

---

Automatic detection of flashing video content in  
videogames  
*Detección automática de contenido de video para medir la  
fotosensibilidad en videojuegos*

---



Trabajo de Fin de Grado  
Curso 2020–2021

**Autor**

Blanca Macazaga Zuazo  
Javier Landaburu Sánchez

**Director**

Guillermo Jiménez Díaz

**Colaborador**

Alberto de Torres Rivera

Grado en Desarrollo de Videojuegos  
Facultad de Informática  
Universidad Complutense de Madrid



Automatic detection of flashing video  
content in videogames  
*Detección automática de contenido de  
video para medir la fotosensibilidad en  
videojuegos*

**Trabajo de Fin de Grado en Desarrollo de Videojuegos  
Departamento de Software e Inteligencia Artificial**

**Autor**

**Blanca Macazaga Zuazo  
Javier Landaburu Sánchez**

**Director**

**Guillermo Jiménez Díaz**

**Colaborador**

**Alberto de Torres Rivera**

**Convocatoria: *Septiembre 2021***

**Grado en Desarrollo de Videojuegos  
Facultad de Informática  
Universidad Complutense de Madrid**

**21 de septiembre de 2021**





# Dedicatoria

*A María, Laura, Celia, Edu, Custo, y un largo etcétera de personas que aguantaron el chaparrón casi diario y aun así no dejaron de hablarme. También a Nacho, Jose Luis, y sobre todo a ti, Ana Belén, por ser la luz que me hace levantarme cada día y me hace seguir adelante.*

*Javi*

*A María, por todas las quejas aguantadas que iban por dos. A mi familia, por todo el apoyo que me han dado al hacer este trabajo y por compartir sus puntos de vista conmigo, ya que siempre me ayudan a mantenerme en el camino correcto. A ti, Paula, que siempre me apoyas y escuchas, y que eres la persona que me ánima a dar lo mejor de mí.*

*Blanca*



# Agradecimientos

A Claudio, por abrirnos la puerta para que esto fuera posible, y por hacernos más amenas todas las reuniones que tuvimos a lo largo del desarrollo. A Alessandro, por ilusionarse a medida que el proyecto iba avanzando, y por la suerte que hemos tenido de tener un manager que se esforzase tanto en ser comprensivo con nosotros y en apoyarnos. A Alberto, ya que no podríamos haber tenido mejor *scrum master*, y por compartir esa ilusión según cumplíamos nuestros objetivos. Y para Guille, por ser un constante apoyo durante este periodo, por todo el tiempo dedicado como tutor y todo el esfuerzo para ayudarnos a sacar este proyecto adelante. Por último, agradecer a nuestras familias y amigos que nos han escuchado (y muchas veces aguantado) y animado siempre que lo necesitábamos.

Blanca y Javi



# Abstract

Along the 21st century, the use of screens has increased exponentially due to the daily use of computers and mobile devices, thus these devices have become a fundamental part of our daily life. Due to their common use, the risk of prevention associated with its use has also grown. One of the most damaging diseases and at the same time most difficult to diagnose is photosensitive epilepsy, which affects almost 50 million people throughout the world. As the vast majority of cases are only diagnosed once they have suffered an epileptic seizure, it is necessary to prevent the conditions that can cause such seizures. In video games, the risk becomes higher as there can be a lot of flashing images along the video game's gameplay.

The objective of this project is to develop a tool that prevents this type of seizures from occurring when viewing audiovisual content (from any audiovisual medium). For this purpose, two types of checks are carried out in accordance with the visual stimuli that cause epileptic seizures in patients with this disease. The first one deals with the detection of luminance and saturated red flashes, and the second one with detection of harmful geometric patterns. Both procedures are based on previous studies carried out by experts, which led to the drafting of international standards for the prevention of epileptogenic content in television broadcasts and, subsequently, all types of audiovisual content.

## Keywords

Photosensitivity, flashes, luminance, patterns, HardingFPA



# Resumen

A medida que avanzamos años del siglo XXI, el uso de pantallas se ha incrementado exponencialmente debido al uso que hacemos diariamente de ordenadores y dispositivos móviles, por lo que se han convertido en una parte fundamental de nuestro día a día. Por ello, también ha crecido la prevención de riesgos asociados a su uso. Una de las enfermedades más dañinas y a la vez más difíciles de diagnosticar es la epilepsia fotosensible, la cual afecta a casi 50 millones de personas a lo largo del mundo. Debido a que la gran mayoría de casos solo son diagnosticados una vez han sufrido un ataque epiléptico, es necesario prevenir las condiciones que puedan provocar tales ataques.

El objetivo de este proyecto es desarrollar una herramienta que permita prevenir que ocurran este tipo de ataques al ver contenido audiovisual (de cualquier medio). Para ello, se llevan a cabo dos tipos de comprobaciones en concordancia con los estímulos visuales que provocan ataques epilépticos a los pacientes con esta enfermedad. El primero se encarga de detectar flashes de luminancia y de rojo saturado, y el segundo de detectar patrones geométrico nocivos. Ambos procesos están basados en estudios previos realizados por expertos, que desencadenaron en la redacción de normas internacionales para la prevención de contenidos epileptógenos en las emisiones de televisión y, posteriormente, todo tipo de contenido audiovisual.

## Palabras clave

Fotosensibilidad, flashes, luminancia, patrones, HardingFPA





# Índice

<b>1. Introduction</b>	<b>1</b>
1.1. Motivation . . . . .	1
1.2. Objectives . . . . .	2
1.3. Work plan . . . . .	3
1.4. Document outline . . . . .	3
<b>2. State of the art</b>	<b>5</b>
2.1. Photosensitive Epilepsy . . . . .	5
2.2. PSE triggers in videos . . . . .	6
2.2.1. Flashing video characteristics . . . . .	7
2.2.2. Pattern characteristics . . . . .	10
2.3. Guidelines . . . . .	13
2.3.1. Ofcom . . . . .	13
2.3.2. ITU . . . . .	15
2.3.3. WCAG . . . . .	15
2.4. Existing tools . . . . .	17
2.4.1. HardingFPA . . . . .	17
2.4.2. PEAT . . . . .	17
2.4.3. Flicker Check . . . . .	18
2.5. Conclusions . . . . .	19
<b>3. Detecting harmful videos</b>	<b>21</b>
3.1. Main pipeline . . . . .	21
3.2. Luminance calculation . . . . .	22

3.2.1.	Luminance at the display . . . . .	22
3.2.2.	Relative luminance . . . . .	22
3.3.	Flash Detection . . . . .	23
3.3.1.	Extract frame values . . . . .	24
3.3.2.	Frame Difference . . . . .	25
3.3.3.	Flashing safe area . . . . .	26
3.3.4.	Value accumulation and transition detection . . . . .	29
3.3.5.	Flash Transition Count . . . . .	30
3.3.6.	Check criteria . . . . .	31
3.4.	Pattern Detection . . . . .	31
3.4.1.	Line detection . . . . .	32
3.4.2.	Pattern Analysis . . . . .	35
3.4.3.	Risk Estimation . . . . .	38
3.4.4.	Check criteria . . . . .	40
3.5.	Conclusions . . . . .	40
<b>4.</b>	<b>Implementation</b>	<b>43</b>
4.1.	Features . . . . .	43
4.2.	Software architecture . . . . .	46
4.2.1.	VideoLib . . . . .	47
4.3.	Technology used . . . . .	48
4.4.	Development process . . . . .	49
4.4.1.	Prototype . . . . .	49
4.4.2.	Performance analysis . . . . .	51
4.4.3.	Final version refactoring . . . . .	51
4.5.	Conclusions . . . . .	53
<b>5.</b>	<b>Evaluation</b>	<b>55</b>
5.1.	Methodology . . . . .	55
5.2.	Real videos . . . . .	57
5.3.	Generated videos . . . . .	60
5.3.1.	Combination of different flashing areas with different frequencies . . . . .	60
5.3.2.	Combination of different flashing red areas with different frequencies . . . . .	62
5.3.3.	Patterns . . . . .	63

5.4. Conclusions . . . . .	64
<b>6. Conclusions and Future Work</b>	<b>65</b>
6.1. Conclusions . . . . .	65
6.2. Future Work . . . . .	66
<b>7. Introducción</b>	<b>69</b>
7.1. Motivación . . . . .	69
7.2. Objetivos . . . . .	70
7.3. Plan de trabajo . . . . .	71
7.4. Estructura de la memoria . . . . .	72
<b>8. Conclusiones y Trabajo Futuro</b>	<b>73</b>
8.1. Conclusiones . . . . .	73
8.2. Trabajo Futuro . . . . .	74
<b>Bibliografía</b>	<b>77</b>
<b>A. Appendix A - Data Visualization</b>	<b>81</b>
<b>B. Appendix B - Contributions</b>	<b>85</b>
B.1. Blanca Macazaga Zuazo . . . . .	85
B.2. Javier Landaburu Sánchez . . . . .	86



# Índice de figuras

1.1.	Frame of the Golden Wonder ad broadcasted in the UK (Youtube, 2021) . . .	2
1.2.	Two frames of the Pokémon episode broadcasted in Japan (Youtube, 2015) . . .	2
2.1.	(a) Striped pattern (b) Circular pattern (c) Checkerboard pattern . . . . .	6
2.2.	Percentage of patients with photosensitive epilepsy exhibiting epileptiform EEG responses to the flicker shown as a function of flash frequency (Harding G.F.A., 1994) . . . . .	7
2.3.	Proportion of patients affected by flashes that differ from screen luminance by various intensities (Binnie et al., 2002) . . . . .	8
2.4.	Proportion of patients affected by flashes as a function of the difference between screen luminance and the luminance of the flash (Wilkins et al., 1980) . . . . .	9
2.5.	Proportion of patients affected according to luminance pattern (Binnie et al., 2002) . . . . .	10
2.6.	Proportion of patients with pattern-sensitive epilepsy at risk from a pattern of bright high-contrast stripes, shown as a function of the overall size of the pattern, expressed in terms of the proportion of the screen that it occupies (Wilkins A., 2005) . . . . .	11
2.7.	Proportion of patients with pattern-sensitive epilepsy at risk from a large pattern, shown as a function of the spatial frequency of the pattern expressed in cycles per degree (Wilkins A., 2005) . . . . .	12
2.8.	Proportion of patients with pattern-sensitive epilepsy at risk from a pattern, shown as a function of the number of bright–dark pairs of stripes (Wilkins A., 2005) . . . . .	12
2.9.	HardingFPA tool GUI . . . . .	18
2.10.	PEAT tool GUI . . . . .	18
2.11.	Hitachi Flicker Check tool GUI . . . . .	19
3.1.	Program flow . . . . .	21

3.2.	Flash Detection Algorithm . . . . .	23
3.3.	Extraction of saturated red pixels into a binary image . . . . .	24
3.4.	Frame sequence from Apex Legends . . . . .	25
3.5.	Absolute luminance difference of frames from figure 3.4 . . . . .	26
3.6.	Frame sequence from It Takes Two! . . . . .	26
3.7.	Absolute red saturation difference of frames from figure 3.6 . . . . .	26
3.8.	Connected components: blob detection . . . . .	28
3.9.	Frame difference . . . . .	28
3.10.	Selected blobs from 3.5 frame sequence . . . . .	29
3.11.	Pattern Detection Algorithm . . . . .	32
3.12.	Canny output with and without noise reduction . . . . .	33
3.13.	Canny output with different thresholds from frame in figure 3.12 . . . . .	34
3.14.	Operations for pattern data extraction . . . . .	35
3.15.	Pattern isolation process . . . . .	36
3.16.	Negative and positive stripes separated into different images . . . . .	37
3.17.	Negative and positive connected components of figure 3.16 . . . . .	38
3.18.	Images of the highlighted light and dark stripe components of a pattern . . . . .	38
4.1.	Highlighted flash area over video frame in grayscale . . . . .	44
4.2.	Images of the highlighted light and dark stripe components of a pattern . . . . .	44
4.3.	Console output of video analysis . . . . .	45
4.4.	IRIS flow . . . . .	46
4.5.	VideoLib class diagram . . . . .	47
4.6.	Accumulated luminance peaks throughout the length of a video . . . . .	50
5.1.	Example of data visualization, in this case the total number of luminance transitions in flashing video of 3Hz . . . . .	56
5.2.	Total number of luminance transitions in Golden Wonder Pot ad . . . . .	57
5.3.	Total number of failed frames in Golden Wonder Pot ad . . . . .	57
5.4.	Total number of red transitions in the Pokémon episode . . . . .	58
5.5.	Total number of failed frames in the Pokémon episode . . . . .	58
5.6.	Total number of pattern failed frames in the White Stripes video . . . . .	59
5.7.	Total number of luminance transitions in the White Stripes video . . . . .	59
5.8.	Total number of luminance transitions in the London Olympics Games ad . . . . .	60

5.9. Total number of failed frames in the London Olympics Games ad . . . . .	60
5.10. Total number of luminance transitions in flashing video of 1.5Hz . . . . .	61
5.11. Total number of harmful frames in flashing video of 1.5Hz . . . . .	61
5.12. Total number of luminance transitions in flashing video of 3Hz . . . . .	62
5.13. Total number of red transitions in red flashing video of 1.5Hz . . . . .	62
5.14. Total number of red transitions in red flashing video of 3Hz . . . . .	63
5.15. Estimated risk of the striped pattern video of 8 pairs of dark/white lines . .	64
5.16. Total pattern failed frames in the circular pattern video of 8 pairs of dark- /white lines . . . . .	64
7.1. Frame del anuncio de Wolden Wonder en Reino Unido (Youtube, 2021) . . .	70
7.2. Dos frames del capítulo de Pokémon emitido en Japón(Youtube, 2015) . . .	70





# Índice de tablas

2.1. Proportion of patients affected by flashes of different screen area (Wilkins et al., 1980) . . . . .	9
5.1. Results of analyzed flashing videos . . . . .	61
5.2. Results of analyzed red flashing videos . . . . .	62
5.3. Results of analyzed pattern videos . . . . .	63



# Introduction

## 1.1. Motivation

Today, the video game industry has become one of the most popular audiovisual media industries in the world, generating global revenues of almost \$180 billion in 2020 (Batchelor, 2020). Due to this popularity and the changes in the consumption habits of society, with a significantly greater presence of the use of screens and mobile devices, the amount of time each person spends in front of a screen throughout the day has increased. In the case of video games, whose main typical consumers are in the young sector of the population, this represents an increase in the risk of suffering from diseases derived from the visual content seen in televisions or screen monitors, as is the case with photosensitive epilepsy.

Photosensitive epilepsy is a neurological disease capable of causing seizures (known as epileptic seizures), caused by visual stimuli such as exposure to rapid changing lights or to geometric patterns (Epilepsy Society, 2020). Epilepsy is more prevalent at an early age, being more common in ages from 7 to 18 years old (Vivir con Epilepsia, 2017), which makes video games a risk factor for possible cases of patients with photosensitive epilepsy.

During the last decades, a series of epileptic incidents caused by the broadcast of photosensitive material prompted the creation and the following updates in the existing regulations on television broadcasts. Given the nature of the disease, it only manifests itself in the face of certain stimuli, and therefore it is difficult to diagnose preventively. Some of the most outstanding events are:

- In 1993, in the UK, a television food advertisement for the Golden Wonder Pot Noodles product<sup>1</sup>, containing a sequence of images with rapid color changes, caused 3 cases of epileptic seizures.
- In 1997, in Japan, the episode 38th of the first season of the animated series Pokémon<sup>2</sup> contains a sequence of fast changes between red and blue colors, besides some flickering images of high luminance, caused 685 cases of epileptic seizures.

Nowadays, video games must give warnings about photosensitive epileptic risks, yet

---

<sup>1</sup>Video URL: <https://www.youtube.com/watch?v=HVQuA9HNhOM>

<sup>2</sup>Video URL: <https://www.youtube.com/watch?v=4LQLvgXLguk>



Figure 1.1: Frame of the Golden Wonder ad broadcasted in the UK (Youtube, 2021)



Figure 1.2: Two frames of the Pokémon episode broadcasted in Japan (Youtube, 2015)

should prevent the risks associated with this disorder. The objective of this project is to develop a tool that analyzes and determines if a video sequence, extracted from a testing session on a video game, is compliant to the existing guidelines and regulations or not, in order to prevent possible harmful content to those who suffer this disease.

## 1.2. Objectives

The main objective of this project is the implementation of a tool that allows to validate video sequences from any media from a photosensitive perspective, following the current guidelines in regard to this issue. The tool aims to provide information and prevent the publishing of harmful photosensitive material.

This project has been developed for Electronic Arts, as a way to provide a more flexible tool to check harmful video content. Given the existence of similar tools on the market, the goal of this project is to create a solution that meets the characteristics of these applications, but that can be updated if the regulations are modified and that can breach the limitations that the existing tools present, such as supporting multiple video resolutions, frame rates and formats.

### 1.3. Work plan

For the development of this project, the organization was carried out as follows:

The chosen methodology used to develop this project is SCRUM, an agile methodology widely used in software development and video games development. Set to 2-week sprints, the members of the development team establish a series of tasks to be carried out, and these tasks are assigned with a priority, as well as an estimate of the time to carry them out. Thus, the tasks are distributed among the team members and carried out during the period called *sprint*. Once that time has elapsed, an analysis of the completed tasks is carried out, proposing new ones in the same way as before, or the conditions of the tasks that could not be completed during the previous sprint are adjusted.

During the first few weeks, the team begins to research on photosensitive epilepsy, its causes, and the effects it has on their patients. At the same time, the research also focuses on clinical studies carried out to detect and diagnose cases of photosensitive epilepsy, as well as what are the characteristics that can trigger an epileptic seizure.

Once the first articles and essays have been studied, the team starts an investigation on the international regulations based on these previous studies, to determine what are the conditions and values that are considered harmful from a photosensitive point of view. The team is also able to familiarize itself with existing tools (such as HardingFPA, section 2.4.1, or PEAT, section 2.4.2), so that it could understand what is the normal flow of use for its subsequent implementation.

The technologies to be used for the development of the tool are determined. In this case, the tool is developed in C#, using the Microsoft Visual Studio development environment, and using OpenCV graphic library.

The first objectives are defined: project skeleton, data extraction from a video, assignment of limit values that define whether a video is harmful or not. From there, there are two ways defined: detection of bright flashes and detection of saturated red flashes.

Once both detections have been defined, there is an investigation about the characteristics of geometric patterns that can trigger an epileptic response. Geometric pattern detection is implemented, and a risk estimation is carried out that determines whether a video is harmful or not.

Once the basic functionalities have been implemented, it is iterated in such a way that each module improves its efficiency, both in terms of resources and time.

### 1.4. Document outline

The structure of this document is defined the following way:

- In Chapter 1, the main objective of this project is introduced, as for its objectives and main motivations.
- In Chapter 2, the concept of Photosensitive Epilepsy(PSE) is defined, its characteristics and harmful effects, as well as the clinic studies carried out during history to determine the triggering conditions of PSE. There is also mention of the international

organizations and the established guidelines to prevent epileptic seizures, based on the results of previous studies.

- In Chapter 3, there is the complete description of the tool's functionality, focusing on the two main features of the tool, the flash detection and the pattern detection.
- In Chapter 4, the project's implementation, technology, software architecture and development are presented.
- In Chapter 5, there is an evaluation of the tool and several results of real cases to determine the compliance of the tool are explained.
- In Chapter 6, a series of conclusions are listed after the realization of this project and its possible future improvements.

## State of the art

This chapter contains the definition and causes of photosensitive epilepsy, the video conditions for triggering photosensitive epilepsy, the research and studies underneath PSE and already existing similar tools to check if a video is considered harmful.

### 2.1. Photosensitive Epilepsy

Photosensitive epilepsy (PSE) is a form of epilepsy, a neurological disorder that can prompt seizures to individuals when triggered. In the case of PSE, the triggers that cause epileptic seizures are visual stimuli, such as flashing lights or rapidly changing geometric patterns (Epilepsy Society, 2020).

The triggers that lead to seizures due to PSE can be both natural or artificial, as they can be caused by natural light, e.g. sunlight shimmering off water, street lights or by the changing visual components in TVs and computer screens. Another factors like tiredness, stress or excitement, or the effect taking up all your field of vision, (frequent states when playing video games), and being very close to a screen during a long time without breaks, may increase the photosensitive risk (Epilepsy Society, 2020).

One in every hundred people have epilepsy, from which 5% have PSE. And from that 5%, 60% experience their first epileptic seizure when watching TV or playing video games, as most people do not realize that they suffer from this disease until they experience their first incident. As PSE is more common in children and teenagers, this two age groups are more vulnerable to the exposure to harmful video content. This disorder is an abnormal response on the electroencephalography (EEG) triggered by the stimuli mentioned before.

Nevertheless, diagnosing epilepsy is not a simple task, as doctors need first to gather lots of different information to assess the causes behind the seizures. Apart from regular tests, such as an EEG, a magnetic resonance imaging (MRI) or blood tests to discard previous pathologies, the diagnosis is based on finding out what happened to the patient before, during and after the seizures (Vivir con Epilepsia, 2017).

Although PSE cannot be cured, the seizures derived from this disorder can be controlled, so the patients can have a normal life with fewer or even the absence of seizures.

The most common treatment, after an epilepsy diagnosis has been made by a specialist, are what is called anti-epileptic drugs (UK NHS, 2021). These type of medicine work by changing the levels of chemicals in the patient’s brain, affecting the neurotransmitters responsible for sending messages, or attach themselves to the surface of neurons and alter the activity of the cell by changing how ions flow into and out of the neurons (Epilepsy Society, 2020).

## 2.2. PSE triggers in videos

As explained in the previous section, PSE can be triggered by natural and artificial conditions. In terms of audiovisual content, harmful videos are characterized by the following:

- Flickering lights, which cause photic stimulation.
- Flickering from or to saturated red colors.
- Spatially periodic patterns, such as stripes, checkerboards or spirals (Figure 2.1).

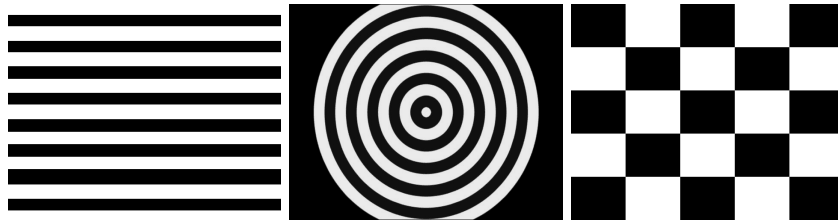


Figure 2.1: (a) Striped pattern (b) Circular pattern (c) Checkerboard pattern

Epileptic seizures can be estimated without inducing them thanks to the called ‘epileptiform’ waves that appear on EEG tests on known or suspected patients of PSE, where they are exposed to visual stimuli using flickering lights from a xenon gas lamp or geometric patterns displayed on a TV screen.

Based on previous researches like (Harding et al., 2005; Binnie et al., 2002) there are certain characteristics regarding flickering or patterned images that induce seizures, including intensity, duration, distance from the source, background illumination, diffusion of the pattern, contrast, type of pattern, color, open versus closed eyes, one versus two-eye viewing, and stage of sleep–wakefulness cycle. The following sections describe how these characteristics were obtained during the experiments developed by some of the most important researchers in the field of PSE, like Harding.

Before explaining some of the results of the studies carried out, let’s introduce first some definitions included in (Harding et al., 2005) to correctly understand them:

- *Photic induced-seizure: A seizure provoked by visual stimulation, whether is a flashing light, or patterns of lines, gratings, checkerboards or other configurations.*



- *Photoparoxysmal response (PPR): An abnormal EEG response to light or pattern, consisting of spikes, spike-waves, or intermittent slow waves. To be classified as photoparoxysmal, the spikes should not be confined to occipital regions and should not be confused with the normal visual evoked response that is phase-locked to the flash.*

### 2.2.1. Flashing video characteristics

A flicker is produced when a light varies over time with respect to its luminance or color, but the commonly used definition of a flash, when talking about audiovisual content, occurs when there is a pair of opposing changes in luminance (i.e., an increase in luminance followed by a decrease, or a decrease followed by an increase). As brightness is a term commonly used to describe the amount of light emitted by a certain source, it cannot be considered for measurement purposes due to it being a subjective perception, different on each person. Instead, the science community introduced the term luminance, described as the luminous intensity of a surface in a given direction per unit of projected area, and it is measured in candelas per square meter ( $cd/m^2$ ), the unit of measurement of the International System of Units (SI).

#### 2.2.1.1. Frequency

The frequency of a flicker is the number of flashes per second. The SI unit for frequency is the Hertz (Hz). According to the studies carried out by Jeavon and Harding (Jeavons y Harding, 1975), as it can be seen in Figure 2.2, the number of patients affected by light of different frequencies, showing the percentage of patients exhibiting a photoparoxysmal EEG response to intermittent visual stimuli. The patients affected by this response fastly increase when the flashes surpass the rate of 3Hz, reaching its top in 16Hz. The results show that the range of what it is considered as ‘harmful frequencies’ 3Hz - 65Hz

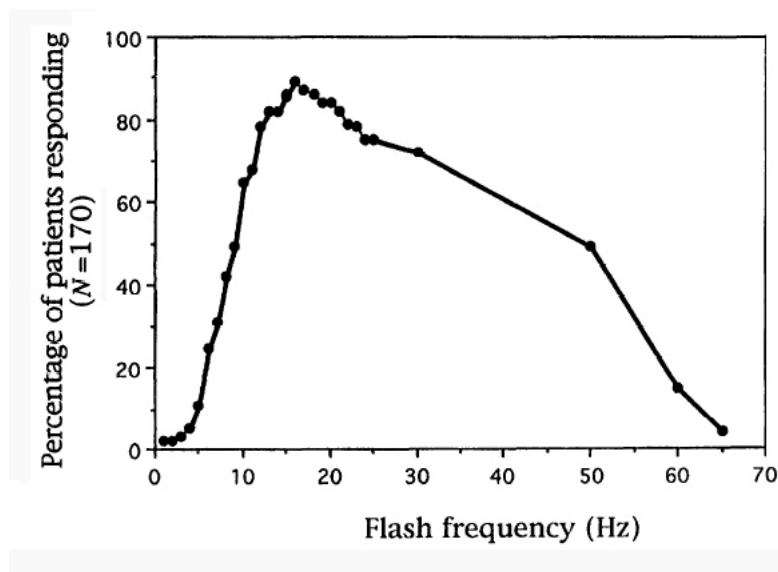


Figure 2.2: Percentage of patients with photosensitive epilepsy exhibiting epileptiform EEG responses to the flicker shown as a function of flash frequency (Harding G.F.A., 1994)

### 2.2.1.2. Intensity

The intensity is the strength of something that can be measured, in this case, the luminous intensity. For flashing content, the flash intensity is the average difference between the two opposing states of a flash (increase / decrease - decrease / increase). But first there is a need to differentiate between the screen intensity (the light emitted per unit area, the screen brightness) from the flash intensity.

