

Balancing automation and user control in a home video editing system

Citation for published version (APA):

Campanella, M. (2009). *Balancing automation and user control in a home video editing system*. [Phd Thesis 2 (Research NOT TU/e / Graduation TU/e), Industrial Engineering and Innovation Sciences]. Technische Universiteit Eindhoven. <https://doi.org/10.6100/IR652921>

DOI:

[10.6100/IR652921](https://doi.org/10.6100/IR652921)

Document status and date:

Published: 01/01/2009

Document Version:

Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy

If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

Balancing Automation and User Control
in a Home Video Editing System

The work described in this thesis was carried out at the Philips Research Laboratories in Eindhoven, The Netherlands. It was financially supported by a Marie Curie Early Stage Training grant (MEST-CT-2004-8201), and it was carried out under the auspices of the J. F. Schouten School for User-System Interaction Research.

© Philips Electronics N.V. 2009

All rights are reserved. Reproduction in whole or in part is prohibited without the written consent of the copyright owner.

ISBN: 978-90-74445-88-7

Cover desing by Henny Herps, Philips CIS Visuals.

Balancing Automation and User Control

in a Home Video Editing System

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Technische Universiteit Eindhoven, op gezag van de rector magnificus, prof.dr.ir C.J. van Duijn, voor een commissie aangewezen door het College voor Promoties in het openbaar te verdedigen op woensdag 7 oktober 2009 om 16.00 uur

door

Marco Campanella

geboren te Brescia, Italië

Dit proefschrift is goedgekeurd door de promotor:

prof.Dr. A.G. Kohlrausch

Copromotoren:

dr. M. Barbieri

en

dr. J. Weda

Contents

1	Introduction	1
1.1	Context and focus of the thesis	3
1.2	Thesis outline	15
1.3	Thesis contribution	15
2	Literature on home video editing	17
2.1	Automatic home video editing systems	17
2.2	Semi-automatic home video editing systems	21
2.3	User behaviors and needs related to home video editing	27
3	Algorithms for automatic home video editing	37
3.1	User requirements	39
3.2	Problem formulation	41
3.3	Solution overview	45
3.4	Low-level feature extraction	47
3.5	Home video segmentation	48
3.6	Automatic home video editing	56
3.7	Zoom detection	64
4	Edit While Watching I	77
4.1	<i>Edit While Watching</i> system	78
4.2	Use test on <i>Edit While Watching</i> : objectives	84
4.3	Test design	85
4.4	Results	88
4.5	Discussion	91
5	Edit While Watching II	93
5.1	Mockups of new interaction paradigms	93
5.2	The second version of <i>Edit While Watching</i>	97
5.3	Use test on <i>Edit While Watching</i> version 2: objectives	106
5.4	Test design	106
5.5	Results	109

5.6	Discussion	116
6	Evaluating the role of intelligent chaptering and the level of user control in video editing	119
6.1	Introduction	119
6.2	Experiment 1: Effect of intelligent chaptering on navigation tasks	123
6.3	Design of Experiment 1	125
6.4	Results of Experiment 1	128
6.5	Discussion of Experiment 1	131
6.6	Experiment 2: Balancing automation and control in editing functions	131
6.7	Design of Experiment 2	135
6.8	Results of Experiment 2	141
6.9	Discussion of Experiment 2	153
7	Conclusions and suggestions for future work	157
7.1	Summary of experimental findings	157
7.2	Some reflections on the experimental methods	160
7.3	Design guidelines	161
7.4	Suggestions for future work	163
	Bibliography	167
A	Questionnaire about use of home videos	173
B	Material for the user study on Edit While Watching	181
C	Material for the user study on Edit While Watching 2	185
D	Material used for the experiments described in Chapter 6	189
	D.1 Material used for experiment about timestamp-based scene overview	189
	D.2 Material used for experiment about semi-automatic editing functions	193
	Summary	199
	Acknowledgments	203
	Curriculum vitae	205

1

Introduction

Since the beginning of history, human beings have invented artifacts and methodologies to expand their cognitive faculties. Devices and methods were created to make us able to store accurate information and share it with others, to reason more precisely and systematically, to learn and develop knowledge more efficiently and effectively, etc. Examples of inventions that extend our cognitive faculties are reading and writing, art and music, logic and arithmetic, encyclopedias and textbooks, science and engineering.

Pictures and videos were originally invented to expand two human psychological functions: perception and memory [Chalfen, 1987] [Norman, 1993]. Since 1825, when photographs were invented, they were seen as very efficient in copying the reality, and in being faithful and lasting representations of the true appearance of persons and things. In 1900, the first photo camera for the mass market, the Kodak Brownie, was launched. Also, in 1932 Kodak introduced a new video tape format (8 mm) that allowed cheap movie films to be sold, enabling average families to buy the necessary equipment for capturing home movies. After the introduction of these mass products, amateur users started capturing and watching more and more pictures and home videos. The acknowledged function of home imagery consisted in having external memories to help people in remembering. However, it became more and more evident that visual documents have also other functions in the context of private and family use. They can be seen as symbols that different people,

or the same people in different contexts, interpret in different ways. When people see their visual memories, they also reinterpret and reinvent their past [Chalfen, 1987]. A man can look at a picture of himself in his childhood with his mother, remembering that moment in time in different ways according to his age or to his experiences. A person can give different meanings to the same picture, depending on whether he/she is alone or with a group of friends. Generally children and parents interpret home imagery quite differently. When home visual documents are seen in groups, usually they are accompanied by a vocal explanation of the author that stimulates more dialogue, more reinterpretation and addition of meanings, or even extra actions like fetching other collections of home imagery (“Ah, why don’t we also watch the time our child made the first steps?”) or capturing more pictures and videos. Visual memories are also seen as good investments that will provide benefit later in time when they will be watched back, bringing enjoyment and recalling tastes, moods, fun and good times. Home imagery is also used for socialization: children are usually introduced to relatives that live far away via pictures. Home imagery leverages also cultural membership: children can witness and learn the parents’ model of life and values by watching their pictures.

Recent technological advances have made pictures and home videos extremely popular. In 1983, Sony launched the first camcorder on the market. Nowadays, 15 million camcorders are sold per year in the world. In conjunction, digital photo cameras with video capturing capability, and mobile phones that capture photo and videos, have been developed and are increasingly sold till today¹. Because of the new video capturing technologies, today the amount of amateur video streams available online is about the same as the amount of online commercial videos. The famous website YouTube is composed for 97% of home videos. Currently thirteen hours of videos are uploaded to YouTube every minute².

The way home imagery is handled and shared made some authors think that visual documents can be seen as symbols that enable a new kind of visual communication [Chalfen, 1987] [Frohlich et al., 2002] [Campanella and Hoonhout, 2008]. People manipulate and share their visual documents to communicate to others a version of their experiences, sometimes this version is a reinterpretation of the original event that people develop according to their actual identity and context. This is one of the reasons that explain why people *edit their videos* and change them according to the meaning they give to that memory, selecting the parts of the video

¹In 2008, more than 116 million digital photo cameras were sold, 15% more than in 2007 (<http://www.cipa.jp/english/data/dizital.html>, May 2009). References for the historical facts mentioned in the introduction are found in the following websites: <http://en.wikipedia.org>, <http://www.tropinature.com/photohist>, <http://inventors.about.com/od/tstartinventions/a/Television.htm>.

²Data on YouTube: <http://www.webtvwire.com>, May 2009.

deemed as relevant to that meaning and cutting away the details that the machine has captured but that are seen as unnecessary.

Video editing technology, however, became affordable and usable by amateurs only relatively late, and it is still problematic. Professional video editing was practiced since the origins of the movie industry, to manipulate the movies' films. Initially, video editing consisted mostly in physically cutting and pasting pieces of movie films. This activity has later been called *linear video editing*. In the seventies, after the diffusion of cheap video recorders, the first systems for *nonlinear video editing* were produced. These systems usually consisted of a PC connected to a bank of video recorders. The PC interface allowed the user to perform *random access* to the video: any frame in the video could be accessed with the same ease as any other. The user could also perform cuts, the system would save these cuts in an editing decisions list (EDL) and apply them altogether at the end of the editing. At the end of the eighties, computers were sufficiently advanced to allow the creation of fully digital nonlinear video editing systems. This is when amateur users began to regularly edit videos. Before that home video editing was done very rarely because of the technological complexity. Nowadays, many home video editing systems are available for personal computers; they are discussed in Section 1.1.2.

