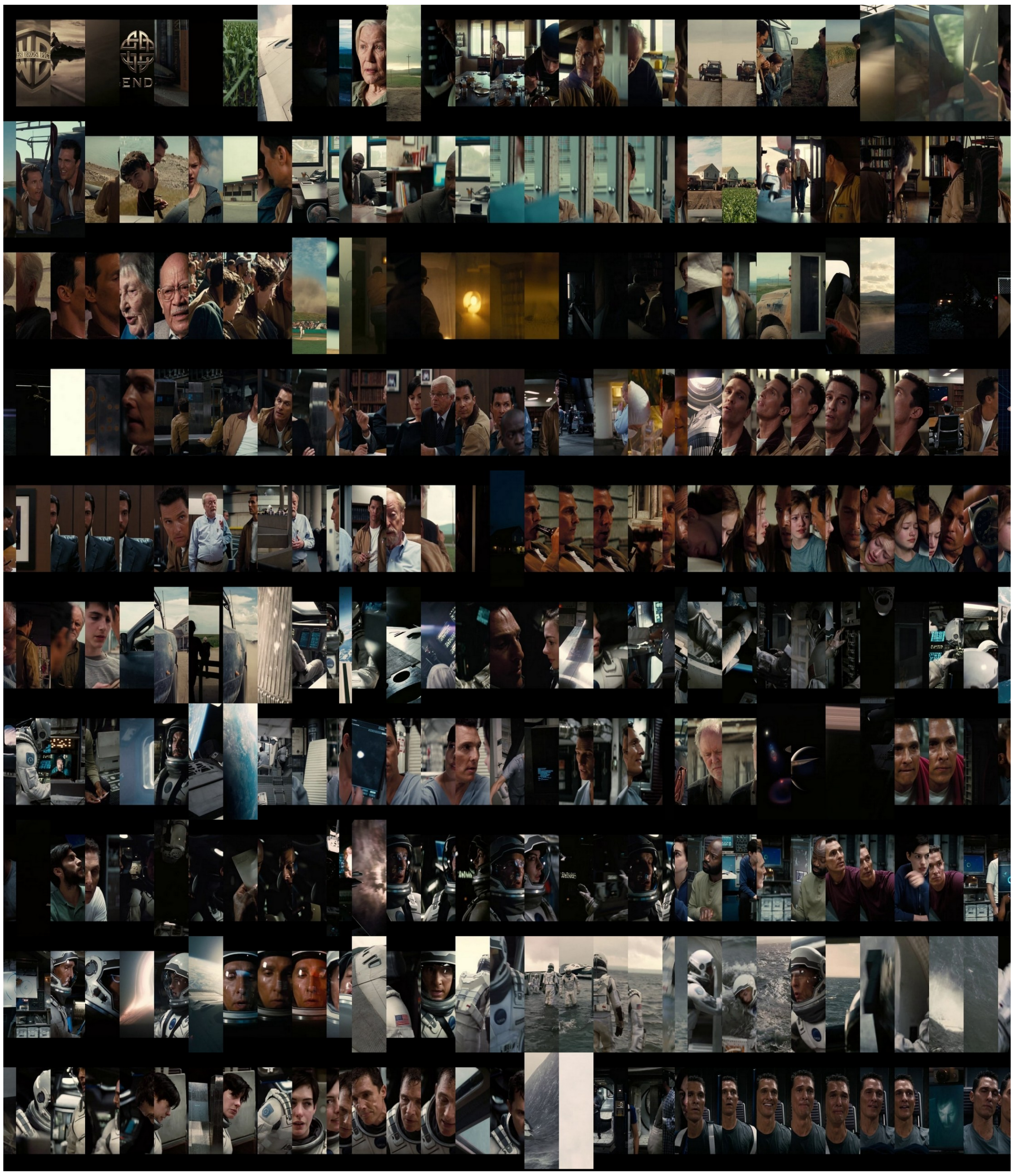


Figure 5.12: First half of the collage made with the slit sequences extracted from the movie "Interstellar" (2014)

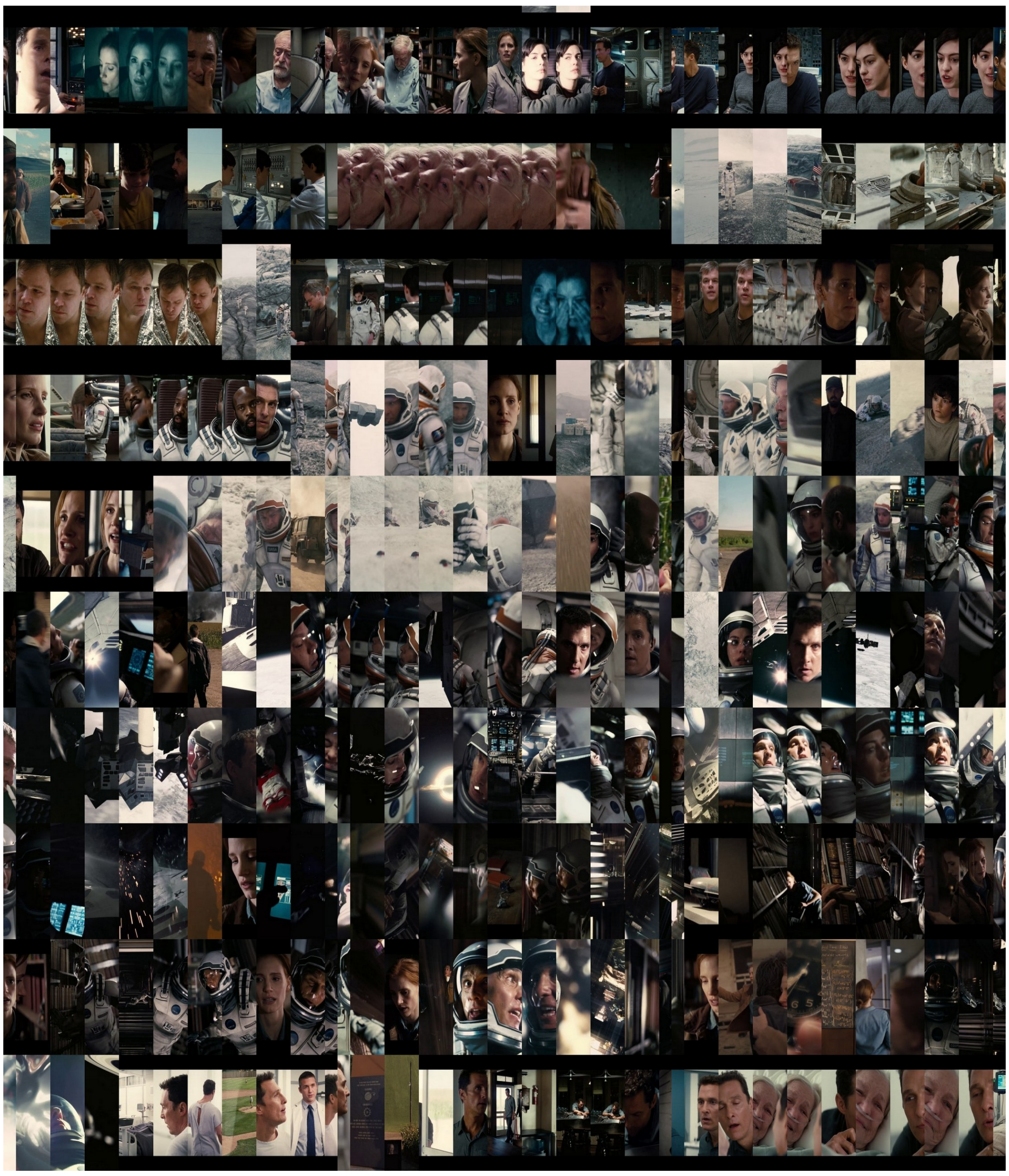


Figure 5.13: Second half of the collage made with the slit sequences extracted from the movie "Interstellar" (2014)

Any of the main shots from figure 5.15 are seen as remarkable, because they represent such a famous moment of this movie. The movie "Pulp Fiction" is well-known for influencing the film industry and even pop culture in general.

The movie "Blade Runner 2049" (2017) is also very impressive with its bright colours and futuristic traces. The figure 5.16 shows 4 main shots from the pink hologram scene, with subfigure 5.16a being arguably the most powerful and prominent one.

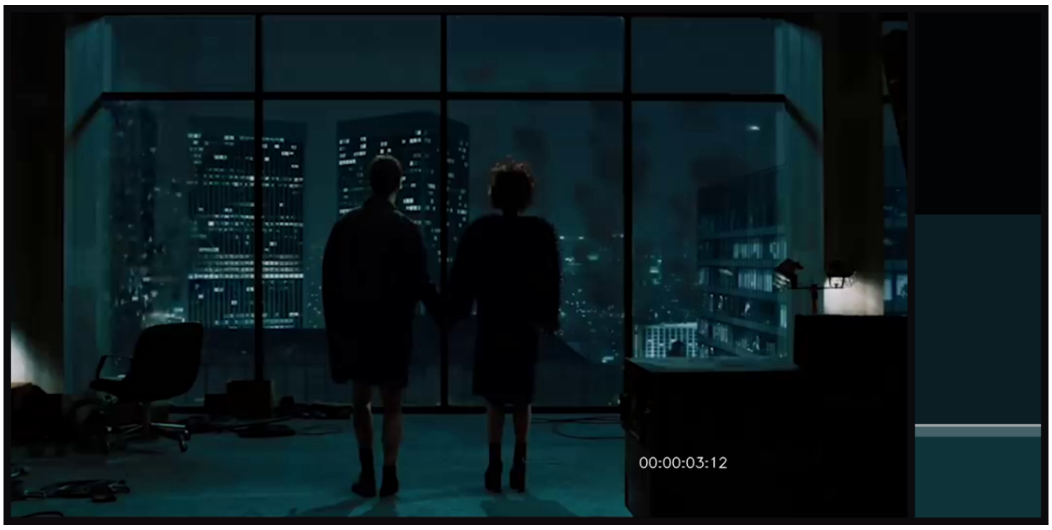


Figure 5.14: Main shot from the ending scene of the movie "Fight Club" (1999)

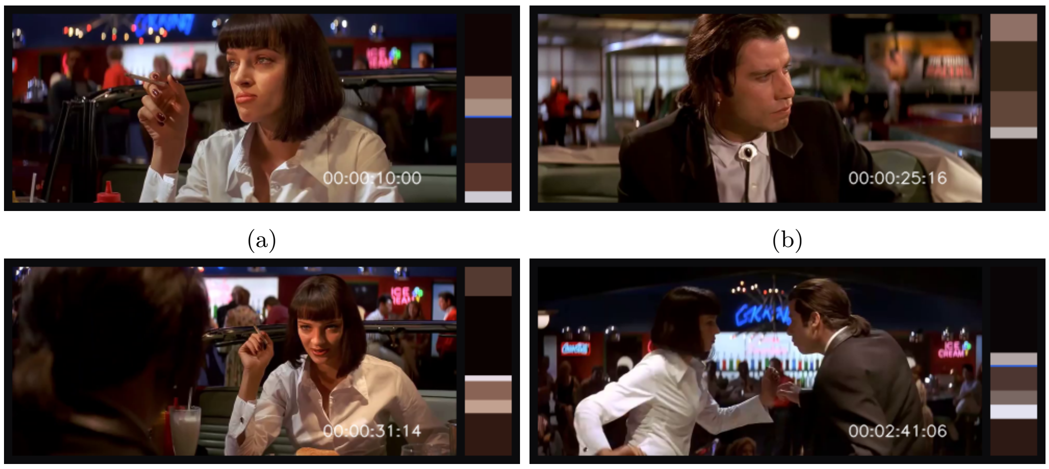


Figure 5.15: Main shots chosen by the user from the dancing scene of the movie "Pulp Fiction" (1994)

