Determining player decisions in computer games automatically from gameplay footage

A.C. van den Brink

Supervisor: Remco Veltkamp
Thesis #: 2710521

Utrecht University, the Netherlands

August 2021

Some computer games present interesting moral dilemmas to the player. However, it is labour-intensive to obtain data of decisions made by players through questionnaires or close observations. This thesis describes an attempt to automatically collect data about decisions players make in games to analyse the behaviour of large numbers of players using gameplay footage. We use simple image features, SURF, and shot boundary detection and conclude that while we cannot accurately determine decisions made by players, we can significantly simplify the problem. This research is based on gameplay footage of "This War of Mine" found on YouTube.

Contents

1 Introduction

1.1 Motivation

Computer games, such as *This war of mine*, present various moral dilemmas to the player. These type of games is seen as potentially valuable for the ability to promote cognitive and affective empathy [1]. Therefore, analysing how a player acts in these games can support peace education and conflict resolution.

Non-serious games are made for entertainment. The incentive for companies to store and expose data about player choices and behaviour is seldom present, as it can be expensive to collect this data for little to no gains to the company. However, non-serious games often dwarf serious games in the sheer volume of players. Due to the volume, they contain valuable data, with both a larger and more diverse set of the population represented. This information can then be used for other studies, such as how games contribute to the remembrance of war stories and tragedies [2].

The traditional way of analysing games is to let people play such games in a controlled environment, where the player's choices and the game states are either recorded by a camera [3], are required to fill in questionnaires [4] or keep a diary while playing [5]. This approach is feasible for a small number of people but becomes unfeasible for large groups. Since it is labour-intensive to analyse the choices made by players in such a game, it is hard to obtain large amounts of data, and hence difficult to draw meaningful conclusions [6]. Automating this process will allow much larger data sets to be considered resulting in greater significance.

Various players share footage of games on platforms such as YouTube. Hence, YouTube offers large data sets of gameplay videos. A bonus is that the videos are not intended for research and are recorded by the players themselves. Therefore the recordings do not contain any bias introduced by the researchers and the data might therefore be a closer match with how those players would react in their natural environment. However, not much research has been done specifically on analysing the player decisions automatically in video footage [7].

This War of Mine is chosen as a case study because it is popular and offers a myriad of moral dilemmas to the player. It is also the subject of many other studies [2, 8, 10], which shows there is a genuine interest in the information this game contains. Automating it could aid in such research, as many of them consider only small subsets of data due to the difficulty of obtaining it[6, 12].

Our objective is to reduce the amount of manual labour involved to gather information from gameplay videos. Thus, facilitating the usage of much larger data sets in other studies. We are specifically interested in information about the state of the game, such as "does the player do X?" or "does situation Y happen?".

1.2 Overview

1.2.1 Research approach

In this thesis, we first attempt to find the answers directly from the video (the top bar in the figure1) with moderate success. This led us to a new approach to first reduce the video to its interesting sections (the bottom bar in 1).

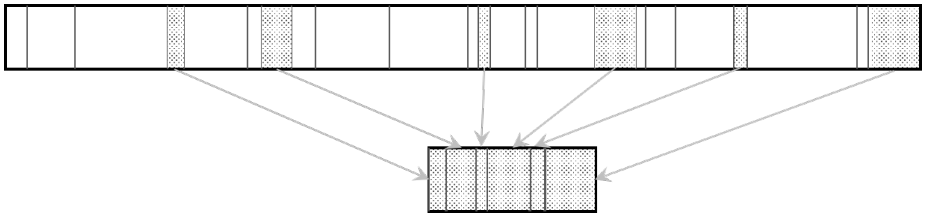

Figure 1: A high-level overview of a typical video

1.2.2 Contents of Thesis

We first describe the game *This War of Mine*. Next, we give an overview of related work. We then describe an attempt to recognise specific choices made by players in various scenarios automatically and conclude with our results.

From the manual annotation required to construct the ground truth for the pilot study, we find that the most labour-intensive part is to find the correct section, rather than conclude the answer from that section. In our chapter on separating game states, we attempt to divide this footage into its respective game states. The actual question can then be answered by a more specific algorithm or a human. This pre-process step greatly reduces the amount of footage that has to be considered. Making it faster to analyse large amounts of footage by only considering relevant sections.

2 Case study "This War of Mine"

This chapter describes the game *This War of Mine* in detail to understand the decisions players make, and consequently the data we attempt to obtain. It highlights the different game states present in the game loop and gives insight into how to solve the problems posed in the pilot study.

2.1 Introduction to the game

This war of mine is a game where the player manages a group of survivors in a war zone with the goal of surviving until the end of the war. The game confronts the player with survival problems, such as famine, heat, shelter, lack of police and safety. The game connects these moral choices with typical consequences. The motto of the game reads: "In war, not everyone is a soldier", which depicts in clear terms what the game is about. The player's primary goal is the survival

of his group by any means necessary. As the war prolongs, resources become scarcer and the choices more drastic. The player has different methods at his disposal for achieving this objective. He could share his food with the people he meets, or steal their food. He also chooses how to spend his resources, creating a good bed to sleep on, or a kitchen to cook food. He could also invest in weaponry, to defend himself or aggressively take resources away from less armed groups. The game becomes more challenging towards the end, as the season shifts from autumn to winter. This presents even more problems like generating enough heat and having thick enough clothes. A good book and chair might be valuable to keep sane during such times, but both of those can be burned to survive the night.

The game features a sequence of states that the player goes through in chronological order. These states represent the day and night cycle that is present in the game. We split this cycle into the two major sections Day and Night, we then further subdivide the Night into four more sections, character select, location select, inventory select and scavenging.

2.2 Game states

2.2.1 Game state: Day

The game starts during the daytime, where the sun is up. During this state, the player can interact with and construct his home, spend his resources on new items or consumables such as food. This is the time that the player takes care of the needs of the group. Morale is raised by telling stories, music, or smoking. Furniture is created or torn apart for firewood and he can listen to the radio to see if there is any light on the horizon. The player must deal with the personal problems that the characters face and greet people at the door asking or trading for food, shelter, or supplies. This phase plays very much like a management game such as the SIMS, where characters have draining statistics such as hunger that can be replenished by spending food items.

Figure 2: The player's "home", where the player spends the day

Note that the scenery is displayed with stains and scratches to create a war-like look-and-feel. This makes it more difficult to automatically detect elements.