Home videos are more and more frequently filmed, watched and shared. However, the technology for editing and manipulating home videos that is currently available on the market is still perceived as time-consuming and difficult, while tools for editing pictures or text are seen as much easier. One of the authors of home videos interviewed during our research work complained:

“I would like a simple way to edit the recordings I make. The editing facilities . . . are extensive but also rather complicated for a person who has not the time / patience and in particular the technical ability to try and figure it all out.”

Making home video editing faster and easier would greatly enhance the possibilities that this new kind of visual communication can offer. This thesis focuses on developing better technology for home video editing.

1.1 Context and focus of the thesis

To define and motivate our research, we first clarify how we use the term “home video” in the thesis. We also explain what people typically do with home videos, and which problems are met most frequently in home video editing.

1.1.1 Definition and characteristic of “home videos”

In the context of this thesis, home videos are audiovisual documents captured by amateur users, or users who do not apply professional methodologies (preproduction, production, postproduction) to capture their videos. With home video, we mean the data composed of a sequence of video frames meant to be displayed one after the other and the associated audio track, regardless of the many media and formats in which a home video can be stored. Home videos are captured with digital camcorders, digital photo cameras, mobile phones or similar devices.

Home videos are captured mainly for documenting people’s lives. This consists in documenting events that are deemed to be special or important, such as holidays, parties, life of newborn children, weddings, etc. The main reasons for capturing home videos are to keep memories of someone’s life and to share someone’s experiences with third parties, such as family and friends [Campanella and Hoonhout, 2008]. Less frequent reasons for capturing home videos are artistic, instructional, or even professional purposes (to register an experiment in a video, for example). In this thesis, we focus on home videos captured with the purpose of documenting people’s lives.

Depending on the capturing device, home videos are recorded in a variety of diverse video formats. Nowadays, nearly all these formats are digital, therefore home videos can be uploaded on a PC as video files. Still, these formats differ greatly among each other. Some formats, for example the DV-AVI, have a signal quality higher than DVD quality. DV-AVI videos are stored on tape cassettes called “MiniDV”. These videos have the same image resolution and frame rate as DVDs, but a higher bitrate than DVDs, since they are less compressed. Many camcorders available on the market store videos on MiniDV cassettes. Other camcorders store videos on hard disks, flash memories or even directly on DVDs. Generally, these camcorders use a video format similar to MPEG-2, the format of DVD videos. Nowadays, also many digital photo cameras and mobile phones have a video capturing functionality. Usually, these devices store videos in formats similar to MPEG-4, and the image resolution and signal quality can be lower or much lower than DVD video. For example, the author’s mobile phone captures video with an image size of 176x144 pixels at a rate of 15 frames per second.

The presence of many different encoding formats for home videos causes problems in using them. In fact, each video format is designed for particular usages and is problematic for other usages. For example, it is very difficult to watch on a TV a home video from a mobile phone, because of its low resolution; it is also difficult to quickly send over the Internet a home video captured with a MiniDV camcorder³.

³The DV-AVI format is in fact quite bulky: 10 minutes of DV-AVI video occupy 2 GB on the hard disk.

In most cases, the encoding format of the home video, determined by the capturing device used, determines strongly the lifecycle of the video [Kirk et al., 2007].

Home videos have technical and perceptual characteristics that make them considerably different from pictures. A photo captures an instant of an event that is evolving, and it is meant to be displayed alone. A video is captured to record also the movement present in the event; to achieve this many frames are employed, and none of them is meant to be watched statically, as a separate picture. Video frames are displayed sequentially with a fast rate, to create in the human eye the impression of continuous movement. The perception of motion requires a display rate of at least 15 frames per second, a satisfactory level of smoothness is reached by employing 25 frames per second. This implies that videos are composed of very long sequences of frames, where each frame is minimally different from the previous and the next. For example, ten seconds of video typically contain 250 similar frames, many more than an average photo album. The sheer number of frames in a video sequence is generally difficult to manage. The situation is complicated by the fact that, since consecutive video frames generally contain a lot of redundant information, compression is applied to save storage space by encoding each frame as a function of a certain number of frames in the temporal neighborhood. This makes accessibility more difficult, since decompressing one frame often requires decoding several neighboring ones.

Many systems for video management try to leverage accessibility to the video by implementing overviews of relevant video clips, generally represented with static frames. However, a single image is not sufficient to give the complete idea of the dynamic event contained in a video clip. Since an overview of static images does not fulfill the “what you see is what you get” principle, users often have to play the clip to properly get all its message. The single frame cannot be seen as the basic element of a movie. The sequence of frames is the basic element of a video, since it conveys in the mind of the viewer a dynamic event with temporal information that is totally lost if video frames are considered separately.

Independently of the format, the typical structure of home videos is the camera take. A *camera take* is a continuous portion of home video that begins when the user presses the “record” button on the capturing device and ends when the user presses the “stop recording” button. Home videos are always partitioned in camera takes. Nowadays, all capturing devices record *timestamp* information for each camera take. A timestamp contains the date and time at which the camera take was captured. Most video editing tools adopt the camera take as the smallest element in which a home video is structured. In the home videos that document people’s experiences, it has been observed that the order of camera takes follows a progression of successive events that are ordered in time. For example, it happens rarely that during holidays people return to the same place and capture the same event

twice. Each event is instead localized in time [Gatica-Perez et al., 2003].

1.1.2 Typical usages of home videos

After having clarified some of the features that are peculiar to home videos, we schematize how people use their video memories. Figure 1.1 displays the workflow of home videos, representing the most common activities that characterize the lifecycle of home videos. These activities are ordered from top to bottom, start-

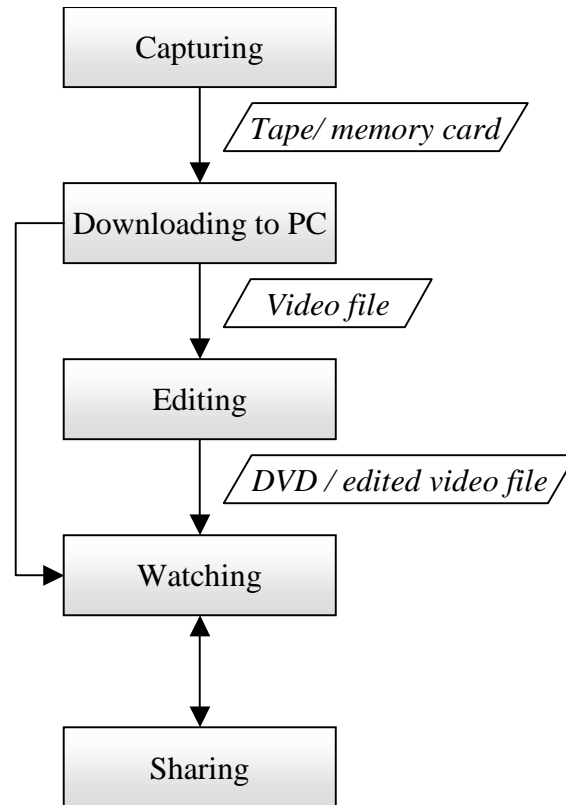


Figure 1.1. Workflow of home videos. The rectangular boxes represent activities performed with the videos, the romboidal boxes represent different storage media for home videos.

ing with the creation of home videos (capturing) and finishing with the end use (watching and sharing). The activities that compose the typical workflow of home videos will be now described, one by one. Data will be collected from the literature about user behaviors with home videos. We also investigated user behaviors and needs related to home videos. We collected insights via an Internet-based survey answered by 181 participants living in the Netherlands and in Italy. The detailed

results of this research are published in [Campanella and Hoonhout, 2008]. Some of our results will be used to draw a picture of the different behavioral aspects related to home videos.

Capturing

Nowadays, the most popular devices used to capture home videos are the digital photo camera (used by 59% of the respondents of our survey), the digital camcorder (57% of the respondents) and the mobile phone (28%). Digital camcorders are used more often by users older than 36 years, mobile phones are preferred more often by younger users. Home videos are used to capture many different kinds of events. In our survey, people indicated as object of their home videos events such as holidays (65% of the respondents), parties or events with friends (58%), family life (46%), artistic events such as concerts (19%), sport events (17%), weddings (13%) and nature scenes (4%). Requirements that users frequently ask from their capturing devices are reliability (if there is an important event to capture, the device should boot and record reliably, since most events are un-repeatable) and video quality (important events should be captured in a good video) [Kirk et al., 2007].