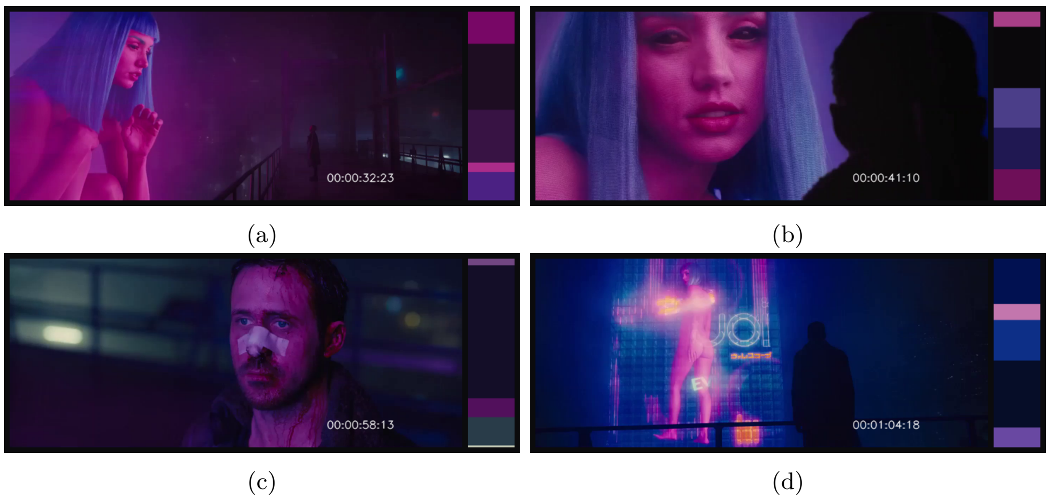
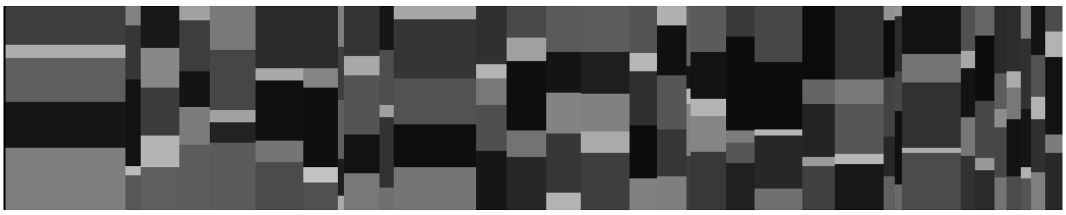
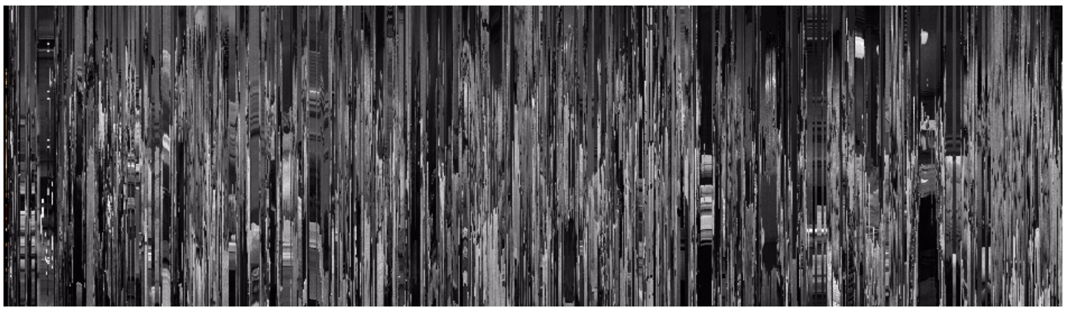


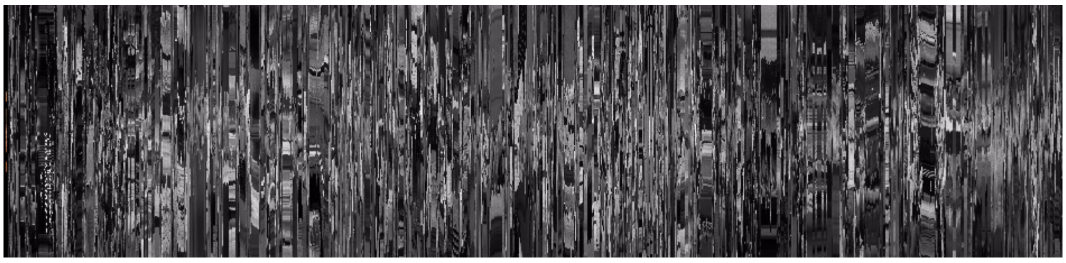
Figure 5.16: Main shots chosen by the user from the pink hologram scene of the movie "Blade Runner 2049" (2017)



(a) Colour palette



(b) Vertical print



(c) Horizontal print

Figure 5.17: Different visual outputs from the black and white movie "12 Angry Men" (1957)

5.3 Black and White Film

The main reason for choosing the movie "12 Angry Men", produced in 1957, was to have an example of a black and white movie. The purpose of this chapter is to demonstrate different examples that make use of all the features implemented in the "Cinemetrics" tool, the black and white mode being one of them.

The informational values of this film can be seen in the first column of table 5.2 (excluding the index). This is the movie with the least segments, highest ASL and consequently the one with the least cuts per minute.

The processing time was the shortest not only because it was the shortest movie, but also because black and white frames and movies are easier to process.

The visual outputs for black and white movies are, as expected, not very conclusive, as it is seen in figure 5.17. With the exception of the slit sequences, which is the only type of visual output where the characters or the setting can be perceptible. Some of these slit sequences are portrayed in the figure 5.18.

These slit sequences describe different phases of the movie. The subfigures 5.18a and 5.18b represent an introduction to the characters where the environment looks calm, and the twelve are getting to know each other.

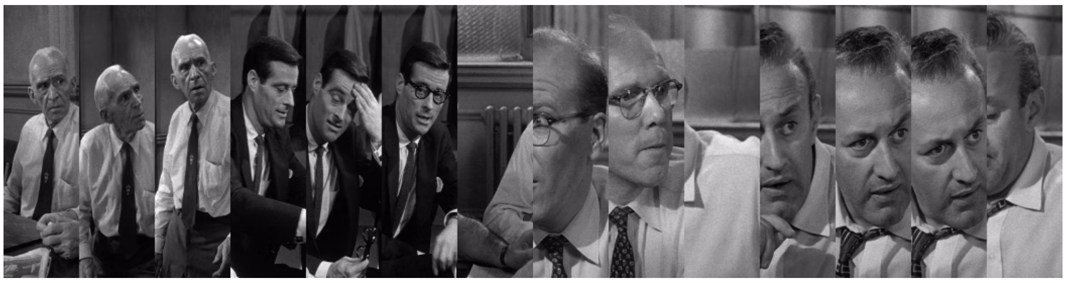
The subfigure 5.18c represents a middle section, the development of the case, where the character played by Henry Fonda is pouring over the subject, rationalizing and analyzing over every hypothesis.

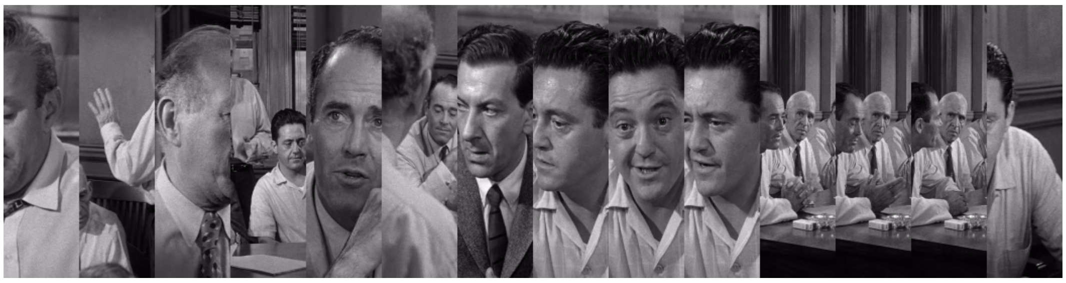
In subfigures 5.18d and 5.18e there are the moments when the drama and intensity are soaring. The characters look nervous and angry, as the title of the movie implies, and the camera focus its center on the face of each jury member. It is easy to recognize the faces and emotions in every character portrayed in these last moments of the movie, due to the way the slits are extracted from the frames.

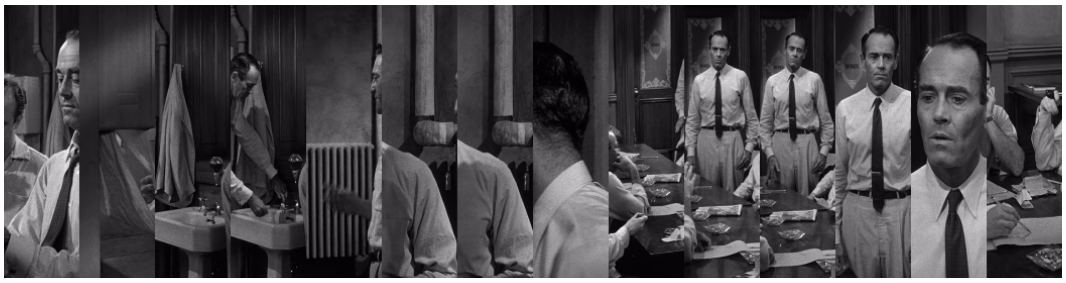
The graph in figure 5.19 is built the same way as the one in figure 5.1, and it shows how low the values of motion are in comparison. This can be explained by the fact that the whole course of this movie takes place in a single room. The second segment of this movie has an impressive length of more than 600 seconds.

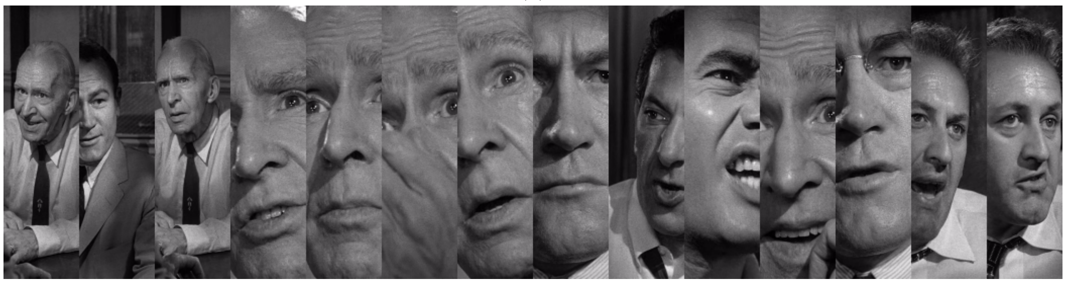
5.4 Short Film

Since the interactive mode forces the user to pay attention and stay aware for the whole process to see which shot transitions detected are actually true, at this point it is easy to realize how laborious this tool can be in that mode. For this reason, and since all the examples and results shown until now consist of feature-length films, the next results will be provided from a short film, more concretely a well-regarded "Pixar" short named "Lou" (2017).