2.2.2 Game state: Night

The second major state the player will be spending his time is during the night, this part of the game is also known as "scavenging" and is where our interest is focused. While the day is used for the player to spend his resources, the night is used for gathering them. The night has three minor parts that are passed in sequence before the actual scavenging starts.

Figure 3: A player selecting characters and location during the night

2.2.3 Character select

The player gets to decide how his characters will spend the night. He can choose to let them sleep, which leaves the house open for invaders to come and steal his resources. He can also let them stand guard to prevent that from happening, but this poses the risk of them dying defending superior armed forces and causes adverse effects due to sleep deprivation. These two things can be mixed and matched as many times as the player has characters. A third option can be chosen only once and sends that character out to scavenge. The player can opt not to send someone to scavenge, which will skip the entire game state of Night.

2.2.4 Location select

If a character is sent out to scavenge, the player can choose the location his character goes to. The game gives the player a vague indication of when and where certain items can be found both by indicator as well as common sense. For instance, the player will find mostly food in a supermarket. The player must assess what he wishes to gather this night, and how able he is to do so. Armed forces might have many supplies, but if the player is not able to overpower or trade for them, he will not be able to get to them. The game also randomizes other scavengers on the map, meaning that if he does not scavenge something this night, it might not be there anymore on the next.

2.2.5 Inventory select

After the player has chosen his location, we are presented with the third sub-state of the game best described as his inventory management. The player is

presented with every item he has available to him in his home and can choose what he wishes to bring along during the scavenge run. It is here that the player decides to bring weapons to fight or supplies to trade or give. The trade-off here is that every character can carry a limited number of items, bringing a shovel or weapon along from your hideout will reduce the number of other items he will be able to bring back provided he does not lose them.

Figure 4: Inventory select (left is taken, right is available)

2.2.6 Scavenging

After these steps of preparation, the player is put in the chosen location with the selected character and equipment. While on-site the player is free to move around in a 2d environment. Within this environment, he can find resources in different areas, as well as interact with other NPC's (non-player characters) to scavenge. During this phase, the game plays like a 2d platformer where the character cannot be directly controlled. The player navigates by clicking somewhere, and the character will automatically go there using the shortest route. The player can interact with containers and collect resources by clicking on the interactive icons that are present throughout the map.

Figure 5: Scavenging with interactable icons

2.3 The part of the game we focus upon

Our research focuses on when the player first enters the supermarket, one of the many locations the player can select from. This has a chance of triggering an event that presents the player with the moral dilemma we are interested in. The player generally comes there to scavenge for food and other use full materials. While the player is present, there is an interaction between an armed soldier and another female scavenger that suggests rape is about to occur. The player can intervene or wait for the scene to play out. Alternatively, the player can ignore the scene altogether and scavenge while the scene plays out.

The dilemma is in that the player does not know the female, and his objective is to gather supplies for his group. Fighting an armed soldier can put a large burden on this group, as the player might get wounded, killed or driven off rendering him unable to collect any supplies this night. The consequence for this can be dire for members of his group. Not intervening does not cause the player any issues, and he is free to scavenge the supermarket without problems, though it is a morally questionable act.

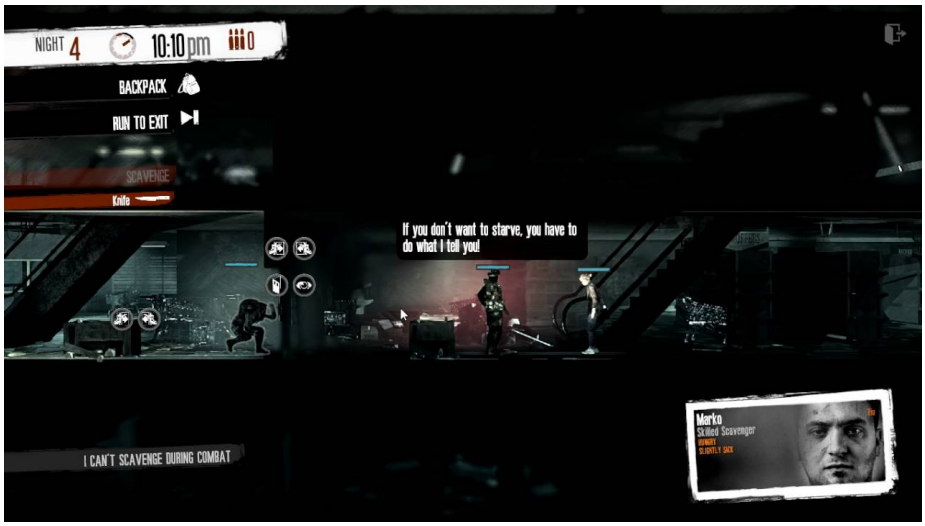

Figure 6: Start of the supermarket scene (player left, soldier middle, woman right)

3 Related work

Computer gaming has become a popular way for young people to spend their free time. Being able to collect data from how games are played can open an analysis of many different types of games for academic research. This information is relevant for other research disciplines like Game and media cultures that want to analyse the behaviour of players. Computer games can be valuable as they contain many different types of scenarios and situations that people would normally not be confronted with in real life. They are also consumed in mass, with a large majority of people in the world gaming regularly and producing large amounts of videos which would provide a large amount of data very quickly. Especially with the large amount of diversity that games offer in both themes, play style, ethnicity and situations, this might bring an answer to many hypothetical questions that could be used for further analysing human thought processes. Analysing footage from YouTube about games has been done in an earlier work where they attempt to find bugs in the game by scanning the footage [22]. This work indicates that analysing footage from YouTube is feasible.

3.1 Game Studies

Various studies exist that attempt to analyse player behaviour for multiple purposes. One study analysed the game "Catch 22", a serious game, developed for supporting juveniles in their ability of moral argumentation[9]. Another study analyses a games' ability to allow players to confront and reflect on difficult occurrences in life[11]. At the same time, this study [10] uses the game "This War

of Mine" to investigate the occurrence of affective learning instances in video game environments.

To analyse the player behaviour in games, such studies [10] are often conducted in controlled environments, where the researcher monitors the player while playing the game and records the choices and interactions. This process is labour-intensive [29] and therefore often small in scale. This creates problems with using small sample sizes, where either the results are insignificant, or cannot properly be generalized.