Downloading to PC

Generally, after capture home videos are downloaded to a PC, for storage or for further processing like creation of a DVD, sharing over the Internet, editing. The details of the downloading operations can change a lot for different video formats and capturing devices. Therefore, users must know and remember how to download videos from each of the capturing devices they possess. Videos coming from MiniDV camcorders are downloadable only via FireWire cables, and the downloading time is equal to the duration of the video. While downloading a video via FireWire, the performances of the PC decrease considerably. Therefore, the user must plan a long time slot to reserve for the downloading operation. Other capturing devices with more compressed video formats allow downloading to PC via USB cables or via memory cards. This is generally easier, since the amount of information to transfer is much smaller. However, short videos from photo cameras or mobile phones are stored on a PC less often than videos from camcorders.

Editing

After a home video is downloaded to the PC, before being watched or shared, a home video may also be edited. Nowadays, home video editing is always performed on a PC. It is done quite frequently: 64% of the respondents of our survey edit at least some of their videos. Videos from camcorders are edited more frequently than videos from digital still cameras or from mobile phones. We define home video editing as the set of the following categories of operations:

- *Selection of video material.* This category includes operations for selecting portions of video (called clips) deemed as important, and for discarding others that the user does not want to keep or to share. The selected portions can be kept in the same order as they were at capture time, or operations may be used for shuffling the clips' order.
- *Enrichment of the video.* This category includes operations for adding extra content or extra features to a home videos. Typical enrichment operations consist in adding video effects or video transitions, adding a music track, inserting text, shapes, pictures, or other video clips, adding an initial menu with DVD chapters, etc.
- *Correction of defects.* Operations that fall in this category are shakiness correction, blur correction, correction of encoding artifacts, or color balancing of video frames with low luminance or low contrast.
- *Video transcoding.* We have seen that home videos can be stored in a variety of different video formats. In the lifecycle of a home video, it is frequently necessary to transcode the video from one format to another. For example, a user that has captured a video with a MiniDV camcorder may transcode the video to MPEG-2 format to create a DVD, and to QuickTime format to have a version of it with reduced framerate that can be sent over the Internet. Some transcoding operations can take hours or dozens of hours.

As we will see, this thesis focuses mainly on the first category of editing operations, although some enrichment operations (second category) are also considered in our work, such as the insertion of video effects and music. Through our survey, we have elicited that the first two categories include the operations that are most frequently used by home video editors [Campanella and Hoonhout, 2008].

Below, a review of the most common video editing applications available on the market is presented. Video editing tools are classified in four categories: professional PC programs, low end PC programs, fully automatic PC tools, and web services.

- **Professional PC programs for video editing.** These programs are well established on the market since around 2000, even if the first version of Adobe Premiere was issued already in 1991. These PC applications are very technical and detailed, and they are meant mainly for semi-professional use. Examples are Adobe Premiere, Pinnacle Studio (2000), Final Cut Pro (1999). All of them allow random access to the video material by browsing a timeline representation of the footage. The user can also browse a storyboard of the video. A storyboard is a panel containing key-frames, each camera take of the home video is represented with a key-frame inside the storyboard. The user plays the different camera takes

by browsing the key-frames in the storyboard. These tools usually offer a great number of editing options: the user can select portions of video with frame detail and enrich them with video transitions, video effects, photos, text, shapes, or even 3D objects and effects (Ulead Cool3D). In these programs, the user does almost all the editing operations manually. Very little automation is involved: typically this automation involves the separation of the raw video material into camera takes and the encoding of the video in DVD format or other formats. Since these tools are based on fine-grained editing operations, users need to be aware of several technical details about video documents (frames, shots, transitions, definition of color filters) and invest a lot of time and effort.

- **Low-end PC tools for video editing.** Other programs are less detailed and meant for consumers: Windows Movie Maker (2000), Apple iMovie (1999), Ulead Video Studio (2000). Although these tools feature simpler interfaces than professional PC programs, the user edits the video through visual elements similar to the ones of the professional tools. Editing is done via browsing frame-by-frame the video represented by a timeline, or by interacting with a storyboard representation of the video.

To make video editing easier for their customers, producers of video editing PC applications started to include functions in their programs that automatically edit the video. Pinnacle Studio introduced the “Smart movie” function in 2004, iMovie introduced “Magic iMovie” in 2005 and Windows Movie Maker came up with the option “autoMovie” in 2006. These functions are explored in the next paragraph.

- **Fully automatic PC programs for video editing.** These programs are more recent than the professional and the low-end ones and started to be on the market in 2001. With these tools, the user just uploads the unedited video in the system, sets some editing parameters like the style or the soundtrack and tells the system to start the automatic editing. The system then edits the video according to pre-defined rules and shows the result to the user. Usually, the calculation of the edited video is performed in a very short time. However, users have very little influence on the “hidden” editing operations. Examples of automatic video editing programs are Muvee autoProducer, ACD Video Magic, the function “smart movie” available in Pinnacle Studio, or the function “autoMovie” in Windows Movie Maker.

- **Web services for video editing.** Since 2006, several services to upload, share and edit videos have also appeared on the web. In these websites, the user can play his/her videos and edit them via simple interfaces with a timeline representation and basic cut functionalities. These web services offer less editing options than PC programs: it is not possible to enrich the video with transitions, special effects or text. However, it is easy to mix pictures and videos from other

users into the user's content. Examples of these services are Jumpcut⁴, Motionbox, Photobucket, Stashspace, RiffTrax Cuts.

In Section 1.1.3 the user satisfaction with the home video editing programs, and the related problems, are presented.

Watching

Once a home video is captured, it is watched mostly around two to five times. People watch home videos on the PC, on the TV or on the capturing device. The PC is used to watch both relatively long home videos coming from digital camcorders, and relatively short videos coming from photo cameras or mobile phones. The TV, on the other hand, is used almost only for long videos coming from a camcorder.

Watching home videos appears to be done mostly in groups. Only 21% of the respondents of our survey watch home videos mostly alone, the rest watches home videos mostly with family or friends. The choice of which home video to watch depends mainly on the type of event recorded in the video. Secondary factors that determine which home video to watch are requests from other people (children and other family members, friends, acquaintances), the level of interest (whether the video is amusing, funny or boring), the accessibility (whether it is easily accessible on a DVD or video file), and other less important factors.

Sharing

Nowadays, many people share their home videos with others, by using Internet, DVDs, USB drives or external harddisks, etc. People share videos quite frequently: 66% of the respondents of our survey distribute copies of at least some of their personal videos. Home videos are shared mainly by giving away DVDs (62% of the respondents of our survey) and secondly via USB storage devices (40%), Internet-based means like e-mails (21%), blogs (7%), YouTube (21%) or other Internet possibilities (for example other websites for sharing videos, 25%). Owners of long videos use DVDs much more frequently than owners of short videos (80% and 49% respectively). Other sharing media, including YouTube and other Internet based-services, are used equally often by owners of long and of short videos.

Recently, it appeared that people capture more and more often short videos, coming from digital still cameras or mobile phones. The sales of these devices with video capturing capability are in fact increasing, while the sales of digital camcorders are steady. The increasing use of short videos is probably the cause of the birth of some recent web tools especially designed for sharing short videos:

⁴Due to prioritization efforts at Yahoo!, the Jumpcut website is announced to be closed on June 15, 2009.

websites for video sharing and the phenomenon of video blogging.

- **Websites for video sharing.** YouTube is perhaps the best known example of a video sharing website. These websites are very recent (they first appeared in 2005) and are growing very quickly in size and number. Recent numbers indicate that more than 300 such sites exist⁵. Most of them host all-purpose videos, however there are video sharing websites specialized for particular topics: college jokes, religious songs, online teaching, motor biking, airplanes, poker playing, etc. Some of them are addressed to a specific region or culture: Germany, Japan, India, arab culture, etc. Most viewed of them, according to the Alexa web traffic ranking⁶, are Yahoo! video, Google Video, YouTube, Video.qq (a Chinese website), Flickr, DailyMotion.

- **Video blogging.** Video blogs are like normal blogs in which people post also home videos; blog entries and messages become therefore multimedial. Video blogging, or vlogging, saw an increase of popularity beginning in 2005. In 2005 and 2006 two conferences of videobloggers were held. Radio stations and television stations are using video blogging as a way to help interact more with listeners and viewers. Also, services to publish online videos almost at capture time are appearing. Qik⁷ is a service to share live on the Internet videos being captured with the mobile phone.

1.1.3 Typical problems of home video editing

In the former sections, a picture of the common behaviors users have with home videos has been drawn. Figure 1.1 shows the typical usages of a home video, organized in a workflow. This thesis focuses on home video editing, which is perceived as the most problematic of the activities involving home videos.