(a)


(b)


(c)


(d)

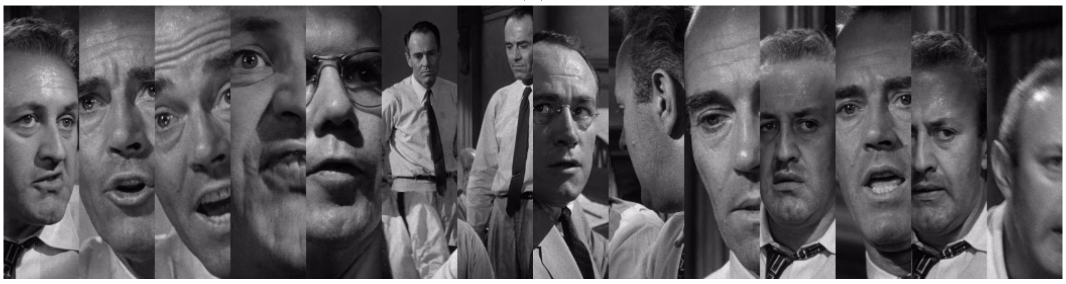

(e)

Figure 5.18: Five slit sequences extracted from the black and white movie "12 Angry Men" (1957)

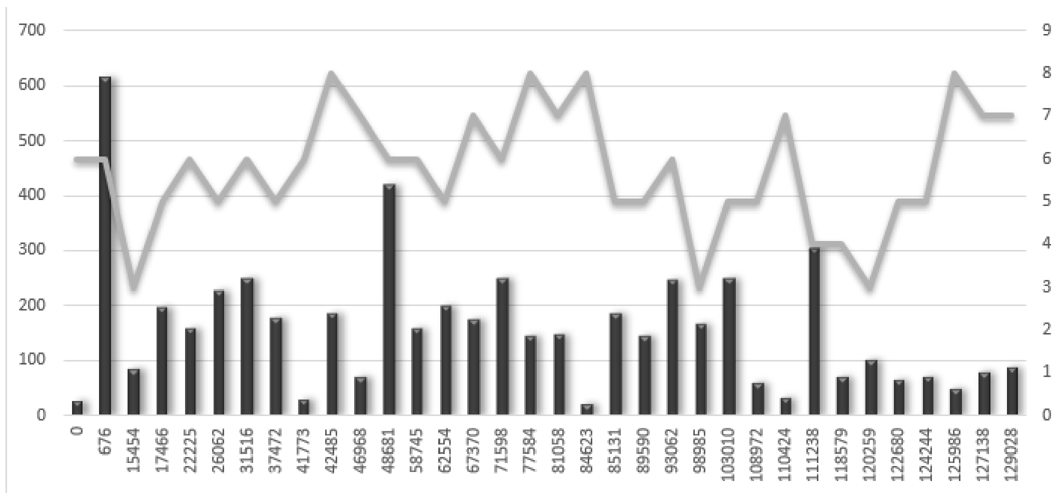


Figure 5.19: Distribution of the segments from the black and white movie "12 Angry Men"

The curious aspect of this movie is that it is animated. So in theory, it wouldn't have shots or camera movements, but nowadays as movies of this nature are becoming more realistic, and their quality and detail have improved, the Visual Effects (VFX) are used in order to give the perception of these recording techniques, to give the film a more realistic and captivating form.

So, even if they are artificial and being reproduced with VFX, the shots and cuts really exist in this type of movies.

Most of the time, the human mind can distinguish what is really a cut from what is not, and the same applies for animated movies. Therefore, while using the interactive mode for this movie, some annotations were made to count and describe how many false cuts were detected in the different instants throughout the film, to try to explain why the tool detected them.

The table 5.8 displays those annotations, where the first row represents instants or percentages of the progression of the movie ("OC" stands for opening credits and "EC" stands for end credits), and the second row the number of occurrences of false detections.

Table 5.6 indicates the total number of frames of the movie, its frame rate, the total number of "fades to black/ white", the duration of the movie and the processing time of the tool in minutes, and the processing ratio which tells how many times the processing time is longer than the duration of the video.

In table 5.7, there are some other informative values like the total number of cuts detected compared to the real and confirmed ones.

And even though the ".xlsx" file doesn't give the MSL value, it was possible to calculate it in the "Microsoft Excel" program with the function "MEDIAN". Firstly, "Microsoft Excel" indicated that the median between the duration values in seconds was 2, and after that, with the values in the unit of frames corresponding to 2 seconds it returned a median of 7 frames. Since the frame rate is 24 FPS, the equivalent of 2 seconds and 7 frames is approximately 2.3 seconds.

Excluding the rare case of the movie "Enter the Void", "Lou" had by far the process of detection with the lowest efficiency. This can be explain because of the rapid succession of frames that typically occur in animated films, hence the reason for having annotated the false detections manually. In other words, there are a lot of moments in this type of movies where the frames change abruptly from one to another, when in reality the action is still happening in the same setting. For example, when the main character kid is chasing "Lou", there are a lot of instants where the script mistakenly detects cuts. Another example is when, in production, they try to simulate a very quick zooming of camera in moments of surprise or when the main character looks back on a memory in his mind. This type of fast zooming can be seen on the instants of 61%, 64% and 65%.

The reason why there are many more false detections in the opening credits compared to the end credits is because the script removes the last 30 seconds of a short film as an estimate for the credits' length.

In table 5.7, by adding the false detections, there are 35 opening credits plus 15 end credits, and the other 66 false cuts. This gives a total of 116 false cuts, and by subtracting the total number of shots detected (250) by the number of real shots (134), the values match.

The shots table of this movie can be seen on the appendix B (it is displayed there for being 3 pages long). By observing the values of motion, both on that table or in the graph in figure 5.20, it is possible to associate instants of high values of motion to the percentages seen in table 5.8. In the graph, the biggest anomaly is the high peak somewhere around the interval between the frames 5500 and 6000.

Contrarily to table 5.5, table 5.9 has two "fades to white", where one of them corresponds to the moment when the main character starts to relive a memory.

The colour palette of this movie, represented in figure 5.21a, it is very engaging and colourful, for the reason that the movie is animated.

In figure 5.22 there are the only two generated slit sequences which summarize this short film in perfection without giving away too much spoilers, also for not showing the frames in their full dimensions.

The main shots represented in figure 5.23, on the other hand, are much more indicative of the plot, and they highlight some of the most important points in the movie, while illustrating different samples of colours.

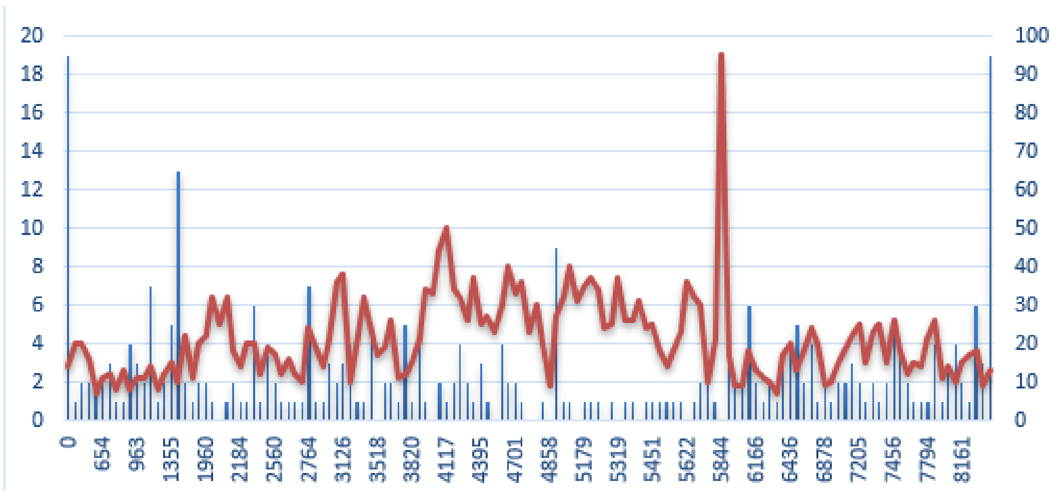
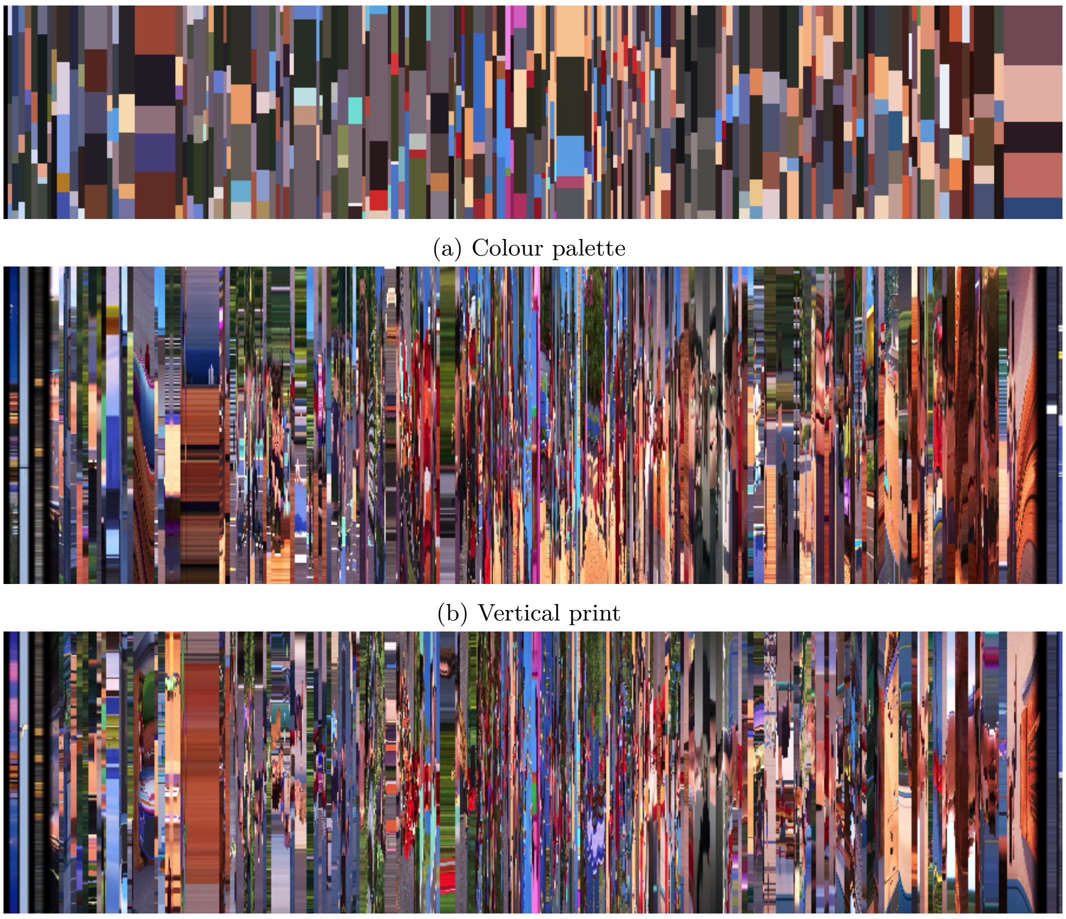


Figure 5.20: Distribution of the shots from the "Pixar" short "Lou" (2017)

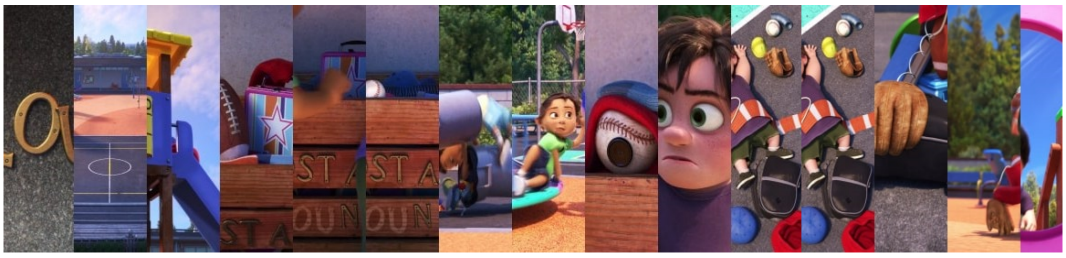


(a) Colour palette

(b) Vertical print

(c) Horizontal print

Figure 5.21: Different visual outputs from the "Pixar" short "Lou" (2017)


(a)

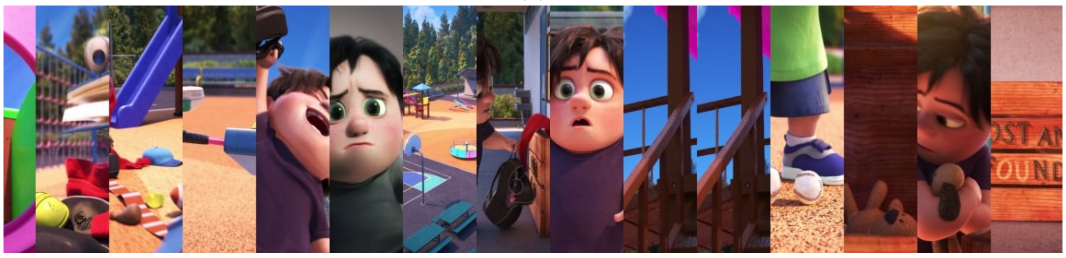

(b)

Figure 5.22: Two slit sequences extracted from the "Pixar" short "Lou" (2017)

Table 5.3: First half of the table of segments extracted by the "Cinemetrics" tool from the movie "There Will Be Blood" (2007)