This War of Mine has been the topic of many other studies [2, 8, 10] due to its cultural impact, and significance to how memories and frames are passed through generations [2]

With large data sets of videos of such games readily available on platforms such as YouTube, the question remains if such questions about games can be answered automatically from footage alone. This would greatly increase the sample size of studies that use games as a medium and require no extra effort from either the creators or the players. Recent studies have attempted to obtain such information from such footage [8], using traditional image detection methods and neural networks to predict player decisions from gameplay footage.

Footage from such data sets is often heavily edited, and poorly filtered. The videos are classified using "tags" that users and creators can assign to them, but these classifications are often too general, or absent. This means that effort must be put into manually filtering these videos, and often need further preprocessing to obtain the relevant footage from the videos.

3.2 Image Recognition

We use methods from the domain of image recognition. Image recognition is a field of computer science that uses various techniques to recognise patterns in images. We give a short overview of related techniques and methods, providing a full overview of this field is beyond the scope of this thesis.

Image features are the simplest features contained in images. For example, colour histograms can be extracted from individual frames and give information about the frequency of colours. This feature is used extensively in many different fields, from analysing sports footage [13] to effects on twitch streams [14]. Colour histograms are computationally cheap, easy to obtain, and frequently used.

A more complex feature is known as the Scale-Invariant Feature Transform (SIFT) [15], which finds a set of descriptive key points (features) of an image that are "largely invariant to changes in scale, illumination and local distortions". While the original usage of SIFT is to detect objects in images, it can also be used to extract information from frames in videos. SIFT features have been used to estimate camera viewpoint in soccer videos [16] by creating a bag-of-visual-words through K-means clustering. An improved version of the SIFT algorithm was later developed known as SURF (Speeded Up Robust Feature) [17]. SURF is more frequently used in the video processing domain due to its faster processing time. Both SIFT and SURF are implemented in the OpenCV library [18].

4 Pilot Study

In this chapter, we describe an algorithm to answer some specific questions posed to us by other researchers from the media and performance studies field, that could aid them in their studies. We use simple image recognition techniques and knowledge of the game to obtain our results.

4.1 Research questions

The research question of the pilot study is based on what information would be relevant for further research. We present a series of questions and attempt to answer them. We then list the potential problems that we face and outline our approach.

We attempt to answer the following questions:

- Q1. What character does the player choose to start his scavenging run?
- Q2. What characters is the player able to choose from?
- Q3. What weapons does the player take along with him?

These questions can give insight into what choices the players make, and what could influence them to make different choices. Interesting examples could be whether players are more likely to intervene if they are armed or to see whether the player is more likely to interfere if he has more (or fewer) characters in his group.

4.2 Approach

The general approach we take to answer these questions is to follow the steps we take as humans to answer these. We start by using simple methods to identify indicators and use logical reasoning along with specifics about the game to answer the research questions. For instance, we count the number of people before and after scavenging, to determine if someone died, since that is the only way to reduce your character count during that game state.

The videos we use are gathered by filtering for tag and title about the scenario using a so-called YouTube scraper and can be found in the appendix. We might need further processing for some of our solutions to work and implement those steps when they are required. Note that the footage does not have clear endings, beginnings, or chronological ordering. As such, we write our solution to work on all footage of sufficient length to contain a full sequence of the following game states in their given order (map selection, inventory selection, scavenging). While there is no guarantee that these sections are present or presented in that order, most footage is generally made while playing the game, without editing the order of the scenario. This implies that almost all footage preserves its chronological order.

We consider a data set of 313 videos obtained by scraping YouTube for videos. The data set is manually filtered for videos that contained footage of

the game we are interested in, and then screened for the actual scene we were looking for (the supermarket).

4.2.1 Q1: What character does the player choose to start his scavenging run?

We attempt to answer this question in a few videos manually, and we quickly noticed that the UI (User Interface) gives a clear and immediate answer to this question. As is visible in figure 7, the UI indicates what character is playing. Answering this question is therefore as simple as finding a frame with the character clearly displayed and identifying who it is.

Figure 7: UI element showing character (bottom right)

Translating this solution to an algorithm, we apply SURF (Speeded Up Robust Features)[17] feature matching to the videos to identify the relevant frames. Template matching is less effective due to the recording quality and scaling of different features in the game.

We iterate over the frames of the video and look for the "RUN TO EXIT" button that is only present during the scavenging state. Once this is found we attempt to detect what character is present in that frame. To detect the button, we used SURF [17], combined with a heuristic based on how well the matched points maintain the width to height ratio of the button. Once we determined a suitable threshold, we then looked for the character portrait that had the most aligning matches. Some character portraits are quite similar, but we did not need any additional steps as there is always exactly one character present. Hence take the best match and ignore all other results.

For this specific step, we use the C++ library OpenCV (version 3.3.0) [18]. We check once every 250 frames of the video if a particular image of the button

is present, using SURF as a feature matcher. The button image is manually retrieved from the video. The matches are then matched using a brute-force k-nearest neighbour matcher to find pairs of the best matches found in the image. We filter out the good matches by taking only those that would not match well with other elements in the image (where the difference between the best and second-best match would be at least 0,8[23]). To combat spurious matches, we look at the corners of the image. Since the image is always square, we check whether the width/height ratio of the original image was preserved in the matches found, to eliminate the effects of scale and resolution. If the ratio does not diverge more than 20% from the original ratio, and at least 40 matches are used in this comparison, we conclude the image is present in that frame. For this algorithm, the default parameters are used that are present in OpenCV [18]. Alterations to the parameters do not produce better results.

4.2.2 Q2: What characters was the player able to choose from?

Our approach to answering this question is similar: find a frame where the player is presented with this choice and find the most matching characters in this frame. We identified the frame by backtracking from the answer of Q1. This way, we can be certain that the work done in answering Q2 provides an answer in relation to Q1. It is therefore a requirement that the chosen character found by answering Q1 must be present in the answer of Q2.

Q2 is more difficult to answer, as we can no longer assume there is exactly one character present. Instead, the number of characters in the frame can range from one to four. Some characters are quite similar, and this makes it hard to distinguish them using SURF. This results in multiple characters being detected by the algorithm even though only one of them is present. We changed the parameters of the heuristic to be stricter to accommodate for this. An alternative solution would be to mask any recognized characters so that no new matches are found with similar data.

We use the frame found in answering Q1 as a baseline and backtrack 100 frames at a time until we find a part of the UI of the character select state. Then we use the same method as is used in answering Q1 to analyse what characters are present at this screen, up to four, but with stricter parameters.