The most frequently reported problem with the modern video editing tools for consumers is that they are way too time-consuming [Lienhart, 1999] [Davis, 2003] [Hua et al., 2004]. While it is reasonably easy to browse a collection of pictures and discard the uninteresting ones, doing the same on a home video proceeds slower and is more complicated. Working with videos has some intrinsic difficulties, caused by the temporal aspect. In Section 1.1.1 some characteristics of videos have been mentioned that make accessibility problematic: the large number of video frames, the slow accessibility to the single frames due to video compression, and the inability of overviews composed by static images to represent dynamic events and temporal information. Furthermore, because of its temporal dimension a video needs to be played back in order to be edited, and effects of

⁵<http://www.reelseo.com/list-video-sharing-websites/>, March 2009.

⁶http://www.alexa.com/site/help/traffic_learn_more, March 2009.

⁷<http://qik.com/>, April 2009

editing operations also need playback to be acknowledged.

To cut an unwanted clip, for example, one has first to find the desired portion of video. With the modern PC tools for video editing, usually the user searches a given moment in a video by browsing a storyboard of frames representing the camera takes, and then by browsing a timeline representation of each single camera take. This involves zooming operations on the timeline, together with fast-forward and fast-rewind operations. Once the desired segment of video is found, the user performs a cut by determining the cut points at frame detail, usually by dragging a slider or by repeatedly pressing “next frame” or “previous frame” buttons. This requires a lot of time and awareness of technical details such as the frame-rate. Once the cut points are decided, the user activates the cut function and needs to play again the edited portion of video to realize the effect of the cut.

Another problem with modern systems for video editing is that they are all PC-based, and perceived as difficult by the users who do not have a high PC expertise. The results from our Internet-based survey have shown that users with low PC literacy edit video significantly less frequently than users with high PC literacy [Campanella and Hoonhout, 2008]. Users with little experience with PCs see video editing tools as very difficult, not only the professional ones, but also the supposedly more user-friendly ones like Windows Movie Maker or Apple iMovie. Furthermore, the fully automatic editing programs are seen as inflexible, since they do not allow authors of home videos to select the video portions to keep or to discard. Home video users perceive the possibility of performing this selection as very important [Girgensohn et al., 2001].

Davis [2003] has argued that the problems of video editing systems stem mainly from the fact that home video editing is too much modeled on professional video editing. The production of professional videos is a complicated and time-consuming process. Usually, there is a pre-production, in which a storyline and a capturing plan (or storyboard) of the video are defined. Then there is the production, where repeatable scenes are captured following rules of media aesthetics. Finally, there is the post-production, where the video is edited by professionals, usually with very advanced techniques. The interfaces for home video editing, based on detailed timeline representations of a video, or on storyboard representations of the camera takes, are very close to the tools professionals use for the post-production. Usually, home video amateurs do not have the time or the economical/cognitive resources to learn and use such detailed methodologies. As a result, video editing applications are perceived as too time-consuming and often too difficult.

1.1.4 Focus of the thesis and research approach

The goal of this project is to find an easy-to-use, effective and efficient solution for home video editing, to enable creators of home videos to quickly and effortlessly author their audiovisual memories. To develop our solution for home video editing, we adopted two main guidelines. We designed a *semi-automatic* tool, and we adopted a *user-centered* approach.

By designing a semi-automatic video editing system, we aim at overcoming the difficulty and the time cost that fully manual tools bring, and at avoiding the low flexibility of the fully automatic editing programs or functionalities. We hypothesize that automatic video content analysis can be employed to make home video editing quicker and easier for the user. For example, a home video can be structured in small video segments that the user can exploit as “atoms” of the editing process, without needing to browse the individual frames anymore. At a higher level of abstraction, a home video can be structured into scenes, where each scene concerns only one event and place. The scene structure has a clear semantic meaning and may help the user in browsing and searching through the video. Automatic video analysis can also be employed to determine which portions of video have higher quality than others and therefore are most suitable for being included in the edited video, helping the user in the process of content selection.

While believing that automation can help home video editing, we also think that a fully automatic editing system will never provide users with the flexibility that they desire. A system that applies a fixed set of rules to calculate how to edit a video will generally not match the subjective preferences a user may have for the video. Most home video authors prefer to select themselves the video clips that they find most valuable, and not to leave this choice completely to the system [Girgensohn et al., 2001]. Furthermore, generally users want to personalize their videos according to their own creativity, enriching them with music, video effects, etc. Therefore, a video editing system should automate the most technical and time-consuming tasks but should also offer to the user interaction means so that the video can be edited according to the user’s creativity and subjective wishes. To maximize the user satisfaction with the system and the objective effectiveness and efficiency, our video editing solution has been designed following a user-centered approach. Two iterations of implementation and user testing have been carried out.

Our home video authoring system, *Edit While Watching*, is designed to provide user-friendly home video editing in the living room. Its interaction uses only a TV set and a remote control instead of a PC monitor, keyboard and mouse. This brings simplicity to video editing, because also users without much experience with PCs can edit their videos. *Edit While Watching* allows a user to modify and enrich his/her video while watching it, seeing in real time the effects of the editing

operations. The two iterations of system implementation and use testing on *Edit While Watching* gave us insights on the user needs and requirements and on which interface elements best support easy and fast video editing. Our research shows that it is possible to provide semi-automatic, user-friendly video editing on a TV via a remote control, to move video editing to a more social environment like a living room and to open up new possibilities of interaction with the users' videos.

Edit While Watching aims at finding a convenient balance between automation and user control. Nowadays, many technologies employ automation to help the user in performing actions. Different technologies set the balance between automation and user control in different ways. For example, a pacemaker regulates the activity of one's heart in a fully automatic way, without any intervention of the user. As second example, the automatic pilot of an airplane also works fully automatically, but the pilot has the option to turn it on or off. As third example, the guidance system of a jet is controlled by the pilot's yoke; however, an automatic system makes sure that the jet follows the direction indicated by the yoke. The pilot has control on the jet's direction, at the same time automation is used to keep the jet on the trajectory.

These three examples clarify that the degree of automation that is applied to a system can vary across a continuum of levels. In designing *Edit While Watching*, we investigated how to balance the degree of automation and user control in several aspects of the system. First of all, the user does not edit directly the video; instead, *Edit While Watching* calculates a first version of the edited video by exploiting content analysis and media aesthetic rules [Zettl, 1999]. Successively, the user refines the automatically edited video via *Edit While Watching*'s interface, based on TV and remote control. To edit the video, *Edit While Watching* provides various functionalities for inserting or removing particular clips. Some of these functionalities also exploit automation: for example, when the user just informs the systems of his or her interest for a certain clip, the system automatically adds more content to it. The selection of the content to be included is again based on content analysis, quality criteria and aesthetic criteria.

Besides evaluating the overall usability of *Edit While Watching*, we studied the balance between automation and user control in different parts of the system's functionalities. An experiment was run to assess how much value automation can add to the overview that the user inspects to browse the video. After introducing an algorithm that structures the overview according to the timestamp information of the video clips, we were able to measure an improvement in the efficiency users have in browsing and searching tasks. Another experiment was aimed at evaluating different combinations of automation and user intervention functions for selecting interesting video clips, to maximize the user satisfaction and the objective editing performance.

1.2 Thesis outline

The rest of this thesis is structured as follows. In Chapter 2 the relevant literature on home video editing is presented and discussed, together with the most common tools for home video editing and sharing available on the market. Assumptions, significant contributions and possible improvements on the state of the art are pointed out. Important information from our investigation of user behaviors and needs related to home videos is also reviewed. Also, user needs for home video editing are distilled to motivate the rest of the work.

In Chapter 3 the algorithmic part of our solution for home video editing, *Edit While Watching*, is described. *Edit While Watching* is a semi-automatic system for video editing where the user can upload his or her video, get a preliminary version of it automatically edited by the system, and refine it via a user interface. This chapter explains how the video is automatically analyzed and edited. Also, a novel algorithm for detecting zoom sequences in home videos is described and tested. In Chapter 4 the user interface of *Edit While Watching* is presented. With our system, the user can edit videos using only a TV set and a remote control. To assess to what extent *Edit While Watching* answers the user needs elicited in Chapter 2, a user study is carried out. The results are reported and discussed in this chapter.