Studies show that not only the amount of flashes during a certain time is enough to trigger a seizure, but the intensity of them. As can be seen on Figure 2.3, the proportions of patients affected by flashes differ from screen luminance by different intensities., where flashes of  $5 \text{ cd/m}^2$  are safe at all screen luminances, but becomes a serious risk when flashes of  $20 \text{ cd/m}^2$  are shown with a screen luminance below  $100 \text{ cd/m}^2$ , and also when the contrast of the flash is the epileptogenic range.

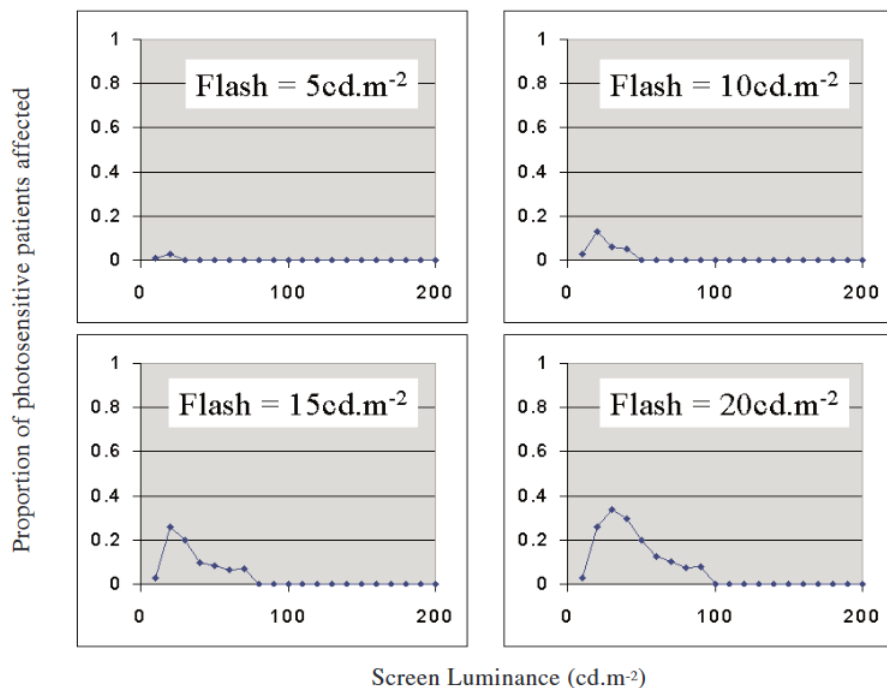


Figure 2.3: Proportion of patients affected by flashes that differ from screen luminance by various intensities (Binnie et al., 2002)

The result of extrapolating all flash intensities results can be seen in Figure 2.4, where the proportion of affected patients by flashes as a function of the difference between screen luminance and the luminance of the flash, increases rapidly for low luminances.

### 2.2.1.3. Area

The flashing area is the size of the screen surface where the flash occurs. Assuming the data given in Rec. ITU-R BT. 500 (Binnie et al., 2002), researchers could calculate the proportion of patients affected by flashes of different size. For example, using a television with a 20-inch screen, viewed at a distance of seven times the screen height (7H) stimulates

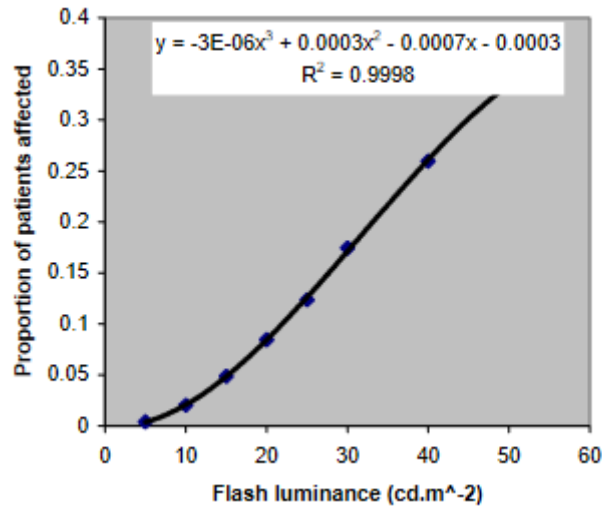


Figure 2.4: Proportion of patients affected by flashes as a function of the difference between screen luminance and the luminance of the flash (Wilkins et al., 1980)

25% of the visual cortex, or using a 60-inch screen viewed at 5H stimulate 34% of the visual cortex.

It has a serious impact on patients, as it is shown on Table 2.1, where the area flashing area of the screen and the proportion of patients affected increase almost proportionally:

Area of the screen flashing	Weighted Average
10%	0.09
12.5%	0.15
25%	0.36
50%	0.64
100%	1

Table 2.1: Proportion of patients affected by flashes of different screen area (Wilkins et al., 1980)

#### 2.2.1.4. Color

As well as previous studies performed of flicker modulating the luminance, some other focused on changing the color of the emitted light. The results of (Binnie et al., 1984) showed that modulating the color and luminance, changing only one class of photoreceptor, was more epileptogenic than the equivalent using white light. Specifically, the different tones of red light showed more brain activity, and further researches would confirm it (Harding, 1998). During this study, subjects EEG activity was recorded while watching the same cartoon, first in black and white, then in color; the second view reflected a clear photoparoxysmal response (PPR) on patients, confirming the premises of the experiment. Later, international broadcasting organizations, such as Ofcom or ITU would include guidelines to prevent epileptic seizures from red flashing images.

### 2.2.2. Pattern characteristics

A geometric pattern is an arrangement of lines or shapes, in which the same shape is repeated at regular intervals over a surface. As can be seen in Figure 2.1, the types of patterns are multiple and can be made by straight lines, circular lines, checkerboards, polka dots. . . , showing a great variety of geometric shapes. For this section, the experiments and results of the works described in (Wilkins A., 2005; Binnie et al., 2002), will provide a clearer understanding of the main characteristics of patterns, and how to understand its main guidelines to avoid harmful sequences.

#### 2.2.2.1. Luminance

In terms of luminance, the effects of its difference works similarly than with flashes, but instead of looking at the difference between its two states, with patterns is necessary to observe the luminance difference between its stripes. Which are divided into two components, or sets, of dark and light stripes (from here also referred to as bright and dark parts of the pattern). Taking a look at the data (Wilkins A., 2005) replotted in Figure 2.5, the number of patients affected by a pattern with log luminance in the range 10-200  $cd/m^2$  shows almost a linear increase.

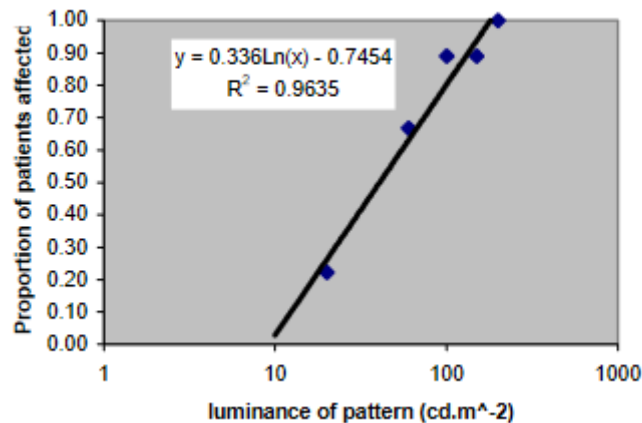


Figure 2.5: Proportion of patients affected according to luminance pattern (Binnie et al., 2002)

#### 2.2.2.2. Duration

All the results of previous studies are given as a proportion of people affected by the visual stimuli, so the responses are measured and calculated in a probabilistic manner. Because of this, the obvious thought is that if a pattern is displayed for a long period of time, the risk of inducing a seizure is bigger. There are no particular studies that have investigated the effects of pattern duration, but the ones that study it show a paroxysmal response around 1 or 2 seconds after visual stimulating the patients. That is the reason why most organizations suggests on its guidelines to prevent PSE seizures that patterns should not be displayed for more than 0.5s (Wilkins A., 2005).

### 2.2.2.3. Size

As it has been showed in Table 2.1, the size of the displayed flashes affected the proportion of patients affected, and these results happen to be the same if the stimulus are patterns. The proportion of affected patients was calculated for the case of a 20-inch screen with 4:3 aspect ratio and 24-inch screen with 16:9 aspect ratio, both of which have a screen height of 0.3m and a preferred viewing distance of 7H; in the same way, they used a 60-inch screen with 4:3 aspect ratio, and a 73-inch screen with 16:9 aspect ratio, at a preferred viewing distance of 5H (Wilkins A., 2005). Figure 2.6 indicates that the difference between the aspect ratio is not significant.

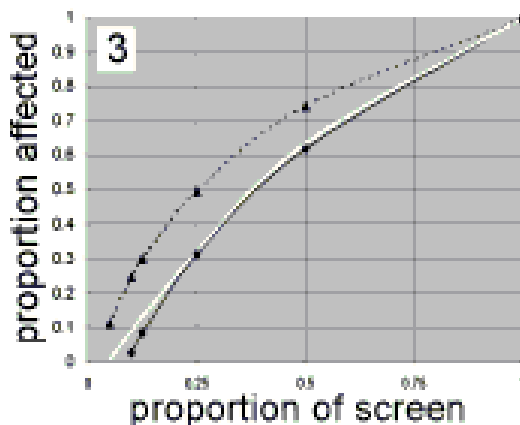


Figure 2.6: Proportion of patients with pattern-sensitive epilepsy at risk from a pattern of bright high-contrast stripes, shown as a function of the overall size of the pattern, expressed in terms of the proportion of the screen that it occupies (Wilkins A., 2005)

### 2.2.2.4. Spatial frequency

The spatial frequency of a pattern is the number of cycles of the pattern (pairs of bright and dark stripes), measured in cycles per degree (cpd) (considering degree as the angle of vision, one of the variables for the visual perception of the size or projection of the size of an object (Wikipedia, 2005)). Wilkins calculated the proportion of affected patients by bright high-contrast patterns (Figure 2.7), depending on the spatial frequency of the pattern, expressed in cycles per 1 degree subtended at the eye.

Considering the spatial frequency and the area of the screen occupied by the stripes together, a proportion of patients affected from the products of the proportions can be estimated, and the results are replotted on Figure 2.8.

It can be observed that the curves show the effects of three different pattern areas, expressed as the percentage of the screen occupied by the pattern. The analysis of the results indicates that the proportion of affected patients approximately double as the number of stripes increases from five to eight, considering patterns of reasonable size. Also, the proportion of affected patients by a five-stripe pairs pattern is close for patterns that occupy the entire screen and those that occupy 25% (Wilkins A., 2005).

These results were used in the development of most of the broadcasting guidelines for

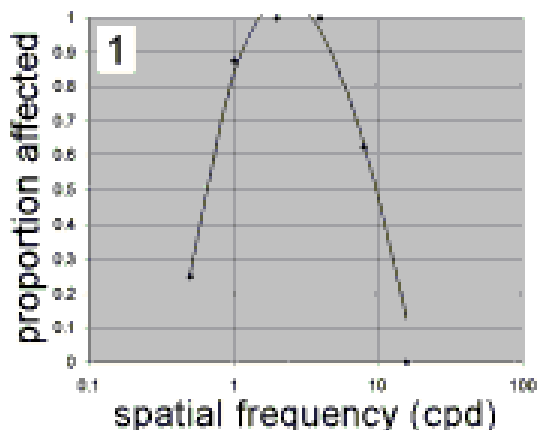


Figure 2.7: Proportion of patients with pattern-sensitive epilepsy at risk from a large pattern, shown as a function of the spatial frequency of the pattern expressed in cycles per degree (Wilkins A., 2005)

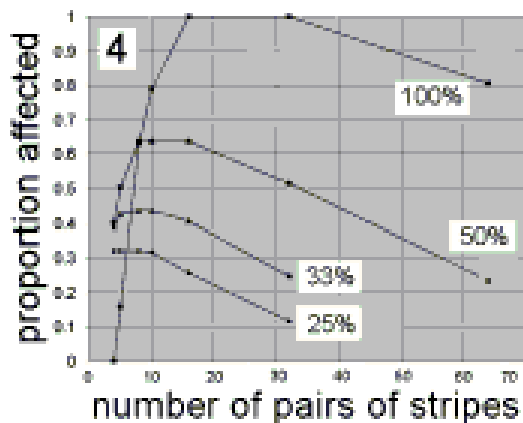


Figure 2.8: Proportion of patients with pattern-sensitive epilepsy at risk from a pattern, shown as a function of the number of bright-dark pairs of stripes (Wilkins A., 2005)

preventing seizures, and defined the limit of 5 pairs of bright-dark pattern as a reasonable boundary.

#### 2.2.2.5. Movement

The results obtained from (Wilkins A., 2005) show that a still pattern is more epileptogenic than if it is drifting continuously in one direction. But it does not imply that the lack of restrictions in this matter has not provoked seizures, as two previous instances show. In one of them, the pattern moved and changed in size in such a way that provoked a flash, and in both cases, the seizures arose when a brief freeze-frame was displayed at the end of a sequence. However, in both cases the patterns had a spatial frequency close to that at which seizures are most readily evoked, and the patterns filled the screen. The similarity between the two examples would seem to suggest that, despite the doubt concerning the exact cause of the seizures, it would be sensible to prohibit large patterns with the worst

spatial frequencies, even when they drift (Wilkins A., 2005).

Because of this uncertainty, Wilkins suggests that, if restricting this behavior is not easy, guidelines should be updated to restrict those that have more than 12 pairs of stripes of any kind (the current guidelines suggest that patterns should not have more than 5 pairs).

## 2.3. Guidelines

In many countries, such as the UK or the USA, the existing guidelines and legislation prevents content creators discriminating on disabled people, epileptic patients included. That is the reason why if an audiovisual content which is produced and broadcasted containing flashing images or geometric patterns, the author must comply with the standards, including, if needed, a disclaimer to provide a warning to possible epileptic people watching that content. Otherwise, the author would be breaking the law, and it could be considered a crime (Epilepsy Society, 2020).

### 2.3.1. Ofcom

The Office of Communications (Ofcom) is a regulatory authority for the broadcasting industry in the United Kingdom. Formerly known as the Independent Television Commission (ITC), this government-approved authority was born after the British government decided to merge their regulatory communications corporations into a single entity. One of its most important tasks is “to provide adequate protection for members of the public from harmful and/or offensive material”, as the Broadcasting Code stands in its Section two: Harm and Offense in rule number 2.12 (Ofcom, 2017b):

*2.12: Television broadcasters must take precautions to maintain a low level of risk to viewers who have photosensitive epilepsy. Where it is not reasonably practicable to follow the Ofcom guidance, and where broadcasters can demonstrate that the broadcasting of flashing lights and/or patterns is editorially justified, viewers should be given an adequate verbal and also, if appropriate, text warning at the start of the programme or programme item.*

This commitment of protecting the viewers from harmful material related with PSE started in 1993 by the ITC until the release of its final version in 2001 (Ofcom, 2017b), where the Broadcasting Code is included, as a consensus statement of a committee set up by the ITC consisting of three medical experts: Prof. Graham F. A. Harding (Cole, 2018), Prof. Colin David Binnie (International League Against Epilepsy, 2019), and Prof. Arnold J. Wilkins (Encyclopedia.com, 2005). The result of this work ended up with the development of several guidelines for broadcasting content, based on the research made by the three experts during their academic career. In their Guidance Notes of the Broadcasting Code, Ofcom provides broadcasters help in interpreting and applying it, including on its Annex 1 a specific note on how to prevent broadcasters to include harmful images for possible viewers with PSE (Ofcom, 2017a):

1. *Flickering or intermittent images and certain types of regular pattern can cause prob-*

lems for some viewers who have photosensitive epilepsy. These guidelines have been drawn up following consultation with leading medical opinion in this area with the aim of reducing the risk of exposure to potentially harmful stimuli.

2. *Television is by nature a flickering medium. In Europe each transmitted picture is refreshed 50 times each second and interlaced scanning generates flicker 25 times each second. It is therefore impossible to eliminate the risk of television causing convulsions in viewers with photosensitive epilepsy. To reduce risk the following guidelines on visual content should be applied when flashing or regular patterns are clearly discernible in normal domestic viewing conditions. It should be noted that the level of any cumulative risk arising from successive sequences of 'potentially harmful' flashes over a prolonged period is unknown. If, as medical opinion suggests, the risk of seizures increases with the duration of flashing, broadcasters should note that it is possible that a sequence of flashing images lasting more than 5 seconds might constitute a risk even when it complies with the guidelines below.*
3. *A potentially harmful flash occurs when there is a pair of opposing changes in luminance (i.e., an increase in luminance followed by a decrease, or a decrease followed by an increase) of 20 candelas per square metre ( $\text{cd}/\text{m}^2$ ) or more. This applies only when the screen luminance of the darker image is below  $160 \text{ cd}/\text{m}^2$ . Irrespective of luminance, a transition to or from a saturated red is also potentially harmful.*
  - a) *Isolated single, double, or triple flashes are acceptable, but a sequence of flashes is not permitted when both the following occur:*
    - 1) *the combined area of flashes occurring concurrently occupies more than one quarter of the displayed screen area; and*
    - 2) *There are more than three flashes within any one-second period. For clarification, successive flashes for which the leading edges are separated by 9 frames or more are acceptable, irrespective of their brightness or screen area.*
4. *Rapidly changing image sequences (e.g. fast cuts) are provocative if they result in areas of the screen that flash, in which case the same constraints apply as for flashes.*
5. *A potentially harmful regular pattern contains clearly discernible stripes when there are more than five light-dark pairs of stripes in any orientation. The stripes may be parallel or radial, curved or straight, and may be formed by rows of repetitive elements such as polka dots. If the stripes change direction, oscillate, flash or reverse in contrast they are more likely to be harmful than if they are stationary. If the patterns obviously flow smoothly across, into, or out of the screen in one direction they are exempt from restriction.*
  - a) *Potentially harmful patterns are not permitted when either of the following two conditions apply:*
    - 1) *the stripes are stationary and the pattern occupies more than 40% of the displayed screen area; or*
    - 2) *the stripes change direction, oscillate, flash, or reverse in contrast and the pattern occupies more than twenty-five per cent of screen area; and in addition to either of the above two conditions applying, when*
    - 3) *the screen luminance of the darker bars in the pattern is below  $160 \text{ cd}/\text{m}^2$  and differs from the lighter bars by  $20 \text{ cd}/\text{m}^2$  or more.*



### 2.3.2. ITU

The International Telecommunications Union (ITU) is a United Nations specialized agency for information and communication technologies, formed in 1865 as the International Telegraph Union, being one of the oldest international organizations still active. Later, in 1932, the International Telegraphic Union approved the change of name to the current International Telecommunications Union, and in 1947 it was integrated into the international management apparatus of the UN to ensure the interoperability of modern communication networks of all countries.

ITU-R (ITU-Radiocommunication) is one of the sectors composing ITU, responsible for managing the international radio-frequency spectrum, developing standards for radio communication systems with the purpose of ensuring effective use of the spectrum. These standards developed by ITU-R (ITU-R Recommendations) are the result of studies carried out involving different radio communications like wireless services, the use of the radio-frequency spectrum, or terrestrial and satellite radio communications broadcasting.

The recommendations related with reducing the risk for possible PSE viewers are mainly grouped in ‘Recommendation ITU-R BT.1702 - Guidance for the reduction of photosensitive epileptic seizures caused by television’ (ITU, 2005), where most of the guidelines on potential harmful flashes from Ofcom are included in this standard. The additions that were included after its first publication in 2005 are:

- *When screen luminance of the darker image is not less than 160 cd/m<sup>2</sup>, a potentially harmful sequence of flashes occurs where there is a difference of more than 1/17 Michelson contrast. For the purpose of this measurement, the Michelson contrast is defined as  $L_{high} - L_{low} / L_{high} + L_{low}$ , where  $L_{low}$  is the luminance of the darker image and  $L_{high}$  is the luminance of the brighter image of a flash (only applicable for HDR programmes only).*
- *Isolated, single, double, or triple flashes are acceptable, but a sequence of flashes is not permitted when both the following occur:*
  - *the combined area of flashes occurring concurrently occupies more than one quarter of the displayed screen area; and*
  - *there are more than three flashes (i.e. six changes in luminance as described above) within any one-second period. For clarification, successive flashes for which the leading edges are separated by 360ms or more are acceptable in a 50Hz environment, or separated by 334ms or more are acceptable in a 60 Hz environment, irrespective of their brightness or screen area.*

In terms of potentially harmful patterns, ITU refers that regular patterns clearly discernible in normal domestic viewing conditions (ITU, 2005) should be avoided, but the patterns’ characteristics are not described as in Ofcom.

### 2.3.3. WCAG

The World Wide Web Consortium (W3C) is an international standards’ organization for internet contents, such as Web, HTML, CSS, etc. in order to ensure growth for the Web.

In December 2008 the organization published the Web Content Accessibility Guidelines 2.0 (WCAG), as an update of WCAG 1.0, published in May 1999, a set of standards for web content developers, web authoring tool developers, web accessibility evaluation tool developers, and others who want or need a standard for web accessibility, including for mobile accessibility (W3, 2005). These guidelines cover a wide range of recommendations for making Web content more accessible, including low visibility or blindness, hearing loss or deafness, learning disabilities, cognitive limitations, movement, speech disabilities, photosensitivity, and a combination of all these (W3, 2021). These guidelines are based on already existing guidelines for the broadcasting industry but adapted for computer screens, where content is viewed from a closer distance (using a larger angle of vision).

Regarding photosensitivity epilepsy, the WCAG 2.0 defines in its section *Guideline 2.3 Seizures: Do not design content in a way that is known to cause seizures*, what web content developers should avoid to prevent causing epileptic seizures in their possible viewers: (W3, 2008).

*2.3.1 Three Flashes or Below Threshold: Web pages do not contain anything that flashes more than three times in any one-second period, or the flash is below the general flash and red flash thresholds. This Success Criterion is based on existing specifications in use in the UK and by others for television broadcast and has been adapted for computer display viewing. The 1024 x 768 screen is used as the reference screen resolution for the evaluation. The 341 x 256 pixel block represents a 10 degree viewport at a typical viewing distance. (The 10 degree field is taken from the original specifications and represents the central vision portion of the eye, where people are most susceptible to photo stimuli). The combined area of flashes occurring concurrently and contiguously means the total area that is actually flashing at the same time. It is calculated by adding up the contiguous area that is flashing simultaneously within any 10-degree angle of view.*

W3 gives a similar definition of what it is considered general flash and red flash thresholds, taking into account the differences of viewing web content in our daily devices (monitor, mobile devices, ...) than the usual TVs or cinema screens:

*a flash or rapidly changing image sequence is below the threshold if any of the following are true:*

- *there are no more than three general flashes and / or no more than three red flashes within any one-second period; or*
- *the combined area of flashes occurring concurrently occupies no more than a total of .006 steradians within any 10 degree visual field on the screen (25% of any 10 degree visual field on the screen) at typical viewing distance*

*where:*

- *A general flash is defined as a pair of opposing changes in relative luminance of 10% or more of the maximum relative luminance where the relative luminance of the darker image is below 0.80; and where "a pair of opposing changes" is an increase followed by a decrease, or a decrease followed by an increase, and*
- *A red flash is defined as any pair of opposing transitions involving a saturated red.*

## 2.4. Existing tools

It is difficult for a human to determine if audiovisual content is compliant to the guidelines or not. Hence, a variety of automatic tools, some which are commercial, exist to help content creators analyze video content and check if it is compliant with the current broadcasting regulations.

### 2.4.1. HardingFPA

The Harding Flash and Pattern Analyser (Harding FPA) is a software tool developed by Cambridge Research Systems Ltd., in collaboration with Professor G.F. Harding and his research group. This tool analyses video content for possible flashing images and spatial patterns that can cause seizures on photosensitive epilepsy patients, and confirms compliance with the Ofcom, ITU-R, JBA (Japan commercial Broadcasters Association) (JBA, 2012) and ISO (*ISO 9241-391:2016 - Ergonomics of human-system interaction — Part 391: Requirements, analysis and compliance test methods for the reduction of photosensitive seizure*) (ISO, 2016) guidelines.

The tool analyses a video frame by frame, providing information in case harmful content is detected, such as luminance flashes, red flashes or spatial patterns, that could induce seizures on PSE sufferers. As it can be seen on Figure 2.9, the tool performs an analysis displaying graphics of the analyzed video, in case some of the above-mentioned events occur during its analysis and generates a report with the compliance result and incident information. The tool allows the user to replay the analysis, browse it throughout the video and also displays images with information on the detected harmful pixels. The tool also provides information about incidents in numerical data through .csv files. Both the reports and .csv files can be used for the authors to correct whatever incident is found during the analysis apart from using the tool through the user interface. Harding FPA also includes in its analysis the possibility of an 'Extended Flash Failure', a specification of Ofcom's second guideline, also included in ITU-R and W3, (*...broadcasters should note that it is possible that a sequence of flashing images lasting more than 5 seconds might constitute a risk even when it complies with the guidelines below*) (Ofcom, 2017a). In this case, consistent flashing lasting more than 4 seconds is considered a failure.