#	Frame	hours	minutes	seconds	frames	Sec	Fra	Motion (%)
0	0	0	0	0	0	33	2	1
1	794	0	0	33	2	177	15	11
2	5057	0	3	30	17	96	13	14
3	7374	0	5	7	6	155	23	15
4	11117	0	7	43	5	129	1	15
5	14214	0	9	52	6	84	14	12
6	16244	0	11	16	20	64	0	22
7	17780	0	12	20	20	45	15	19
8	18875	0	13	6	11	4	21	25
9	18992	0	13	11	8	249	9	13
10	24977	0	17	20	17	303	6	10
11	32255	0	22	23	23	225	20	10
12	37675	0	26	9	19	256	12	11
13	43831	0	30	26	7	134	22	19
14	47069	0	32	41	5	213	2	7
15	52183	0	36	14	7	118	20	14
16	55035	0	38	13	3	35	6	24
17	55881	0	38	48	9	49	13	22
18	57070	0	39	37	22	139	1	14
19	60407	0	41	56	23	124	14	15
20	63397	0	44	1	13	112	0	17
21	66085	0	45	53	13	100	1	14
22	68486	0	47	33	14	103	16	18
23	70974	0	49	17	6	116	15	21
24	73773	0	51	13	21	102	2	13
25	76223	0	52	55	23	29	9	11
26	76928	0	53	25	8	255	8	12
27	83056	0	57	40	16	197	23	15
28	87807	1	0	58	15	1	12	40
29	87843	1	1	0	3	9	22	27
30	88081	1	1	10	1	16	20	24
31	88485	1	1	26	21	87	16	23
32	90589	1	2	54	13	50	2	20
33	91791	1	3	44	15	86	14	19
34	93869	1	5	11	5	118	18	9
35	96719	1	7	9	23	138	4	16

Table 5.4: Second half of the table of segments extracted by the "Cinemetrics" tool from the movie "There Will Be Blood" (2007)

#	Frame	hours	minutes	seconds	frames	Sec	Fra	Motion (%)
36	100035	1	9	28	3	92	18	15
37	102261	1	11	0	21	136	17	17
38	105542	1	13	17	14	41	9	18
39	106535	1	13	58	23	362	4	15
40	115227	1	20	1	3	155	6	12
41	118953	1	22	36	9	129	20	14
42	122069	1	24	46	5	208	1	6
43	127062	1	28	14	6	168	20	11
44	131114	1	31	3	2	153	6	10
45	134792	1	33	36	8	166	9	15
46	138785	1	36	22	17	77	23	17
47	140656	1	37	40	16	40	15	22
48	141631	1	38	21	7	113	20	21
49	144363	1	40	15	3	130	5	15
50	147488	1	42	25	8	205	4	6
51	152412	1	45	50	12	217	1	7
52	157621	1	49	27	13	68	6	16
53	159259	1	50	35	19	74	16	16
54	161051	1	51	50	11	147	21	18
55	164600	1	54	18	8	376	17	19
56	173641	2	0	35	1	64	9	17
57	175186	2	1	39	10	189	18	15
58	179740	2	4	49	4	104	9	13
59	182245	2	6	33	13	81	17	13
60	184206	2	7	55	6	110	2	10
61	186848	2	9	45	8	73	4	7
62	188604	2	10	58	12	68	11	6
63	190247	2	12	6	23	82	0	8
64	192215	2	13	28	23	69	11	9
65	193882	2	14	38	10	212	21	15
66	198991	2	18	11	7	104	18	14
67	201505	2	19	56	1	160	22	11
68	205367	2	22	36	23	212	8	12
69	210463	2	26	9	7	192	14	15
70	215085	2	29	21	21	21	3	19
71	215592	2	29	43	0	227	10	7

Table 5.5: List of the first 15 "fades to black" detected in the movie "Inception" (2010)

#	Frame No.	Instant (min.)	sec.	fra.	Duration (sec.)	fra.
1	0	0	0	0	3	4
2	383	0	15	23	1	19
3	749	0	31	5	2	16
4	889	0	37	1	2	19
5	7128	4	57	0	2	19
6	9229	6	24	13	0	19
7	9249	6	25	9	0	1
8	9252	6	25	12	0	22
9	9584	6	39	8	0	2
10	9587	6	39	11	0	22
11	9610	6	40	10	0	1
12	9614	6	40	14	0	1
13	9616	6	40	16	0	1
14	9884	6	51	20	0	22
15	9907	6	52	19	2	4

Table 5.6: Some informational values annotated with the "Cinemetrics" tool for the "Pixar" short "Lou" (2017)

Frames	Frame rate	Fades	Duration	Proc. time	Proc. ratio
9685	24	8	7	45	6.43

Table 5.7: Other informational values annotated with the "Cinemetrics" tool for the "Pixar" short "Lou" (2017)

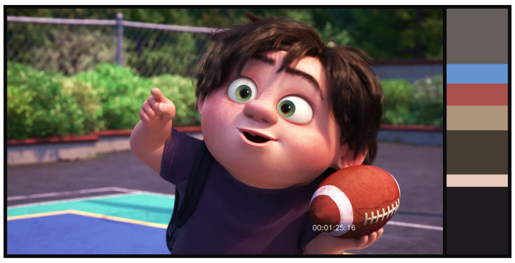
Cuts/ min.	ASL (sec)	MSL (sec)	Real	Detected	Efficiency (%)
19	3	2.3	134	250	54

Table 5.8: The number of false cuts annotated (lower row) and the instants where they occurred (upper row)

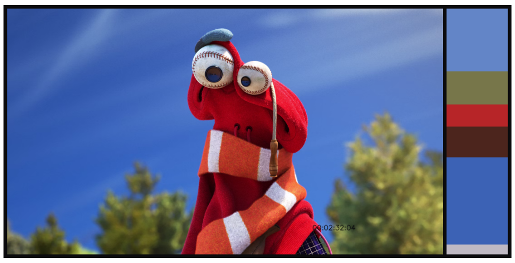
OC	36	45	46	47	49	50	51	54	58	61	63	64	65	68	88	EC
35	5	10	12	9	1	1	4	3	1	1	1	3	3	2	10	15

Table 5.9: List of "fades to black/ white" extracted by the "Cinemetrics" tool from the "Pixar" short film "Luo" (2017)

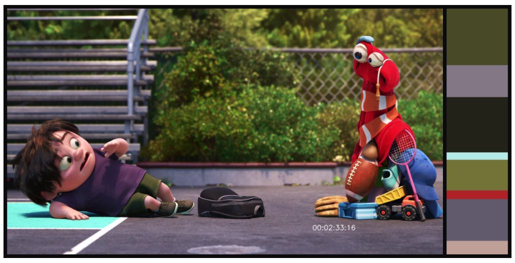
#	Frame	minutes	seconds	frames	Sec	Frs	B / W
1	0	0	0	0	1	2	Black
2	114	0	4	18	0	3	Black
3	128	0	5	8	2	9	White
4	206	0	8	14	1	9	Black
5	336	0	14	0	1	7	Black
6	6083	4	13	11	0	13	White
7	8766	6	5	6	2	10	Black
8	8924	6	11	20	0	1	Black



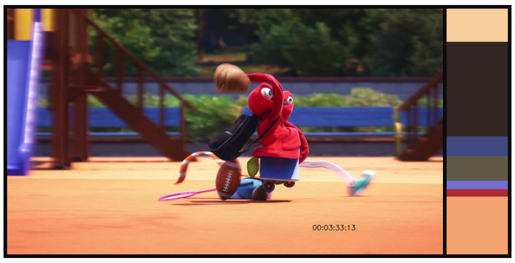
(a)



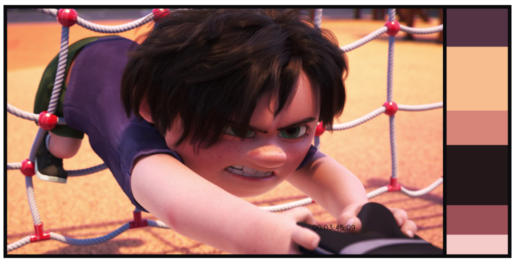
(b)



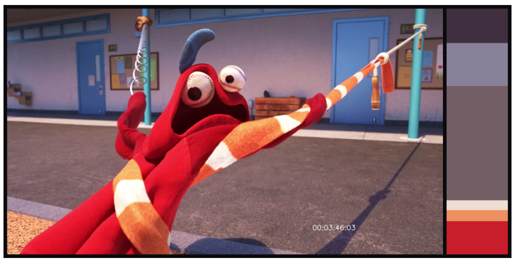
(c)



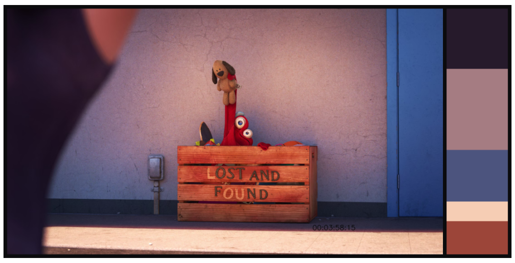
(d)



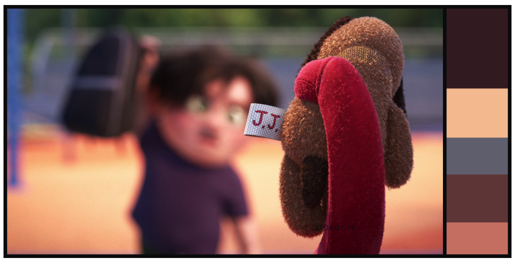
(e)



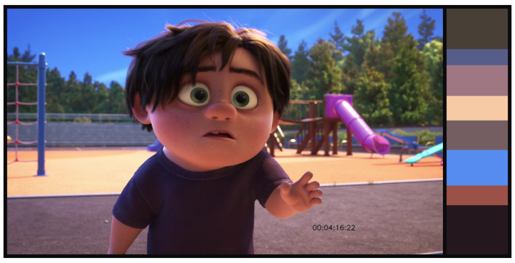
(f)



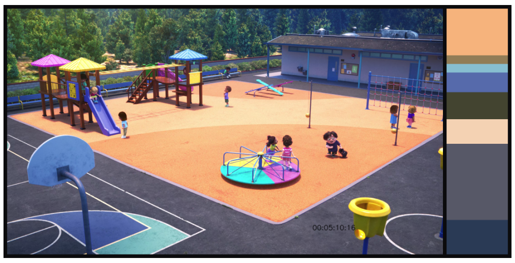
(g)



(h)



(i)



(j)

Figure 5.23: 10 main shots chosen by the user from the "Pixar" short "Lou" (2017)

Chapter 6

Conclusions and Future work

Even though this thesis can be seen as somehow extensive due to the quantity of information that needed to be transmitted to better contextualize the project, it has a rich, expressive and visual component revolving around the theme of data visualization, which makes it much more interesting. The development of this project involved a lot of research in this field and also in computer vision.

The first big breakthrough of this project was being able to convert the outdated script to the most recent version of "OpenCV" ("cv2"). The best part about implementing code in Python or other programming language is the problem solving, since the development or improvement of algorithms require a sense of autonomy, patience, and a greater attention to details. To get to the root of each problem encountered there was a lot of code reviewing, and trial and error (debuging). This means that for a programmer it is important to be methodical, to organize the thoughts and ideas in the best and most optimal way.

Since this project is based on generating outputs, it is essential to embrace the errors and flaws that one might encounter in the discussion of results, because to learn and improve the way of thinking towards any subject, it is crucial to detect where the flaws are, why they exist, what is their cause. The subject of creating new appealing visual representations can be sometimes too abstract in a way that in most cases there are no definite answers to many of the problems raised in this project.

The cinematographic industry and multimedia streaming platforms are getting more relevant and innovative, as this subject and technology are getting closer and more connected.

It is imperative for society to keep up with technology, and with this being said, data designers and software engineers have to start thinking about how they can already automate processes for future multimedia technologies, the same way that they are now trying to adapt old formats by transferring and handling data in databases and platforms.

The following list refers to possible improvements that could be made in this project; more concretely, features that could be added to the "Cinemetrics" tool:

- **Face recognition** – One of the best possible improvements for this tool would be to implement a feature of face recognition, with the most obvious drawback being the fact that it would prolong the estimated time of video processing even more. To prevent this from happening, an idea would be to just apply this function to a smaller set of frames; for example, to the first frame of each segment or shot, depending on the duration of the movie.