In Chapter 5 a second version of *Edit While Watching* is developed to overcome the pitfalls of the first version. The second version is evaluated by inviting users to edit their personal videos with it, according to their wishes. The results of the evaluation are reported and discussed. In Chapter 6 the balance between automation and user control in *Edit While Watching* is questioned and investigated. Different levels of automation and user control for browsing and editing functionalities are compared in order to find the balance between system intervention and user intervention that optimizes the objective performances and the user satisfaction. Finally, in Chapter 7 the conclusions of this work are presented and suggestions for further developments are discussed.

1.3 Thesis contribution

The main contributions of this thesis can be summarized as follows.

- Insights in user behavior and needs related to home videos, based on a survey including 181 users.
- A novel algorithm for fast and precise zoom detection in home video sequences.
- *Edit While Watching*: a semi-automatic tool for home video editing that works with a TV set and few keys on a remote control. With *Edit While*

Watching, even users not experienced in video editing can author their videos easily and in a short time.

- Optimization of the balance between automation and user control in functionalities for selecting or discarding video clips. This balance is tuned to optimize the objective performances and the user satisfaction.
- For the first time, an evaluation methodology that combines the users subjective evaluation and objective measurements of performances and usage patterns has been applied to assess the usability of home video editing systems.

2

Literature on home video editing

In this chapter, the most relevant studies on home video editing present in the literature are reviewed and compared.

2.1 Automatic home video editing systems

In this section the literature related to automatic video editing is reviewed.

One of the earliest studies on automatic home video editing has been written by Lienhart [1999]. The author has made some assumptions on home video usage:

- Home video collections easily sum up to many hours.
- Most of them are never touched or watched again.
- Raw video footage is unedited and lacks in quality, therefore is not appealing to watch.
- Video editing is too time consuming.
- Video editing is rarely done.
- Users would like different versions of their videos for different people.

These assumptions have been used as motivation for creating a system for automatically producing video abstracts. This system organizes the home video clips in clusters, according to the clips' timestamps. For example, clips filmed in the

same hour fall in the same cluster. In the case of analog material, the system detects the timestamp by using character recognition on the lower-right corner of the video frames. The system then selects video material in order to uniformly represent each cluster. Furthermore, the clips are shortened to their most interesting parts. The procedure for selecting these interesting parts is motivated by the observation that during important events or actions, the sound is mostly clearly audible over a longer period of time than is the case with less important content. The system presented in [Lienhart, 1999] has been tested with 10 users; 5 video abstracts have been created and evaluated by the users by means of Likert-scale questions. These questions concerned the suitability of a video abstracting system for real scenarios, and the quality and usefulness of the video abstract. The users' answers have been positive. The main outcome of this test is that:

“There is a tremendous demand for abstracting home video material automatically.” [Lienhart 1999, p. 40]

Lienhart [1999] has looked into how to automatically edit home videos. However, he has presented only *assumptions* about user needs for home video editing, without studying these needs by looking at real users.

After this study, others have tried to develop intelligent algorithms to address the problem of automatic editing of home videos. All the studies on automatic home video editing have been motivated by the assumptions presented by Lienhart [1999], with small variations.

Aner-Wolf and Wolf [2002] have attempted to summarize home videos of weddings by matching photo collections taken at the same event. They assume that the photo album of the wedding can be used as a storyboard for the wedding video: the pictures, in fact, will contain the most relevant moments of the wedding. Besides synchronizing the pictures with the video shots, they also automatically add video transitions and video effects to deliver an aesthetically pleasant final video. Aner-Wolf and Wolf [2002] do not report any evaluative study of their summarization algorithm.

Yip et al. [2003] have proposed a system for automatic home video editing, assuming that owners of home videos do not have the time or the editing skills to edit their videos. Their application is called “Automatic Video Editor” and edits a home video automatically into a condensed and supposedly interesting mini-movie. The system works by picking out segments from the source videos whose color properties contrast with each other the most, in order to try to maximize the informativeness of the automatic summary. The systems disregards jerky segments and allows to tune the amount of faces, close ups, indoor/outdoor scenes to be present in the edited video according to the user's interest. Special effects are aligned with the video to make it more appealing. These effects are: slow motion, fast motion,

and picture in picture (PiP). Yip et al. [2003] describe only a qualitative evaluation of the Automatic Video Editor, performed with “a dozen of Carnegie-Mellon students”. The only reported result of this evaluation is that the criteria for editing the video should vary according to the audience and the source material. People appearing in the video can be more interested in clips about themselves, while people not in the video may be more interested in clips with nature or scenery.

Oami et al. [2004] have made an attempt to investigate with users which video content is considered interesting and which not, in order to build a better algorithm for automatic home video summarization. They have looked at modeling the “user interest” for semantic elements in videos, such as objects (main entities in the video), scenes (compositions of objects) and events (happenings with special meaning). They have run an experiment with 12 users and a set of 50 video clips. The users were invited to evaluate the importance of objects, of scenes and of events for each clip. The importance was judged by choosing between low, medium and high. The correlation between the importance judgment of different users was 0.70 on average. This result is interesting, however it is not clear why the user interest for a certain clip should depend only on objects, scenes and events and not on the presence of people, what is said, the story, the recipient of the video, the mood of the user, or the subjective value of an event for a specific user. In fact, the authors also report that other variables influence the user interest for a video: owners of the home videos evaluated the elements in the clips in a significantly different way compared to people not acquainted with the videos.

Hua et al. [2004] have described a system for automatic home video editing called AVE (Automatic Video Editing). This system analyzes the raw video footage trying to detect the viewer’s *attention* to a certain video clip. The user’s attention is calculated as a function of low-level features such as object motion, camera motion, faces, audio and speech. After the analysis, the AVE system composes a video summary out of the raw footage, aiming to ensure that:

1. clips with the highest “attention level” are included in the summary,
2. clips are well aligned with music tempo,
3. selected clips uniformly represent all parts of the raw footage.

The problem of creating a video summary that satisfies as much as possible these three criteria is formalized as a search problem and solved with a suitable algorithm. The summaries produced by AVE have been evaluated by 10 users who majored in arts. The evaluation has shown that the AVE-produced videos were judged as better than randomly produced ones and as good as videos manually edited by a non-professional user.

In another paper [Hua and Zhang, 2004], the authors have shown how to automatically combine aesthetically appealing video effects to particular portions of

video content. This is one of the few attempts in automating not only the selection of content, but the enrichment of the video with effects or other elements.

In yet another study [Mei et al., 2005], the authors have shown how to automatically evaluate the *quality* of specific portions of home video. To assess the quality of home videos, the following factors are checked for: shakiness and jerkiness of camera motion, low contrast of images, presence of blur, low luminance, bad orientation of the camera. By means of an evaluative study with 10 users, video summaries composed with clips having the best quality have been shown to be more pleasant and equally informative for the users compared with video summaries created considering the viewer's attention, according to Hua et al. [2004].

The AVE system contains clever algorithms for automatically selecting the best portions of a home video and enriching them with music and video effects. However, the authors of AVE have based their work on the same assumptions about user needs contained in [Lienhart, 1999], without verifying these assumptions. Therefore, it is not clear to what extent AVE helps the users in their real needs.

Achanta et al. [2006] have proposed a system to automatically convey or accentuate emotions in home videos. They assume that owners of home videos would like to convey more effectively a certain intent or emotional message in their personal movies. They present a system called "IntentMaker". This system applies film grammar rules [Zettl, 1999] to modify a number of low-level features in a video with the aim of conveying one of four emotions: cheer, serenity, gloom and excitement. At the request of the user, IntentMaker modifies the luminance, the saturation, the motion, the rhythm, the music and the video transitions of a home video, in order to convey a certain emotion. A user evaluation of the videos edited by IntentMaker has been run with ten users. The users were asked to watch four videos and to indicate which one of the four emotions was mainly conveyed by each video. Three of the four videos were judged to convey mainly the desired intent, although the paper does not say whether the results are statistically significant. The emotion was more effectively conveyed with the addition of proper music. Conveying emotions that are opposite to the emotion originally present in the clip led to poorer results. The work of Achanta et al. [2006] is interesting, because users understand easily emotions like cheer or gloom and editing videos by manipulating emotions can be easier and more satisfactory than directly editing the videos. However, we still do not know how important it is for the users to be able to convey emotions with their home videos, and whether other needs have higher priority.