Despite being the reference tool of this kind of analysis, Harding FPA is not free from defects, the tool fails to detect any kind of transitions that occur in the first 4 frames of a video, it can also miss luminance or red saturated transitions from time to time.

### 2.4.2. PEAT

The Trace Center's Photosensitive Epilepsy Analysis Tool (PEAT) is a software tool developed by Trace Research & Development Center, in the College of Information Studies (University of Maryland, USA), to identify possible seizure risks in web content applications. It is a free downloadable tool<sup>1</sup> that helps content creators determine if their video content is harmful or if its in compliance with the Web Content Accessibility Guidelines 2.0 (W3, 2008a). PEAT is capable of detecting general flash failures, red flash failures, and extended flash warnings. It does not detect geometric patterns. As it can be seen in

---

<sup>1</sup><https://trace.umd.edu/peat/>

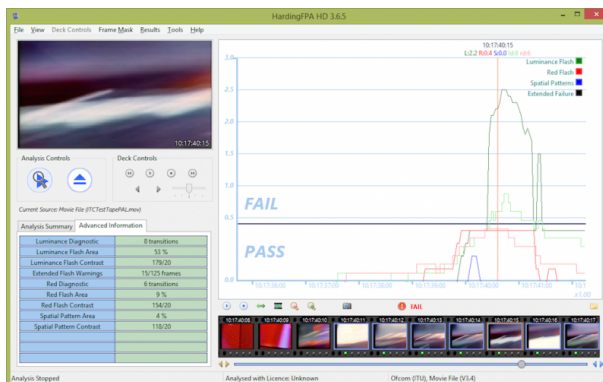


Figure 2.9: HardingFPA tool GUI

Figure 2.10, PEAT also provides graphical information about the possible harmful events that take place in the analyzed videos, being able to move frame by frame to accurately check where the analysis is failing.

PEAT was created to provide a way for web designers and developers to ensure that their web content is not harmful for possible PSE viewers, however, the tool can't be used to analyze commercial products. PEAT uses a modified version of the proprietary analysis engine from Harding FPA, thus it is not permitted to use it for commercially or to analyze non web content. As it also happens with HardingFPA, this tool has some issues detecting transitions in the beginning of the videos, as it does not detect transitions during the first 10 frames.

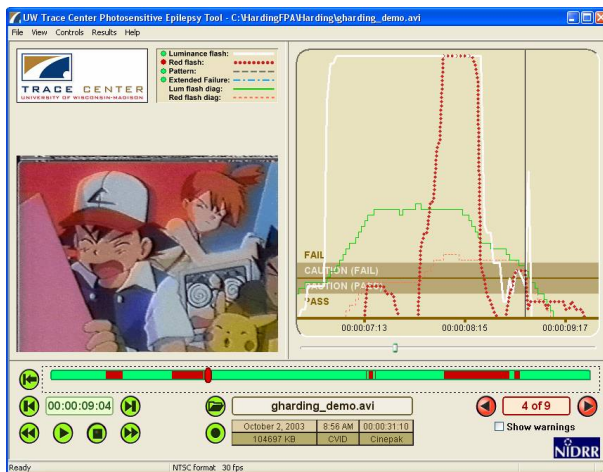


Figure 2.10: PEAT tool GUI

### 2.4.3. Flicker Check

Hitachi Flicker Check is a software tool developed by Hitachi Solutions Ltd., a specialized IT company from Japan, founded in 1970. Hitachi started developing video processing solutions in 1995, providing a tool that automatically detects harmful effects that can be easily verified by its authors. It supports ITU-R and JBA guideline, and performs additional tests searching for contrast changes and subliminal images (Hitachi-Solutions Inc., 2007).

This tool also performs an analysis with graphical information of the analyzed video and clicking in the plot leads to the suspected harmful segment where the video can be viewed for later verification (note Figure 2.11). These segments indicate the guideline violations, which are categorized and shown with icons, so the content creators can easily recognize where the incorrect sections are displayed and their type.



Figure 2.11: Hitachi Flicker Check tool GUI

## 2.5. Conclusions

As explained in this chapter, PSE is a serious issue that anyone could be affected by. Even without suffering from PSE, every person has their own threshold for photosensitive epilepsy, and could experience a seizure if triggered. For this reason, and thanks to the research of experts in the field, it has been possible to develop certain regulations that prevent the appearance of conditions that can trigger an epilepsy attack. These guidelines have been key for adjusting the conditions for this project, they have been included as threshold parameters that help determine whether video content is susceptible to triggering attacks. Whenever these regulations are updated, the tool can be updated to the new thresholds if needed.

In addition, the possibility of using existing tools (such as HardingFPA or PEAT) has served to constantly compare the results of this project, focusing on how to comprehend the metrics, read the results and improve them.



# Chapter 3

## Detecting harmful videos

After researching how videos can induce seizures in PSE patients, a tool has been designed to detect possible harmful video content. This chapter describes the algorithms used to detect the videos that are not in compliance with the guidelines from section 2.3. It covers all the steps that need to be executed in order to detect harmful flashes and geometric patterns, and the criteria to verify if the guidelines for photosensitivity have been violated.

### 3.1. Main pipeline

A video is a visual media product featuring moving images, with or without audio. In terms of harmful video content, in regard to PSE, all the frames must be analyzed. Frames are the sequence of images that are displayed through the length of the video. Thus, the detection tool will read the video frame by frame in order to detect possible flashes and patterns to check compliance with the guidelines.

The whole process is represented in figure 3.1. The tool's input is a video file. For each frame in the video, the luminance values are extracted. Once the luminance has been obtained, the program runs both the flash and pattern detection algorithms in order to generate the required data to check the video's compliance. When the program ends, the pass or fail result is generated.

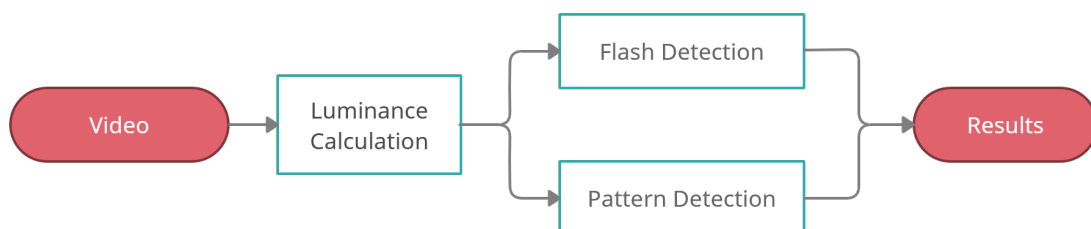


Figure 3.1: Program flow

In the next sections, each step from the pipeline will be explained.

## 3.2. Luminance calculation

Brightness is the subjective measurement of light. It is the perception of each individual to the amount of light emitted by an object. However, although it is possible to adapt the brightness of a digital screen, brightness has no measurement unit. As it does not have any defined range of values.

Luminance ( $Y$ ) is the photometric measure of the luminous intensity per unit area of light travelling in a given direction. It can be thought as the objective measurement of brightness. It's international system unit is candelas per square meter ( $cd/m^2$ ).

The luminance calculation is the first step of the tool's pipeline, in which a video frame in RGB is received to convert the RGB values into luminance values. Thus, luminance is calculated per each frame in the video, and each pixel that is part of the frame has its own luminance value. The conversion is done by applying a formula to the color space values in every pixel. As a frame is essentially a pixel matrix, where the pixel information varies depending on the color space of the image, hence the pixel values can be used for the conversion. From this point on, images will also be referred to as matrices.

There are two available formulas to use for this conversion, both of which return a float matrix of the corresponding luminance values of the frame.

### 3.2.1. Luminance at the display

This conversion establishes a relationship between the luma component of a pixel and the voltage of a TV screen. Luma is the weighed sum of the RGB values of a pixel, luma is gamma compressed, assuming a gamma value of 2.2. Therefore, the measurement for this type of luminance is in  $cd/m^2$ . Firstly, the frame is converted from RGB to the luma-chroma ( $Y'CrCb$ ) color space. Then formula 3.1 is applied to the pixels, the resulting values range from 0 to 200 in  $cd/m^2$ . The guidelines allow the assumption of this luminance range.

$$Y(cd/m^2) = 413.435(0.002745 * Y' + 0.0189623)^{2.2} \quad (3.1)$$

However, this method can be inaccurate if the incorrect conversion to  $Y'CrCb$  is chosen. There are different formulas depending on the video encoding standard. If this formula is used to calculate the luminance, the encoding standard needs to be taken into account to correctly convert RGB to  $Y'CrCb$ .

### 3.2.2. Relative luminance

Relative luminance are normalized luminance values which are calculated from linear components. RGB, the frame's default color space, is non-linear as it suffers from gamma correction. As a result, the first step is to convert the color space to the linear sRGB color



space with formula 3.2.

$$\begin{aligned}
 &\text{if } R_sRGB \leq 0.04045 \text{ then } R = R_sRGB/12.92 \text{ else } R = ((R_sRGB + 0.055)/1.055)^{2.4} \\
 &\text{if } G_sRGB \leq 0.04045 \text{ then } G = G_sRGB/12.92 \text{ else } G = ((G_sRGB + 0.055)/1.055)^{2.4} \\
 &\text{if } B_sRGB \leq 0.04045 \text{ then } B = B_sRGB/12.92 \text{ else } B = ((B_sRGB + 0.055)/1.055)^{2.4}
 \end{aligned} \tag{3.2}$$

Once the frame has been converted to sRGB, formula 3.3 is applied to obtain the luminance values where R, G and B are the different color components from the sRGB color space. The resulting values range from 0 to 1.

$$Y = 0.2126 * R + 0.7152 * G + 0.0722 * B \tag{3.3}$$

This method is the default luminance method used in the tool, as it does not require the knowledge of the encoding standard in a video.

### 3.3. Flash Detection

The shift in luminance or red saturation cannot be obtained from frame to frame. The variation can be prolonged throughout several frames. To properly detect transitions, and analyze videos for flash detection, it is required to evaluate the changes considering that variations above the thresholds can happen in extended periods of time. As a result of this requisite, the flash variation values must be accumulated frame by frame. Meaning that, if the accumulated difference reaches the threshold, a transition will have occurred.

Having these considerations in mind, the algorithm is composed of the following steps, based on (Carreira et al., 2015), for both luminance and red saturation flashes:

1. Extract the values of each frame (section 3.3.1).
2. Get the difference between each consecutive pair of frames (section 3.3.2).
3. Get the maximum average difference above the threshold area (section 3.3.3).
4. Accumulate the average difference by sign (positive or negative) (section 3.3.4).
5. Check if a new transition has occurred (section 3.3.4).
6. Register and track amount of transitions (section 3.3.5).
7. Check if the transitions in second are above the thresholds (section 3.3.6).



Figure 3.2: Flash Detection Algorithm

### 3.3.1. Extract frame values

In the case of luminance flashes, the threshold values are either  $20cd/m^2$  or a change in 10% relative luminance (0.1). The values of each frame are calculated as described in section 3.2 to get the current luminance matrix.

The red saturation matrix is also calculated with the frame in the sRGB color space from section 3.2. When it comes to color theory, saturation is the range from pure color to gray. Saturation is sometimes referred to as color intensity or chroma.

As defined by W3's guideline 2.3.1 W3 (2008b), saturated red color transitions occur when:

$$\blacksquare R/(R + G + B) \geq 0.8 \quad (3.4)$$

$$\blacksquare (R - G - B) * 320 \geq 20 \quad (3.5)$$

Condition 3.4 is the mathematical definition for saturated red where R, G and B are the color components from sRGB. Whenever a pixel meets this condition, the pixel is considered to be saturated red. Condition 3.5 is the minimum needed value of change for a red saturation transition to take place. Hence, it is the threshold value for red saturated transitions.

As transitions from, or to, saturated red must comply with condition 3.4, a binary mask is created by applying a threshold using function 3.6. This operation converts the relevant information (saturated red pixels) to white, and the rest are set to black, thus constructing a binary image. The process can be observed in figure 3.3.

$$redMask = \begin{cases} 255 & \text{if } redSat(x,y) \geq 0.8 \\ 0 & \text{otherwise} \end{cases} \quad (3.6)$$

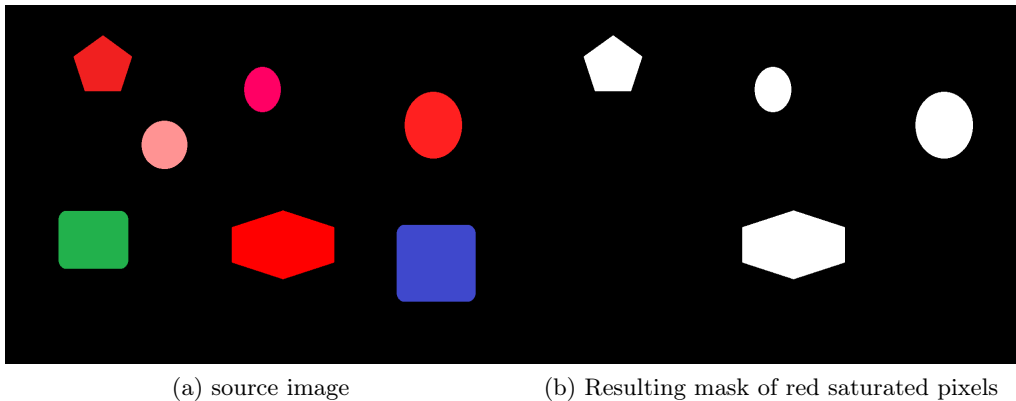


Figure 3.3: Extraction of saturated red pixels into a binary image

Once the mask has been obtained, the change value of the current frame's red saturation is calculated using formula 3.5. This operation is performed filtered with the *redMask*, as

to only operate in the red saturated pixels. The resulting matrix, *redCoefficient*, is of type float and stores the red saturated pixel values. A threshold operation is done to ensure that all values are positive by function 3.7.

$$redCoefficient = \begin{cases} redCoefficient(x, y) & \text{if } redCoefficient(x, y) \geq 0 \\ 0 & \text{otherwise} \end{cases} \quad (3.7)$$

### 3.3.2. Frame Difference

To detect changes throughout the video, the variation among frames must be obtained. The frame difference is performed on the flash value matrices, meaning that the matrices used are the luminance values matrix and the red saturated values matrix from the previous section. The frame difference is computed for each pair of consecutive frames. The difference values can be:

- Positive: increase of the flash values.
- Negative: decrease of the flash values.
- Null: no variation of the flash values.

When there is no change between frames, it means that the flash trend has not changed. Thus, the variation trend can only be either incremental or in decline.

To get the variation trend, the first step is to subtract each pair of consecutive frames (equation 3.8). For both luminance and red saturation, the subtraction is done using the frame difference of their respective values defined in section 3.3.1 and results in another two matrices of type float, *lumDiff* and *redDiff*. The frame difference of the first frame in the video is never computed, as it marks the initial state of the flash values in the video.

$$frameDiff = frame_n - frame_{n-1} \quad (3.8)$$

Figure 3.4 is a frame sequence of gameplay footage from Apex Legends video game (Respawn Entertainment, 2019) in grayscale. The process of applying equation 3.8 using the luminance matrices can be observed in the images from figure 3.5 however, the images are displayed as the absolute luminance frame difference as negative values cannot be displayed in images.

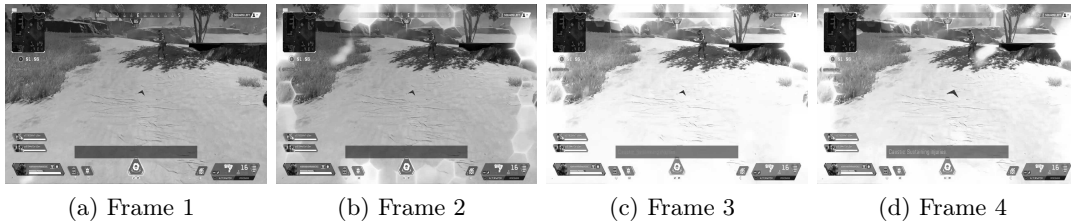


Figure 3.4: Frame sequence from Apex Legends

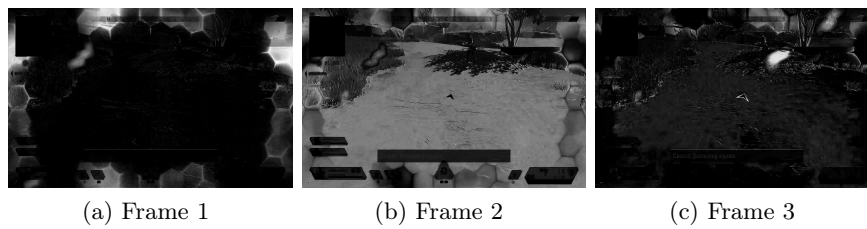


Figure 3.5: Absolute luminance difference of frames from figure 3.4

The process can also be observed for red saturation in figure 3.6, using the absolute frame difference for easier visualization as well. The images represent the red saturation absolute difference of the gameplay footage from *It Takes Two!* (Hazelight Studios, 2021)

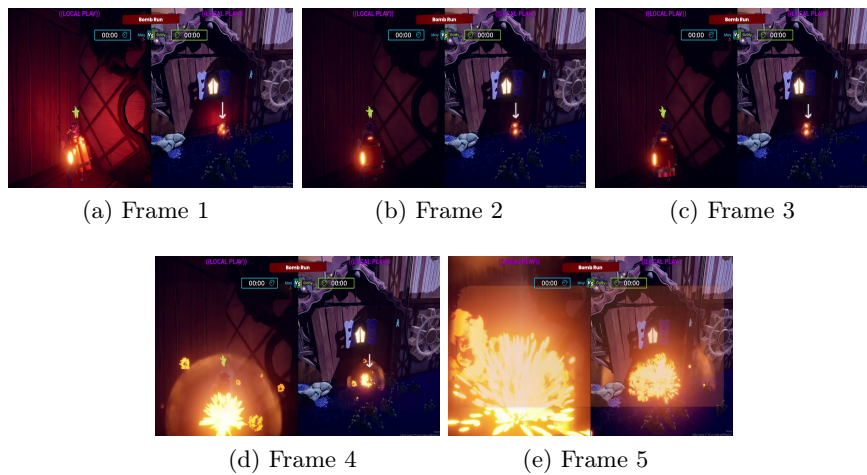
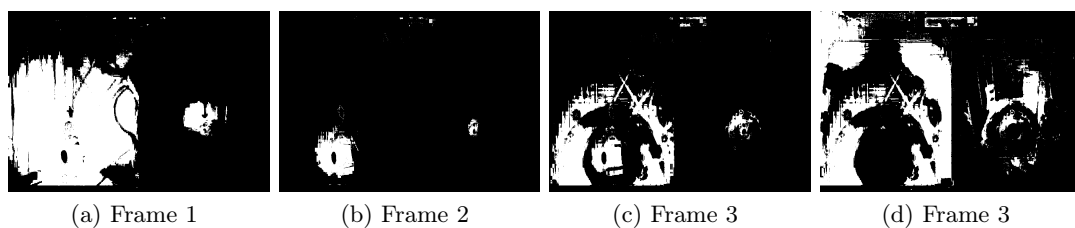
Figure 3.6: Frame sequence from *It Takes Two!*

Figure 3.7: Absolute red saturation difference of frames from figure 3.6

### 3.3.3. Flashing safe area

It is expected to have variations in luminance and red saturation throughout a video, even above the flash thresholds. These changes are not considered harmful unless they also overpass a certain flash area.

Thus, at this point in the algorithm, it cannot be known if the changes in value from the frame difference matrices are harmful or not. Though it can be known if there has been enough variation that might trigger the flash thresholds. Therefore, with the difference

values of each flash type, it is possible to check for the variation area and then set the flash trend mentioned in section 3.3.2.

The standard allowed flashing area is up to 25% of the screen display. If the area is equal or above, then flashing may be occurring. However, the parameters used to calculate the percentage are part of the common viewing conditions of Rec. ITU-R BT.500-11 BT5 (2002). This document was published in 2002 and only considers TVs as screen displays.

In the current time, media is consumed in multiple devices. Video games are played using TVs, screen monitors and even mobile phones. All this different screen types have different sizes and common viewing distances. But the one that tends to be viewed the farthest away from, is the TV. This results in being the device that allows the biggest safe area, meaning that 25% of the screen display is not a suitable parameter for the flash area size.

The allowed flashing area is equivalent to 0.006 steradian, or one fourth of a  $10^{\circ}$  visual field in pixels. The W3's guidelines offer the area covered by  $10^{\circ}$  degree visual field in a screen of 15-17 inch viewed at 22-26 inches. These screen parameters are the most common for PC monitors.

The resulting area is of 341 x 256 pixels (on a display resolution of 1024 x 768). It allows obtaining the area of any video resolution using cross multiplication. The criteria indicates that the area is one fourth of the resulting rectangle. This means that to obtain the maximum average of variation for both the positive and negative values, a sliding window algorithm should be executed calculating the averages that are above the area inside the rectangle, which results in very slow performance as the sliding window moves through the image pixel by pixel. However, in (Wilkins A., 2005) it is explained that this check can be performed in the whole screen without needing to analyze it by sections, as when viewing a screen the diagonal is covered by our central vision. As a result, the variation can happen at any point of the screen display. Seeing as the check is performed in the whole screen at once, the area is set to a  $10^{\circ}$  degree visual field, which is roughly 10% of the screen display.

As explained in this section, the aim of this step in the flash detection algorithm is to confirm if the variation from frame to frame has reached a certain area size. If so, then flashing may be occurring. To perform this validation, a series of steps, common for both flash types, are carried out:

1. Split the frame difference into two binary matrices for positive and negative values.
2. Detect continuous areas of variation for positive and negative values.
3. Calculate the average frame difference of the positive and negative values.
4. Set the flash trend.

The connected components algorithm, in computer vision, is an algorithm that establishes relationships between pixels when the neighboring pixels are non-zero positive values, making a component out of a set of neighboring pixels. Components are also referred to as blobs in computer vision. In figure 3.8. the source image is a binary image with different white rectangles, each representing a different component. Image (b) is the result of the blob detection done by the connected components algorithm, which correctly distinguishes each rectangle and properly visualizes them with different colors.

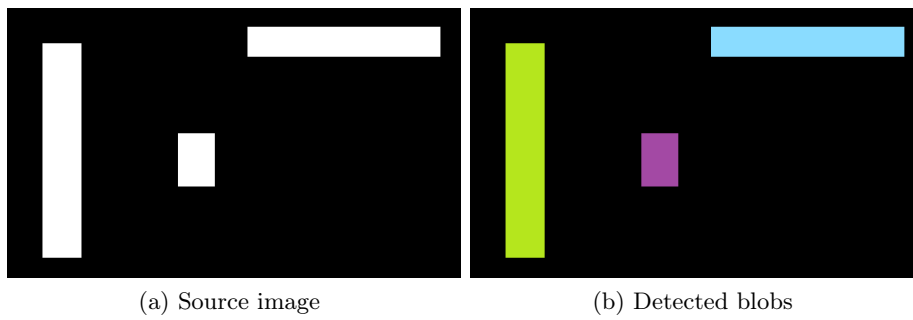


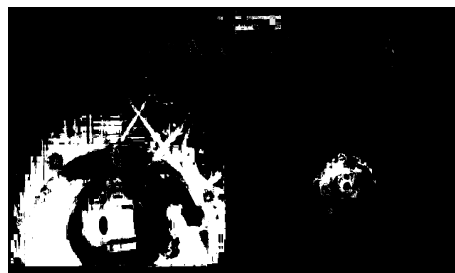
Figure 3.8: Connected components: blob detection

Nevertheless, this algorithm only works with positive values and the matrix used as input must be a binary matrix. Hence, the need to split the negative and positive values into two different binary matrices, where non-zero values are represented as white.

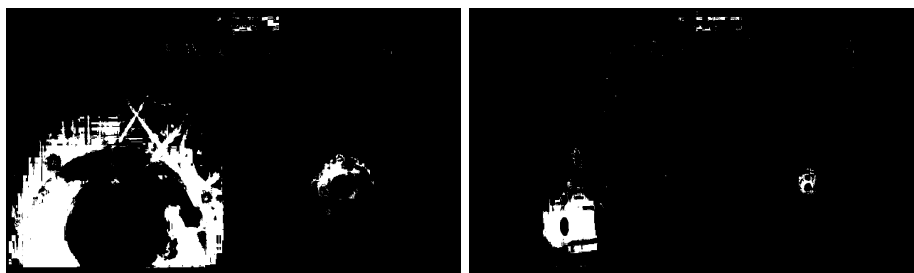
The frame positive difference matrix is set using function 3.9 whereas the negative difference is set using function 3.10. The process can be observed in figure 3.9, where frame 3 from the frame sequence of figure 3.7 was taken and its values were split into two matrices, representing its positive and negative difference.

$$positiveDifference = \begin{cases} 255 & \text{if } frameDifference(x,y) > 0 \\ 0 & \text{otherwise} \end{cases} \quad (3.9)$$

$$negativeDifference = \begin{cases} 0 & \text{if } frameDifference(x,y) > 0 \\ 255 & \text{otherwise} \end{cases} \quad (3.10)$$



(a) 3.7 Frame 3



(b) Positive Difference

(c) Negative Difference

Figure 3.9: Frame difference

Before running the connected components algorithm on both matrices, the non-zero pixels of the binary matrices are counted, if the minimum area of variation has not been reached the next steps are skipped.

Once the blobs are detected, each blob is isolated into a different matrix with the positive or negative frame values from the difference matrix that correspond to the blob area. The sum of the positive or negative difference is then computed for all blobs and stored along the blob area. Afterwards, this data is looped to select the blobs with the highest sums and areas until the minimum safe area has been reached, then the average difference is calculated. In figure 3.10, the blob selection results can be observed. The selected components from the frame sequence of figure 3.5 are highlighted in the different images.