  This feature could compare the frames to a database with portrait photos of famous actors one by one, but the biggest disadvantage of this method would be the overload of data that would have to be stored and processed.

  So the best option would be to somehow try to gather a concrete number of frames for each main character (e.g. 5 maximum characters with 5 frames for each one) and this can be easily done with an interaction which, similar to the "Main shot" checkbox, would ask the user in the GUI if they wanted to select that frame as a character identifier.

  If hypothetically, this feature asked the user for the name of the character and actor or actress each time a frame was selected, this feature wouldn't be a proper face recognition because there wouldn't be algorithms or any type of processing or computing involved. So instead, it would be more convenient and less dull for the user to just be required for a large enough amount of frames with characters in it (preferably portraits).

  With this set of pictures it would be possible to categorize them, or in simpler words, to group and associate photos for the different characters. This could be achieved with the use of specific algorithms, neural networks (maybe a CNN) or another type of deep learning model. The convenience of this process is that it would be done only in the end of the code, after going through all the frames of the movie, and even if it took too much time, it would be easy to just pass this function onto another script, since the collection of photos could be saved considering that there wouldn't be many (maximum of 25 pictures according to the example previously mentioned).

  The missing piece here would be how to assign each group into the character denomination. This problem is easy to solve because even though the tool wouldn't be able to name the character or the actor by itself, it can give them a label like a letter or a number (e.g. character 1, character 2, etc.) and then later, in the website for example, any user who came across that "fingerprint" could name them in order to complete it.

- **Optimize tool to reduce the processing time** – The biggest flaw that this tool has, and that has been mentioned many times in this thesis, is the extensive estimated time of movie processing; it can be two or three times longer than the movie duration, depending on which mode is used (automatic or interactive). This flaw should definitely be improved since it takes a lot of patience and attention for the user to operate the tool especially when using the interactive mode.

  To achieve this goal there would need to be a deeper and centralized study in the processing power and average times for each Python and "OpenCV" function used in this script. Or to use a computer with better hardware such as the CPU.

- **Optimize the threshold** – The value of the threshold for boundary scene detection could be optimized with experiments by testing many different movies to change it to a more specific value.

- **Audio feature** – This tool would be even more complete with a feature that could extract the audio and display it visually with the use of frequency graphs or other visualizations that describe loudness, musics, audio moods. One concrete example would be to simply overlap a frequency graph on any of the outputs like the colour scheme, the slit sequences, or the horizontal and vertical prints. But this would have to be adjusted or adapted in order to be visible at naked eye, since, considering the size of a movie, there would be the need to adjust its scale or to select smaller portions.

- **Diagonal print** – As another visual output, besides the vertical and horizontal prints, it would be interesting to compare how the prints would look by extracting both diagonals of the frames. In "OpenCV", there is not a direct function to extract a diagonal line of an image. The closest thing would be to rotate the image a certain number of degrees, depending on the aspect ratio of the movie, and then to extract the vertical or horizontal line normally.

- **Animated motion feature** – The last idea for future improvement of this tool is to display the values of motion for each shot/ segment in an animated form. This idea is based on the movement portrayed in the "fingerprints" of the *Frederic Brodbeck*'s "Cinemetrics" application, where each coloured segment moves towards the inside and the outside of the circle. The same could be applied to the colour palette or to other visual outputs by moving each section alternately, where one would go up, while at the same time, the next section would go down, and so on. These sections would move at different speeds according to the percentage of motion (considering that these values are given in the shots table), where the maximum speed would be around 50% (since the cut detection threshold is 0.48), and 1 would be a speed almost imperceptible to the naked eye. If by any chance there were shots with a value 0 of motion, then the section wouldn't move at all.

This could be implemented with "Chart.js", a data visualization tool already mentioned in chapter 2, which allows the user to create animated visualizations; or "Chartist.js" which is a similar tool [16].

Another option would be to generate a Graphics Interchange Format (GIF) file, but for it to be fluid, without sections jumping abruptly up or downwards, the file would be too large and with a long duration, considering that to make all the sections cross the middle point at the same time, the least common multiple would need to be calculated between a big sample of different values of motion (one for each section).

The best option would probably be to use the CSS language in a website like the prototype described in the appendix A, to animate the visualizations that could be displayed with a click on a checkbox.

References

[1] Tableau, "What is data visualization? definition, examples, and learning resources." Available at `https://www.tableau.com/learn/articles/data-visualization`. [Cited on pages 5 and 17]

[2] H.-C. Hege, "Data visualization: foundations, techniques, and applications," in *HACIA LA TRANSFORMACIÓN DIGITAL. Actas del I Congreso Internacional de Ingeniería de Sistemas*, (Universidade de Lima, Peru), pp. 27–34, Sept. 2018. [Cited on page 5]

[3] M. Sridharan, "Data visualization: History and origins." Available at `https://thinkinsights.net/digital/data-visualization-history/`, Jan. 2017. (Last accessed in 12/10/2021). [Cited on page 5]

[4] A. Barletta, B. Moser, and M. Mayer, "Visualization and control techniques for multimedia digital content," U.S. Patent 8 209 623, Jun. 2012. [Cited on pages ix, 6, and 7]

[5] W. Zhu, P. Cui, Z. Wang, and G. Hua, "Multimedia big data computing," *IEEE MultiMedia*, vol. 22, no. 3, pp. 96–c3, 2015. [Cited on pages 7 and 8]

[6] J. Bulao, "How much data is created every day in 2021?." Available at `https://techjury.net/blog/how-much-data-is-created-every-day/`, Oct. 2021. (Last accessed in 18/10/2021). [Cited on page 7]

[7] M. Dhande, "What is the difference between ai, machine learning and deep learning?." Available at `https://www.geospatialworld.net/blogs/`, Mar. 2020. [Cited on pages ix and 8]

[8] J. Luo, D. Joshi, J. Yu, and A. Gallagher, "Geotagging in multimedia and computer vision—a survey," *Multimedia Tools and Applications*, vol. 51, p. 187–211, Jan. 2011. [Cited on page 8]

[9] K. T. Y. Leung, A. Dembicz, and A. Stevenson, "Exploring ethical dilemma in big data analytics: A literature review," *Technium: Romanian Journal of Applied Sciences and Technology*, vol. 2, p. 43–48, Jun. 2020. [Cited on page 9]

[10] Tableau, "Data is beautiful: 10 of the best data visualization examples from history and today." Available at `https://www.tableau.com/learn/articles/`

best-beautiful-data-visualization-examples/. [Cited on pages ix, 9, 10, 11, 12, and 16]

[11] G. Bhatia, P. K. Dutta, C. Canipe, and J. McClure, “Covid-19 vaccination tracker.” Available at https://graphics.reuters.com/world-coronavirus-tracker-and-maps/vaccination-rollout-and-access/. (Last accessed in 21/10/2021). [Cited on pages ix and 13]

[12] A. Glivinska, “The 25 best data visualizations of 2020 [examples].” Available at https://visme.co/blog/best-data-visualizations/, Aug. 2021. [Cited on page 12]

[13] M. Bonera, S. Perozzi, F. Fracascio, G. Zerbini, D. Berto, F. Pontiroli, G. Bertolotti, and X. Nguyen, “Codex atlanticus.” Available at https://codex-atlanticus.it/. [Cited on pages ix and 13]

[14] O. Velarde, “The 25 best data visualizations of 2019.” Available at https://visme.co/blog/best-data-visualizations-2019/, Jan. 2020. [Cited on page 14]

[15] L. Manovich, M. Stefaner, M. Yazdani, D. Baur, D. Goddemeyer, A. Tifentale, N. Hochman, and J. Chow, “Selfiecity.” Available at https://www.selfiecity.net/, 2014. [Cited on pages 15 and 16]

[16] C. Chapman, “A complete overview of the best data visualization tools.” Available at https://www.toptal.com/designers/data-visualization/data-visualization-tools. [Cited on pages 16, 17, and 108]

[17] B. Marr, “The 7 best data visualization tools available today.” Available at https://www.forbes.com/sites/bernardmarr/2017/07/20/the-7-best-data-visualization-tools-in-2017/, July 2017. [Cited on page 17]

[18] O. Velarde, “Top 9 data visualization tools for 2021.” Available at https://visme.co/blog/data-visualization-tools/, Sept. 2020. [Cited on page 17]

[19] Harkiran, “10 best data visualization tools in 2020.” Available at https://www.geeksforgeeks.org/10-best-data-visualization-tools-in-2020/, July 2021. (Last accessed in 18/10/2021). [Cited on page 17]

[20] T. Public, “About - what is tableau public?.” Available at https://public.tableau.com/en-us/s/about. [Cited on page 17]

[21] D. D. Livera and G. Shah, “What are you watching on netflix?.” Available at https://public.tableau.com/app/profile/dinushki.de.livera/

viz/04_30_2021FinalNetflixCollabViz/NetflixViz, Apr. 2021. [Cited on page 18]

[22] K. Scott, "The movies of the decade." Available at https://public.tableau.com/app/profile/kimly.scott/viz/Themoviesofthedecade/MoviesoftheDecade, Jan. 2020. [Cited on page 18]

[23] Amazon.com, "Internet movie database (imdb)." Available at https://www.imdb.com/. [Cited on pages 18, 30, 40, and 79]

[24] R. Sleeper, "Blockbuster." Available at https://public.tableau.com/views/BLOCKBUSTER/BLOCKBUSTER?:showVizHome=no. [Cited on page 18]

[25] K. Flerlage, "Movie money." Available at https://public.tableau.com/app/profile/kevin.flerlage/viz/MovieMoney_15813400492150/MovieMoney, Feb. 2020. [Cited on pages ix, 18, and 19]

[26] G. Goldestan, "Disney movies income (1937-2016)." Available at https://public.tableau.com/app/profile/gandes.goldestan/viz/DisneyMovieIncome1937-2016DataDNAChallenge/Dashboard, May 2021. [Cited on pages 18 and 19]

[27] IBM, "What is computer vision?." Available at https://www.ibm.com/topics/computer-vision. [Cited on pages 20, 21, 22, and 23]

[28] SAS, "Computer vision." Available at https://www.sas.com/en_us/insights/analytics/computer-vision.html. [Cited on pages 20 and 21]

[29] B. Marr, "7 amazing examples of computer and machine vision in practice." Available at https://www.forbes.com/sites/bernardmarr/2019/04/08/7-amazing-examples-of-computer-and-machine-vision-in-practice/, Apr. 2019. [Cited on page 20]

[30] G. Boesch, "The 12 most popular computer vision tools in 2021." Available at https://viso.ai/computer-vision/the-most-popular-computer-vision-tools/, June 2021. [Cited on pages 21 and 22]

[31] J.-R. Jiang, J.-E. Lee, and Y.-M. Zeng, "Time series multiple channel convolutional neural network with attention-based long short-term memory for predicting bearing remaining useful life," *Sensors*, vol. 20, no. 1, 2020. [Cited on pages ix and 23]

[32] IBM, "What is overfitting?." Available at https://www.ibm.com/cloud/learn/overfitting. [Cited on page 24]

[33] K. Nassau, "Colour." Available at https://www.britannica.com/science/color, July 1999. [Cited on pages 24 and 25]

[34] I. D. Foundation, "Color theory." Available at https://www.interaction-design.org/literature/topics/color-theory. [Cited on pages 24 and 25]

[35] T. Scully, "What is rgb lighting? top 5 rgb led strips and lights." Available at https://www.ledsupply.com/blog/rgb-lighting-guide-to-the-top-5-rgb-led-strips-lights/, Mar. 2021. [Cited on page 25]

[36] Miki, "Hsl color model decomposition in blender." Available at https://meshlogic.github.io/posts/blender/materials/nodes-hsl-color-model/, Aug. 2016. [Cited on page 25]

[37] G. Smith, "Color palettes from famous movies show how colors set the mood of a film." Available at https://digitalsynopsis.com/design/cinema-palettes-famous-movie-colors/, July 2015. [Cited on pages 26 and 27]

[38] R. Radulescu, "Movies in color." Available at https://moviesincolor.com/, Feb. 2013. [Cited on page 27]