4.2.3 Q3: What weapons did the player bring?

For this question, we take the same approach as the first two questions. Since this question also bears a close relation to answering Q1 and Q2, we use unique UI elements to find the matching item selection frame, while backtracking from the frame of used by answering Q1. Finding the correct frame is simple, but it is quite difficult to recognise the individual items on that frame, unlike the characters we recognised from previous frames.

Figure 8: Simple item (left) and complex item (right)

Figure 8 shows two examples of images we attempted to detect. There is a clear difference in the fidelity and complexity of the items, which made it difficult to find parameters that work for the general case. We attempt to cut the items out of their context in the hopes to improve our results. We mask out all individual items from the scene and detect them one by one. This masking is done using edge detection to find the semi-bordered regions of each item and cutting them out of the original image using the best matches from template matching. This works well in some instances, but we find quite some exceptions.

Figure 9: The area to recognise items from

As shown in 9, the shovel (first item) has several status indicators that merge with the borders we are trying to recognise. The top part of the shovel is also

occluded by a health bar. This makes it challenging to recognise due to the lack of distinct features for SURF[17]. Another problem is the cracks that are present throughout the background. Aesthetically this makes it look like an old worn piece of paper but makes it more difficult to filter out the areas using edge detection due to the edges becoming obfuscated by the noise. A threshold on intensity is also ineffective, due to the background often being brighter than the borders we wish to detect. Pattern matching also fails for the same reason as items are problematic to detect, a noisy background and overlay elements.

4.3 Results

Our findings for all three questions can be found in table 1,

Table 1: Results of Pilot study (34 videos analysed)

	Correct	Incorrect	Partial
Q1	25	9	-
Q2	12	9	13
Q3	9	19	6

- Our algorithm has a success ratio of 73% when attempting to answer Q1. We notice that the incorrect results stem from the same videos, where it fails to recognise the correct frames. Upon closer inspection, we observe that those videos are recorded on the PS4, which uses a slightly different interface and makes our algorithm fail to detect the proper frame. This question cannot have partial answers, since we are guaranteed to only have one character present.

- Our algorithm seems to have trouble finding the complete results when attempting to answer Q2. The number of results that is at least partly correct is also 73%, which indicates it is good at finding the correct frame but inaccurate at finding the correct results within that frame. Considering that our approach to answering Q1 has a high success ratio, this is likely because the algorithm cannot assume a certain number of answers and does not currently consider positional data. Hence portraits that look similar are both be considered "present" even though only one of them is. Another observation is that our approach to answering both Q1 and Q2 have the same number of incorrect results. This is because the algorithm that attempts to answer Q2 builds upon the answers of Q1.

- Our algorithm performs poorly when attempting to answer Q3 with a low success rate (26%), and only reaches 44% when including partial results. This is because of the problems mentioned earlier of using SURF[17] to recognise low-feature objects on a noisy background. This question requires a different method to answer correctly.

During the construction of the ground truth, a data set of 313 videos was used. We manually filter those for all occurrences with the "supermarket scenario" which left only 34 videos remaining, which were analysed by the algorithm. Filtering out the videos where the scenario occurs for this data set is more labour-intensive than manually answering the questions. This is because the occurrence of the scenario is scarce, and YouTube filters cannot distinguish between levels or game states. Filtering the scenario correctly prior to answering these questions is a better way of minimizing human labour to obtain these results, although likely both are required for large sets of videos.

We also notice that a few results contain a different answer than what the algorithm searches for. The algorithm searches for the first occurrence of a specific image in the UI, though for some videos the supermarket scenario is not the first scenario played by the player. The results of answering Q1 and Q2 are therefore expected to be slightly better if we process the data further to exclude all non-supermarket scenarios.

4.4 Discussion

YouTube videos have a large spread of quality and recording methods. Even after filtering on videos uploaded in 720p the diversity is still considerable. For instance, some people upload 480p with black borders such that the video becomes 720p while others scale 480p up to 720p, so it becomes blurred, yet 720p.

Another common trend for you-tubers is to record themselves while playing. They hide some part of the screen (commonly UI) and display their webcam feed there instead. This means the specific parts of the video that the algorithm attempts to find can be occluded by the webcam feed, which makes such videos especially hard to recognise.

The videos are recorded with different versions of the game. Games try to reach as large an audience as possible, and therefore many of them feature the same game with other translations. This makes it so that text cannot be used for detection, and also scales certain elements (especially UI) to fit the text. This means important elements can be occluded in the Dutch version, whereas they are not in the English version.

Games tend to have many modifiers applied to elements such as health bars. These occlude part of the entity they belong to and whatever is behind them, making it hard to detect objects that mix and match elements, such as health bars and status indicators. This is especially true if those status indicators are represented by text alone, such as if the characters are thirsty or tired.

Lastly, some videos are edited after recording them. For instance, the screen might zoom in on the selected items during the 'item select' state. This makes it more apparent to the viewer which items are selected but makes it hard for the algorithm to find patterns and images that are now either scaled or at different positions than it expects.

The method used so far is effective at answering Q1 (what character the player chooses to scavenge with) but is ineffective for answering Q2 and Q3 due

to the increased amount of noise introduced for these elements, as well as increased difficulty in recognizing the correct frame that contains the answers. We conclude that a different approach is required to answer more complex questions for this game.

In our construction of the ground-truth for this pilot study, we find that the most labour-intensive process is not to conclude the result of a given video but to find the section of the video where the result can be observed. On average, about 90% of our time during manual annotation was spend searching for a relevant section, while concluding the answer from that section is fast. This is because most of those questions are immediately obvious to humans from a small amount of footage, yet it can be difficult for an algorithm to detect.

5 Separating game states

5.1 Research question

The most labour-intensive process in the pilot study is to filter all videos containing our scenario of interest. After filtering the 313 for videos that do not contain any gameplay footage at all, we were still left with 274 videos, of which only 34 contain the scenario of interest (the supermarket).

The general objective of this thesis is to reduce the manual labour required to acquire ground-truth-data similar to what is created in the pilot study, in order to make an analysis of larger data sets feasible. In this chapter, we shift our focus to the process of filtering relevant video sections from large data sets such as YouTube by pre-processing the data into game states.

We focus on reducing the most labour-intensive process of our pilot study. Therefore, our research question for this chapter is:

- Q4: Can we automatically subdivide gameplay footage from YouTube into sections that represent different game states, such as "gameplay", "inventory", "menu".