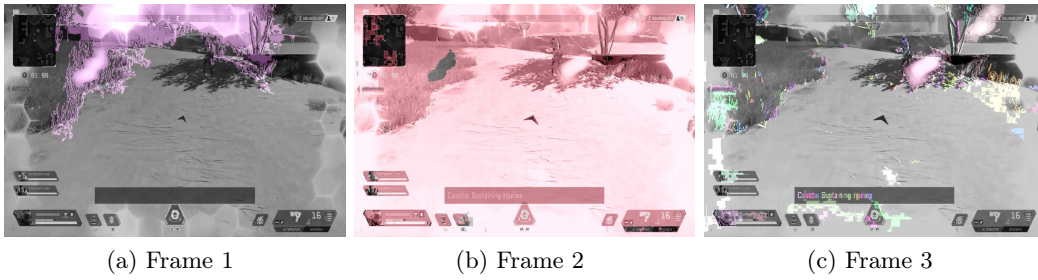


Figure 3.10: Selected blobs from 3.5 frame sequence

As the flash trend can only be either incremental or in decline, it is set by the highest absolute variation in between frames, which means that the flash trend, the average difference, for both luminance and red saturation is set through condition 3.11. Hence, the flash safe area check returns the value of the flash trend (for each flash type), whether it is positive or negative.

$$frameAverageDifference = \begin{cases} positiveAverageDifference & \text{if } [positiveAverageDifference] > \\ & [negativeAverageDifference] \\ negativeAverageDifference & \text{otherwise} \end{cases} \quad (3.11)$$

### 3.3.4. Value accumulation and transition detection

The flash trend is received as a float variable, the *frameAverageDifference*, calculated in section 3.3.3. Because flash variations can take place throughout multiple frames, the frame average difference must be accumulated to detect changes between frame sequences. The increase or decrease of flash values can be gradual.

The accumulation procedure is described in algorithm 1 and follows the options explained in section 3.3.2 regarding the possible flash value tendencies. Thus, the average difference is accumulated if the flash trend remains the same or if there has been no variation among consecutive frames. Otherwise, the flash accumulation is reset to the new average difference value.

This procedure also takes note that the compliance criteria requires to check the flash rate in any second of the video, hence the fact that, although the average difference is

accumulated by sign, it is also stored in a list. This way, whenever a full second of the video has been analyzed, the latest average difference value is removed from the list and subtracted from the total accumulated value. This process remains as long as the list size equals the amount of frames in one second, meaning that if the flash trend is reset (the tendency has changed) or the list has not reached the size of frames in one second, this step is skipped.

A transition occurs whenever conditions 3.12 and 3.13 are true, which is when the flash tendency remains the same. Or, when condition 3.14 is true and thus the flash tendency has changed. These conditions prevent counting multiple times transitions that take place through several frames, thus this step returns a boolean that is only true if a new transition has occurred.

Transition conditions:

$$\begin{aligned} & \blacksquare (avgDiffAcc \leq 0 \ \&\& \ lastAvgDiffAcc \leq 0) \\ & \quad || (avgDiffAcc \geq 0 \ \&\& \ lastAvgDiffAcc \geq 0) \end{aligned} \tag{3.12}$$

$$\blacksquare [lastAvgDiffAcc] \geq Threshold \tag{3.13}$$

$$\blacksquare [avgDiffAcc] \geq Threshold \tag{3.14}$$

---

**Algorithm 1** Flash value accumulation & transition detection
 

---

**Result:** newTransition

**if** *avgDiffInSecond.Count* == *avgDiffInSecond.Capacity* **then**

  | *lastAvgDiffAcc* -= *avgDiffInSecond*[0];

  | *avgDiffInSecond.Remove*(0);

**end**

**if** *SameSign*(*lastAvgDiffAcc*, *avgDiff*) **then**

  | *avgDiff* += *lastAvgDiffAcc*;

**else**

  | *avgDiffInSecond.Clear*();

**end**

*avgDiffInSecond.Add*(*avgDiff*);

*newTransition* = *IsFlashTransition*(*lastAvgDiffAcc*, *avgDiff*);

*lastAvgDiffAcc* = *avgDiff*; //new start acc value

**return** *newTransition*;

---

### 3.3.5. Flash Transition Count

The transition count allows knowing the amount of flash transitions in any given second of the video. When a new transition is detected, as explained in section 3.3.4, it is registered. So it is always possible to know the amount of transitions that have occurred in the last second.



To keep track of the amount of transitions in any given second of the video, all transitions are registered in 3 different lists, *luminanceTransitions*, *redTransitions* and *totalTransitions*. The lists *luminanceTransitions*, *redTransitions* only serve to better understand and visualize the data of the transitions of each type of flash, as they represent their corresponding flash transitions data. All lists are the size of the amount of frames in one second of the video. The list *totalTransitions* is the common flash transitions data, it is the list that is used to later check the compliance with the guidelines.

The transition lists serves as a way to easily keep track of all the transitions that have occurred from that moment in the video to exactly one second before. When a new transition is registered, the total transition sum is increased, and then it is subtracted by all transitions that have already been discarded, all transitions that are no longer part of the last second in the video.

In regard to the extended failure, a similar procedure is followed. There is a list of the size of the frames in 5 seconds of the video, that registers transitions whenever the transitions in second are between 4 and 6, which corresponds to flashing between 2 and 3 Hz.

To easily access the current second's transition count, the total value of transitions that have occurred in the video, minus the transitions that have passed the current second's window, are stored in an integer value that corresponds to the amount of transitions that have occurred in the last second, *transitionsInSecond*.

### 3.3.6. Check criteria

The guidelines estate that no flashing may occur at more than 3Hz, and that extended flashing for 5 seconds can also be harmful. As a flash is defined as a pair of opposing transitions, any given second of the video that surpasses 6 transitions will fail against the compliance criteria. Also, any period of time that reaches at least 4 seconds of flashing between 2 and 3 Hz will be an extended flash failure. With the information of the lists from section 3.3.5, it is easy to check the compliance criteria.

Anytime that the *transitionsInSecond* variable surpasses 6 transitions, flashing is at more than 3Hz and thus a flash incident has occurred, hence the frame is marked as a flash fail. If not, anytime the *transitionsInSecond* variable is above 4 transitions, which means flashing at least at 2Hz, the amount of frames in the extended failure list is increased, if it reaches the size corresponding to 4 seconds of the video, an extended failure incident will have occurred, and thus the frame is marked as an extended failure.

Two boolean variables exist to set them to true in case a flash incident or an extended failure incident occur. This way, both types of flash incidents can be reported to the user when the video analysis ends.

## 3.4. Pattern Detection

As described in section 2.2.2, there are multiple variations to geometric patterns. In order to analyze a pattern, its information is required. Patterns can be reduced to sets of lines, straight or circular, that are composed of similar discernible stripes with similar luminance values and that occupy a certain area of the screen display.

The pattern detection algorithm, based on (Rodrigues, 2015), is a series of operations to extract pattern information, for each video frame, that consists in:

1. Detect probable pattern lines (section 3.4.1).
2. Analyze possible patterns from detected lines (section 3.4.2).
3. Estimate risk (section 3.4.3).
4. Check compliance criteria (section 3.4.4).

As with the flash detection algorithm, the video is read frame by frame, and each frame goes through the pattern detection flow. Once all the pattern data has been generated (area, number of stripes and average stripe luminance), a risk can be estimated from a series of formulas based on the proportion of affected patients from the studies of Wilkins et al. (Wilkins A., 2005). Afterwards, the compliance criteria is checked to give the pass or fail result of the video, depending on if the guidelines have been violated or not.

When running the pattern detection, the video frame comes in the RGB color space. The very first task is to convert it to grayscale (a single channel matrix with values ranging from 0 to 255), as it is needed for the different steps in the algorithm.



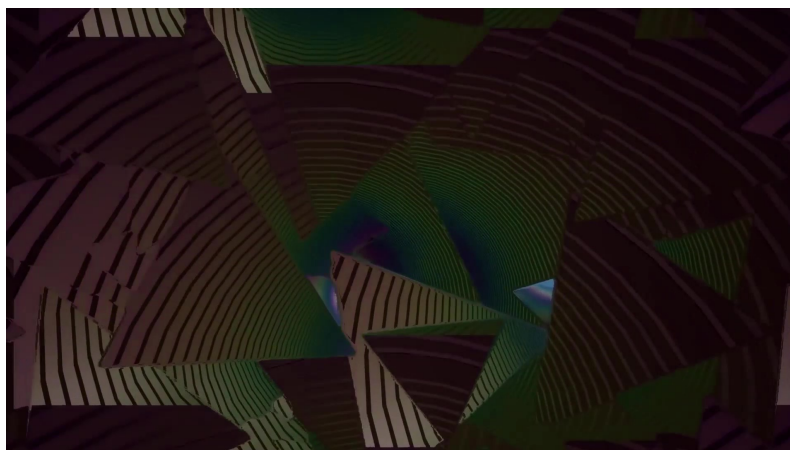
Figure 3.11: Pattern Detection Algorithm

### 3.4.1. Line detection

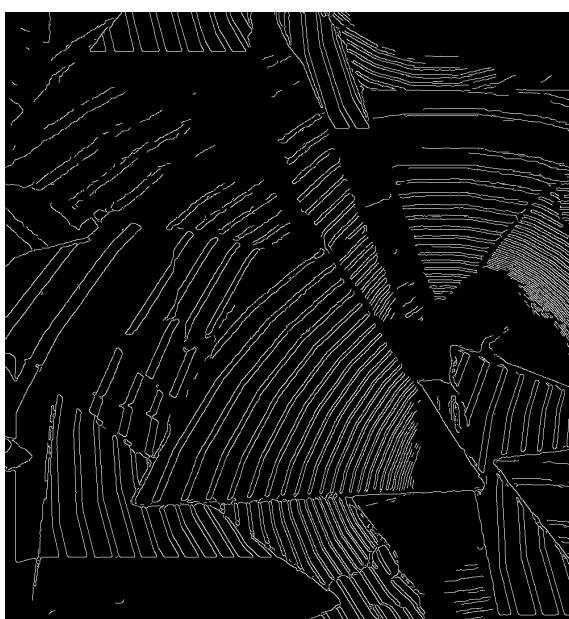
As mentioned in the previous section, patterns are sets of lines. Thus, the first step is to detect lines that are likely part of a pattern. As the lines forming a pattern can be either straight or circular, the detection is divided into two different methods. One that covers straight line detection, the other detects circular segments. It is normal for a video frame to contain lines, so the detection phase focuses on selecting probable pattern lines from those detected. If pattern lines have been detected. The pattern analysis is then executed to analyze the line regions.

Before running the line detection, two pre-processing steps are carried out and applied to the grayscale frame, noise reduction and edge detection. To properly detect lines in an image, edge detection must be performed, otherwise the results are most likely to be poor as the used detection methods expect the input to be the result of an edge detection algorithm. Reducing the noise in an image improves the edge detection significantly, as can be seen in figure 3.12, the edges, which are the white lines in the image, are smoother and less noisy. If the image is not filtered, it could provoke issues due to worse line detection.

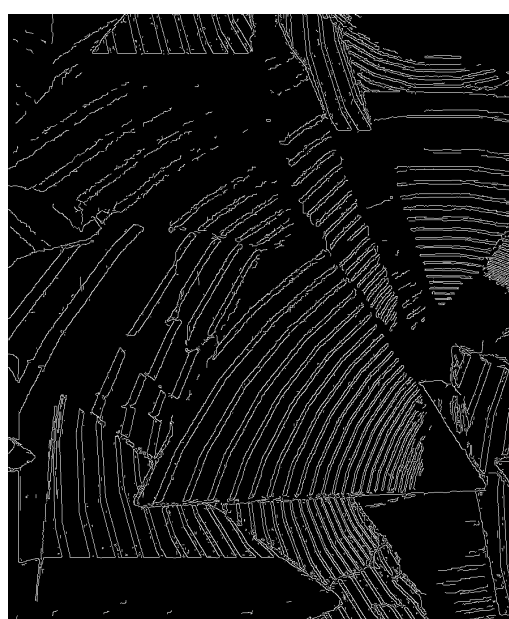
A median filter is a digital filter that smooths and blurs an image, which results in noise reduction of the said image. A kernel size of 5 gave the best results for this operation. The filter is applied to the grayscale frame and stored in a separate matrix. Afterwards, the Canny edge detection algorithm is performed on the blurred resulting matrix, as to extract the image's structural information into a binary matrix containing the edges of the frame.



(a) It Takes Two! frame with a curved lined pattern



(b) Canny output with noise reduction



(c) Canny output without noise reduction

Figure 3.12: Canny output with and without noise reduction

Canny edge detection has three different parameters: the lower and upper thresholds and the aperture size. It can be assumed that the aperture size is always three, but the lower and upper thresholds cannot be static, otherwise the canny edge detection won't work for all video frames, as not all frames have the same intensity values in grayscale. Figure 3.13 is an example of the result of the Canny algorithm with the wrong threshold parameters and the best values for that particular image. It can be observed that, with the correct values, the edge detection is improved drastically. Thus, it is very important to apply the correct values.

To apply the correct values, the threshold parameters must be assigned automatically, as it is not possible to define them for all the frames of one video and, most importantly, because it would be very verbose and tedious to set global values for all the frames in a video for each video to analyze manually. Also, this would still not result in having the best values.



(a) Canny output with wrong threshold values (b) Canny output with best threshold values

Figure 3.13: Canny output with different thresholds from frame in figure 3.12

The method that offered the best results for assigning automatically the values, is to calculate the median of the frame's grayscale values, and then apply the formulas 3.15 and 3.16 to obtain the thresholds. By doing this, the threshold parameters are always set for each frame depending on its grayscale intensity values. The sigma value is set to 0.33. In the example from figure 3.13, the incorrect parameters were set to 100 and 150 whereas using this method, the thresholds were set to 14 and 27 and hence the result was improved.

$$cannyTh1 = \text{Max}(0, (1 - \text{sigma}) * \text{median}) \quad (3.15)$$

$$cannyTh2 = \text{Max}(255, (1 + \text{sigma}) * \text{median}) \quad (3.16)$$

#### 3.4.1.1. Straight line detection

The straight line detection extracts straight line information from the canny output of section 3.4.1. When this method is executed, it returns a set of lines in the form of both points in the image that compose the line. With this information, lines can be selected and discarded per the line criteria. The line criteria is used to determine if the detected lines are part of a pattern, thus similarity relationships are made between lines as to obtain probable patterns.

The line selection criteria which determines if lines form a potential pattern are:

1. Minimum length: the line detection method detects only lines of a minimum length, this length varies depending on the video resolution.
2. Orientation: the detected lines are grouped by their orientation, lines of a similar or near angle are grouped together as it is assumed that they are, approximately, parallel lines.
3. Midpoint deviation of parallel lines: this step serves as another line grouping operation to ensure that lines that comply an angular relation, are actually related.
4. Minimum number of lines: if a line group has the minimum number of lines required, then the group of lines is a potential striped pattern.

After all the steps are performed on the detected lines, the groups that meet the criteria are returned as to run the pattern analysis on them.

### 3.4.1.2. Circular line detection

The circle line detection is performed on the canny output from section 3.4.1, and only if no straight lines have been detected. To obtain the information of a curved line, the contours of the line must be detected. When the contour detection is executed, points from an edge are extracted and stored in a vector of vectors. Meaning that each contour has a position in the vector and stores all its points in the vector of that position. Circular lines (contours) also have their own criteria to check if a potential pattern has been detected.

To verify if the contour is circular or curved, the contours' points are approximated to a polygonal shape to check its number of vertices. Approximating the contour to a polygonal shape allows to obtain an estimated number of vertices from the shape. Curved lines are composed of many vertices, meaning that contours that have few are most likely not curves. The minimum number of vertices needed for a contour to be considered a curve or a semicircular shape is 10, otherwise the contour is discarded.

Also, the center of mass for each detected contour is computed. Circular shapes contain the center of mass inside their shape, if not, the contour is discarded.

Because the canny output can provide somewhat noisy edges, even when reducing the noise in the frame, the two mentioned checks (center of mass and number of vertices), can be incorrect as the contour vertices can be miss-estimated. Dilating the Canny image provides a solution to this problem. Dilation is a morphological operation that, in this case, expands the edges and makes them thicker, which results in higher curve line detection.

When the circular line detection is done, and all contours have been either kept or discarded, the contour information is returned as to run the pattern analysis on it.

## 3.4.2. Pattern Analysis

The pattern analysis serves as a process to extract information to gain the needed parameters for the risk estimation. Using the line information from the previous section, pattern information can be gathered from the video frame.

The pattern is first separated from the rest of the frame, then a series of operations allow collecting data on the number of stripes, their luminance and the area the pattern occupies, as displayed in figure 3.14.



Figure 3.14: Operations for pattern data extraction

### 3.4.2.1. Pattern Isolation

The sets of lines from the section 3.4.1 are used to make a binary image to apply as a mask to the frame.

First, the convex hull of each set of lines is obtained. The convex hull is the polygon that will cover the entirety of a set of lines, which allows setting the region to white as to

form the mask.

Second, once the binary mask has been formed, it is applied to the grayscale frame as to extract only the region of interest from the frame, the pattern. This operation results in a matrix which information is the extracted pattern in grayscale.

The process can be observed in figure 3.15, in which the pattern is extracted into a new image leaving the background behind (to emphasize the process, the video frame has been displayed in RGB instead of grayscale).

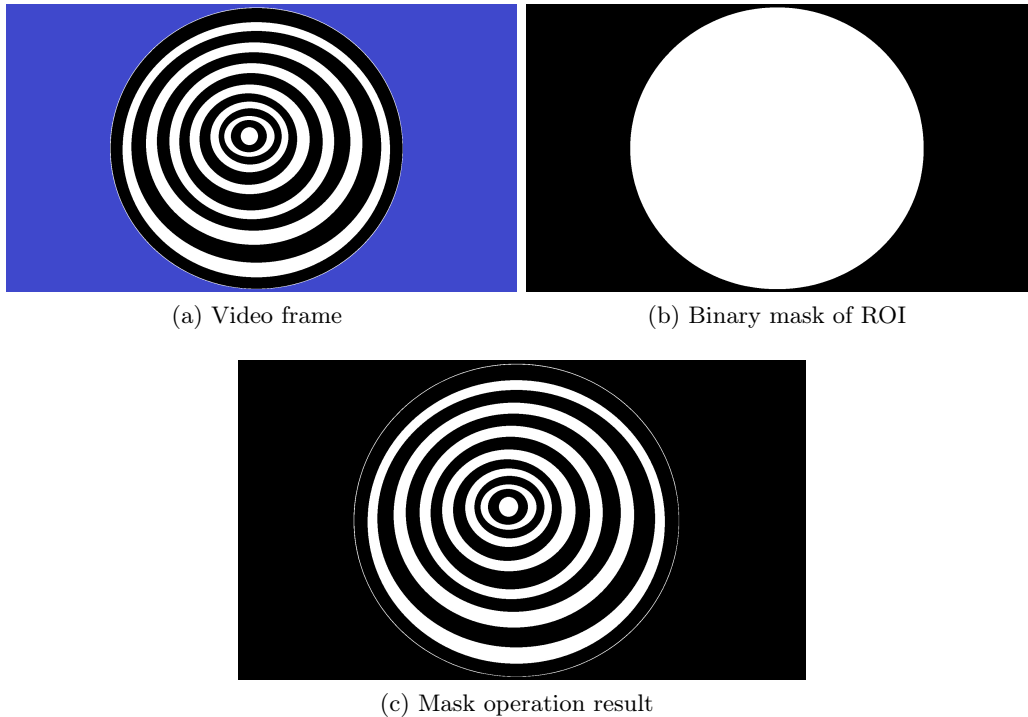


Figure 3.15: Pattern isolation process

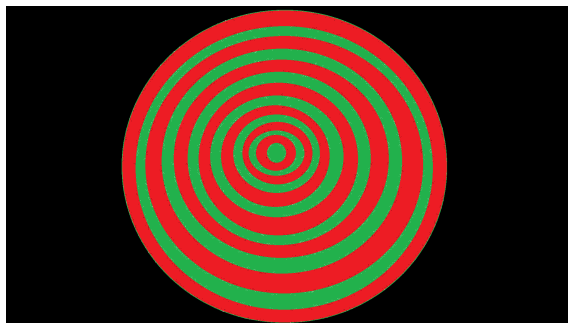
#### 3.4.2.2. Stripe separation

With the regions of interest isolated, the dark and light stripes of the patterns need to be split into separate matrices. An Otsu threshold operation is used for this task, it is applied to the isolated pattern matrix. The Otsu algorithm tries to find a threshold value to separate foreground and background by maximizing inter-class variance of the pixel intensity. In this case, the light stripes of the pattern will be considered the foreground and the dark stripes the background.

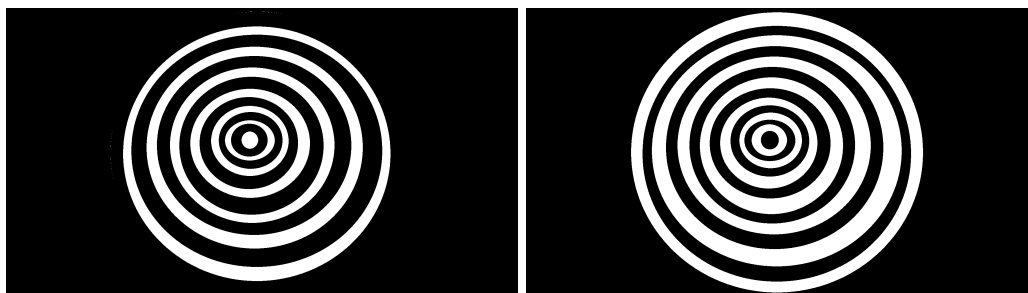
Therefore, the Otsu threshold operation will return the binary image of the positive components, the light stripes, from the pattern. By inverting this image, the negative components can be obtained. However, it is important to perform a bitwise AND operation on this image with the mask from section 3.4.2.1. This ensures having the correct components for the dark stripes, otherwise, the background could be part of the dark components. The stripe separation process can be observed in figure 3.16.

Afterwards, both images are eroded. Erosion is a morphological operation that applies

a structuring element to an input image. Erosion can be used to achieve different goals, such as removing noise from an image, or isolating individual elements in an image. In this case, element isolation is the desired effect. When obtaining the negative and positive components, some stripes could be connected, eroding prevents issues regarding separate stripes that appear to be a single one. The stripe separation is, then, a process to obtain two matrices that store the light and dark stripes respectively.



(a) Positive and negative components highlighted as green and red respectively of an isolated pattern



(b) Positive components

(c) Negative components

Figure 3.16: Negative and positive stripes separated into different images

### 3.4.2.3. Connected Components

As explained in section 3.3.3, the connected components algorithm establishes relationships between non-zero neighboring pixels. At this point the stripes of the pattern have been separated into two different images (one with the light stripes, the other with the dark ones), the connected components algorithm serves as a way to detect each stripe of the pattern and access the region it occupies for both the negative and light stripe matrices. In figure 3.17 the result of the connected components can be observed, each stripe is colored differently as to inform that it is its own component.

After running the algorithm, each component is checked to confirm if the minimum area has been reached. If the component is very small, it is discarded. If not, its area and its average luminance is calculated and stored. Thus, this process provides two lists with the area and average luminance of each stripe.

The detected connected components result is also used to gather pattern incident information that can be viewed after the analysis. Figure 3.18 displays the pattern stripes drawn over the original video frame, which helps users understand the incident.

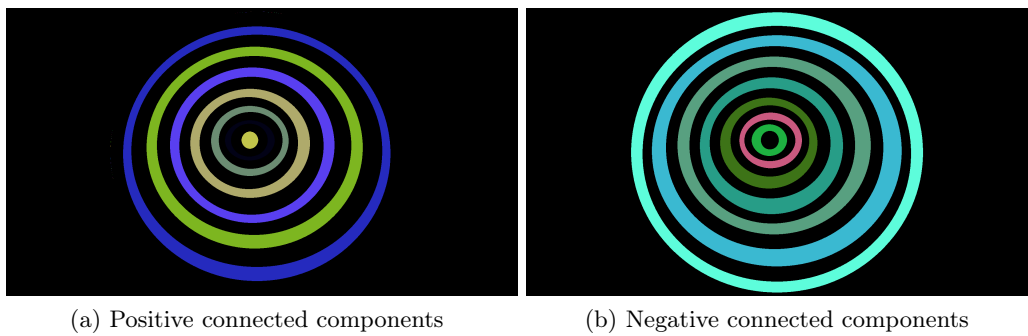


Figure 3.17: Negative and positive connected components of figure 3.16

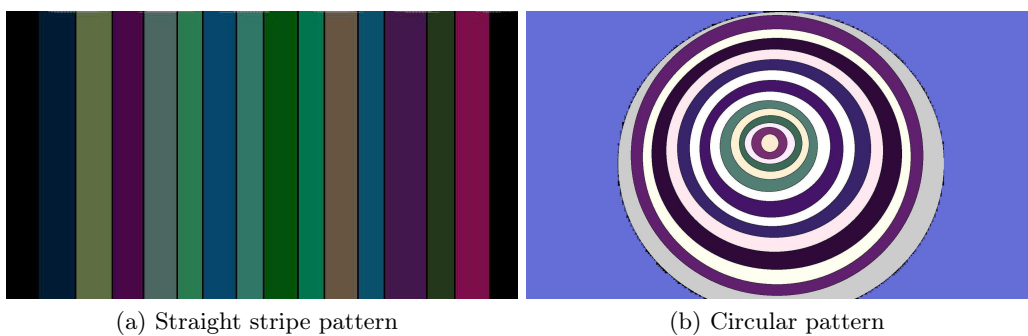


Figure 3.18: Images of the highlighted light and dark stripe components of a pattern

#### 3.4.2.4. Similar components

After the connected components step, the average luminance and area of the stripes has been collected. To finish gathering the information for the risk estimation, all the stripe information (for both positive and negative components separately) is compared against the rest of stripes' information to group them by similarity in luminance. If the stripes don't overpass the luminance flash threshold between their average luminance values, then they are considered similar. Once all stripes have been checked, the group with the highest number of stripes is selected.

This operation results in having the amount of dark and light stripes, the sum of their total average luminance and the sum of their total occupied area, which is needed for the risk estimation.