Wang et al. [2007] have approached the problem of automatic home video editing by implementing an "information-aware" selection of content. They assume that home video users are interested in *who* is in the video and *where* the event in the video takes place. In other words, the users' interest for a specific video

scene can be evaluated according to these two elements of information: subjects and locations. The key to produce a good edited video was assumed to consist in automatically understanding the *who* and the *where* in the audiovisual document. The authors have built a system that automatically summarizes home videos according to these criteria, and have evaluated it with users. The videos produced by their algorithm were perceived as better than randomly edited videos and close to manually-edited videos. However, this evaluation has been conducted with a limited number of users (five). Also, they conclude that users are not only interested in subjects and locations of a video, but also in video quality and events experienced by the owners of the video. Finally, Wang et al. [2007] do not clarify the question about what the user needs for home video editing are. Table 2.1 summarizes the studies on automatic home video editing systems that have been reviewed in this section.

2.2 Semi-automatic home video editing systems

The *semi-automatic* approach to home video editing is an alternative to full automation. In the semi-automatic systems, most of the difficult and time-costly tasks of home video editing are automated. Typically these tasks include the detection of meaningful cut-points in the raw footage, the selection of interesting parts of the raw material, and the composition of an initial version of the edited video. The user is offered the possibility to change and refine the edited video by means of interfaces with suitable functions for selecting interesting clips, refining cut points, adding video effects, transitions and music. The idea behind semi-automatic tools is that a fully-automatic system for home video authoring will never meet the subjective wishes of users for an edited video; on the other hand, fully manual systems are definitely too complex and time-consuming. Therefore it is better to study a clever compromise between automation and user control. In this way, users can reach the result they desire without dealing with technical and time-consuming tasks. Here we survey the semi-automatic solutions published in the literature.

In [Girgensohn et al., 2000] the “Hitchcock” tool for semi-automatic authoring of home video is described. Hitchcock automates some technical and time-consuming tasks in home video editing, like structuring the raw video footage in small segments, selecting the segments that are suitable to be included in the edited video and choosing where to place the cut points of these suitable segments. Hitchcock composes an initial version of the edited video, successively the user has the freedom to refine the selection of the video segments, and to change the length of the edited video and of the single segments. Hitchcock works on a PC, and its interface represents the video segments as frames grouped in piles (Figure 2.1). The user can switch between time-based grouping and color similarity-based grouping.

Table 2.1. Summary of studies on automatic video editing.

Article reviewed	Main contributions
[Lienhart, 1999]	System for automatic video summarization, designed specifically for home videos. The assumptions on user needs for video editing are not verified.
[Aner-Wolf and Wolf, 2002]	Summarizing wedding videos by matching photos of the same event. No evaluative study performed.
[Yip et al., 2003]	Home video summaries based on maximizing informativeness and minimizing jerkiness. Summarizing criteria should be user-dependent.
[Oami et al., 2004]	Investigating how much users are interested in objects, scenes and events. Correlation among different users' interest is 70%.
[Hua et al., 2004]	Automatic video editing based on modeling the user's <i>attention</i> to clips. Automatically produced videos appear to be as pleasant as videos edited by a non-professional.
[Hua and Zhang, 2004]	How to automatically combine video effects to particular clips.
[Mei et al., 2005]	Automatic video editing by evaluating the <i>quality</i> of clips. Introduction of quality criteria specific for home videos.
[Achanta et al., 2006]	System to automatically convey emotions in home videos. Successful in three videos out of four.
[Wang et al., 2007]	Automatic selection of video content based on interesting people and places. Video summaries produced are almost as pleasant as manually edited videos.

Hitchcock's usability has been evaluated by means of a use test with 9 users [Girgensohn et al., 2001]. The system appeared to be quite usable; however, users were sometimes not satisfied with the fact that low-quality clips were automatically discarded: in some cases users were interested in clips with particular content, even if poor in quality. Also, users often wanted to have more control on the clip duration and on the cut points. The authors conclude that automation in video editing is useful, but it must be introduced in ways that still enable user control,

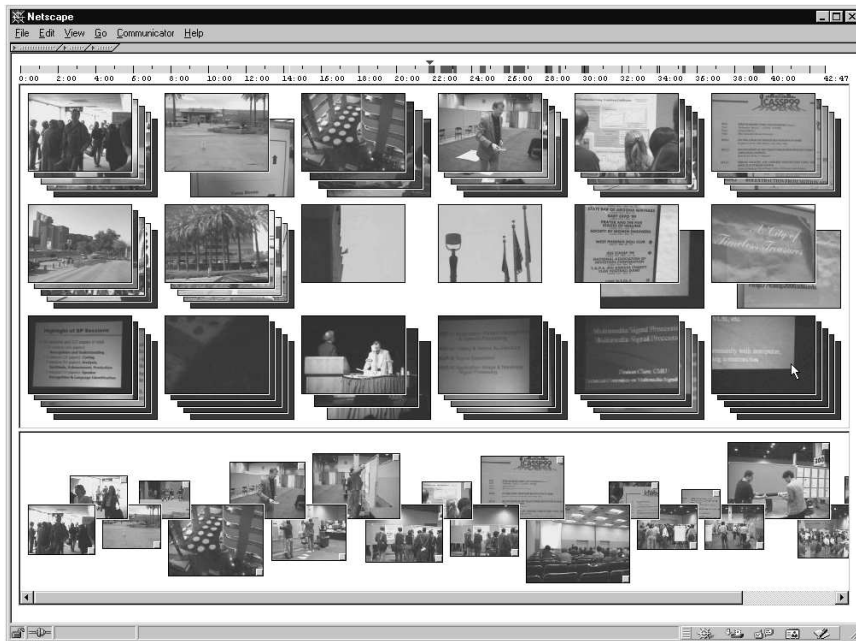


Figure 2.1. Hitchcock user interface [Girgensohn et al., 2001].

e.g. as default choice or suggested choice.

Lockerd and Mueller [2002] have proposed an interesting synergy between the user and the system, with the aim of producing an edited video which contains the moments that are most interesting for the user. They propose to collect information about the emotions and the arousal of the user at *capture* time, to be used to automatically produce a better edited video. To reach this goal, they have implemented LAFCam (Leveraging Affective Feedback Camcorder). LAFCam is shown in Figure 2.2.

LAFCam detects laughter of the camera operator, records skin conductivity, and films the camera operator's face. The recording of the operator's face adds another perspective that can be inserted in the edited video. The authors propose also a video editing application to edit the videos captured with LAFCam. The edited video can be produced automatically, by selecting all the video clips whose corresponding skin conductivity was above a threshold value. The user can adjust this threshold value via a slider in the interface. It is commonly assumed in literature that high values in skin conductivity correspond to highly aroused emotions, like joy, enthusiasm or fright. Lockerd and Mueller [2002] do not report any evaluation to assess to what extent the videos captured with LAFCam and edited accordingly satisfy the initial user requirement: to efficiently select the most inter-



Figure 2.2. The camcorder “LafCam” implements three affective input channels [Lockerd and Mueller, 2002].

esting clips in the unedited video footage.

Casares et al. [2002] have proposed the “Silver” system for semi-automatic home video editing. Silver exploits an audio-text transcriber to detect the words that are pronounced in the video. These words are shown aligned with the video in a visualization system composed of three timelines with different levels of detail (camera take, shot, frame detail). Silver helps users cutting the video material at convenient points, at shot boundaries or word boundaries. The user can also manipulate parts of the video stream by doing cut and paste operations on the text transcript of the same stream. An evaluative test on Silver has been done involving graduate students with a computer science background. The users were invited to perform a series of tasks on the same video material. Some aspects of the Silver interface were perceived as difficult to understand, such as the difference between the three timelines or the feedback after a delete operation. Also, the “smart help” in editing provided by Silver did not significantly improve the time to complete editing tasks. The idea of editing a video by cutting and pasting the words pronounced in it is interesting, however it is not clear how helpful it is for real home videos and for users without a computer science background.

Adams and Venkatesh [2005] have gone beyond the idea of automating only editing operations after the capture of a home video. They argue that a semi-automatic system for home video authoring should intervene already at pre-capture and at capture time, not only when the home video is already captured. They assume that home video amateurs do not possess the knowledge to shoot aesthetically

acceptable videos. This knowledge would help them in better conveying their experiences and in obtaining a more enjoyable result. Therefore, they developed a system that prepares a filmic/narrative structure for the user's video and gives *shot suggestions* to the user during video capturing, according to this structure. Successively, the system edits the so-obtained raw footage by injecting convenient media aesthetics and presents the final result to the user. These shot suggestions can be constructed by letting users choose between templates of events, as it happens in the IMCE system [Adams et al., 2005]. Alternatively, users can be asked simple questions about event, place and characters being shot; this is how the MediaTE system [Adams and Venkatesh, 2005] works. A use test on this latter system shows that movies obtained with the assistance of MediaTE are generally preferred to videos shot without aid. However, half of the users refused to execute most of the shot suggestions that were provided by the system. It is not specified why users did not want to attempt the shot suggestions. Users expressed the desire to employ MediaTE again, although some found the system to be suited for "lengthy projects with easily repeatable scenes". Therefore it is difficult to say to what extent the MediaTE system meets the user needs for home video editing.