[39] H. BUNDLE, "Video file resolution information." Available at https://support.humblebundle.com/hc/en-us/articles/205166827-Video-File-Resolution-Information. [Cited on pages ix and 28]

[40] T. Labs, "Color extraction." Available at https://labs.tineye.com/color/. [Cited on pages ix and 29]

[41] TinEye, "About." Available at https://tineye.com/about. [Cited on page 29]

[42] R. Lewis, "Imdb." Available at https://www.britannica.com/topic/IMDb, Oct. 2017. [Cited on page 30]

[43] Fandango, "Rotten tomatoes." Available at https://www.rottentomatoes.com/. [Cited on page 30]

[44] R. Ventures, "Metacritic." Available at https://www.metacritic.com/. [Cited on page 30]

[45] Letterboxd, "Letterboxd." Available at https://letterboxd.com/. [Cited on pages 30 and 31]

[46] Y. Tsivian, "Cinemetrics." Available at http://www.cinemetrics.lv/, 2005. [Cited on pages 33, 40, 43, 79, and 123]

[47] T. U. of Chicago, "Yuri tsivian." Available at https://arthistory.uchicago.edu/faculty/profiles/tsivian. [Cited on page 33]

[48] A. Wermer-Colan, "Digital video and image analytics." Available at https://guides.temple.edu/video-image-analysis-and-visualization, Mar. 2019. [Cited on page 34]

[49] D. Brunner, "Frame rate: A beginner's guide." Available at https://www.techsmith.com/blog/frame-rate-beginners-guide/, Mar. 2017. [Cited on page 34]

[50] B. Salt, "Starword - data method: Statistical style analysis." Available at http://www.starword.com/Data_Method/data_method.html. [Cited on pages ix, 35, 37, and 38]

[51] S. Allen, "Cinemetrics quantifies creativity in film." Available at https://news.uchicago.edu/story/cinemetrics-quantifies-creativity-film, May 2015. [Cited on pages 35 and 40]

[52] M. staff, "Film 101: What is a shot list? how to format and create a shot list." Available at https://www.masterclass.com/articles/film-101-what-is-a-shot-list-how-to-format-and-create-a-shot-list, Aug. 2021. [Cited on page 35]

[53] B. A. Halim and T. Faiza, "Shot boundary detection: Fundamental concepts and survey," in *CITSC*, 2019. [Cited on pages 35 and 36]

[54] S. Emory, "What does it mean when a film fades to white?." Available at https://www.vice.com/en/article/z4q3a9/death-or-glory-what-does-it-mean-when-a-film-fades-to-white, Apr. 2015. [Cited on page 36]

[55] T. tropes, "Fade to white." Available at https://tvtropes.org/pmwiki/pmwiki.php/Main/FadeToWhite. [Cited on pages 36 and 37]

[56] S. Follows, "How many shots are in the average movie?." Available at https://stephenfollows.com/many-shots-average-movie/, July 2017. [Cited on pages ix, 40, 41, and 42]

[57] B. W. Cogley, "Cut - a cinemetrics viz." Available at https://public.tableau.com/app/profile/bridget/viz/Cinemetrics/Home, Aug. 2017. [Cited on page 40]

[58] P. Etemesi, "The best movies that were made to look like a single take (including 1917), ranked." Available at https://screenrant.com/1917-best-movies-single-take/, Jan. 2021. [Cited on pages 40 and 41]

[59] A. Wilkinson, “Why sam mendes made 1917 look like it was shot in a single, continuous take.” Available at `https://www.vox.com/culture/2019/12/30/21021190/1917-movie-sam-mendes-interview-one-shot`, Jan. 2020. [Cited on page 40]

[60] G. Savchenko, “Hidden cuts in birdman shown in one video.” Available at `https://birdinflight.com/news/industry/20170406-birdman-hidden-cuts.html`, Apr. 2017. [Cited on page 41]

[61] C. Hooton, “Taken 3 took 14 camera cuts in 6 seconds to show liam neeson jumping a fence.” Available at `https://www.independent.co.uk/arts-entertainment/films/news/taken-3-took-14-camera-cuts-in-6-seconds-to-show-liam-neeson/jumping-a-fence-a6890791.html`, Feb. 2016. [Cited on pages 42 and 78]

[62] Z. Sharf, “‘there will be blood’: What you learn about paul thomas anderson by counting all 678 shots — watch.” Available at `https://www.indiewire.com/2017/07/there-will-be-blood-editing-678-shots-cuts-video-1201861014/`, July 2017. [Cited on pages 42, 43, and 79]

[63] J. Butler, “Shot logger.” Available at `http://www.shotlogger.org/`, Aug. 2007. [Cited on pages 43 and 123]

[64] M. Svanera, M. Savardi, A. Signoroni, A. B. Kovács, and S. Benini, “Who is the film’s director? authorship recognition based on shot features,” *IEEE MultiMedia*, vol. 26, no. 4, pp. 43–54, 2019. [Cited on pages ix, 43, and 44]

[65] S. Benini, M. Svanera, N. Adami, R. Leonardi, and A. B. Kovács, “Shot scale distribution in art films,” *MultiMedia Tools and Applications*, vol. 75, p. 16499–16527, 2016. [Cited on pages ix, 44, and 45]

[66] G. Halter, R. Ballester-Ripoll, B. Flueckiger, and R. Pajarola, “Vian: A visual annotation tool for film analysis,” *Computer Graphics Forum*, vol. 38, pp. 119–129, 06 2019. [Cited on pages ix, 44, 45, and 46]

[67] F. Brodbeck, “Cinemetrics.” Available at `http://cinemetrics.fredericbrodbeck.de/`, July 2011. [Cited on pages ix, x, 46, 47, 48, 50, and 51]

[68] F. Brodbeck, “freder - cinemetrics.” Available at `https://github.com/freder/cinemetrics/`, Nov. 2011. [Cited on pages 49, 51, and 73]

[69] B. Dawes, “Cinema redux.” Available at `https://brendandawes.com/projects/cinemaredux`, 2004. [Cited on pages x, 52, and 53]

[70] "Moviebarcode." Available at https://moviebarcode.tumblr.com/. [Cited on pages x, 52, and 54]

[71] C. Clark, "The colors of motion." Available at https://thecolorsofmotion.com/. [Cited on pages x, 52, 53, and 55]

[72] OFFC, "'sensing 'the shining'." Available at http://www.offc.co/web/sensingtheshining.php. [Cited on pages x, 53, and 56]

[73] OpenCV, "Histograms - 1 : Find, plot, analyze !!!." Available at https://docs.opencv.org/master/d1/db7/tutorial_py_histogram_begins.html. [Cited on page 58]

[74] OpenCV, "Histograms - 3 : 2d histograms." Available at https://docs.opencv.org/master/dd/d0d/tutorial_py_2d_histogram.html. [Cited on page 58]

[75] A. Rosebrock, "Opencv and python k-means color clustering." Available at https://www.pyimagesearch.com/2014/05/26/opencv-python-k-means-color-clustering/, May 2014. [Cited on pages xv, 65, and 66]

[76] B. Kumar, "Python tkinter checkbutton – how to use." Available at https://pythonguides.com/python-tkinter-checkbutton/, Dec. 2020. [Cited on page 69]

[77] M. Nasrin, "12 angry men (1957, united states) directed by: Sidney lumet." Available at http://www.cinemetrics.lv/movie.php?movie_ID=13734, May 2013. [Cited on page 79]

[78] KFA, "Enter the void (2009, united states) directed by: Gaspar noe." Available at http://www.cinemetrics.lv/movie.php?movie_ID=26883, May 2020. [Cited on page 79]

[79] D. Boterbergh, "Inception (2010, usa) directed by: Christopher nolan." Available at http://www.cinemetrics.lv/movie.php?movie_ID=7742, Mar. 2011. [Cited on page 79]

[80] R. D. Kokes, "Interstellar (2014, united states) directed by: Christopher nolan." Available at http://www.cinemetrics.lv/movie.php?movie_ID=18543, July 2015. [Cited on page 79]

[81] H. Carlsen, "Spider-man: Homecoming (2017, united states) directed by: Jon watts." Available at http://www.cinemetrics.lv/movie.php?movie_ID=27475, Apr. 2021. [Cited on page 79]

[82] R. Zahed, "Eight teams and 1,190 shots go into 'blade runner 2049's' visual effects." Available at `https://www.latimes.com/entertainment/envelope/la-en-mn-crafts-blade-runner-fx-20180220-story.html`, Feb. 2018. [Cited on page 79]

[83] B. Frazer, "Interstellar: First reactions and tech specs for exhibition." Available at `https://www.studiodaily.com/2014/10/interstellar-first-reactions-and-tech-specs-for-exhibition/`, Oct. 2014. [Cited on page 85]

[84] F. B.V., "Framer." Available at `https://www.framer.com/`. [Cited on pages xi, 120, and 122]

Appendix A

Website Prototype

A.1 Django Web Application

One of the initial ideas of this project was to build a web application with the "Cinemetrics" tool embedded in it. The first step was to choose the most suitable web framework of Python for the tool and for movie data navigation. It came down to the two most famous web frameworks for the Python programming language, "Django" and "Flask".

The chosen option would end up being "Django" since it seems to be a better fit due to being a high-level framework, which is more complex compared to "Flask", but it has more features, functions and a better online support overall. Besides that, it is also the most used web framework for Python, which makes it more interesting to learn and more probably to one day be useful for other web applications.

The front-end of this application would be a platform, a repository, or a forum to share the generated outputs from the tool and other important information (e.g. year, director, genres, cast, characters, movie cover, etc.) from movies. This supplementary information could be extracted with a script directly from a movie database like "IMDb".

The database would have all this data about movies where any person, besides the users, would be able to navigate through the website's content by looking into any movie individually or by comparing two (or more) movies and their visual and informative outputs, side by side. They could also use a search engine to look for specific movies by filtering or ordering the database by year, first letter, genre or any other information, like other online movie platforms do.

The steps to run the "Django" server, and to load and edit the project's webpages are the following:

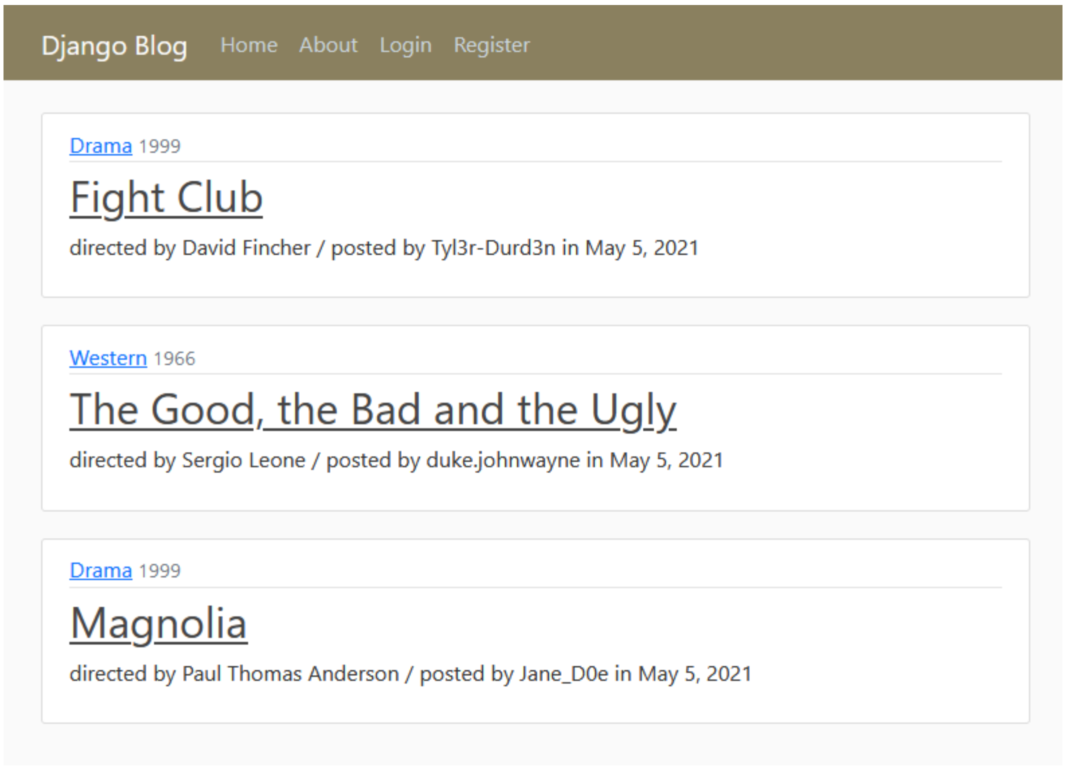
1. Open command prompt;
2. Move to the directory named `"Scripts"` inside the created directory for the "Django" environment;
3. Execute the file `"activate.bat"` to activate the environment
4. Move to the source code directory (`"src"`);
5. Open the "Visual Studio Code" code editor to manage and edit all the files from the source code (Python, HTML, and CSS files) by typing `"code ."`;
6. Run the command line `"py manage.py runserver"`;
7. Open a web browser and type in the "Django" project Internet Protocol (IP) address (`"http://127.0.0.1:8000/"` or `"http://localhost:8000/"`);
8. Press `"Ctrl+C"` to quit the server at anytime.