#### 3.4.3. Risk Estimation

This step consists in estimating the risk of a pattern via the pattern's data. In particular, the amount of dark and light stripes, their total average luminance and total occupied area (the sum of the area of both the dark and light stripes). The risk estimation is fully based on (Wilkins A., 2005), in which a series of functions represent the affected portion of patients under the risk estimation pattern data.



The pattern estimated risk is set as:

$$\text{PatternRisk} = \text{AreaRisk} * \text{LuminanceRisk} \quad (3.17)$$

### 3.4.3.1. Area risk

The area risk is computed by obtaining the cycles per degree proportion risk and area proportion risk.

To calculate the cycles per degree proportion risk, the spatial frequency of the pattern must be obtained as in equation 3.18, the visual angle is assumed to be  $10^0$  and  $nStripes$  is the sum of the number of light and dark stripes.

$$\text{SpatialFrequency} = nStripes / \text{visualAngle} \quad (3.18)$$

Then, the cycles per degree proportion risk can be calculated as follows:

$$y(x) = 0.382 \cdot \log(x)^3 - 2.020 \cdot \log(x)^2 + 1.285 \cdot \log(x) + 0.839 \quad (3.19)$$

where  $x \in [0, 1]$  is the spatial frequency and  $y$  the cycles per degree proportion risk

The area proportion risk is calculated with equation 3.20, in which the area occupied by the pattern is used to obtain the video area proportion of the occupied pattern.

$$y(x) = -0.708x^2 + 1.792x - 0.083 \quad (3.20)$$

where  $x \in [0, 1]$  is the proportion of the occupied screen area

Formula 3.20 will set a pattern as harmful once it reaches an area of 30% of the screen display. As the tool checks a smaller flashing area due to the assumed viewing distances, an alternative value for the formula is provided to check a smaller pattern area. This value is part of the default values used by the tool which result in formula 3.21.

$$y(x) = -0.708x^2 + 3.3x - 0.083 \quad (3.21)$$

Lastly, the area risk is computed:

$$\text{AreaRisk} = \text{CyclesPerDegreeRisk} * \text{AreaProportionRisk} \quad (3.22)$$

### 3.4.3.2. Luminance risk

The luminance risk is computed by obtaining the Michelson contrast risk and the pattern luminance risk.

Michelson contrast:

$$\text{MichelsonContrast} = (\text{lightLum} - \text{darkLum}) / (\text{lightLum} + \text{darkLum}) \quad (3.23)$$

where *lightLum* is the average luminance of the lighter stripes and *darkLum* the average luminance of the darker stripes.

Michelson contrast risk:

$$y(x) = -0.7874 * x^{(-0.3136)} + 2.309 \quad (3.24)$$

where  $x$  is the Michelson contrast

Luminance pattern risk for relative luminance:

$$y(x) = 0.336 * \ln(x) + 1.03523 \quad (3.25)$$

where  $x \in [0, 1]$  is the average luminance of the pattern

Luminance pattern risk for luminance at the display:

$$y(x) = 0.336 * \ln(x) - 0.745 \text{ where } x \text{ is the average luminance of the pattern} \quad (3.26)$$

#### 3.4.4. Check criteria

Harmful patterns should not be displayed any longer than half a second of a video continuously. The check criteria for patterns follows the same procedure as the flash criteria check, using the transition count (3.3.5).

A harmful pattern is determined by its estimated risk, which would be above 0.4. The guidelines' objective is to protect 60% of the population, as the estimated risk is computed based on affected proportions of patients, any risk above 0.4 would not be protecting 60% of the population.

Thus, whenever the risk is above 0.4 the frame is marked as containing a harmful pattern. Throughout the video, the count of harmful pattern frames is kept in track so that whenever the count is equal to the amount of frames in half a second of the video, the last frame in the count and all continued harmful frames are set as failed.

A boolean variable exists to be set to true if a harmful pattern sequence is detected, as to later inform the user of the pattern incident.

### 3.5. Conclusions

In this chapter, the tool's main pipeline was described. The tool receives a video file to analyze through the two workflows it has, the flash and pattern detection algorithms.

Firstly, the luminance values of the frame are extracted. Afterwards, for each frame in the video, the flash detection begins by obtaining the red saturation values of the frame, calculating the frame difference for both red and luminance flashes, and then the flash transitions are counted and registered. Whenever the amount of transitions that have occurred in the last second are above the thresholds, the video fails due to flashing.

---

In all frames of the video, and after running the flash detection on the frame, the pattern detection is performed. The median of the frame values in grayscale is computed to obtain the best parameters for the Canny edge detection. Then, the straight line detection is run. If there are enough valid straight lines, the pattern analysis is done on the region of straight lines. If not, the circular line detection is run to check if there are circular valid lines. The pattern analysis separated the dark and light stripes of pattern, obtains their area and average luminance as well as the number of stripes. Thus, all the data to perform the risk estimation for the pattern has been obtained. With the risk estimation, the pattern compliance criteria can be checked. A video will be considered harmful due to pattern exposure when harmful patterns persist for at least half a second of the video continuously.

In the next chapter, the implementation of the tool will be explained as to allow understanding on how these algorithms have been defined and programmed.



# Implementation

This chapter describes the implementation of IRIS, the tool described in chapter 3. It covers the tool's features, its software architecture, the technology used and the development methodology.

## 4.1. Features

IRIS is a console application that analyzes videos to check compliance with Ofcom's, ITU-R's and W3's photosensitivity guidelines and output the results of the analysis. The list of features included in the tool are:

- **Multiple video analysis:** the videos analyzed by the tool are all located in the same "TestVideos" directory. The tool reads the file information in the folder and then proceeds to open each video sequentially to run its analysis. This design was chosen as it is more comfortable for the user to have all the videos in the same place, and then simply launch the tool to get the results when running it locally. The process is represented in figure 4.4. It can be observed that IRIS checks if there are any videos in the "TestVideos" folder, or even if it exists, as to inform the user to add videos to the directory and rerun the tool. When there are valid videos in the folder, the tool opens a video, analyses it generating and collecting data for the user, and then generates a pass or fail result. If there are more videos in the folder, the process is repeated.
- **Cross-platform:** the tool runs both on Windows and Linux operating systems thanks to the used technology (section 4.3), allowing the tool to run locally on both systems. This feature is important as it allows the tool to be used as the backend project for a web application running on Linux machines.
- **Progress information:** when analyzing a video, the tool displays information regarding the success of reading a video as well as the progress percentage while analyzing it. It also displays the total elapsed analysis time of a video.
- **Data persistence:** the tool collects data generated by the video analysis and stores it on files. This helps users understand the incidents that have occurred throughout a

video as it includes a set of information per video frame (section 4.2.1). The current tool's implementation enables persistence to .csv files.

- Flash images: whenever a flash transition occurs, a screenshot is made and saved to provide more details for flash incidents. These images are stored in the JPEG format and are a representation of the video frame in grayscale with the flash area highlighted in color as can be seen in figure 4.1



Figure 4.1: Highlighted flash area over video frame in grayscale

- Pattern images: whenever the pattern risk is above the threshold, a screenshot is made and saved to provide more details for pattern incidents. These images are stored in the JPEG format and are a representation of the video frame in grayscale with the pattern stripes highlighted in color as can be seen in figure 4.2.

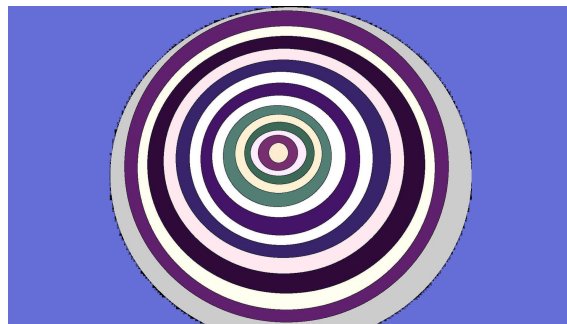
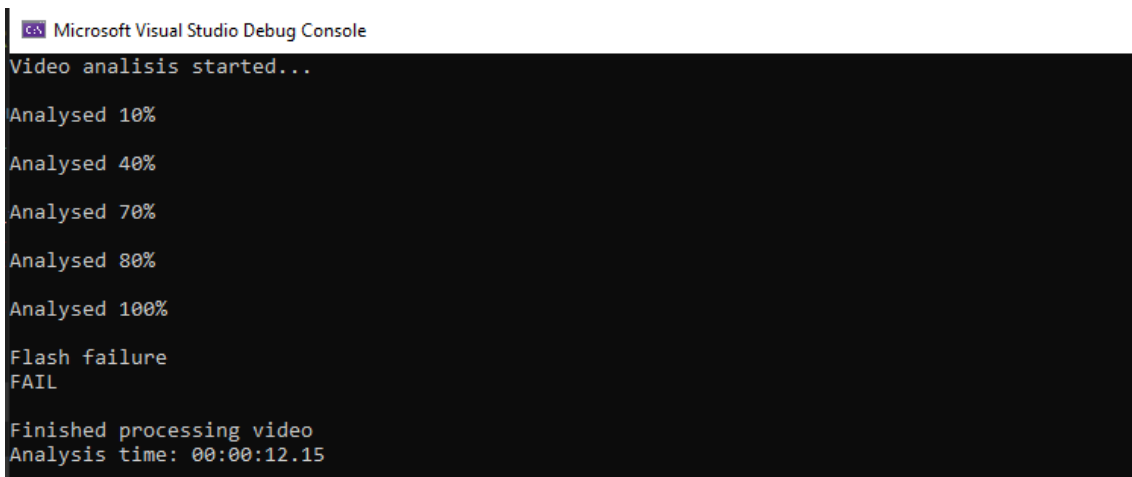


Figure 4.2: Images of the highlighted light and dark stripe components of a pattern

- App settings files: a set of configuration files exist to store all the constant values and parameters from the tool. This enables changing the settings without having to change the code or recompiling the project. This feature also allows versatility for the tool. For example, in the case of the risk estimation, only the luminance type needs to be checked as to load the correct luminance risk formula values, as the code remains the same for formula's calculation. As mentioned, all the constant values used in IRIS are defined in the configuration files and should be modified there. Some notorious parameters are:
  - LuminanceType: allows setting the luminance values either to relative luminance or luminance at the display.
  - AreaProportion: allows setting the flash area percentage.
  - LowArea: if true, the values for the area risk set a harmful pattern at 15% of the screen display. If not, it is set to 30% of the screen display.

- Screenshot saver parameters: allow adjusting the alpha, beta and gamma values for the merging of the frame with the flash area and the pattern stripes. The JPEG quality of the generated image is also a parameter that can be modified.

- Result files: when running the tool, a "Results" directory is created in which all files generated by the tool are saved. There is a folder per video that contains a "Screenshots" folder, in which subsequent folders for luminance flashes, red saturated flashes and harmful patterns exist. There is also a directory in which the logs .csv file from the data persistence is located. In this directory a "*VideoName*\_output.txt" file is located too. This file is used to persist certain analysis information such as the analysis time, to compare the length of the video analysis between different executions as this file is not overwritten, instead information is added to the existing one. This file has been used to store certain information while developing the tool.
- Result console output: after the analysis of a video, the tool displays a pass or fail results. But it also indicates the exact detected issue(s), a message per failure type is displayed in the console. Thus, it can be easily known what type of incidents (flash failure, flash extended failure or pattern failure) have occurred, as can be seen in figure 4.3.



```
Microsoft Visual Studio Debug Console
Video analysis started...
Analysed 10%
Analysed 40%
Analysed 70%
Analysed 80%
Analysed 100%
Flash failure
FAIL
Finished processing video
Analysis time: 00:00:12.15
```

Figure 4.3: Console output of video analysis

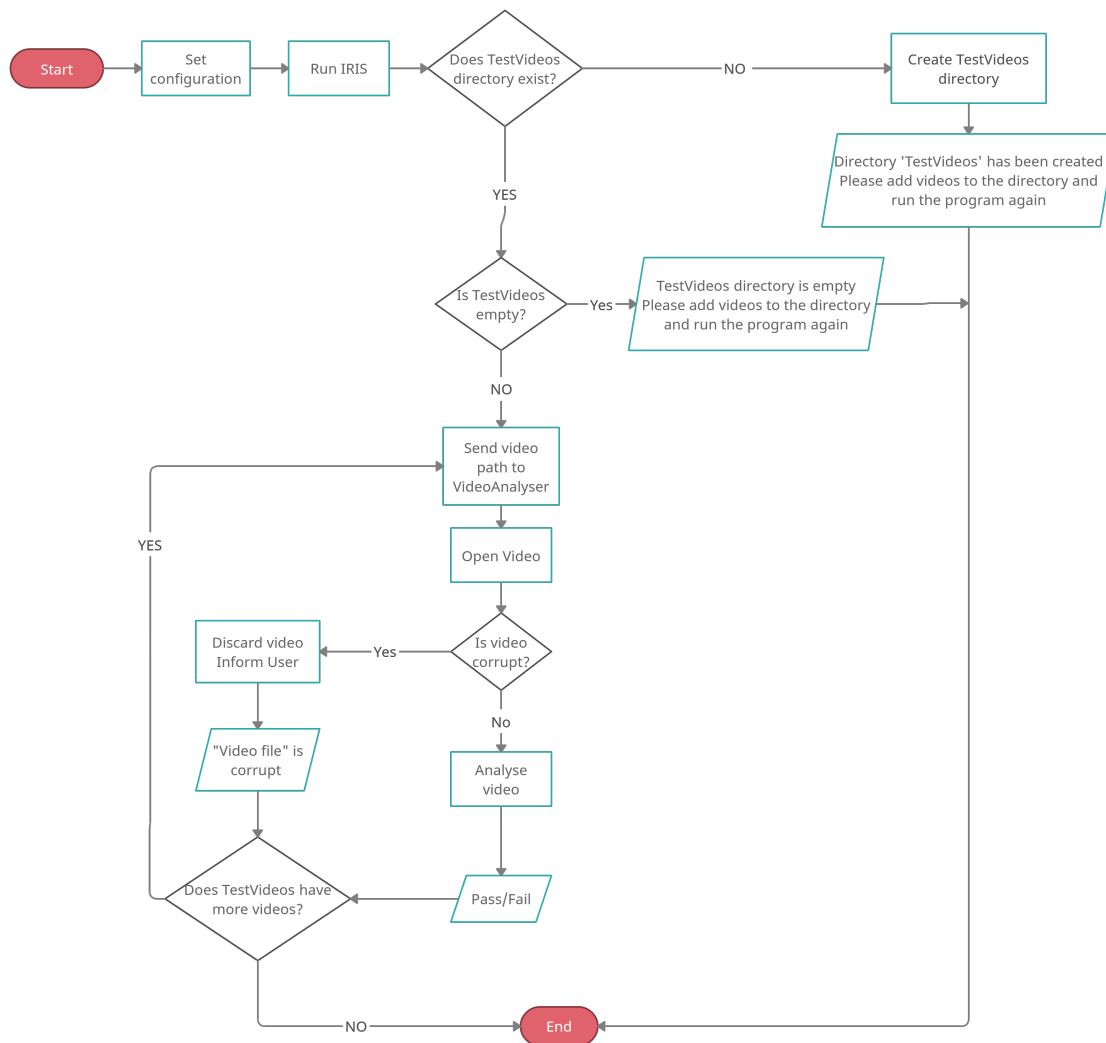


Figure 4.4: IRIS flow

## 4.2. Software architecture

IRIS is composed of four different projects. A console application project (IrisApp), a library to read videos from a folder (IrisLib), a library to analyze videos (VideoLib) and finally, a unit test project (VideoLib.Tests) for the VideoLib library project.

- IrisApp is a console application project that loads the configuration files to feed them into the other projects via IrisLib. It is the start point of the tool. The configuration files are a set of appsettings.json files (for different stages such as development, local, stage) that have the values for all the constant parameters in the tool, which allows changing parameters without needing to recompile the tool.
- IrisLib is the library project that checks if there are any video files in the "TestVideos" directory, in which case, it will initialize VideoLib with the current configuration read in IrisApp and send the video file paths, as can be seen in figure 4.4.
- VideoLib is a project library in charge of opening a video file, analyze it and output



the results of the analysis, as well as collecting and persisting the data related to flashes and patterns for later viewing. It is the tool's core library.

- A test project, `VideoLib.Tests`, was created to ensure a good coverage on the implementation of `VideoLib`. The test project has a total of 111 unit tests, providing a 97.41% code coverage.

### 4.2.1. VideoLib

As mentioned, `VideoLib` is the library project where the video analysis has been implemented. Its structure can be viewed in figure 4.5.

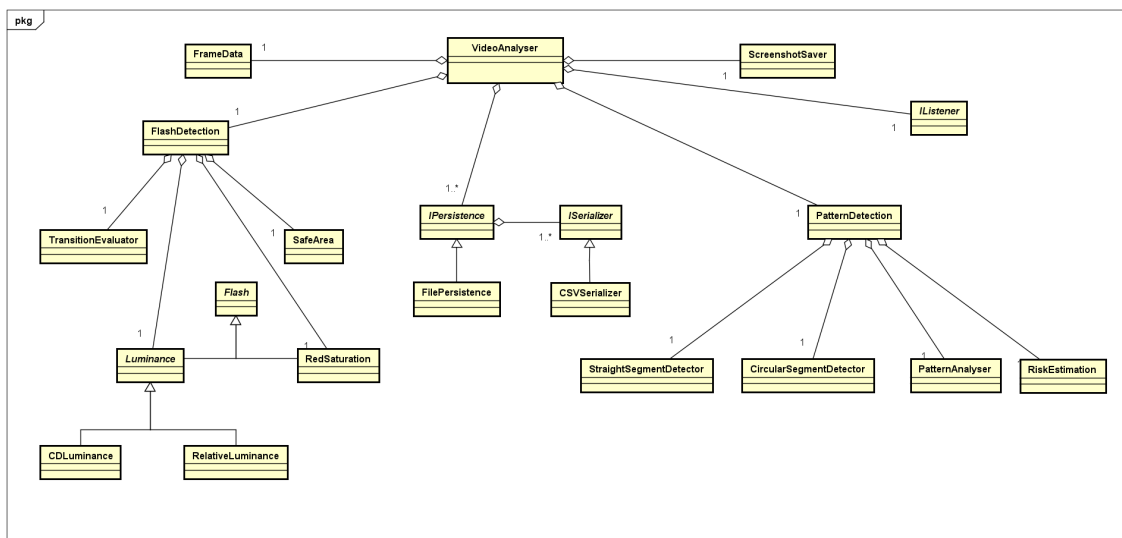


Figure 4.5: VideoLib class diagram

From the class diagram, it can be observed that most of the classes are self-explanatory as to their function, due to the names used for each class, most sections from the previous chapter explain at least one of these classes. The classes that were not previously explained in the last chapter are the persistence and serializer related classes, the *IListener* class and the *FrameData* class.

The *IListener* class defines two methods. One that prints messages on the console, another one that stores desired messages in an "output.txt" that is generated per video. This way, important messages can be displayed in the console and other can be preserved in files.

The persistence and serializer classes are used to serialize a video's frame data and then store it in a file.

The *FrameData* class is a simple class that declares the attributes that are persisted for each frame in an analyzed video. The attributes will be read by the serializer for later persistence, so attributes can be added or removed without having to modify the serializer code. This data allows better understanding on the incidents that occur in a video. The persisted data attributes are:

- Frame index

- Time stamp
- Average luminance difference
- Average luminance accumulated difference
- Average red difference
- Average red accumulated difference
- Pattern risk
- Luminance transitions
- Red Transitions
- Total transition count
- Flash failed frames
- Pattern failed frames

After the analysis, graphs can be plotted as seen throughout chapter 5, to better understand the evolution of the incidents throughout the length of the video using the frame data. The failed frames attributes for flashes and patterns easily indicate which sections of the video have failed. The flash transition attributes allows knowing how many transitions have occurred in the last second, both for each flash type and the combined total transition count. In the case of patterns, the pattern risk indicates if the frame had a harmful pattern in it (pattern risk above 0.4). If needed, the evolution of flashes can be tracked with the accumulated and frame difference attributes, as it allows understanding how the frame values have evolved. And the frame index and time stamp allow to easily view the frame either by its frame position or time position in the video.

### 4.3. Technology used

IRIS' goal is to open a video, read the video frames, analyze its information and then output results for the video's compliance with photosensitivity guidelines, which means that for developing the tool, some data and operations needed are:

- Access video data (frame rate, frame pixel data, etc).
- Matrix operations.
- Blob detection, sets of non-zero neighboring pixels.
- Edge detection.
- Line detection.

OpenCV (OpenCV) is a popular open source computer vision library that includes a wide variety of image processing and machine learning algorithms to perform tasks related to computer vision. As the OpenCV library is used to work with images and videos, it

fully covers all the tool's necessities. All video and image processing is done using this library.

OpenCV is written natively in C++ but offers full support in Python and Java too. However, as the tool was developed in partnership with Electronic Arts (EA, 1982), a request from the company was that, if possible, the development of the tool should be using C# and the .NET Core framework (NetCore).

Microsoft's .NET Core (NetCore) is a free, open source, cross-platform framework for the C# language. Making it very easy to develop applications that can run on different operating systems. An ideal solution for the tool as it is to become the backend project of a web application, to be run on Linux machines.

Thus, the original OpenCV library in C++ was discarded as an option. Instead, a C# wrapper was used to develop the tool, in particular, the OpenCVsharp wrapper (OpenCVsharp). This wrapper was chosen for its similar syntax to the original library and because it supports different OS by providing a runtime package per platform.

For the test project, the MS Test framework was used to develop all the units tests for VideoLib.

## 4.4. Development process

The development can be divided into three different phases:

1. Prototype
2. Performance analysis
3. Final version refactoring

### 4.4.1. Prototype

To begin the development and further understand how to analyze videos to check the guidelines' compliance, a prototype was developed following the implementation from (Carreira et al., 2015). This prototype allowed luminance flash transition detection using luminance at the display, detection of transitions to, or from, saturated red using the Y'CrCb color space for the threshold values and circular and straight line pattern detection. However, this prototype presented some limitations.

Firstly, as explained in section 3.2, luminance at the display uses the luma component of Y'CrCb color space for its calculation. Moreover, OpenCV only has implemented the conversion to Y'CrCb BT.601, which is not suitable for HD resolutions. To prevent conversion issues, as there are currently three different conversion formulas for the Y'CrCb color space, and to not have to keep track of the different conversions, this method was discarded. Instead, the relative luminance formula from section 3.2.2 is used.

Another issue found is that although the tool could detect transitions to, or from, saturated red colors and non-saturated red colors, it could not detect transitions in-between red saturated colors. This problem was tested with a set of videos run on the Harding FPA

and PEAT programs to confirm that the tool was failing to detect these transitions. Thus, the implementation was changed to that of section 3.3.1.

For the flash values, the persisted data provided information on the average frame difference and the average accumulated frame difference (only for luminance, red saturation did not use such values). As it can be seen on figure 4.6, these values are not very readable nor user-friendly. To allow content creators to better understand the incidents in a video, it was important to find a way to present information so that it would be clear when and why a video had failed.

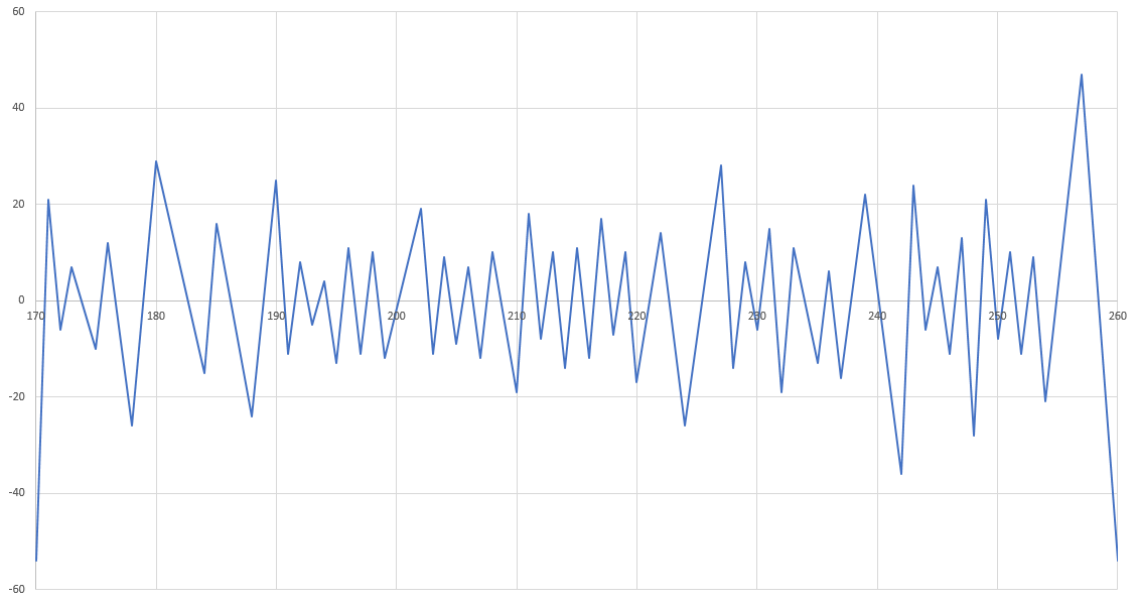


Figure 4.6: Accumulated luminance peaks throughout the length of a video

In this prototype, the flashing safe area was set to 25% of the screen display. The method to check the flashing area was done by calculating the average flash values of the highest and lowest elements of the frame difference matrix. This proves to be lacking considering the flash safe area is too big, and that it does not prioritize the neighboring pixels of the highest and lowest flash values. Later it was changed to a sliding window algorithm to follow W3's flash criterion.

Regarding patterns, only one type of pattern can be detected and analyzed per frame, furthermore, only simple patterns can be detected. Regarding straight line pattern detection, this is not an issue as OpenCV's "FastLineDetector" gives good results. However, circular line detection is more complicated and this implementation misses complex patterns of this type when simply using OpenCV's "FindContours" method. Also, this implementation only informed of the pattern risk, but not if a video failed the guidelines due to harmful patterns.