If we can achieve that division, we can reduce the footage that has to be considered by the pilot study. Since it is immediately obvious what game states are relevant to answer such questions. This reduction would allow us to analyse only a small subset of the data since only specific game-states have relevant information.

5.2 Approach

Movies are often composed of several cuts, transitions, and segments. Humans find such concepts easy to understand, and therefore questions such as "forward to the next section" or "edit section five" are frequent occurrences, especially during production. To answer those questions, algorithms have been developed to alleviate the requirement of manual labour in identifying the sections for such tasks. These algorithms do not yet produce perfect results, but that impacts us

less because we consider large data sets. Missing a few transitions is not likely to affect our results in a significant way.

Gameplay videos often consist of many state transitions that can be easily identified by humans. Games are just state machines with a finite number of states and transitions between them. Several states can be uniquely identified with a common classifier such as "main menu" or "in game". Our approach focuses on filtering the different states from gameplay videos. This allows further processing to only consider the states of interest, severely reducing the amount of footage that needs to be considered and reducing the variance in such sections.

We use an SBD (shot boundary detection) algorithm[13] in our attempt to separate these states since switches between game states are nearly always aligned with transitions. The current state of SBD algorithms defines a differentiation between instant transitions and gradual transitions. Instant (cut) transitions are defined by a sudden change from frame to frame; such transitions are often abrupt and are considered easy to detect. Gradual transitions are known as "fades", "fade to black" is an example of such a fade that often occurs in our game and are considered harder to detect than cuts. We have taken an algorithm published here [13] because it is recent and achieves good results. It attempts to find both transitions using various simple metrics and smart applications. An overview of the algorithm can be seen in 10. We implement this algorithm to recognise the different states in the game.

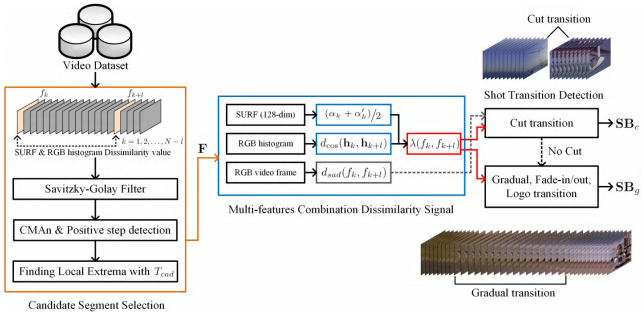

Figure 10: The process of the SBD algorithm[13]

5.3 Implementation

To detect gradual and instant transitions, an inter-frame distance is defined. The idea behind this distance is that when a gradual transition occurs, the difference between two consecutive frames is not large, but if you compare the current frame to twenty frames later, the difference will be large. The inter-frame distance for this experiment has been set to five, in accordance with the original implementation.

We then define two separate measures by which we determine if frames are "similar" or not. The first of such measures is done through SURF[17] descriptors. We take the descriptors of the current and current-n frames and

compare how many descriptors match. This is done by ranking the descriptors from most matching to least matching and only considering the matches where the model image and the target image refer to one another as their top match. To translate this value into a scalar between zero and one, we then divide the number of matching descriptors by the total number of descriptors.

The second measure is known as a histogram comparison. We take the histogram for all colour channels (RBG) and perform a Hellinger distance [19] calculation on them to retrieve a scalar value indicating how well they match.

A score is then computed by multiplying both scores together. The resulting score is high when both the descriptors and colours match, and low when one of them does not. If both measures are uncertain if a cut has happened (both scores are medium), it is not counted as such, due to the multiplication still resulting in a medium value.

We apply a Savitzky-Golay filter[20] to this score to smooth the signal and remove outliers.

Finally, we use positive step detection[13] to detect and compute the results, grouping each consecutive positive step as a single transition, and concluding they are separate transitions if they contain negative steps in between.

The algorithm and annotation data are both frame-specific, we could therefore establish a measure of how many frames are correctly predicted. However, when manually annotating, it is hard to determine what the start and endpoints of transitions are. Therefore, frame specific data on gradual transitions would achieve poor results on a frame-by-frame basis, even though the transition itself was found. Information in games is also not displayed exclusively on the first or last frame, being a few frames off will therefore not influence any information such sections might contain. For this reason, we choose to measure whether sections are correctly identified while allowing some degree of disagreement with the ground truth. Specifically, a transition is found if any of its frames are within a two-frame window of the ground-truth.

This means that we attribute detected sections to specific transitions in the ground-truth even if sections do not overlap. It also means that if the algorithm finds the same section twice or finds only one section when in fact there are two or more, we record an error in our results.

The results in table 2 show lower performance than expected. This is because of how the results are gathered. We observe in the data that the transitions correlate closely with the low values of the algorithm. The problem stems from positive step detection. This step detection acts as a threshold value to determine when a frame is significantly different. For example, a screen that does not move starts at a high value, but as we add more frames to the step detection, the value becomes lower and lower. The longer the frame does not move, and thus the lower the positive step detection is, the more sensitive the detection becomes to transitions. We observe that cuts and transitions happen frequently in our data and due to background-movement, lightning strikes and other visual effects the step detection fails to stabilize sufficiently to detect the next transition.

Unlike the original algorithm, we can make assumptions about the video

data we are given. All transitions are predefined and limited in number. We modified the algorithm to act on a threshold, rather than step detection and conclude that any frame with a score of 0.2 is a transition, without considering the general characteristics or trends of the video. Our results for this are shown in 3 and show significant improvements.

We explore if the complexity of this algorithm is required for good results. Our current method takes approximately 4 hours to run on 14 videos. We established another set of results made by adjusting the algorithm to use only one of its measures, in the hope we can simplify the algorithm further and reduce processing time. The first feature is the colour histogram [21]. This is the most basic feature we use and is fast to compute. We retained the characteristics of the SBD algorithm, but use a threshold of .4 to obtain our results. The second feature is SURF [17]. SURF descriptors are widely used in image recognition techniques and could be advanced enough to achieve good results on their own, without relying on colour histograms as the second metric. We use a threshold of 0.5 for our SURF descriptor as this produces the best results.