After any significant changes, some commands of migrations need to be made in order to update the Structured Query Language (SQL) database.

The figure A.2 shows an auxiliary page named `"admin"` that comes with any new project of "Django" by default, and it helps to directly manage many things from the website such as users and any content, like the movies in this case. The section of the administration page displayed is the one where the administrator can add a movie manually by writing the title of the movie and the name of the director, and by choosing the year when the movie was released, and the genres. Every movie can have more than one genre and 24 different genres were defined in the file `"models.py"` with the model and form field named `"django-multiselectfield"`. This field allows users to choose multiple options (choices) with checkboxes. The choices marked are stored to the database as "CharField" of comma-separated values.

This idea was afterwards discarded due to the fact that the "OpenCV" library for python has some constraints and complications to integrate with the "Django" framework or with any other Python web framework.

Therefore, it wouldn't be so relevant to create the website since the main goal of the project was to build the tool and generate outputs with it. And, in this case, the website would be just a platform to submit these outputs in a submission form for the files (shots table, information table, "fades to black/ white" table, and the



Drama 1999

Fight Club

directed by David Fincher / posted by Tyl3r-Durd3n in May 5, 2021

Western 1966

The Good, the Bad and the Ugly

directed by Sergio Leone / posted by duke.johnwayne in May 5, 2021

Drama 1999

Magnolia

directed by Paul Thomas Anderson / posted by Jane_D0e in May 5, 2021

Figure A.1: Screenshot of the main page from the unfinished "Django" project

visual representations of the movie), making it a very separated and distinct thing from the tool.

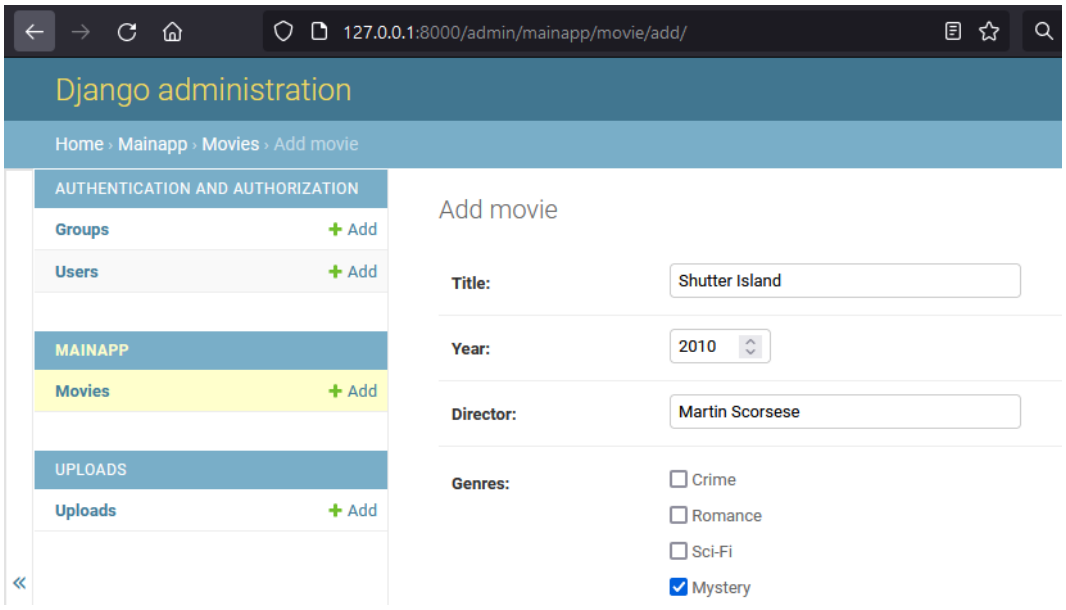
With this being said, the next section will just show a visual example and prototype of a webpage with some suggestions.

A.2 Webpage Concept Suggestion

Even though the development of this website was never concluded, the main concepts and ideas to build it, will be described in this section.

One of the first suggestions would be to have a "movie fingerprint completion" indicated by the three examples in figure A.3. This concept is about indicating the level of each submission made by users. In other words, there would be the lowest level (1/3 - blue fingerprint) where any user could just give information and metadata about a movie from other movie databases.

The middle level (2/3 - red fingerprint) would be the outputs generated with the "Cinemetrics" tool in the automatic mode. And the highest level (3/3 - green fingerprint), which is supposed to be the most complete and precise representation of any movie, consists of the outputs generated from the interactive mode, where the number of shots and all the other values are more precise and can be seen as


127.0.0.1:8000/admin/mainapp/movie/add/
Django administration
Home › Mainapp › Movies › Add movie
AUTHENTICATION AND AUTHORIZATION
Groups + Add
Users + Add
MAINAPP
Movies + Add
UPLOADS
Uploads + Add
«

Add movie

Title:	Shutter Island
Year:	2010
Director:	Martin Scorsese
Genres:	☐ Crime ☐ Romance ☐ Sci-Fi ☑ Mystery

Figure A.2: Screenshot of the built-in administration page from the unfinished "Django" project

GT data. Besides that, the highest level also has the main shots, as it is illustrated in figure A.4.

The different subsections or webpages of a movie correspond to the different outputs generated from the tool, while in the lowest level, there is only the section of information. This section could also show the ratings from different websites to give a feedback about the movie.

Besides these ratings, the users are also able to give reviews (rating and/or commentaries) to the generated visualizations themselves, like the scale of five stars displayed in figure A.4, to tell how good these visual and graphical representations actually represent the movie, or to evaluate them aesthetic wise.

Depending on the level of a movie, any user could complete a "fingerprint" or make it go up a level by using the tool (depending on the mode). For a user to add a movie to the database from zero, they have to, at least, submit the information on a form for the lowest level.

The prototype in figure A.4 was made with "Framer" [84], a free designing tool for applications, websites or any other type of visual interface.

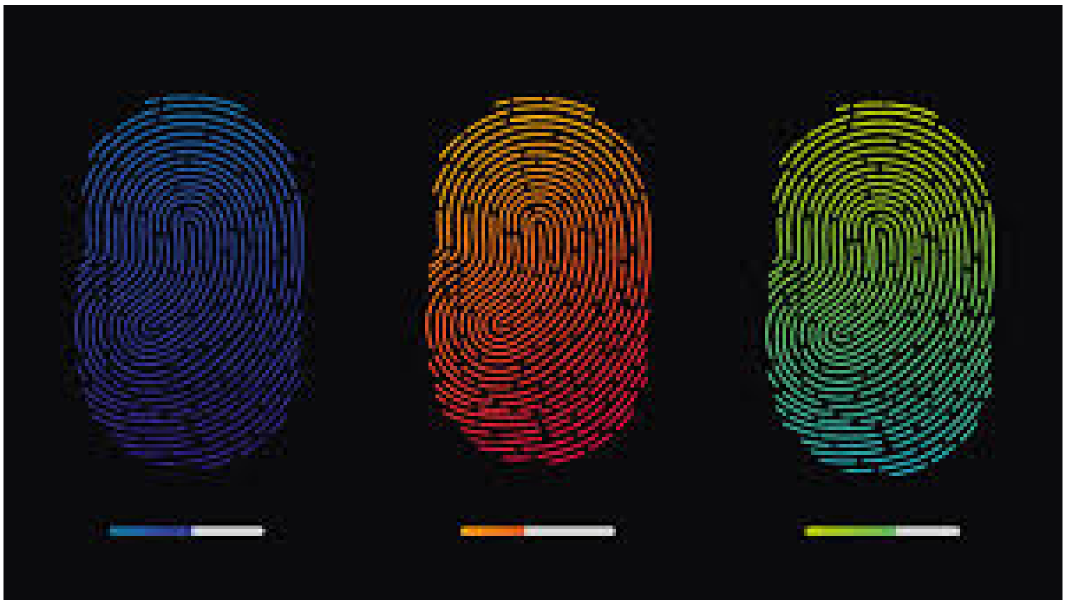
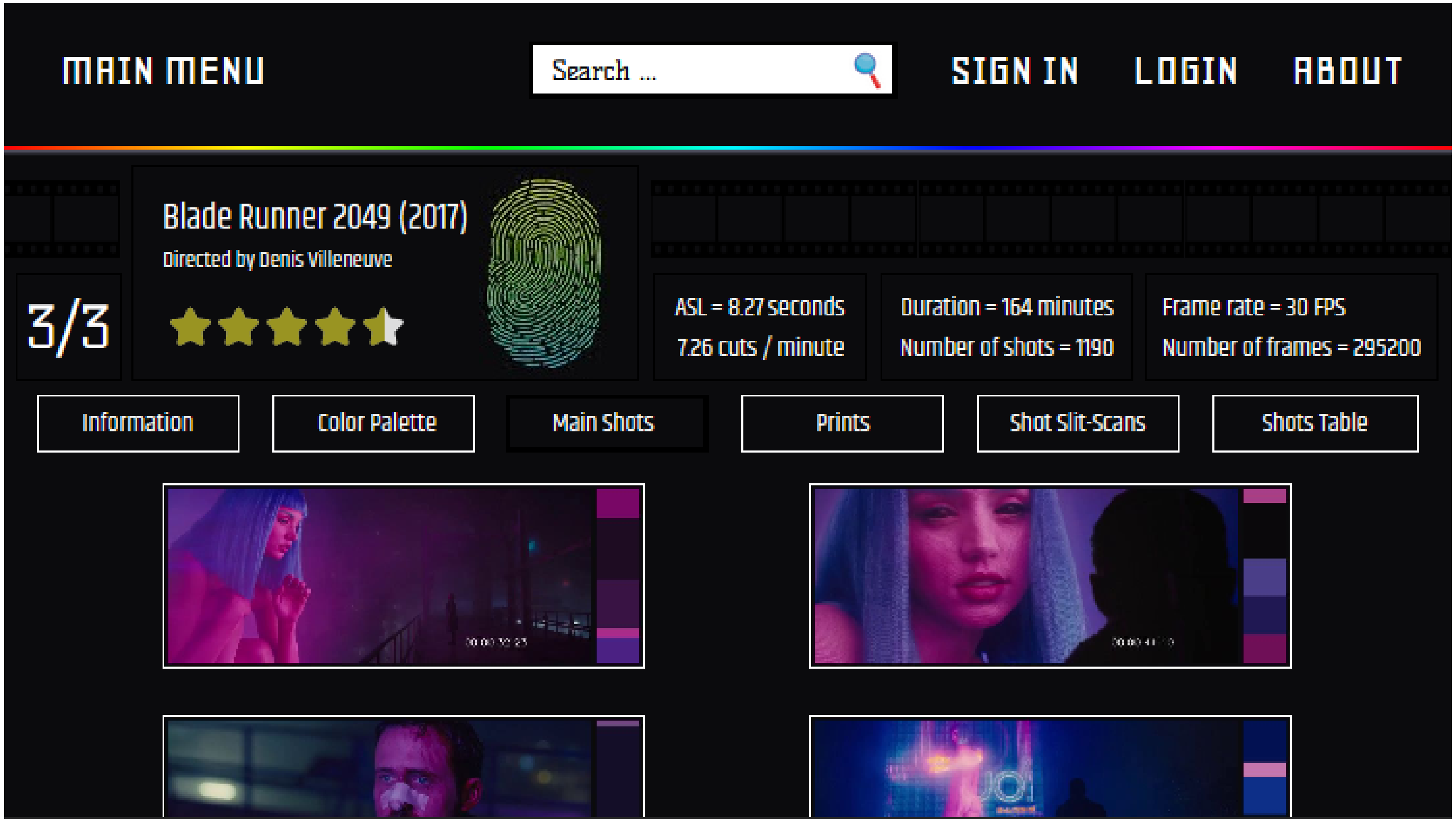


Figure A.3: Images of "fingerprints" to indicate movie completion



Blade Runner 2049 (2017)
Directed by Denis Villeneuve

ASL = 8.27 seconds
7.26 cuts / minute

Duration = 164 minutes
Number of shots = 1190

Frame rate = 30 FPS
Number of frames = 295200

Figure A.4: Prototype of a webpage made to display the outputs generated from a movie with the "Cinemetrics" tool [84]

There could be also a feature that would compare the values of the movies such as the ASL, MSL, and number of shots, to others from references or databases like the one from the "Cinemetrics" website [46], the one from "Shot Logger" [63], or the one from Barry Salt, to see how efficient the tool (automatic mode) or the users' accuracy (interactive mode) is.