Lastly, this prototype requires analyzing the entire length of the video, generate all the data related to flashes and patterns, and then check the compliance criteria. Meaning that the compliance criteria checks cannot be run during the video analysis, but only afterwards. This would limit the tool towards the possibility of analyzing streaming videos, as if the streaming is too prolonged in time, a memory overflow could occur if too many frames need to be analyzed. This issue also affected data persistence, as the data from figure 4.6 needed to be added after the flash criteria check.

### 4.4.2. Performance analysis

During the tool's development, different options to improve the performance were explored. When developing the prototype, the usage of OpenCV built-in functions was favored as, for example, looping entire matrices is very time expensive, whereas OpenCV's methods are well optimized and faster than any manual matrix operation.

For the luminance conversion, and later for the conversion from RGB to sRGB, a look-up table procedure was used instead of converting every value per frame. Pixel data ranges from 0 to 255, the conversion formulas are applied to those values once and then stored in the app settings JSON files for all the other executions of the tool. This information is read once when analyzing a video, and then the frames can be fastly converted to either luminance at the display or sRGB values.

Later on, upon completion of the prototype, a performance analysis was done on the tool as there were some apparent bottlenecks that were slowing the analysis time considerably. Visual Studio's performance profiler was used to generate reports and analyze the tool's CPU usage.

After this analysis, certain problems were detected:

- The sliding window algorithm, even when using threads, was incredibly time consuming.
- The minimum line length parameter for the fast line detector needed to depend on video resolution. In high resolution videos, the line length was too small, and the fast line detector took 80% of the analysis execution time.
- Obtaining the average luminance of the pattern stripes was also very time-consuming due to having to loop the matrix.
- Saving the screenshots also took considerable time, even more so when saving them as .png file

### 4.4.3. Final version refactoring

Gathering the information from the previous sections, certain modifications needed to be made to the first prototype in order to have a correct final version of the tool.

- As previously mentioned, the relative luminance calculation was implemented. However, the tool can run the analysis using luminance at the display by changing the luminance type in the configuration files. This method though is not recommended as it is not supported *per se*.
- In the case of the saturated red flash values, section's 3.3.1 procedure was implemented, following the same principles as the luminance flashes which allowed creating the *Flash* abstract class to reduce duplicate code. For red saturation, the old implementation was fully deprecated.
- The *SafeArea* class was implemented to get the highest or lowest average frame difference for the flash values. There are two possible methods to perform this task.

The explained method in section 3.3.3 explains the task's procedure using connected components. But if needed, a sliding window algorithm was also implemented in case it was necessary to follow W3's flash area criteria strictly.

- In the connected components step for the pattern detection, the OpenCV function was changed from "ConnectedComponentsWithStats" to "ConnectedComponentsEx". This last function allows filtering detected blobs from the original image into a separate one, which allowed to separate all the blobs and compute their average luminance using OpenCV's functions instead of looping through the matrix. This method is also used in the safe area check.
- When implementing the sliding window algorithm for the safe area, a small safe area rectangle needed to be calculated to be used as the sliding window. The rectangle varies in size depending on the video resolution. This calculation serves to set the minimum line length for straight lines in patterns too, as it will be then set depending on video resolution.
- The way compliance criteria was checked, which was by generating all the flash and pattern data and checking it after the video has been analyzed, was modified to allow compliance checks throughout the analysis of the video. Which means that the compliance criteria is checked in analysis time.
- This last modification also helped to improve the readability of incident data. In the prototype, attempting to understand incidents based on persisted data was a hard task, as it wasn't very readable. When refactoring the way that flashes were detected as per section 3.3.5, the attributes for flash transitions count could be added, which resulted in a much easier to understand data.
- The *FlashFailedFrames* and *PatternFailedFrames* attributes were added to the *FrameData* class, as they also make it easier to understand incidents in the video by quickly signaling the failed frames by their type.
- The messages of the console output also observed the addition of always printing the failure type as to be clear of the detected incidents.
- In the pattern detection, the use of the median filter and canny edge detection was implemented to improve, mainly, the circular line detection.
- Screenshots were set to be saved asynchronously using the Task library. Screenshots were set to be stored in .jpeg format and with a 50% quality to reduce execution time.
- Figure 4.5 is the class diagram of the refactored version of the tool. The flash and pattern detection classes were created to better encapsulate the code, just as the pattern detection algorithm was divided into different classes for the same reason.
- The flash and pattern detection classes were created to better encapsulate the code, just as the pattern detection algorithm was divided into different classes for the same reason. Figure 4.5 is the class diagram of the refactored version of the tool.

## 4.5. Conclusions

In this chapter, the tool's implementation was described. The tool consists of four different projects developed using C with the .Net Core 3.1 framework. To read videos and perform operations on matrices, as well as use computer vision algorithms, the OpenCV computer vision library was included in the project. In particular, the OpenCVsharp wrapper for the C# language. A least of features the tool has was provided in this chapter too.

IRIS' software architecture was discussed, in which a special emphasis went to the VideoLib library project, as it is where the main work has been done. The *FrameData* class was also explained more in depth as to show the information the tool provides when analyzing a video.

The tool's development process has had three different phases. Firstly, a prototype following published papers was done. While developing the prototype, comparisons with existing tools were made to detect possible problems. With these tests, it was found that the luminance conversion of the frame values was incorrect and that red saturation transitions were being missed. Other issues were found related to missing detected lines in the pattern detection algorithm. After finishing the prototype, a performance analysis was run on the tool to detect further issues. Once done, a series of tasks were defined in order to refactor the tool. Some of the advantages of refactoring the tool is that it allowed having better line detection, setting the flash area with a parameter and providing more user-friendly information about the incidents in the tool.

In the next chapter, an evaluation of the developed tool is presented. The results of the tool are contrasted and compared against those of existing tools, as to showcase their differences.





# Chapter 5

## Evaluation

This chapter contains the obtained results after performing the analysis on different videos, comparing the obtained results with another tool used for checking possible PSE content in audiovisual content: HardingFPA (section 2.4.1). The results can be visualized through different plots generated after each analysis.

### 5.1. Methodology

The main objective of the evaluation is to compare the results obtained by this tool with the ones provided by HardingFPA and determine if the tool is able to provide correct results on a video analysis, as well as provide the correct information that leads to each result. For this purpose, the guidelines explained in chapter 2 serve as reference to set the threshold values on the different checks the tool performs when analyzing a video.

In terms of luminance and saturated red transitions, as it has been explained, the tool continuously analyses a video while checking the compliance criteria for flashes. The frequency is known by the amount of transitions that have occurred in the last second (section 3.3.5), and it is considered that a flash frequency above 3Hz is harmful. Also, the flashing area cannot surpass 10% of the screen display.

For the pattern analysis evaluation, apart from checking its luminance values, the analysis checks the number of pairs of lines forming the pattern, with a valid limit of 5 bright/dark paired lines.

After both checks are done, the tool determines if the analyzed video contains epileptic content, with a result of PASS or FAIL, informing too of the exact incident type.

The evaluation has been performed using two categories of videos. On the one hand, the first checks were done on real videos that caused seizures when they were broadcasted the first time. On the other hand, in order to check the different official guidelines from international broadcasting organizations, another set of videos was created manually, including luminance flashing videos, red saturated flashing videos, and harmful pattern videos. For these type of videos, more than 100 videos were created during the development of IRIS.

All these videos were also analyzed using the tool HardingFPA (only available to the

developers of this tool) in order to make sure that the results obtained were similar. HardingFPA has been the main reference during the course of the project, as a goal of the tool was to at least fail anything that HardingFPA did. As the tool follows the same guidelines as HardingFPA, the threshold parameters are similar too, as well as the generated data, which helps users become familiar with the developed tool.

As described in a previous chapter, IRIS generates a CSV file which includes every frame data of the analyzed video. As previously explained (section 4.2.1), the CSV file contains information about the flash values, the number of transitions until in each second, or which frames are considered as failed, etc. A Python script was implemented (Appendix A) with the aim of having a more comfortable data visualization method. This enables the end user to easily visualize which moments of the video are harmful, as well as contrasting the failed frame information with the incident type as to understand the issue.

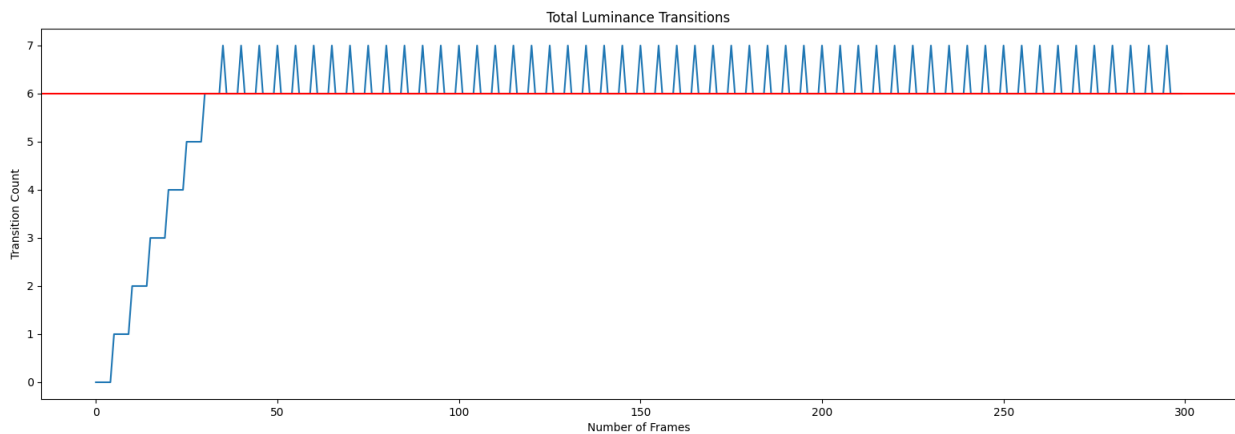


Figure 5.1: Example of data visualization, in this case the total number of luminance transitions in flashing video of 3Hz

Figure 5.1 is an example of how the data is displayed: the X axis being the number of frames of the video, and the Y axis being a luminance transition counter. Also, the limit allowed up to 6 transitions is indicated with a red line, since, as it was established previously, the maximum frequency is 3Hz (remember that each flash is an increase followed by a decrease, or a decrease followed by an increase of luminance). In the example, the transitions counter increases until it reaches the allowed threshold around frame number 30, where it stabilizes during the rest of the video between 6 and 7 transitions per second, so it can be assumed that this video would be considered a FAIL. The given result from IRIS confirms it (note table 5.1). In the same way, other data can be visualized, like the number of saturated red transitions, estimated risk of patterns, the number of failed frames, etc.

## 5.2. Real videos

As it is mentioned in chapter 1, several broadcasting content shown on TV caused paroxysmal responses on viewers:

- 1993: Golden Wonder Pot ad<sup>1</sup>

This advertisement included quick changing images displaying a lot of different luminance and color changes. As can be seen in figure 5.2, the graph displayed indicates how the video includes a big amount of flashes, as we can see the total number of luminance transitions overpassing the threshold set in 6 transitions (3Hz). There are also red harmful transitions included in this video, but the total number of failed frames due to harmful (luminance and red) transitions are displayed in figure 5.3, so the result of the analysis is FAIL.

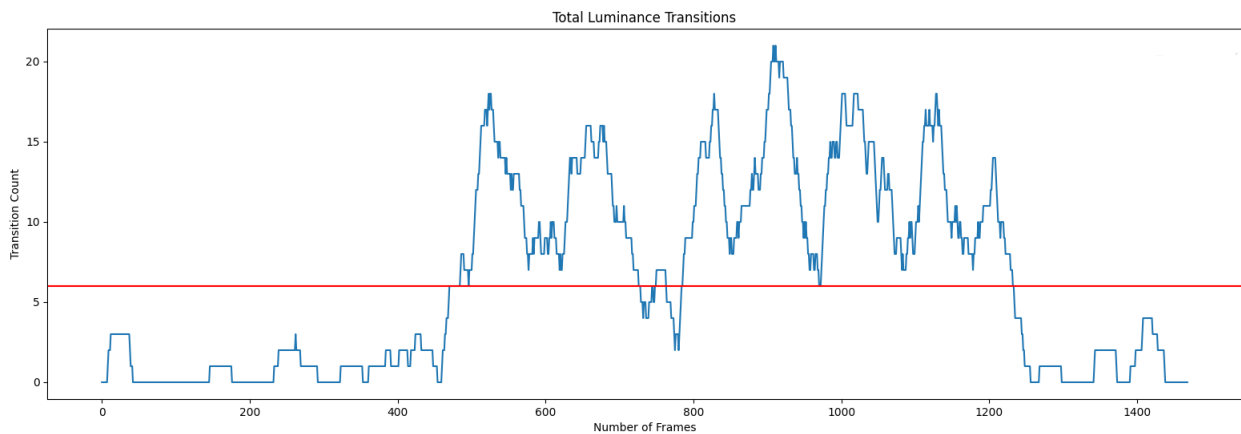


Figure 5.2: Total number of luminance transitions in Golden Wonder Pot ad

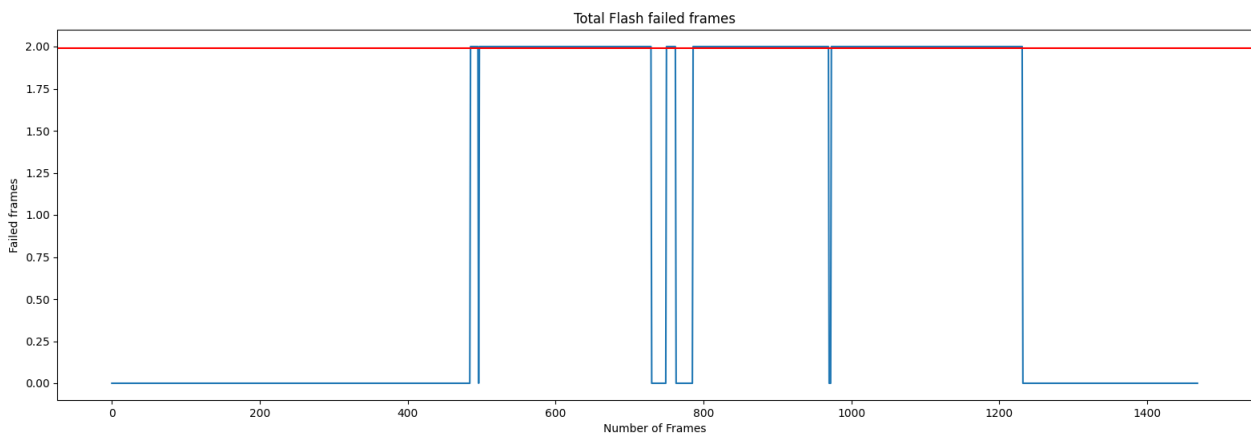


Figure 5.3: Total number of failed frames in Golden Wonder Pot ad

<sup>1</sup>Youtube (2021) Video URL: <https://www.youtube.com/watch?v=HVQuA9HNhOM>

- 1997: Pokémon episode<sup>2</sup>

This Pokémon episode broadcasted only in Japan due to the damage caused to a big amount of children included a long sequence of fast changes between red and blue backgrounds, apart from several flashing images of the explosion that occurs during the sequence. The plot of figure 5.4 shows the sudden increase of harmful red transitions around frame 450, which resulted in the above-mentioned outcome of several seizures on children. The results displayed in figure 5.5 indicate some failed frames before these sequence, due to a few harmful luminance flashes, but the problem is highlighted at the red/blue sequence, with a big amount of dangerous frames, resulting in a FAIL.

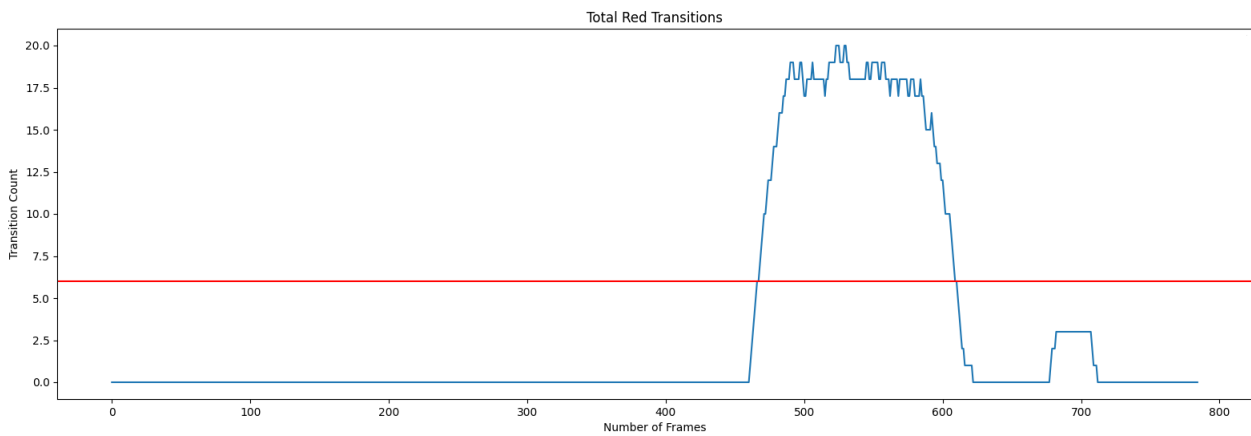


Figure 5.4: Total number of red transitions in the Pokémon episode

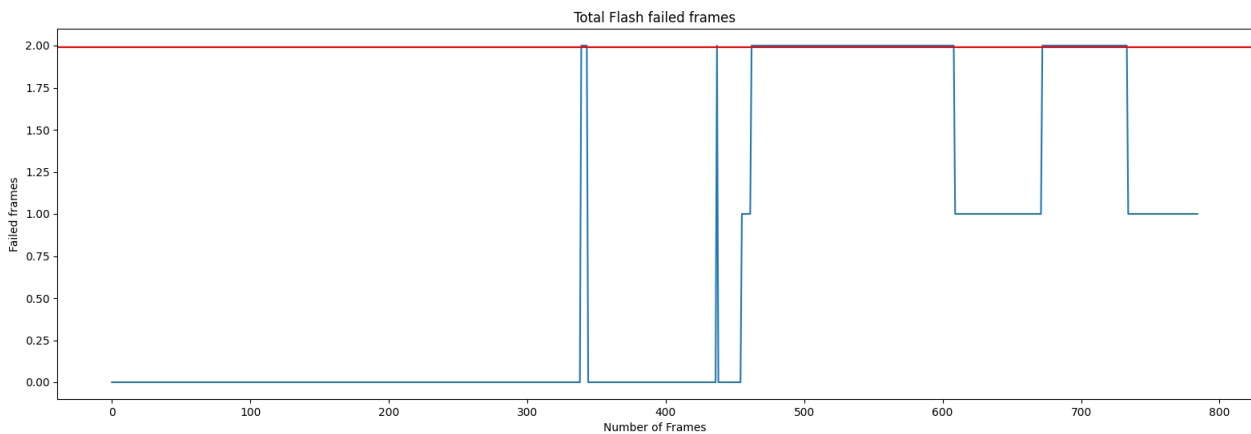


Figure 5.5: Total number of failed frames in the Pokémon episode

- 2003: The White Stripes - Seven Nation Army - music video<sup>3</sup>

<sup>2</sup>Youtube (2015) Video URL: <https://www.youtube.com/watch?v=4LQLvgXLguk>

<sup>3</sup>Youtube (2008) Video URL: <https://www.youtube.com/watch?v=0J2QdDbelmY>

The official video clip of the song Seven Nation Army of the band The White Stripes, included a continuous triangular pattern moving towards the viewer, with changing red, black and white colors. The results of the analysis of figure 5.6 shows that, although it could seem that the video contains moving patterns during the whole sequence, there is barely one harmful frame detected due to patterns. Instead, the video result is FAIL due to the flashing analysis, as it is shown in figure 5.7.

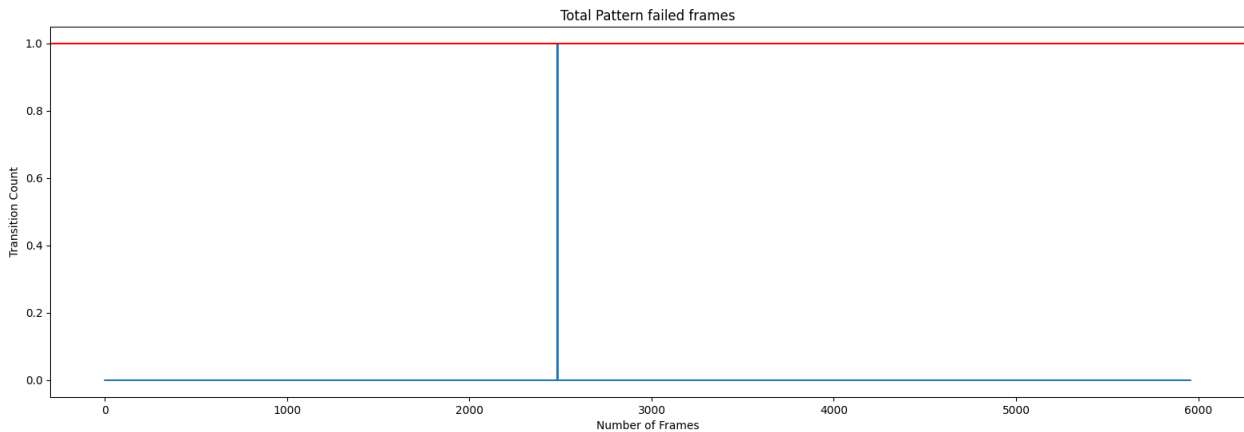


Figure 5.6: Total number of pattern failed frames in the White Stripes video

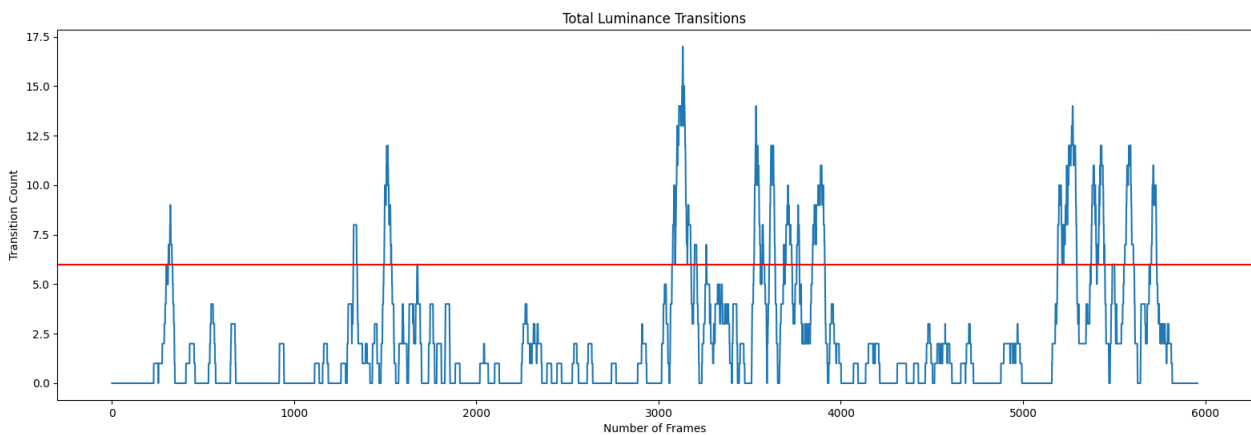


Figure 5.7: Total number of luminance transitions in the White Stripes video

- 2012: London Olympics Games ad<sup>4</sup>

A promotional video of the 2012 London Olympics Games showed several flashes and fast changing colours. As can be seen in figure 5.8, the graph displayed shows a lot of changes in luminance transitions, with a big part of the sequence between values 4 and 6 (caution area), and some other transitions above the allowed threshold. The

<sup>4</sup>Youtube (2010) Video URL: <https://www.youtube.com/watch?v=vs0hfhSje9M>

analysis ends with a result of FAIL, as can be seen in figure 5.9 where multiple frames are set as failed frames.