5.4 Results

Due to the smoothing of the sg-filter in the algorithm and the fluctuating nature of frames due to lightning, it is hard to determine precisely the length of any section. Therefore, we did not categorize the type of transition from the algorithm. However, this categorization is done for the ground-truth. This means the algorithm might conclude a "cut" transition is 2 frames in length, but this is still counted as a successful detection. All transitions in the ground-truth that last longer than one frame are classified as gradual transitions.

The video can be found by appending the video link to
"https://www.youtube.com/watch?v=".
For example "https://www.youtube.com/watch?v=2pxqR50zuwU"

5.4.1 Method: SBD

The following table shows the results of the algorithm as described in the SBD paper [13].

Video link	Instant	Gradual	Missed Instant	Missed Gradual	Error
2pxqR50zuwU	7	5	0	3	1
3aW86Mu1NLY	3	0	0	0	2
4mBmrOUbdn4	4	1	0	1	3
5Ycj-PL5iXw	47	2	0	0	4
8CEc13avHZo	8	2	0	1	2
azERn9omeqA	4	0	0	0	1
FiLluNqxU5o	7	1	0	0	3
golINEQ2P1g	0	0	0	0	1
HqgJQ752BwY	12	8	2	3	3
ieFDOlXy7hU	9	6	0	1	6
KHrB85JeiAQ	0	0	0	0	1
niCofFBdKYQ	11	11	3	0	0
tbIwrX-KwRw	0	5	0	1	1
Xjspn70c9rQ	4	1	0	1	1

Table 2: Prediction results based on SBD Positive step detection

Table 2 shows the results by running the algorithm from [13]. The algorithm finds 158/174(90%) of the total transitions correctly but also makes quite some errors, putting its total accuracy at 75%. We observe that the missed transitions are often because the positive step detection fails to recognise sections. Image 11 shows the positive step detection from the algorithm, the original score and the ground-truth. It shows the fluctuations that happen in the positive step signal which cause incorrect results. The original score however closely matches the ground-truth, which led us to believe that a simple threshold on the combined score would lead to better results. The results for this threshold approach can be seen in table 3.

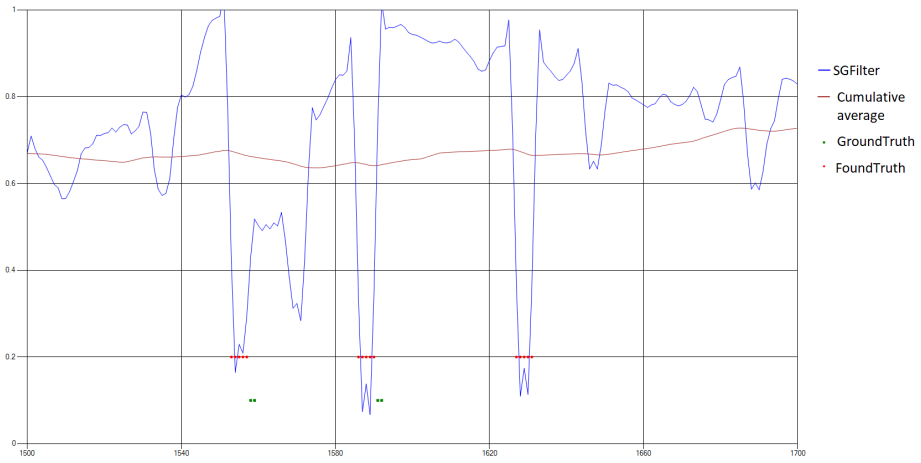

Figure 11: The positive step function, the ground-truth, and the combined score of SURF + histogram after the sg-filter

5.4.2 Method: Threshold

The following table shows our results of the SBD-algorithm after we have modified the positive step detection to a threshold function.

Video Link	Instant	Gradual	Missed Instant	Missed Gradual	Error
2pxqR50zuwU	7	5	0	0	1
3aW86Mu1NLY	3	0	0	0	1
4mBmrOUbdn4	2	3	0	0	0
5Ycj-PL5iXw	47	2	0	0	3
8CEc13avHZo	8	2	0	0	2
azERn9omeqA	4	0	0	0	0
FiLluNqxU5o	7	1	0	0	2
golINEQ2P1g	0	0	0	0	0
HqgJQ752BwY	9	7	0	0	6
ieFDOlXy7hU	9	6	0	0	5
KHrB85JeiAQ	0	0	0	0	0
niCofFBdKYQ	11	11	0	0	0
tbIwrX-KwRw	0	5	0	0	0
Xjspn70c9rQ	4	1	0	0	0

Table 3: Prediction results based on SBD score without the positive step detection (Threshold = 0.2)

The results for this method are shown in table 3 and have no missed sections, finding 174/174(100%) transitions, with a total accuracy of 87%. We do see that false positives are still detected. We observe that these false positives stem from

a lack of features in the loading screen, which causes large fluctuations in scores by fading text in and out.

Next, we observe how the individual measures contribute to the detection of these sections. We hope that by further reducing the complexity of this algorithm we might produce more accurate results.

The following two tables 4 and 5 show the results the algorithm obtains when using exclusively the histogram score and the SURF score respectively.

5.4.3 Method: Histogram

The following table contains our findings when using exclusively the color histogram score from the algorithm.

Video Link	Instant	Gradual	Missed Instant	Missed Gradual	Error
2pxqR50zuwU	7	5	2	0	1
3aW86Mu1NLY	3	0	0	0	1
4mBmrOUbdn4	4	1	2	0	0
5Ycj-PL5iXw	47	2	0	0	2
8CEc13avHZo	8	2	0	0	2
azERn9omeqA	4	0	0	0	0
FiLluNqxU5o	7	1	1	0	2
golINEQ2P1g	0	0	0	0	0
HqgJQ752BwY	12	8	2	0	2
ieFDOlXy7hU	9	6	1	0	5
KHrB85JeiAQ	0	0	0	0	0
niCofFBdKYQ	11	11	0	1	0
tbIwrX-KwRw	0	5	0	0	0
Xjspn70c9rQ	4	1	0	0	0

Table 4: Results based on histogram data (Threshold = 0.4)

We notice worse performance in table 4 then we see in table 3, finding 166/174(95%) of all transitions with a total accuracy of 85%. Indicating that the SURF score and the histogram score complement each other to allow for more accurate results. The performance of the histogram score alone is still better than the original SBD algorithm including the positive step detection, indicating that the histogram score is already a good method of detecting state changes.

5.4.4 Method SURF

The following table contains our findings when using exclusively the SURF score from the algorithm.