Here's a list of keywords that could help name the website: Film; Movie; Shot; Cut; Print; Reel; Motion; Colour; Chroma; Hue; Fade to Black; Cinemetrics; Cinema; Theater; Spool; Roll; Projector; Reveal; Database; Vision; Visualization; Audiovisual.

Background colour:

- RGB: (11, 11, 13)
- Hex: `#0B0B0D`

This was chosen to be used as the main colour for all types of interface related to the website, because it is the colour presented in the background of the "fingerprints" image, and also in the contours or outlines of the main shots generated with the tool.

Appendix B

Interactive Mode

B.1 Flowchart

The flowchart in figure B.1 represents the logical order that the script (version 2.0) of the "Cinemetrics" tool, follows to decode each frame and to detect cuts with the acknowledgement and interaction with the user.

B.2 Shots Table from Lou (Pixar Short)

The tables B.1, B.2 and B.3, extracted with the "Cinemetrics" tool from the "Pixar" short film named "Lou" (2017), are displayed in this appendix for being too extensive and tiresome to be represented in any middle part of the thesis, and also for being another example of how these table of shots are usually structured.

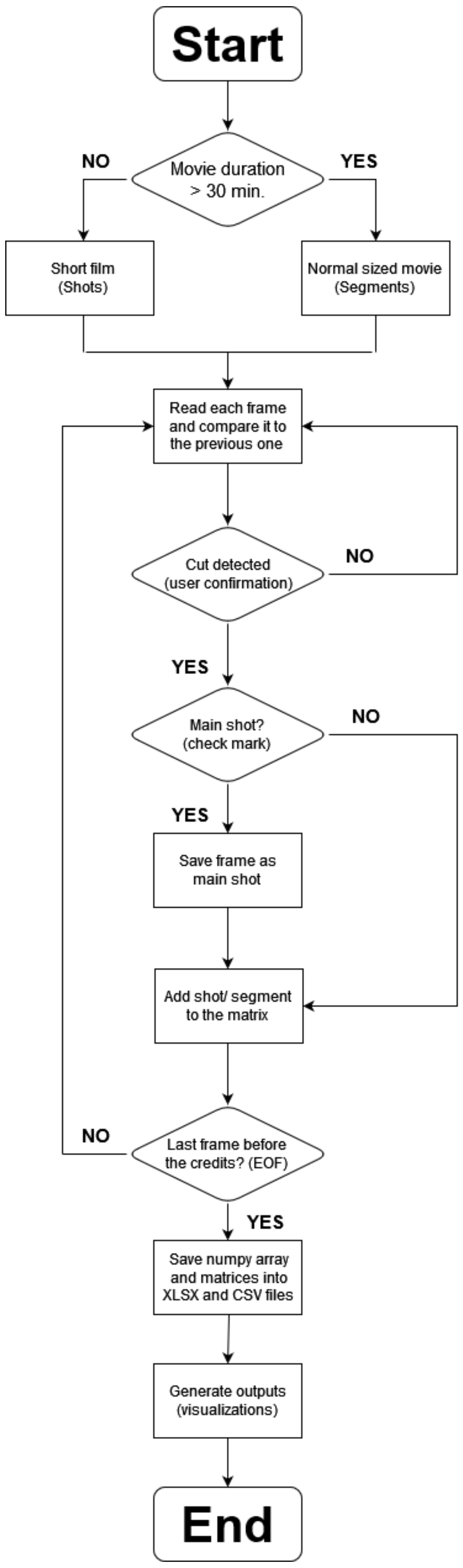

Start
Movie duration > 30 min.
NO
YES
Short film (Shots)
Normal sized movie (Segments)
Read each frame and compare it to the previous one
Cut detected (user confirmation)
NO
YES
Main shot? (check mark)
NO
YES
Save frame as main shot
Add shot/ segment to the matrix
Last frame before the credits? (EOF)
NO
YES
Save numpy array and matrices into XLSX and CSV files
Generate outputs (visualizations)
End

Figure B.1: Flowchart describing the interactive mode of the "Cinemetrics" tool

Table B.1: First third of the table of shots extracted by the "Cinemetrics" tool from the "Pixar" short film "Lou" (2017)

#	Frame	hours	minutes	seconds	frames	Sec	Frs	Motion (%)
0	0	0	0	0	0	19	1	14
1	457	0	0	19	1	1	17	20
2	498	0	0	20	18	2	9	20
3	555	0	0	23	3	2	1	16
4	604	0	0	25	4	2	2	7
5	654	0	0	27	6	2	4	11
6	706	0	0	29	10	3	2	12
7	780	0	0	32	12	1	13	8
8	817	0	0	34	1	1	19	13
9	860	0	0	35	20	4	7	8
10	963	0	0	40	3	3	0	11
11	1035	0	0	43	3	2	5	11
12	1088	0	0	45	8	7	7	14
13	1263	0	0	52	15	1	13	8
14	1300	0	0	54	4	2	7	12
15	1355	0	0	56	11	5	8	15
16	1483	0	1	1	19	13	18	10
17	1813	0	1	15	13	2	7	22
18	1868	0	1	17	20	1	11	11
19	1903	0	1	19	7	2	9	20
20	1960	0	1	21	16	2	22	22
21	2030	0	1	24	14	1	2	32
22	2056	0	1	25	16	0	15	25
23	2071	0	1	26	7	1	19	32
24	2114	0	1	28	2	2	22	18
25	2184	0	1	31	0	1	5	14
26	2213	0	1	32	5	1	16	20
27	2253	0	1	33	21	6	17	20
28	2414	0	1	40	14	1	19	12
29	2457	0	1	42	9	4	7	19
30	2560	0	1	46	16	2	10	17
31	2618	0	1	49	2	1	10	12
32	2652	0	1	50	12	1	6	16
33	2682	0	1	51	18	1	21	12
34	2727	0	1	53	15	1	13	10
35	2764	0	1	55	4	7	11	24
36	2943	0	2	2	15	1	6	19
37	2973	0	2	3	21	1	4	14
38	3001	0	2	5	1	3	4	21
39	3077	0	2	8	5	2	1	36
40	3126	0	2	10	6	3	15	38
41	3213	0	2	13	21	4	9	10
42	3318	0	2	18	6	1	0	22
43	3342	0	2	19	6	1	11	32
44	3377	0	2	20	17	5	21	23

Table B.2: Second third of the table of shots extracted by the "Cinemetrics" tool from the "Pixar" short film "Lou" (2017)

#	Frame	hours	minutes	seconds	frames	Sec	Frs	Motion (%)
45	3518	0	2	26	14	0	23	17
46	3541	0	2	27	13	2	14	19
47	3603	0	2	30	3	2	1	26
48	3652	0	2	32	4	1	12	11
49	3688	0	2	33	16	5	12	12
50	3820	0	2	39	4	1	13	15
51	3857	0	2	40	17	5	23	21
52	4000	0	2	46	16	1	23	34
53	4047	0	2	48	15	0	21	33
54	4068	0	2	49	12	2	1	44
55	4117	0	2	51	13	1	1	50
56	4142	0	2	52	14	2	16	34
57	4206	0	2	55	6	4	0	32
58	4302	0	2	59	6	2	7	26
59	4357	0	3	1	13	1	14	37
60	4395	0	3	3	3	3	3	25
61	4470	0	3	6	6	1	20	27
62	4514	0	3	8	2	0	17	23
63	4531	0	3	8	19	4	11	30
64	4638	0	3	13	6	2	15	40
65	4701	0	3	15	21	2	6	33
66	4755	0	3	18	3	1	9	36
67	4788	0	3	19	12	0	21	23
68	4809	0	3	20	9	0	17	30
69	4826	0	3	21	2	1	8	20
70	4858	0	3	22	10	0	20	9
71	4878	0	3	23	6	9	0	27
72	5094	0	3	32	6	1	7	32
73	5125	0	3	33	13	1	10	40
74	5159	0	3	34	23	0	20	31
75	5179	0	3	35	19	1	0	35
76	5203	0	3	36	19	1	12	37
77	5239	0	3	38	7	1	4	34
78	5267	0	3	39	11	0	13	24
79	5280	0	3	40	0	1	15	25
80	5319	0	3	41	15	0	22	37
81	5341	0	3	42	13	1	12	26
82	5377	0	3	44	1	1	8	26
83	5409	0	3	45	9	0	18	31
84	5427	0	3	46	3	1	0	24
85	5451	0	3	47	3	1	16	25
86	5491	0	3	48	19	1	8	18
87	5523	0	3	50	3	1	8	14
88	5555	0	3	51	11	1	12	18
89	5591	0	3	52	23	1	7	23

Table B.3: Last third of the table of shots extracted by the "Cinemetrics" tool from the "Pixar" short film "Lou" (2017)

#	Frame	hours	minutes	secons	frames	Sec	Frs	Motion (%)
90	5622	0	3	54	6	0	13	36
91	5635	0	3	54	19	1	14	32
92	5673	0	3	56	9	2	6	30
93	5727	0	3	58	15	2	23	10
94	5798	0	4	1	14	1	22	21
95	5844	0	4	3	12	0	2	95
96	5846	0	4	3	14	3	0	17
97	5918	0	4	6	14	2	6	9
98	5972	0	4	8	20	2	1	9
99	6021	0	4	10	21	6	1	18
100	6166	0	4	16	22	2	2	13
101	6216	0	4	19	0	1	9	11
102	6249	0	4	20	9	2	3	10
103	6300	0	4	22	12	1	23	7
104	6347	0	4	24	11	3	17	17
105	6436	0	4	28	4	4	7	20
106	6539	0	4	32	11	5	23	13
107	6682	0	4	38	10	2	6	18
108	6736	0	4	40	16	4	6	24
109	6838	0	4	44	22	1	16	20
110	6878	0	4	46	14	2	15	9
111	6941	0	4	49	5	1	16	10
112	6981	0	4	50	21	2	16	15
113	7045	0	4	53	13	2	22	19
114	7115	0	4	56	11	3	18	22
115	7205	0	5	0	5	2	12	25
116	7265	0	5	2	17	1	15	15
117	7304	0	5	4	8	2	16	23
118	7368	0	5	7	0	1	16	25
119	7408	0	5	8	16	2	0	15
120	7456	0	5	10	16	5	0	26
121	7576	0	5	15	16	4	9	18
122	7681	0	5	20	1	2	4	12
123	7733	0	5	22	5	1	2	15
124	7759	0	5	23	7	1	11	14
125	7794	0	5	24	18	1	6	21
126	7824	0	5	26	0	4	22	26
127	7942	0	5	30	22	1	15	11
128	7981	0	5	32	13	3	5	14
129	8058	0	5	35	18	4	7	10
130	8161	0	5	40	1	2	17	15
131	8226	0	5	42	18	1	11	17
132	8261	0	5	44	5	6	18	18
133	8423	0	5	50	23	3	5	9
134	8500	0	5	54	4	19	8	13