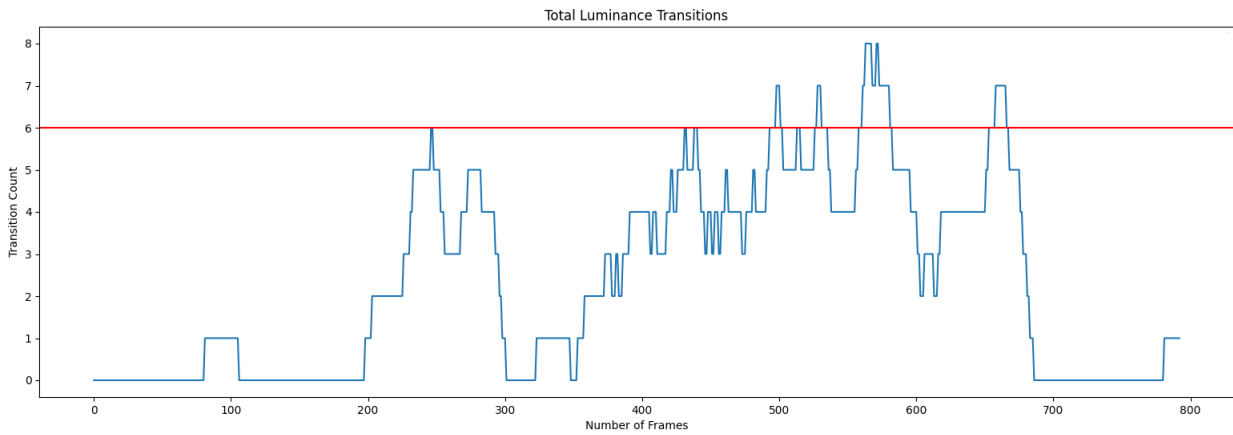


Figure 5.8: Total number of luminance transitions in the London Olympics Games ad

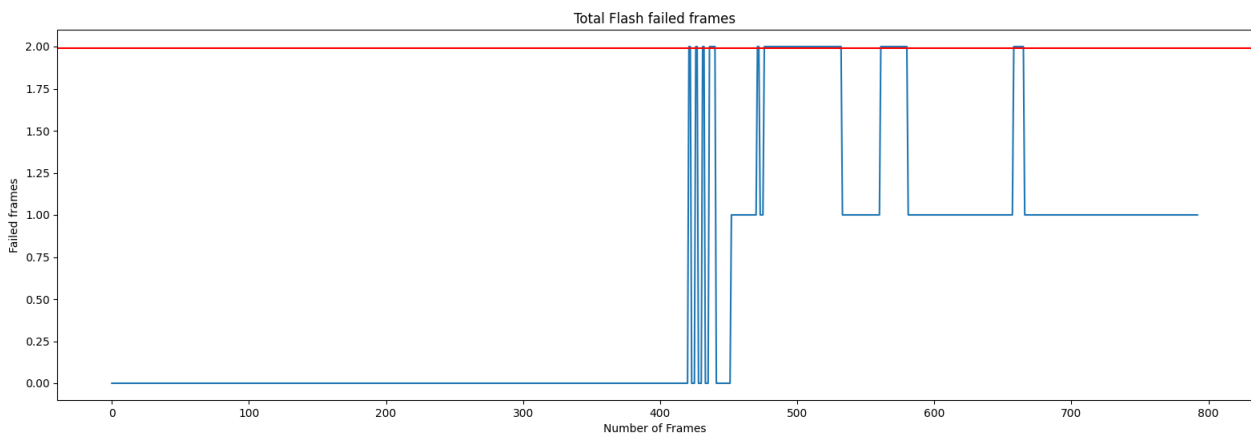


Figure 5.9: Total number of failed frames in the London Olympics Games ad

## 5.3. Generated videos

### 5.3.1. Combination of different flashing areas with different frequencies

These videos were made with a length of 10 seconds, 30 frames per second, and contains an area corresponding to the peak of luminance ( $200 \text{ cd/m}^2$ ) flickering in a black background ( $0 \text{ cd/m}^2$ ):

When taking a look at figure 5.10, it can be observed that the number of luminance transitions do not surpass the threshold level, indicating that frequency is not considered as harmful. Figure 5.11 shows not a single harmful frame included in the video sequence, so the result of the analysis on 1.5Hz frequency videos is PASS.

Area of the screen flashing	Freq: 1.5Hz	Freq: 3Hz	Freq: 5Hz
25%	PASS	FAIL	FAIL
15%	PASS	FAIL	FAIL
100%	PASS	FAIL	FAIL

Table 5.1: Results of analyzed flashing videos

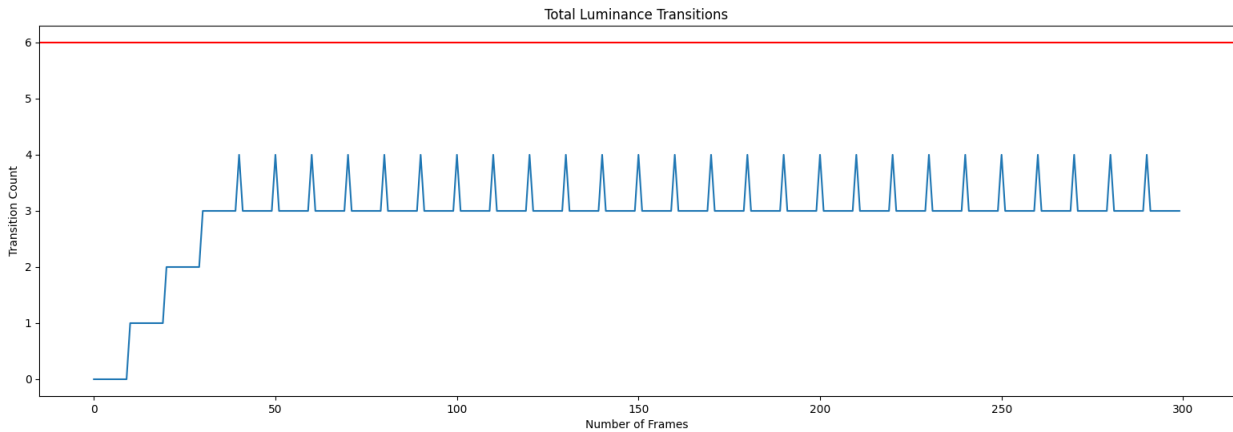


Figure 5.10: Total number of luminance transitions in flashing video of 1.5Hz

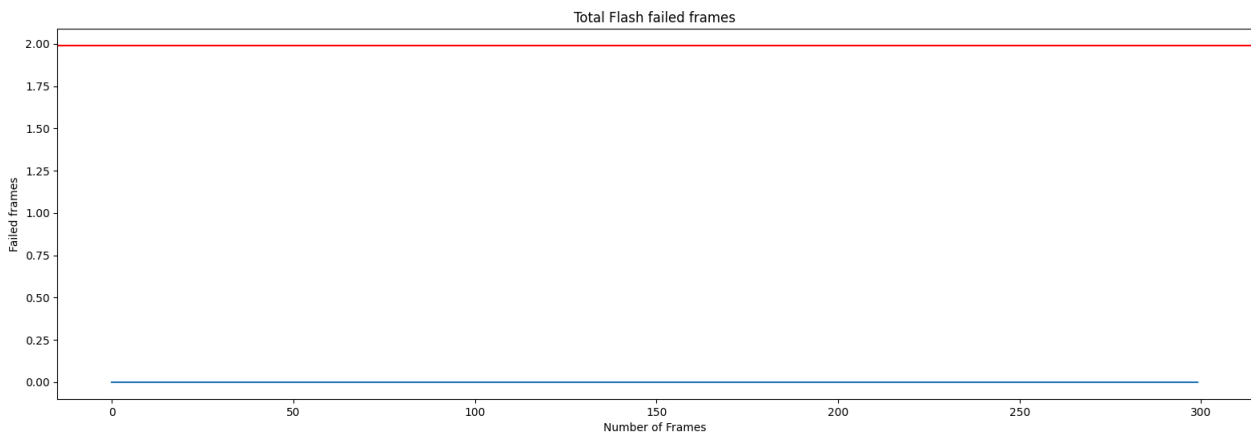


Figure 5.11: Total number of harmful frames in flashing video of 1.5Hz

The analysis of these videos indicates how the frequency of the flashes affects on their final results, as can be seen on table 5.1. When a sequence of flashes is above the allowed threshold (3Hz) the result of the analysis will be a FAIL, as it is shown in figure 5.12.

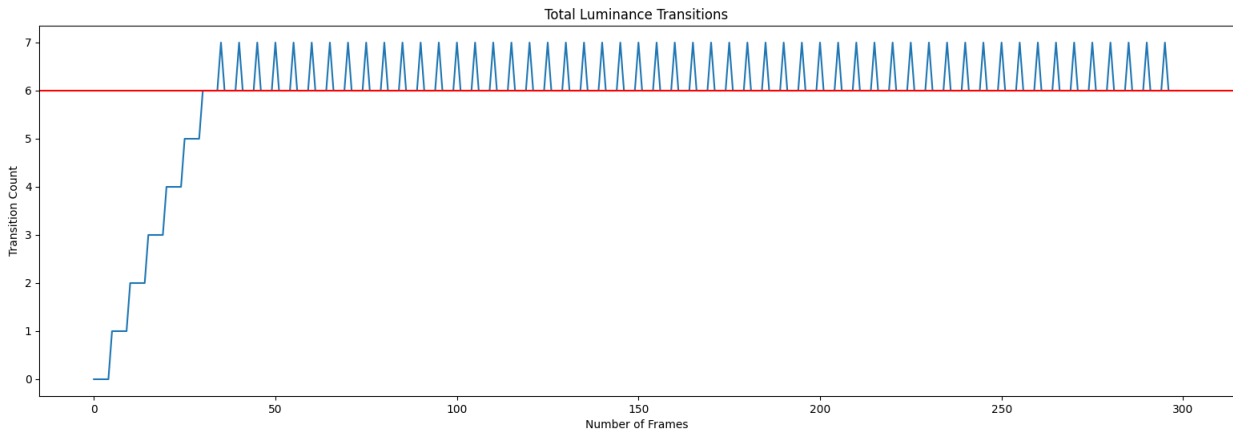


Figure 5.12: Total number of luminance transitions in flashing video of 3Hz

### 5.3.2. Combination of different flashing red areas with different frequencies

These videos were made with at length of 10 seconds, 30 frames per second, and contains an area corresponding to transitions of red colours. The following table 5.2 summarizes the obtained results of the analysed videos of red flashing areas.

Area of the screen flashing	Freq: 1.5Hz	Freq: 3Hz	Freq: 5Hz
25%	PASS	FAIL	FAIL
15%	PASS	FAIL	FAIL
100%	PASS	FAIL	FAIL

Table 5.2: Results of analyzed red flashing videos

The transition count of a 1.5Hz frequency video are shown in figure 5.13 where the values do not reach the harmful threshold and stay among 3 and 4 transitions per second, so the result will be a PASS (table 5.2).

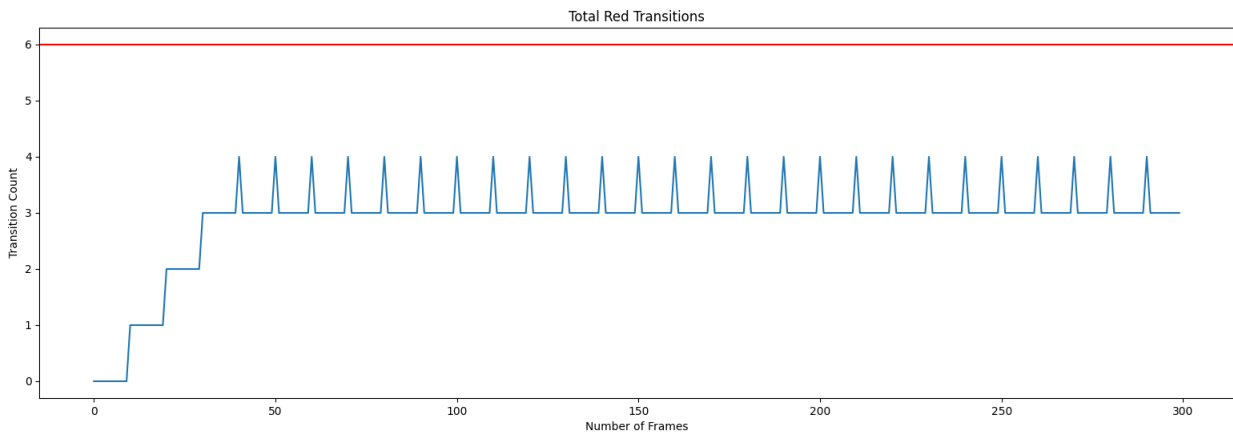


Figure 5.13: Total number of red transitions in red flashing video of 1.5Hz



As it happened with the luminance transitions, whenever a video sequence contains colour transitions involving saturated red above the allowed threshold, the analysis indicates there are harmful conditions included, as we can observe in figure 5.14. When the frequency is set to 3Hz, the transition counter stabilizes between the values 6 and 7 throughout the whole sequence, indicating the video result is FAIL (table 5.2).

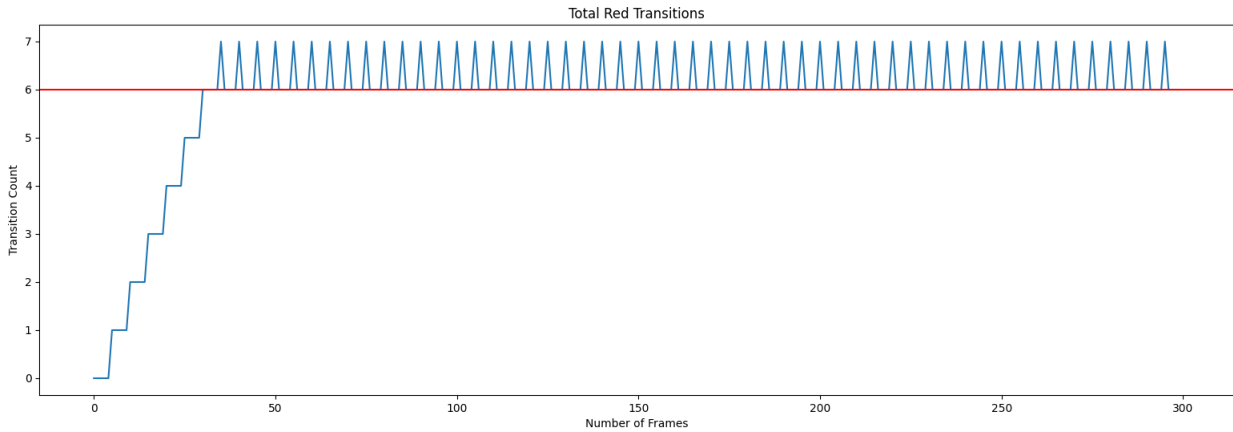


Figure 5.14: Total number of red transitions in red flashing video of 3Hz

### 5.3.3. Patterns

These videos were made with at length of 10 seconds, 30 frames per second, and contains different types of static patterns having a different number of lines. The following table shows the calculated estimated risk of inducing a seizure:

Type of pattern	pairs of lines: 4	pairs of lines: 5	pairs of lines: 8
Static striped pattern	PASS	PASS	FAIL
Static circular pattern	PASS	PASS	FAIL

Table 5.3: Results of analyzed pattern videos

As can be seen in table 5.3, both types of patterns are allowed when the number of components is not higher than 5 pairs of dark/white lines, but figure 5.15 shows the estimated risk set above 0.8 during the whole sequence. This value indicates the video is much above the allowed risk value for a long period of time, resulting in a FAIL as it is shown in figure 5.16.

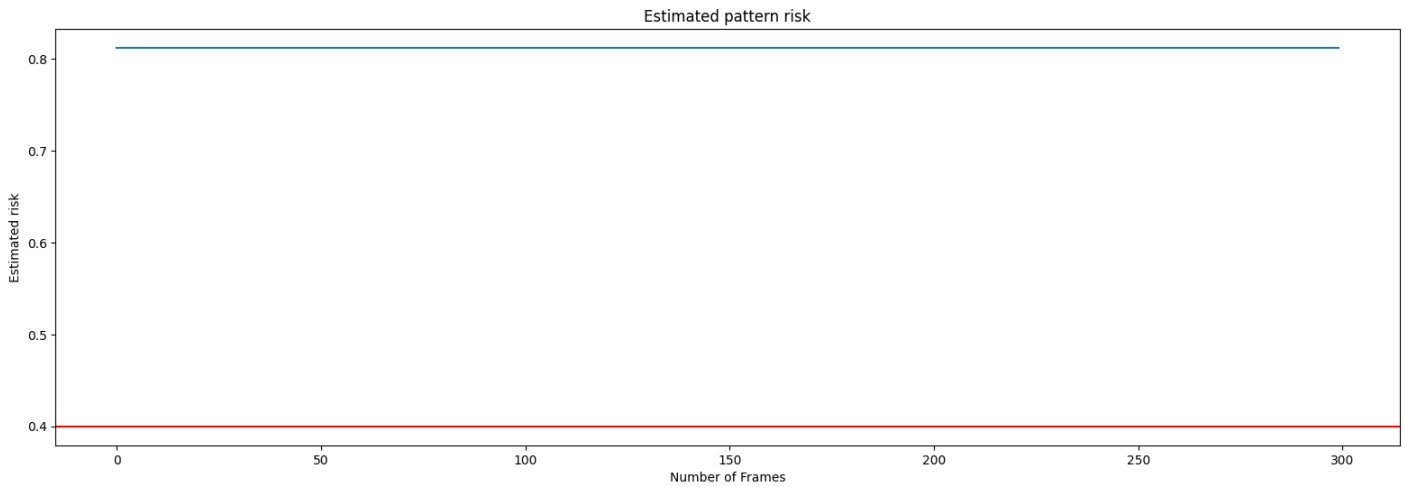


Figure 5.15: Estimated risk of the striped pattern video of 8 pairs of dark/white lines

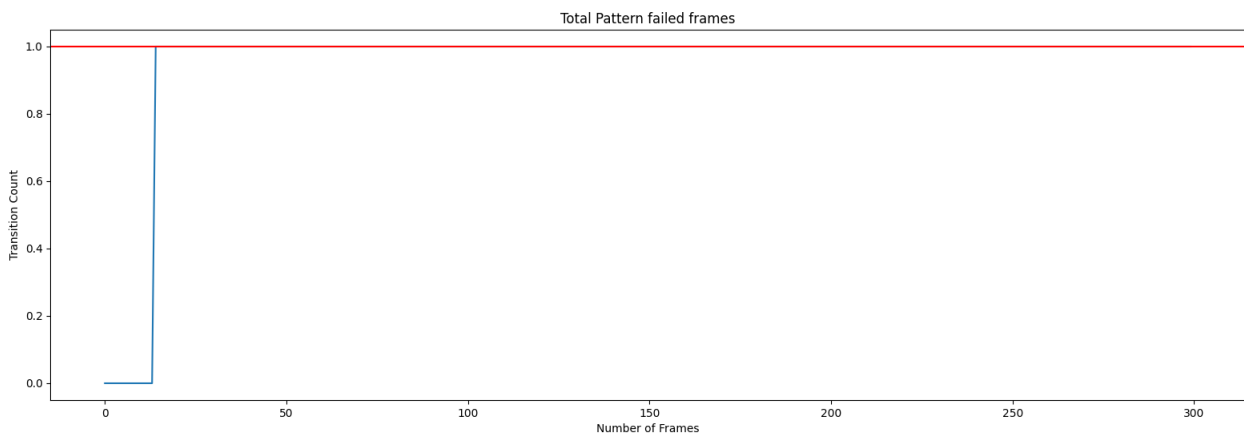


Figure 5.16: Total pattern failed frames in the circular pattern video of 8 pairs of dark/white lines

## 5.4. Conclusions

As have been explained in this chapter, the evaluation of IRIS was always done against HardingFPA, the reference tool for checking possible PSE content in videos. All the obtained results, both the real broadcasted videos and the ones generated manually to check the explained guidelines, have the same results. Therefore, it can be said that the evaluation was very successful. Once both tools got the same results, the next step was replicating the data visualization, in order to have similar ways of interpreting the obtained data.

Despite this, it is necessary to emphasize the difference in performance between the two tools, HardingFPA being much faster, so the developer team started working on improving the performance of the tool once the main functionalities were implemented.

## Conclusions and Future Work

### 6.1. Conclusions

This project was born with the intention of providing a solution to a problem increasingly present in every audiovisual content development (especially video games, where the appearance of flashing and changing images is very present): the health risk prevention.

The main objective of this work was to develop IRIS, a tool that helps detect photosensitive elements in videos, susceptible to being harmful to users with photosensitive epilepsy that could cause them epileptic seizures. In order to achieve this, other existing tools were studied, e.g. HardingFPA, with the aim of trying to understand and improve both their usage and functionalities.

Due to the complexity of the studied topic, an important barrier was to assimilate all the information that was necessary to understand. However, once the theoretical bases on which these tools are based were established, the tool's core was defined as well as the thresholds for the compliance criteria that were extracted from the guidelines.

In order to have a correctly organized development, methodologies learned during the course of the degree were used, such as SCRUM, which allowed to adjust each short-term objective in a realistic and detailed way.

It was possible to develop a part of the prototype that included the detection of luminance flashing images in only 2 months of development, with the aim that it could be tested against HardingFPA and check if the results obtained by both tools were similar. In just over 4 months, the main objective was achieved, a tool with the basic functionalities in detecting photosensitive elements such as flashes or geometric patterns.

Given the professional nature of this project, since it has been developed within a video game company (Electronic Arts), it was necessary that the obtained results were the same as those of the tool it was compared with (HardingFPA), complying with the standards for the photosensitive prevention, and it is ready to be used internally by the organization.

In order to ensure the effectiveness of IRIS, more than 100 different videos were executed during its the development of the tool, comparing them with the results of existing tools such as HardingFPA, which helped to validate results and detect issues in the developed tool.

Afterwards, the focus was set on improving the efficiency of the tool, due to the long execution times that the tool carried out during its analyses. To do this, the detection thresholds were refined little by little to make them more precise, while ensuring that the results obtained were correct.

Once the performance analysis finished, a series of tasks to refactor the tool into the final version for this project were designed and completed to provide an improved version of the original prototype.

It is considered that the objectives set at the beginning of the project have been met, as IRIS is able to validate video content against photosensitive guidelines.

As the tool has undergone multiple updates during its development, iterating and improving both the effectiveness of the results and its efficiency in runtime, it is considered that the tool is still susceptible of further improvement.

## 6.2. Future Work

Although the developed tool has met the defined objects at the beginning of the project, there is still room from improvement and additions.

- Implementation of a frontend project: providing a UI for users would allow them to navigate a more accessible environment. The idea is, given the commercial nature of this tool, to research and implement the functionalities that end users would use, with a design that allows them to quickly become familiar with the tool and by having a similarity with the previously used tool in the company to ensure a smooth transition to the new one.
- Real time analysis: even though the tool has undergone performance analysis and efforts have been carried out to reduce the video analysis time, it's execution time is far from reaching real time analysis. An analysis on the current implementation, led by an optimization task could improve the performance and allow real time analysis of a video, which would reduce the analysis times from the tool.
- High dynamic range (HDR) support: many video games are played enabling the HDR feature in screen displays. HDR, offers better luminance ranges on the highest and lowest luminance values. Higher and lower luminance values can be an issue, as it can result in more flash transitions in video content. Supporting analysis of HDR content would help prevent photosensitive issues whenever the video content uses this technology.
- Virtual reality (VR) support: VR headsets create the illusion that the user is completely seeing a virtual environment through its own eyes. Yet in VR headsets actually split the source image into two individual images as to create a 3D perspective. Also, VR headsets use lenses in-between the screen and the user's eyes to better distort the image for a more lifelike perception. These are peculiar characteristics for a screen type, yet more video games are being developed for virtual reality and seeing as there is little to none visual distance to the screen display, VR video games and experiences can be a potential serious issue for PSE patients. Research on this topic would allow implementing support for this type of screens.

These improvements were raised and considered once the core functionality of the prototype was implemented. Despite this, the following work done on the tool was centered on improving its performance and finishing the refactoring for the project's final version.

Thus, these proposed improvements could not be carried out due to lack of time, but in any case they are realistic enough to keep them as possible future modifications and further improve the tool's features.



# Introducción

En este capítulo se introducen los objetivos y motivaciones que se llevaron a cabo para la realización de este proyecto, una herramienta capaz de analizar y detectar las posibles condiciones que desencadenan en un ataque epiléptico fotosensible a partir de un video. A su vez, se describe la tecnología utilizada, el plan establecido durante el desarrollo de la herramienta y la descripción de esta memoria.

## 7.1. Motivación

A día de hoy, la industria del videojuego se ha convertido en uno de los medios de contenido audiovisual más populares del mundo, generando unos ingresos globales de casi 180 mil millones de dólares en el año 2020 (Batchelor, 2020). Debido a esa popularidad y los cambios en los hábitos de consumo de la sociedad, con una presencia significativamente mayor del uso de pantallas y los dispositivos móviles, ha supuesto un aumento en la cantidad de tiempo que una persona pasa delante de una pantalla a lo largo del día. En el caso de los videojuegos, cuyo principal consumidor habitual se encuentra en el sector joven de la población, supone un aumento en el riesgo de padecer enfermedades derivadas de las emisiones que produce un televisor o monitor, como es el caso de la epilepsia fotosensible.

La epilepsia fotosensible es una enfermedad neurológica capaz de provocar convulsiones (conocidas como ataques epilépticos), provocado por estímulos visuales tales como la exposición a imágenes luminosas o cambios rápidos de patrones de figuras geométricas en movimiento (Epilepsy Society, 2020). La epilepsia tiene una mayor presencia a edades tempranas, siendo esta más común en edades de los 7 a los 18 años (Vivir con Epilepsia, 2017), lo que convierte a los videojuegos en un factor de riesgo para los posibles casos de pacientes con epilepsia fotosensible.

Durante las últimas décadas se desarrollaron una serie de incidentes epilépticos provocados por la emisión de material fotosensible, lo que incitó una actualización en la normativa existente sobre las emisiones de televisión, ya que dada la naturaleza de la enfermedad, esta solo se manifiesta ante determinados estímulos, y por tanto, difícil de diagnosticar preventivamente. Algunos de los eventos más destacados:

- En 1993, en el Reino Unido, un anuncio televisivo de comida del producto Golden

Wonder Pot Noodles<sup>1</sup>, que contiene una secuencia de imágenes con cambios rápidos de colores, provocó 3 casos de ataques epilépticos.



Figure 7.1: Frame del anuncio de Wolden Wonder en Reino Unido (Youtube, 2021)

- En 1997, en Japón, en el episodio número 38 de la primera temporada de la serie de animación Pokémon<sup>2</sup>, que contiene una secuencia de cambios rápidos entre colores azul y rojo, además de imágenes intermitentes de alta luminosidad, provocó 685 casos de ataques epilépticos.



Figure 7.2: Dos frames del capítulo de Pokémon emitido en Japón(Youtube, 2015)

En el desarrollo de videojuegos es necesario garantizar la seguridad de los usuarios al ser expuestos a imágenes estimulantes o muy luminosas que puedan afectar a personas con epilepsia. El objetivo de este TFG es el desarrollo de una herramienta que permita analizar y medir la fotosensibilidad de una secuencia de vídeo extraída durante el testing de un videojuego, con el fin de poder determinar si contiene secuencias de escenas potencialmente peligrosas para este tipo de enfermedades.

## 7.2. Objetivos

El objetivo fundamental de este proyecto es la implementación de una herramienta que permita validar secuencias de video incluidas en medios audiovisuales desde una perspectiva fotosensible, siguiendo las normas vigentes en lo que respecta a prevención de riesgos

<sup>1</sup>Video URL: <https://www.youtube.com/watch?v=HVQuA9HNhOM>

<sup>2</sup>Video URL: <https://www.youtube.com/watch?v=4LQLvgXLguk>



fotosensibles. La herramienta tiene como objetivo proporcionar información y prevenir la publicación de material fotosensible dañino en medios audiovisuales.