Hua and Li [2006] have proposed the LazyCut system for semi-automatic home video editing. LazyCut is an extension of the AVE system for automatic video editing proposed in [Hua et al., 2004]. LazyCut is designed to address the fact that video editing is complex and time-consuming, and also to offer more flexibility than a fully automatic video editing system. To tackle these issues, LazyCut supports the user in composing an edited video out of his or her videos and photos. The editing uses *templates* of possible final videos. A template includes a rough storyline, an editing method and editing parameters like style and music. For example, a template can specify that a video should be composed by an initial chapter, a body part with three sub-chapters and a final chapter. Via a UI, the user drags and drops selected videos and photos in each chapter of the template storyline. Successively, the video is automatically edited with the same criteria used by the AVE system [Hua et al., 2004], with the additional constraint that the succession indicated by the user in the storyline should be preserved. A user evaluation of LazyCut has been carried out, inviting ten users to evaluate edited home videos produced randomly, manually, with AVE and with LazyCut. The results show that videos produced with LazyCut were preferred over the videos produced in other ways. Videos produced with AVE and those produced manually were equally well liked and were preferred over videos produced randomly. Unfortunately, the usability and the utility of LazyCut have not been evaluated.

Takeuchi and Sugimoto [2007] have proposed a system for semi-automatic video editing. This system tries to learn the user's subjective preferences for video content by analyzing the user's personal photo library. Takeuchi and Sugimoto

[2007] criticize the automatic approach to home video editing, arguing that automatic algorithms apply a fixed set of editing rules for all users, without taking into account personal preferences of each user. According to these authors, these preferences can be learned by the system by analyzing the personal photo collections that most users have on their PCs. The system proposed in [Takeuchi and Sugi-

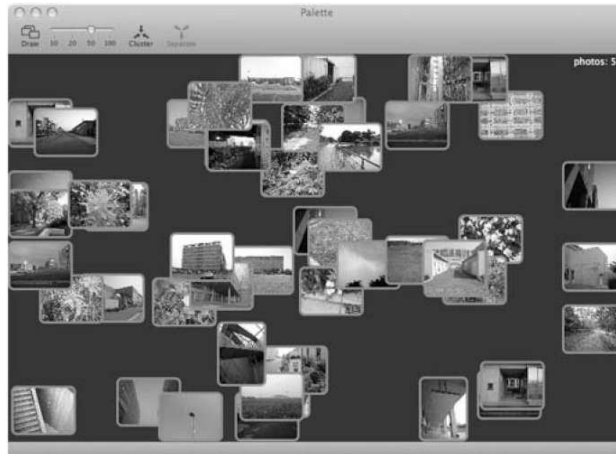


Figure 2.3. Interface for photo clustering [Takeuchi and Sugimoto, 2007].

moto, 2007] works as follows. At first, the user organizes his/her pictures into a small number of categories, eight for example. By using the interface shown in Figure 2.3, the user can drag a subset of his/her pictures into clusters and label these clusters. The categorization of the remaining pictures is performed automatically by the system. Successively, the user's video is segmented and each segment is matched to one of the categories labeled by the user. Each segment receives also an importance score; this score increases if the photo category associated with the segment is big and if the segment neatly belongs to it. Then the system produces an edited video by assembling the video segments with the highest importance score. The user can influence the importance of the photo categories he or she created, via the interface shown in Figure 2.4. This interface is interesting, because it allows the user to specify preferences for the edited video by means of intuitive semantic concepts like the labels of the picture categories. For example, if the user gives more importance to the photo category of "landscapes", the edited video will contain more video segments classified as landscapes.

The user evaluation of the system in [Takeuchi and Sugimoto, 2007] has been carried out among nine students of the University of Tokyo. The study was aimed at assessing how well the summarized video reflects each student's preferences. The users were invited to capture videos of the University's main campus, then

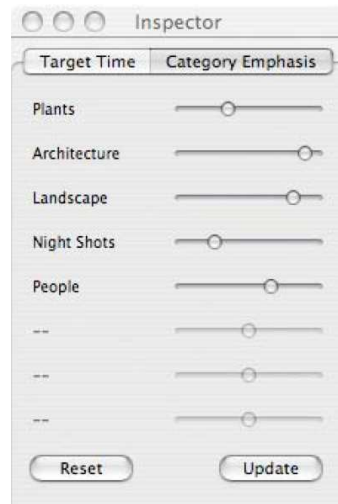


Figure 2.4. Interface for weighting the photo categories [Takeuchi and Sugimoto, 2007].

they were invited to say which points of the videos were most interesting to them (the “stress points”). Successively, each user categorized the *same* set of pictures and performed the summarization. Video summaries produced by the proposed system retained the stress points slightly better than randomly generated summaries (47.7% against 44.6%), but not enough to claim statistical significance.

Unfortunately, Takeuchi and Sugimoto [2007] have not evaluated their system in a more realistic scenario, for example by letting the users categorize *their* own photo collections. Table 2.2 summarizes the papers on semi-automatic home video editing systems that have been reviewed in this section.

2.3 User behaviors and needs related to home video editing

So far, we have presented an overview of automatic and semi-automatic systems for home video editing. These solutions contain intelligent algorithms and interaction paradigms; however, all these studies are motivated by *assumptions* about the user needs, without verifying what users wish to do with their audiovisual memories. At the time of writing, we are aware of only one study [Kirk et al., 2007] about how people use their home videos and what their needs and problems are. This study is reviewed in the next Section (2.3.1). As already mentioned in the introduction (Section 1.1.2), we also investigated the behavioral aspects concerning home videos, via an Internet-based survey. The results of the survey that are relevant to home video editing are presented in the Sections 2.3.2 and following.

Table 2.2. Summary of studies on semi-automatic video editing.

Article reviewed	Main contributions
[Girgensohn et al., 2000] [Girgensohn et al., 2001]	Video automatically summarized, then edited via a PC-based interface with color-based clip grouping. User evaluation shows good usability and need for more user control.
[Lockerd and Mueller, 2002]	Emotions and arousal of the user are collected at <i>capture</i> time to better edit the video. No user evaluation done.
[Casares et al., 2002]	Video editing PC interface with three timelines with increasing detail. Semi-automatic clip selection functions based on audio transcript.
[Adams and Venkatesh, 2005] [Adams et al., 2005]	The system gives shot suggestions at capture time, to build a nice storyline. Movies produced with shot suggestions are perceived as better than movies shot without aid.
[Hua and Li, 2006]	The user composes a video out of his/her clips and photos, by using <i>templates</i> of storylines. Videos created this way are preferred over automatically or randomly edited videos.
[Takeuchi and Sugimoto, 2007]	The system learns the user's subjective preferences for content from the user's photo collection.

2.3.1 Early study on usage of home videos

Kirk et al. [2007] point out a “technology push” in the studies on automatic and semi-automatic systems we have just reviewed. They assert that all the studies in the literature

“suffer from a common lack of concern with user behavior, what one might call “usefulness” (as against a more pedantic concern with whether users can use the software in question). All offer limited evaluation studies, designed to ratify that the system built can be used, or used after a period of initial training; not whether it is useful technology in terms of supporting users in accomplishing the kinds of tasks they wish to perform [Kirk et al. 2007, p. 62]”.

Kirk et al. [2007] have made a first step in looking into user behavior and needs for home videos. They have run an interview with 18 members of 12 families in their homes and a focus-group with 7 teenagers. The aim of their study was to collect information about the whole lifecycle of home videos: choosing the device to capture video, capturing, editing, and consuming.

The main finding of these interviews is that two different ways of working with home videos can be distinguished: the *lightweight* and the *heavyweight* videowork. In the lightweight videowork, users exploit mobile phones and digital still cameras to spontaneously shoot short clips, without pre-planning, to capture on-the-fly interesting events. These short clips are usually not edited, are shown on the capture devices or shared via Internet. On the other hand, in the heavyweight videowork, camcorders and digital still cameras are used to shoot long clips of important events, in a pre-planned way. These longer videos are edited more often, in order to convey the user's creativity and in order to obtain a tangible result, like a DVD.