Video Link	Instant	Gradual	Missed Instant	Missed Gradual	Error
2pxqR50zuwU	7	5	0	0	1
3aW86Mu1NLY	3	0	0	0	0
4mBmrOUbdn4	4	1	0	0	0
5Ycj-PL5iXw	47	2	0	0	0
8CEc13avHZo	8	2	0	0	2
azERn9omeqA	4	0	0	0	0
FiLluNqxU5o	7	1	0	0	0
golINEQ2P1g	0	0	0	0	0
HqgJQ752BwY	12	8	1	4	2
ieFDOlXy7hU	9	6	0	0	1
KHrB85JeiAQ	0	0	0	0	0
niCofFBdKYQ	11	11	2	4	0
tbIwrX-KwRw	0	5	0	5	0
Xjspn70c9rQ	4	1	0	0	0

Table 5: Prediction results based on SURF data (Threshold=0.5)

Table 5 shows 158/174(91%) of total transitions found with an accuracy of 86%, which is still not an improvement over using both histogram and SURF data. The SURF score and histogram score performs similar in total, indicating that both are good individual measures. But the main difference lies in its ability to detect gradual transitions. We notice that the histogram score finds more transitions overall, both correct and incorrect (189 vs 180). We also observe that the histogram score is more accurate at predicting gradual transitions, while the SURF score is better at predicting instant transitions. We believe this is because the game contains many text elements that change drastically across instant transitions, while gradual transitions often fade to a solid colour. This gradually makes the SURF features disappear, without causing a huge spike in changes, whereas the colour can shift quite drastically on a frame-by-frame basis during such a transition.

5.5 Discussion

The results in table 2 from the SBD algorithm shows that the original algorithm from the paper has some issues finding instant and gradual transition because of its positive step detection. This is likely due to the short duration of certain segments and videos. The positive step detection is a function that detects changes larger than average, assuming that most of the scene is stable when no transition occurs. Some videos contain many transitions in comparison to normal frames, and many sudden changes. This causes the average to shift up, and therefore miss certain transitions. The errors are mostly caused by text-overlays, cut-scenes and mid-video pauses for an explanation, causing the step detection to adapt to lower values, only to find non-transitions afterwards.

The results in table 3 show that after removing the positive step detection

the algorithm does not miss any transitions. This makes it useful as no manual effort is required to scan the video file for potentially missed cases. It appears all the false positives are detected in one particular state transition; the loading screen with flashing text. Fortunately, this screen does not appear very often and does not provide any interesting data. A filter based on the supplied transitions would therefore not miss any information the game conveys, even if it cuts the loading state into more parts than it actually contains. Furthermore, these results highlight the advantage games have over movies for containing a set of predetermined transitions. We can therefore achieve more accurate results using simple thresholds because we can make assumptions about how and when transitions might occur.

Table 4 and 5 give an indication of how the different measures cooperate to find the transitions. We observe that most of the false positives and negatives do not overlap, showcasing that they complement each other well. Furthermore, we observe that the SURF metric contains fewer false positives, which might suggest it can be weighted more when attempting to remove false positives.

6 Conclusions

We show how the manual labour required for answering questions regarding footage can be reduced by applying algorithms and game knowledge. The pilot study shows us that while answering questions directly is difficult to do automatically, we can significantly reduce the labour required by analysing simpler features, such as state transitions.

We show a method of pre-processing YouTube videos using the structure of games as finite state machines to filter videos for interesting sections. Questions from the pilot study were clearly defined in the "Night" game state sometimes even specific menus within that state. Rather than analysing the entire video, our approach simplifies the problem to consider a small section of the video and achieves good accuracy.

The absence of a ground-truth required us to construct it by hand as without large enough sample sizes, modern learning algorithms do not tend to perform well. Our approach provides a way to construct such a ground-truth with a fraction of the labour required. This data can then be used either directly or used by other algorithms for training.

All games are essentially finite state machines. Hence, the methods described in this thesis could be applicable to other 2D games, as nearly all of them have menus, options, and various game states that are easily identified by humans and separated by transitions. These states are easily recognized by humans, allowing them to easily identify what game state is of interest to them. 3D games might prove more challenging using this method, due to large and abrupt changes on-screen that can be caused by rotating the camera in an environment, without changing the state.

For future research, we recommend improving our methods used during the pilot study. Once large data sets are available, we also recommend using learn-

ing algorithms to replace our methods from the pilot study as such algorithms are more suited to analyse complex questions. We also recommend further improving on our current method. One might be able to automatically group the separate "cuts" of videos, allowing for users to quickly browse all parts of the video of the "main menu" in order to further reduce the manual labour required. One could also analyse how well this algorithm translates to further genres/game-types and could be improved upon to allow all types of games to be separated by state, including 3D games.

7 References

References

[1] Paul Darvasi. *How digital games can support peace education and conflict resolution.* Working paper, November 2016, https://www.gcedclearinghouse.org/sites/default/files/resources/170025eng.pdf

[2] de Smale, Stephanie *Ludic Memory Networks : Following Translations and Circulations of War Memory in Digital Popular Culture.* University Utrecht, 2019-09-27 https://www.narcis.nl/publication/RecordID/oai%3Adspace.library.uu.nl%3A1874%2F384896/uquery/stephanie%20de%20smale/id/2/Language/NL

[3] B. Arend, S. Heuser, V. Maquil, H. Afkari, P. Sunnen *'BEING A SPACE MINING CREW': HOW PARTICIPANTS JOINTLY DISCOVER THEIR COMPLEMENTARY RESOURCES WHILE ENGAGING INTO A SERIOUS GAME AT AN INTERACTIVE TABLETOP (ITT).* Conference: 12th International Conference on Education and New Learning Technologies https://orbilu.uni.lu/bitstream/10993/43955/1/edulearn20_%27being%20a%20space%20mining%20crew%27.pdf

[4] Philip Hammond, Holger Pötzsch *Memory, Militarism and the Subject of Play.* ISBN: 9781501351150, 12-12-2019 https://www.bloomsbury.com/uk/war-games-9781501351150/

[5] Jesse Fox, Michael Gilbert, Wai Yen Tang *Player experiences in a massively multiplayer online game: A diary study of performance, motivation, and social interaction.* First Published April 11, 2018 Research Article https://journals.sagepub.com/doi/abs/10.1177/1461444818767102