Este proyecto ha sido desarrollado para Electronic Arts (EA, 1982) con el objetivo de proporcionar una herramienta más flexible que permita comprobar videos con posible contenido dañino. Dada la existencia de herramientas similares ya existentes en el mercado, la principal idea de este proyecto es proporcionar una solución que reúna las características de estas plataformas, pero con la posibilidad de ser fácilmente modificada, como añadir resoluciones, frecuencias o formatos de video, en caso de que las normativas internacionales en prevención de riesgos epilépticos añadan o modifiquen dichas reglas.

### 7.3. Plan de trabajo

A la hora de llevar a cabo el desarrollo de este proyecto, la organización se formó de la siguiente manera:

La metodología utilizada para la realización de este proyecto es SCRUM, metodología ágil muy utilizada en el entorno del desarrollo de software y el desarrollo de videojuegos. Provistos en periodos de tiempo de 2 semanas (llamados *sprints*), los integrantes del equipo de desarrollo establecen una serie de tareas a realizar y se les asigna una prioridad, así como una estimación del tiempo para llevarla a cabo. Así, se reparten las tareas entre los integrantes del equipo y llevan a cabo durante el periodo denominado *sprint*. Una vez transcurrido ese tiempo, se realiza un análisis de las tareas completadas, planteando nuevas del mismo modo que anteriormente, o se ajustan las condiciones de las tareas que no han podido ser completadas durante el *sprint* anterior.

Durante las primeras semanas, el equipo inicia la investigación acerca de la epilepsia fotosensible, sus causas y los efectos que produce en sus pacientes. A su vez, la investigación también se centra en los estudios clínicos llevados a cabo para saber detectar y diagnosticar casos de epilepsia fotosensible, así como cuáles son las características que pueden desencadenar un ataque epiléptico.

Una vez estudiados los primeros artículos y ensayos, se investiga sobre las normativas internacionales sustentadas en estos estudios previos, para determinar cuáles son las condiciones y los valores que se consideran dañinos dentro de un punto de vista fotosensible. El equipo también es capaz de familiarizarse con herramientas ya existentes (como HardingFPA, sección 2.4.1, o PEAT, sección 2.4.2), de forma que pudiera entender cuál es el flujo normal de uso para su posterior implementación.

Se determina cuál van a ser las tecnologías a utilizar para el desarrollo de la herramienta. En este caso, la herramienta está desarrollada en C#, utilizando el entorno de desarrollo Microsoft Visual Studio, y el uso de la librería gráfica OpenCV.

Se definen los primeros objetivos: esqueleto del proyecto, extracción de datos a partir de un video, asignación de valores límite que definen si un video es dañino o no. A partir de ahí, se definen dos vías: detección de flashes luminosos y detección de flashes de rojos saturados.

Una vez definidos ambas detecciones, se investigan las características de patrones geométricos que puedan desencadenar una respuesta epiléptica. Se implementa la detección de patrones geométricos y se lleva a cabo una estimación de riesgo que determina si un

video es dañino o no.

Una vez implementadas las funcionalidades básicas, se itera de forma que cada módulo mejore su eficiencia, tanto en recursos como en tiempo.

## 7.4. Estructura de la memoria

Este documento está estructurado de la siguiente manera:

- En el capítulo 1, se introduce el objetivo general de este proyecto, así como sus objetivos y motivaciones principales.
- En el capítulo 2, se introduce el concepto de epilepsia fotosensible, sus características y efectos adversos, así como los estudios clínicos realizados a lo largo de la historia para determinar las condiciones que los provocan. También se enumeran las organizaciones y normativas establecidas para la prevención de posibles ataques, basadas en los resultados de dichos estudios previos.
- En el capítulo 3, se realiza la descripción completa de la funcionalidad de la herramienta, ahondando en las dos funciones principales de la herramienta, la detección de flashes y patrones geométricos dañinos.
- En el capítulo 4, se presenta la implementación del proyecto, junto con las tecnologías usadas, la arquitectura y el desarrollo de la herramienta.
- En el capítulo 5, se realiza una evaluación de la herramienta, así como una serie de comprobaciones sobre casos reales para determinar la validación o no de la herramienta, y la exposición de estos resultados.
- En el capítulo 6, se enumeran una serie de conclusiones tras la realización de este proyecto y las posibles mejoras de cara al futuro.

## Conclusiones y Trabajo Futuro

Este capítulo contiene la traducción al español del capítulo 6.

### 8.1. Conclusiones

Este proyecto nació con la intención de proporcionar una solución a un problema cada vez más presente en el desarrollo de contenido audiovisual (en especial a los videojuegos, donde la aparición de imágenes muy luminosas y cambiantes está muy presente): la prevención de riesgos para la salud.

El objetivo de este trabajo era ser capaces de realizar una herramienta que ayude a la detección de elementos fotosensibles en videos, susceptibles de ser dañinos para usuarios con epilepsia fotosensible y provocar problemas en ellos. Para ello, se partió de la experiencia con otras herramientas ya existentes con el objetivo de tratar de replicar tanto su uso como su funcionamiento.

Debido a la complejidad del tema en cuestión, gran parte de la primera problemática fue asimilar toda la nueva información que era necesaria para comprender. Sin embargo, una vez establecidas las bases teóricas en las que están basadas estas herramientas, se definió la base de la herramienta y los umbrales de los criterios de cumplimiento, extraídas de las normativas estudiadas.

Para un desarrollo organizado correctamente, se utilizaron metodologías aprendidas durante el transcurso de la carrera, como SCRUM, que permitieron ajustar cada objetivo a corto plazo de manera realista y detallada.

Con tan solo 2 meses de desarrollo, se consiguió desarrollar una primera versión que contenía la detección de imágenes con flashes, con el objetivo de que pudiera ser testado contra HardingFPA y comprobar si los resultados obtenidos por ambas herramientas era similar. En poco más de 4 meses se alcanzó el objetivo principal, una herramienta con las funcionalidades básicas en detección de elementos fotosensibles como flashes o patrones geométricos.

Dada la naturaleza profesional de este proyecto, ya que ha sido realizado en un entorno profesional dentro de una empresa de videojuegos (Electronic Arts), era necesario que

los resultados obtenidos fueran los mismos que los de la herramienta con la que se ha comparado (HardingFPA), cumpliendo con los estándares para la prevención de riesgos en materia fotosensible, y que estuviera lista para ser utilizada a nivel interno por parte de la organización a la que pertenece.

Para asegurar la efectividad de la herramienta, se ejecutaron más de 100 videos diferentes durante todo el desarrollo de la herramienta, comparándolos con los resultados por herramientas ya existentes como HardingFPA, lo que permitió validar los resultados obtenidos y detectar posibles errores de la herramienta.

A partir de ese momento, el objetivo cambió a mejorar la eficiencia de la herramienta, debido a los largos tiempos de ejecución que la herramienta realizaba durante sus análisis. Para ello, poco a poco se fueron afinando los umbrales de detección para hacerlos más precisos, a la vez que se aseguraba que los resultados obtenidos fueran correctos.

Una vez realizados los análisis de rendimiento, se establecieron y completaron una serie de tareas para refactorizar la herramienta en su versión final, con el objetivo de obtener una versión mejorada del prototipo inicial.

Por tanto, dados los objetivos planteados al comienzo del proyecto, los casos a analizar por la herramienta y el flujo de trabajo que lleva a cabo, se considera que se han cumplido con las expectativas generadas.

Debido a que la herramienta ha sufrido múltiples actualizaciones durante su desarrollo, iterando y mejorando tanto la efectividad de los resultados como su eficiencia en tiempo de ejecución, se considera que la herramienta es susceptible de seguir mejorando.

## 8.2. Trabajo Futuro

A pesar de que la herramienta cumple con los objetivos definidos al comienzo del proyecto, es posible continuar con su mejora y añadido de funcionalidades, como las siguientes:

- Implementación de la interfaz: proporcionar una interfaz de usuario permitiría un entorno más accesible para los usuarios. La idea es, dada la naturaleza comercial de la herramienta, investigar e implementar funcionalidades que el usuario final pueda utilizar, con un diseño que les permita familiarizarse rápidamente con la herramienta y que sea similar a la utilizada anteriormente por la compañía, para asegurar la correcta transición al uso de esta nueva herramienta.
- Análisis en tiempo real: a pesar de pasar por un análisis de rendimiento y los esfuerzos llevados a cabo para reducir el tiempo de análisis, el tiempo de ejecución de la herramienta está lejos de alcanzar un análisis en tiempo real. Un estudio de la implementación actual, junto a un trabajo de optimización podría mejorar el rendimiento y permitir en análisis en tiempo real, lo que reduciría los tiempos de ejecución.
- Soporte para imágenes de Alto Rango Dinámico (HDR): muchos de los videojuegos actuales permiten el uso de pantallas con HDR, ofreciendo mejores rangos de luminosidad en sus valores tanto altos y bajos. Esto puede suponer un problema, ya que puede dar lugar a mayores transiciones de flashes en un video. Incorporar el análisis

de imágenes HDR podría prevenir problemas fotosensibles cuando el video analizado utilice imágenes con esta tecnología

- Soporte para realidad virtual (VR): las gafas de realidad virtual crean la ilusión de que el usuario contempla un entorno virtual a través de sus propios ojos. Sin embargo, los dispositivos actuales separan la imagen origen en dos imágenes para crear la perspectiva 3D del usuario. Además, las gafas de realidad virtual utilizan lentes entre la pantalla y los ojos del usuario que distorsionan mejor la imagen y permiten lograr una percepción más realista. Estas características son peculiares para una pantalla convencional, sin embargo, se están desarrollando más videojuegos para realidad virtual y, dado que hay poca o ninguna distancia visual a la pantalla, los videojuegos y las experiencias de realidad virtual pueden ser un problema potencialmente grave para los pacientes con epilepsia fotosensible. La investigación sobre este tema permitiría implementar soporte para este tipo de pantallas.

Estas mejoras fueron planteadas y consideradas una vez las funcionalidades básicas del prototipo fueron implementadas. A pesar de ello, el trabajo llevado a cabo a continuación se centró en mejorar el rendimiento y acabar la refactorización de la versión final del proyecto.

Todas estas mejoras planteadas no pudieron llevarse a cabo por falta de tiempo, pero en cualquier caso son lo suficientemente realistas como para mantenerlas como posibles modificaciones a futuro y mejorar aún más la eficiencia de la herramienta



# Bibliografía

- Electronic Arts. Disponible en <https://www.ea.com>. [Online accessed: 20-06-2021].
- Rec. itu-r bt.500-11. Disponible en [https://www.itu.int/dms\\_pubrec/itu-r/rec/bt/R-REC-BT.500-11-200206-S!!PDF-E.pdf](https://www.itu.int/dms_pubrec/itu-r/rec/bt/R-REC-BT.500-11-200206-S!!PDF-E.pdf). [Online: accessed 25-April-2021].
- BATCHELOR, J. “gamesindustry.biz presents... the year in numbers 2020”. *gameindustry.biz*, Disponible en <https://www.gamesindustry.biz/articles/2020-12-21-gamesindustry-biz-presents-the-year-in-numbers-2020>.
- BINNIE, C., ESTEVEZ, O., KASTELEIJN-NOLST TRENITÉ, D. y PETERS, A. Colour and photosensitive epilepsy. *Electroencephalography and Clinical Neurophysiology*, vol. 58(5), páginas 387–391, Disponible en <https://www.sciencedirect.com/science/article/pii/0013469484901342>.
- BINNIE, C. D., EMMETT, J., GARDINER, P., HARDING, G. F. A., HARRISON, D. y WILKINS, A. J. Characterizing the flashing television images that precipitate seizures. *SMPTE Journal*, vol. 111(6-7), páginas 323–329, 2002.
- CARREIRA, L., RODRIGUES, N., ROQUE, B. y QUELUZ, M. P. Automatic detection of flashing video content. En *2015 Seventh International Workshop on Quality of Multimedia Experience (QoMEX)*. 2015.
- COLE, J. Graham Frederick Anthony Harding. 2018. Disponible en <https://history.rcplondon.ac.uk/inspiring-physicians/graham-frederick-anthony-harding> (último acceso, April, 2021).
- ENCYCLOPEDIA.COM. Wilkins, arnold j(onathan). Disponible en <https://www.encyclopedia.com/arts/culture-magazines/wilkins-arnold-jonathan>. [Online; accessed 3-May-2021].
- Epilepsy Society, 2020. Photosensitive epilepsy. 2020. Disponible en <https://epilepsysociety.org.uk/photosensitive-epilepsy> (último acceso, March, 2021).
- HARDING, G. “TV can be bad for your health”. *Nature Medicine*, vol. 4, pp. 265-267, 1998.
- HARDING, G., WILKINS, A. J., ERBA, G., BARKLEY, G. L. y FISHER, R. S. Photic- and pattern-induced seizures: Expert consensus of the epilepsy foundation of america working group. *Epilepsia*, vol. 46(9), páginas 1423–1425, Disponible en <https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1528-1167.2005.31305.x>.

- HARDING G.F.A., J. P. *Photosensitive Epilepsy. Clinics in Developmental Medicine*. Mac Keith Press, London, UK, 1994.
- HAZELIGHT STUDIOS. It takes two! Disponible en <https://www.ea.com/es-es/games/apex-legends>.
- HITACHI-SOLUTIONS INC. Video flash pattern analysis flicker check. Disponible en <https://www.hitachi-solutions.com/pdf/flickercheck.pdf>. [Online; accessed 17-April-2021].
- INTERNATIONAL LEAGUE AGAINST EPILEPSY. Colin david binnie 1938 - 2019. Disponible en <https://www.ilae.org/about-ilae/in-memoriam/in-memoriam-recent-and-archive-deaths/colin-binnie-1938-2019>. [Online; accessed 15-April-2021].
- ISO. “iso 9241-391:2016 - ergonomics of human-system interaction — part 391: Requirements, analysis and compliance test methods for the reduction of photosensitive seizure”. *ISO*, Disponible en <https://www.sis.se/api/document/preview/920133/>.
- ITU. “recommendation itu-r bt.1702 - guidance for the reduction of photosensitive epileptic seizures caused by television”. *ITU-R Recommendations*, Disponible en [https://www.itu.int/dms\\_pubrec/itu-r/rec/bt/R-REC-BT.1702-0-200502-I!!PDF-E.pdf](https://www.itu.int/dms_pubrec/itu-r/rec/bt/R-REC-BT.1702-0-200502-I!!PDF-E.pdf).
- JBA. Japan commercial broadcasters association. Disponible en <https://www.j-ba.or.jp/category/english>. [Online; accessed 17-April-2021].
- JEAVONS, P. M. y HARDING, G. F. A. *Photosensitive epilepsy : a review of the literature and a study of 460 patients*. Heinemann [for] Spastics International Medical Publications ; J. B. Lippincott Co London : Philadelphia, 1975. ISBN 0433172010.
- OFCOM. “ofcom: Guidance notes - section 2: Harm and offence”. *Ofcom: Guidance Notes*, Disponible en [https://www.ofcom.org.uk/\\_\\_data/assets/pdf\\_file/0023/104657/Section-2-Guidance-Notes.pdf](https://www.ofcom.org.uk/__data/assets/pdf_file/0023/104657/Section-2-Guidance-Notes.pdf).
- OFCOM. “the ofcom broadcasting code”. *Ofcom Broadcasting Code*, Disponible en [https://www.ofcom.org.uk/\\_\\_data/assets/pdf\\_file/0005/100103/broadcast-code-april-2017.pdf](https://www.ofcom.org.uk/__data/assets/pdf_file/0005/100103/broadcast-code-april-2017.pdf).
- RESPAWN ENTERTAINMENT. Apex legends. Disponible en <https://www.ea.com/es-es/games/apex-legends>.
- RODRIGUES, N. Automatic detection of epileptogenic video content. En *Automatic Detection of Epileptogenic Video Content*. 2015.
- UK NHS, 2021. Epilepsy. 2017. Disponible en <https://www.nhs.uk/conditions/epilepsy/> (último acceso, March, 2021).
- Vivir con Epilepsia, 2017. Epilepsia fotosensible, o como las imágenes pueden provocar crisis. 2017. Disponible en <https://vivirconepilepsia.es/epilepsia-fotosensible> (último acceso, March, 2021).
- W3. Web content accessibility guidelines (wcag) overview. Disponible en <https://www.w3.org/WAI/standards-guidelines/wcag/>. [Online; accessed 15-April-2021].



W3. Three flashes or below threshold: Understanding sc 2.3.1. Disponible en <https://www.w3.org/TR/UNDERSTANDING-WCAG20/seizure-does-not-violate.html>. [Online; accessed 4-April-2021].

W3. Web content accessibility guidelines (wcag) 2.0. Informe técnico, World Wide Web Consortium (W3C), 2008a.

W3. Web content accessibility guidelines (wcag) 2.2. Disponible en <https://www.w3.org/WAI/WCAG21/Understanding/three-flashes-or-below-threshold.html#dfn-flash>.

W3. Web content accessibility guidelines (wcag) 3.0. Informe técnico, World Wide Web Consortium (W3C), 2021.

WIKIPEDIA. Angle of view. Disponible en [https://en.wikipedia.org/wiki/Angle\\_of\\_view](https://en.wikipedia.org/wiki/Angle_of_view). [Online; accessed 23-April-2021].

WILKINS, A., BINNIE, C. y DARBY, C. Visually-induced seizures. *Progress in Neurobiology*, vol. 15(2), páginas 85–117, 1980. Disponible en <https://www.sciencedirect.com/science/article/pii/0301008280900040>.

WILKINS A., H. G., EMMET J. Characterising the patterned images that precipitate seizures, and optimising guidelines to prevent them. *Epilepsia*, vol. 46(8), páginas 1212–1218, Disponible en [https://www.researchgate.net/profile/Arnold-Wilkins/publication/7686482\\_Characterizing\\_the\\_Patterned\\_Images\\_That\\_Precipitate\\_Seizures\\_and\\_Optimizing\\_Guidelines\\_To\\_Prevent\\_Them/links/59e3a576aca2724cbfe3aeb/Characterizing-the-Patterned-Images-That-Precipitate-Seizures-and-Optimizing-Guidelines-pdf](https://www.researchgate.net/profile/Arnold-Wilkins/publication/7686482_Characterizing_the_Patterned_Images_That_Precipitate_Seizures_and_Optimizing_Guidelines_To_Prevent_Them/links/59e3a576aca2724cbfe3aeb/Characterizing-the-Patterned-Images-That-Precipitate-Seizures-and-Optimizing-Guidelines-pdf).

YOUTUBE. The white stripes - seven nation army (official music video). Disponible en <https://www.youtube.com/watch?v=0J2QdDbelmY>. [Online; accessed 01-Jun-2021].

YOUTUBE. London 2012 olympics seizure. Disponible en <https://www.youtube.com/watch?v=vs0hfhSje9M>. [Online; accessed 01-Jun-2021].

YOUTUBE. Banned pokemon seizure scene. Disponible en <https://www.youtube.com/watch?v=4LQLvgXLguk>. [Online; accessed 15-April-2021].

YOUTUBE. Pot noodle advert 1993 uk. Disponible en <https://www.youtube.com/watch?v=HVQuA9HNh0M>. [Online; accessed 15-May-2021].



# Appendix A

## Appendix A - Data Visualization

This appendix contains the code used for data visualization. In order to generate graphs with the obtained information of the analysis, there is only needed to execute the Python file with the name of the .csv file generated after the analysis of the tool.

```
# Locate the csv data file in the same directory as this .py file  
# and execute the command "python ReadCSV.py filename", being 'filename' the csv file.
```

```
import pandas as pd  
import os  
import matplotlib.pyplot as plt  
import sys
```

```
filename = sys.argv[1]  
cwd = os.getcwd()
```

```
def showPlot(filename):
```

```
    #read data from csv  
    data = pd.read_csv(cwd + "\\\" + filename, sep=',')  
  
    # Plot Luminance Transitions  
    plt.figure(figsize=(16, 9))  
    x = range(len(data['Frame']))  
    plt.plot(x, data['LuminanceTransitions'])  
    plt.axhline(y=6, color='r', linestyle='-', label='threshold')  
    plt.legend(loc='best', bbox_to_anchor=(1,1))  
  
    plt.xlabel('Number of Frames')  
    plt.ylabel('Transition Count')  
    plt.title('Total Luminance Transitions')  
    plt.show()
```

```
# Plot Red Transitions
plt.figure(figsize=(16, 9))
x = range(len(data['Frame']))
plt.plot(x, data['RedTransitions'])
plt.axhline(y=6, color='r', linestyle='-', label='threshold')
plt.legend(loc = 'best', bbox_to_anchor=(1,1))

plt.xlabel('Number of Frames')
plt.ylabel('Transition Count')
plt.title('Total Red Transitions')
plt.show()

# Plot Pattern Risk
plt.figure(figsize=(16, 9))
x = range(len(data['Frame']))
plt.plot(x, data['PatternRisk'])
plt.axhline(y=0.4, color='r', linestyle='-', label='threshold')
plt.legend(loc = 'best', bbox_to_anchor=(1,1))

plt.xlabel('Number of Frames')
plt.ylabel('Estimated risk')
plt.title('Estimated pattern risk')
plt.show()

# Plot Flash Failed Frames
plt.figure(figsize=(16, 9))
x = range(len(data['Frame']))
plt.plot(x, data['FlashFailedFrames'])
plt.axhline(y=1, color='r', linestyle='-', label='threshold')
plt.legend(loc = 'best', bbox_to_anchor=(1,1))

plt.xlabel('Number of Frames')
plt.ylabel('Failed frames')
plt.title('Total Flash failed frames')
plt.show()

# Plot Pattern Failed Frames
plt.figure(figsize=(16, 9))
x = range(len(data['Frame']))
plt.plot(x, data['PatternFailedFrames'])
plt.axhline(y=1, color='r', linestyle='-', label='threshold')
plt.legend(loc = 'best', bbox_to_anchor=(1,1))

plt.xlabel('Number of Frames')
```

```
plt.ylabel('Transition Count')  
plt.title('Total Pattern failed frames')  
plt.show()
```

```
return 0
```

```
showPlot(filename)
```



## Appendix B - Contributions

This appendix contains the made contributions to this project:

### B.1. Blanca Macazaga Zuazo

Throughout the development of this project, I have worked to achieve the objectives set at the beginning of the project in collaboration with my colleague. My contributions to the project are:

- Continuous research on the selected topic throughout the development of the tool.
- Researched the technology to be used, OpenCV, to verify if it would support the needs of the project by implementing small tests to validate the library.
- Researched the different color spaces and their differences to understand the luminance and red saturation conversions, as it later served to discard the usage of the Y'CrCb color space.
- Researched existing tools and comprehended their functionality in order to improve the tool, as the transition count implementation for flashes, and the adaptation for the pattern risk check too, has been inspired by existing tools.
- Continuously created videos to test the tool and test existing tool for comparison, validation and feature understanding.
- In the prototype, collaborated defining the logic and implemented the luminance conversion, collaborated in the luminance frame difference and area check implementation, implemented the flash detection and the criteria check.
- In the prototype, collaborated in planning the logic for the red saturated flash detection and implemented the red difference for the deprecated method.
- Implemented the straight line detection.
- In the pattern analysis, implemented pattern isolation, connected components and the similar components functions. Also collaborated in the stripe separation.

- Run several performance analyses to detect and correct detected bottlenecks.
- Refactored the project to use the `textitResourcesTracker` class to reduce memory usage in the tool after detecting the miss-management in used dynamic memory.
- Implemented the risk estimation class.
- Defined the logic and implemented the *SafeArea* class.
- Defined and implemented the file persistence and the `.csv` serializer.
- Defined all the logic and implemented the refactoring for the final version.
- Updated the `VideoLib.Tests` project increasing the number of tests to 111 and setting the code coverage to 97%.
- Completed a total of 61 user tickets for the tool's development.
- Wrote chapters 3 and 4, collaborated in chapter 6 and contributed with minor corrections to chapters 1, 2 and 5.

## B.2. Javier Landaburu Sánchez

For the development of this project, and given the magnitude that it supposed from the beginning, we decided to work together during the course of its development, with weekly meetings that would allow us a common point of view of the project. My first steps were to research about the chosen topic, trying to understand the problem of photosensitivity, its previous studies and the use of existing tools. This ended up with a small common bibliography for the development of the tool.

At the same time, we both investigated which technologies were best suited for obtaining image and video data. Once we decided which technologies and tools to use, we began to identify the first steps to take. Since the first objective was to obtain data related to flashes, my first objective was to research on the different data format of digital image representation and its differences (RGB, YUV, YCrCb, ...). This ended up implementing the first version of saturated red colors recognition.

Given the increase in data that the project was gradually including, I included a configuration file that would contain all the used parameters and variables, so its modification would not include changes in the code.

Once the first version of the flash recognition was completed, I started to investigate the geometric pattern recognition, which meant in differentiating between linear and circular pattern recognition, so both implementations were carried out separately. I ended up doing a first implementation of both functions.

To test the tool, I created multiple videos manually that covered all the areas of analysis of the tool: changes in luminance, colors, geometric patterns, areas on the screen, intensities, time... To validate the obtained results, the `HardingFPA` tool was used at the same time, having at least two analysis per video.

In order to improve the performance of the tool, I investigated the possibility of migrating the project to a `C++` environment, using the original graphics library (`OpenCV`),



so we could use the hardware acceleration provided by the computer's graphic card. After several prototypes and tests, this possibility was rejected due to the low profit in execution time and the high cost of refactoring the entire project.

To visualize data from the CSV generated by the tool, I developed a small Python script (see Appendix A) that generates graphs of the obtained data, which were later introduced in chapter 5.

At the same time that some functionalities of the tool were already implemented, I made a small documentation that contained a user guide for the first versions, in order to be able to test the tool with videos.

Given the professional nature of this project, I stopped having access to the tool's code 3 months before its completion, so my colleague Blanca continued to improve the performance of the tool, refactoring large parts of the code.

Finally, for the realization of this report I have written chapters 1, 2, 5, 6, 7, 8 and Appendix A.