[6] Markus Mühling, Ralph Ewerth, Thilo Stadelmann, Bernd Freisleben, Rene Weber and Klaus Mathiak *Semantic Video Analysis for Psychological Research on Violence in Computer Games.* Conference: Proceedings of the 6th ACM International Conference on Image and Video Retrieval, CIVR 2007, Amsterdam, The Netherlands, July 9-11, 2007 https://www.researchgate.net/publication/221368622_Semantic_video_analysis_for_psychological_research_on_violence_in_computer_games

[7] Varuni Wickramasinghe, Katherine M White, Daniel Johnson *Predictors of Players' Decisions to Help Others in Video Games*. Cyberpsychology, Behavior, and Social NetworkingVol. 23, No. 4. https://pubmed.ncbi.nlm.nih.gov/32031868/

[8] S.A.A. den Broeder *Automatic semantic analysis of gameplay videos of 'This War of Mine'*. Master thesis, March 2019 http://dspace.library.uu.nl/handle/1874/379280

[9] Jacqueline Schuldt née Krebs *Moral Dilemmas in Serious Games*. ICAICTE At: Sanya, Hainan, ChinaVolume: 2013 https://www.researchgate.net/profile/Jacqueline_Schuldt_Nee_Krebs/publication/261947749_Moral_Dilemmas_in_Serious_Games/links/0f317535fc258a4923000000/Moral-Dilemmas-in-Serious-Games.pdf

[10] Cassinelli James. *This Emotion of Mine: a diary study on affective learning in videogaming environments*. Master Thesis, December 2019 http://essay.utwente.nl/80248/1/Cassinelli_MA_BMS.pdf

[11] Gareth Schott *That Dragon, Cancer: Contemplating life and death in a medium that has frequently trivialized both*. DiGRA International Conference (Vol. 14, pp. 1–10) https://researchcommons.waikato.ac.nz/bitstream/handle/10289/11325/30_DIGRA2017_FP_Schott_Dragon_Cancer1.pdf?sequence=2&isAllowed=y

[12] Christopher Moser and Xiaowen Fang *Narrative Control and Player Experience in Role Playing Games: Decision Points and Branching Narrative Feedback* . Conference: International Conference on Human-Computer Interaction https://www.researchgate.net/publication/300588610_Narrative_Control_and_Player_Experience_in_Role_Playing_Games_Decision_Points_and_Branching_Narrative_Feedback

[13] Sawitchaya Tippaya, Suchada Sitjongsataporn, Tele Tan, Masood Mehmood Khan, Kosin Chamnongthai. *Multi-Modal Visual Features-Based Video Shot Boundary Detection*. IEEE Access (Volume: 5), 21 June 2017 https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7954599

[14] Wei-Ta Chu, Yung-Chieh Chou *Event Detection and Highlight Detection of Broadcasted Game Videos*. 2015 ACM. ISBN 978-1-4503-3747-2/15/10 https://www.cs.ccu.edu.tw/~wtchu/papers/2015HCMC-chu.pdf

[15] D.G. Lowe *Object recognition from local scale-invariant features*. Computer vision, 1999. The proceedings of the seventh IEEE international conference on volume 2, pages 1150-1157. https://www.cs.ubc.ca/~lowe/papers/iccv99.pdf

[16] R. A. Sharma, V. Gandhi, V. Chari, and C. Jawahar. *Automatic analysis of broadcast football videos using contextual priors*. Signal, Image and Video Processing, 11(1):171-178, 2017.

https://www.researchgate.net/publication/304002665_Automatic_analysis_of_broadcast_football_videos_using_contextual_priors

[17] H. Bay, T. Tuytelaars, and L. Van Gool. *Surf: Speeded up robust features*. European conference on computer vision, pages 404-417. https://www.researchgate.net/publication/225761164_SURF_Speeded_up_robust_features

[18] OpenCV image recognition library https://opencv.org/

[19] Hellinger distance algorithm https://en.wikipedia.org/wiki/Hellinger_distance

[20] Savitzky-Golayfilter (sg-filter) https://nl.wikipedia.org/wiki/Savitzky-Golayfilter

[21] Color Histogram https://en.wikipedia.org/wiki/Color_histogram

[22] Dayi Lin, Cor-Paul Bezemer, Ahmed E. Hassan *Identifying gameplay videos that exhibit bugs in computer games*. Empirical Software Engineering volume 24, pages4006–4033(2019) https://link.springer.com/article/10.1007/s10664-019-09733-6

[23] Distinctive Image Features from Scale-Invariant Keypoints https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf

[24] Stephanie de Smale, Martijn J. L. Kors, Alyea M. Sandovar. *The Case of This War of Mine: A Production Studies Perspective on Moral Game Design*. Research Article, August 2017 https://journals.sagepub.com/doi/pdf/10.1177/1555412017725996

[25] The game: This war of mine. https://store.steampowered.com/app/282070/This_War_of_Mine/

[26] Markus Mühling, Ralph Ewerth, Thilo Stadelmann, Bernd Freisleben, Rene Weber, Klaus Mathiak *Semantic Video Analysis for Psychological Research on Violence in Computer Games*. Proceedings of the 6th ACM International Conference on Image and Video Retrieval, CIVR 2007, Amsterdam, The Netherlands, July 9-11, 2007 https://www.researchgate.net/publication/221368622_Semantic_video_analysis_for_psychological_research_on_violence_in_computer_games/link/5c70fd78299bf1268d1e30aa/download

[27] Jonghwan Mun, Paul Hongsuck Seo, Ilchae Jung, Bohyung Han *MarioQA: Answering Questions by Watching Gameplay Videos*. 2017 IEEE International Conference on Computer Vision (ICCV), 22-29 Oct. 2017 https://openaccess.thecvf.com/content_ICCV_2017/papers/Mun_MarioQA_Answering_Questions_ICCV_2017_paper.pdf

[28] Alfredo Nantes, Ross Brown and Frederic Maire *A Framework for the Semi-Automatic Testing of Video Games* . AIIDE 2008 https://www.aaai.org/Papers/AIIDE/2008/AIIDE08-033.pdf

[29] Mila Bujić, Mikko Salminen, Joseph Macey, Juho Hamari *"Empathy machine": how virtual reality affects human rights attitudes*. ISSN: 1066-2243, 30 June 2020 https://www.emerald.com/insight/content/doi/10.1108/INTR-07-2019-0306/full/pdf?title=empathy-machine-how-virtual-reality-affects-human-rights-attitudes