Kirk et al. [2007] invalidate one of the assumptions accepted by most studies in the literature: home video collections easily increase and become unmanageable. The interviewed families, in fact, collected over several years generally 10-20 hours of raw video footage. Also, the assumption that users would like to convey a narrative or filmic structure to their audiovisual documents, stated in [Adams et al., 2005], is not confirmed by the structure of DVDs people usually edit for themselves. These DVDs are more intended to be easily browsable to support users in skipping across interesting moments while telling about the event to friends. Camcorder users consider the time of uploading the video to the PC as significantly hampering their videowork. Encoding the final result on a DVD is perceived as one of the most important requirements for a video editing system. Another finding of this study [Kirk et al. 2007, p. 67] is that users edit their videos in many diverse ways, depending on the video content, on the purpose of the video and on the user's creativity and personality. Therefore, a home video editing tool should satisfy a vast set of requirements, allowing users to do precise cuts, or to add effects, music, transitions and titles, or to quickly produce a musical summary from random clips of the raw footage.

Kirk et al. [2007] have presented useful insights about user behavior and needs for home video editing. Unfortunately, their study is based on only a small number of respondents. Moreover, no other independent study has been done to confirm or invalidate the results. Therefore, we wanted to obtain a more precise, more complete and statistically more relevant picture of the user behavior and needs, surveying a larger number of people.

2.3.2 Deeper investigation into behavioral aspects of home video editing

In [Campanella and Hoonhout, 2008] our research on the usage of home videos has been published. The objective of this research was to study how and why people capture, watch, share and edit home videos, and what the needs and wishes are users have related to their video memories. Additionally, we wanted to assess to what extent the assumptions proposed by Lienhart (p- 17) are true. To address these research questions, we created an Internet-based survey, which is reported in Appendix A. In the rest of this chapter, we review those results of our survey that are relevant to home video editing.

The questionnaire, an English and an Italian version, was distributed via bulletin boards and mailing lists at Universities and the Eindhoven High Tech Campus. Anyone who had at least a basic experience in shooting home videos with any device was invited to take part in the survey. The questionnaire was answered by 181 people, 67% of which were living in the Netherlands, the remaining 33% in Italy. 69% of the respondents were male. The age of the respondents was distributed as shown in Table 2.3.

Table 2.3. Age distribution of participants in the Internet-based survey.

Age	Percentage of participants
Younger than 20	4%
21-25 years old	34%
26-35 years old	34%
36-50 years old	18%
51 years old or older	9%

Given the distribution channels we employed, it is not surprising that the majority of the respondents is relatively young, and that many of the respondents indicated to have a background related to technical disciplines. Table 2.3 shows that 73% of the respondents are younger than 36 years. However, the number of respondents in our sample older than 36 years (49 respondents) still allowed us to obtain relevant insights in how home video use might change when people's lifestyle is changing due to getting older and, e.g., having families.

It is also possible that people who capture home videos are mostly younger than 36 years, and that we did not have many respondents older than 36 years simply because they did not match our selection criterion (having at least a basic experience in video capturing). We were not able to check whether home videos are captured more often by people younger than 36 years than by people older.

We asked the participants to rate their level of experience with PCs on a scale from one to nine (question 24 of Appendix A). We decided to divide the sample into two groups of participants: users of basic PC applications like Internet, email

and word processing (54 respondents), and users of more advanced PC applications, or IT professionals (107 respondents). The groups were created as follows: the first group was formed by all the respondents who placed their level of experience with PCs from one to four in the question 24; the second group included everyone who scored from six to nine. The 20 respondents that rated their PC experience with five in question 24 were not inserted in any of the two PC expertise groups.

The fact that our sample included more respondents with a high level of PC experience will probably be due to the technical background of many of our participants, among which the questionnaire was distributed. We have not been able to verify if our sample differs in this respect from the general population of home video users.

In order to determine whether particular participant characteristics result in different response patterns, we have split the respondents into different user groups and studied the results for these groups:

- Respondents older than 36 years (49 people).
- Respondents younger than 36 years (132 people).
- Respondents with low PC expertise (54 people).
- Respondents with high PC expertise (107 people).
- Respondents that capture mostly short (less than 15 min) videos (101 people).
- Respondents that capture mostly long (more than 15 min) videos (87 people).

Given the characteristics of the data set, a Chi-square test for independence was applied [Siegel and Castellan, 1988] to analyze the answer patterns of the subgroups as mentioned, with $\alpha = 0.05$.

2.3.3 Results relevant for home video editing

Our respondents edit their videos quite often. 64% of the respondents edit at least some of their videos. The frequency of editing home videos changes with the acquaintance users have with computers. Of those people with high computer expertise, 76% edit some of their videos, against 46% of users with low computer expertise (Figure 2.5). The difference in editing frequency among the two groups of PC expertise is statistically significant ($\chi^2 = 16.25$, $df = 3$, $p = 0.001$). Although our sample of participants may be biased towards people with high PC experience, we observe that even among users with low PC skills the editing frequency is surprisingly high, in contrast with what is commonly assumed in literature¹. The editing frequency changes also with the duration of home videos; 79% of owners of long

¹See page 17 and following for the assumptions made by Lienhart [1999] and others.

videos do some editing, against 49% of creators of short videos. The difference in editing frequency among users with different video duration is statistically significant ($\chi^2 = 17.83$, $df. = 3$, $p = 0.0004$). The editing frequency does not change significantly with the participants' age.

We asked people who never or rarely edit their videos whether there is a reason for their low editing activity. Only 19% of them said that home videos do not need editing. The others complained about the amount of time required for video editing (77%) or about its difficulty (40%).

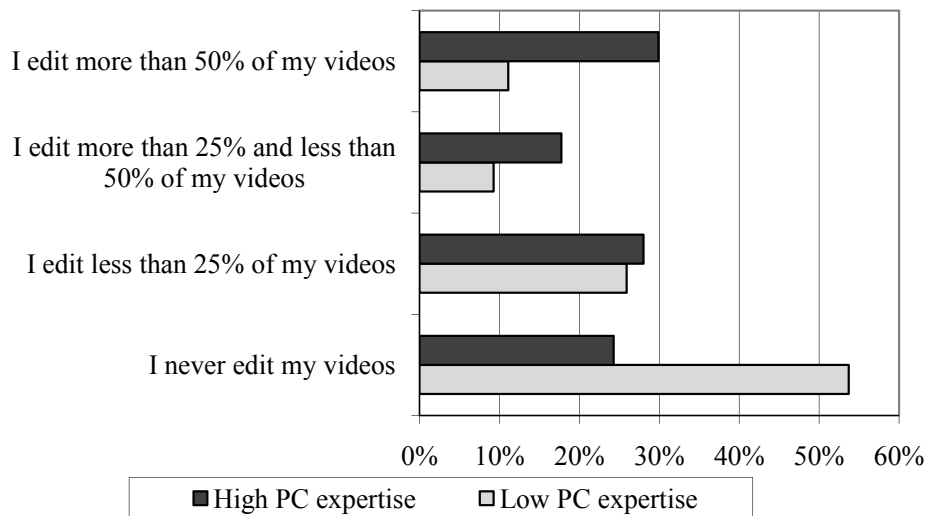


Figure 2.5. Distribution of editing frequency in the two groups with different PC expertise.

Our survey contained a number of questions that only had to be answered by the 64% of respondents with at least some experience in video editing (questions 17 - 20 in Appendix A). These questions concerned the motivations for doing video editing, the time needed, the tools used and their main limitations.

89% of respondents editing videos said that the main reason for editing is to remove uninteresting parts: the raw video is normally too long. 67% of video editors enrich their videos by adding music and video effects, to increase the attractiveness of the video. Other motivations are to add a personal touch to the video (41%), to have a nice video memory to keep (24%), to have different versions of a video to give to different people (15%). Owners of short videos plan significantly less often to keep videos as a memory (14% against 31% of owners of long videos, $\chi^2 = 4.52$, $df. = 1$, $p = 0.03$).

Many respondents who edit videos stated that video editing takes a lot of time: 40% of video editors need ten or more times longer for editing than the total duration of the raw material. We did not find any dependencies between time spent on editing and editing reasons. We asked also about the editing tools that were used most often. 66% of respondents used (semi-)professional tools like Adobe Premiere or Pinnacle Studio. Of respondents, 40% use low-end tools like Windows Movie Maker, Apple iMovie, Ulead Video Studio and similar programs, generally seen as a bit more user-friendly. Respondents using editing tools embedded on hard disk recorders or DVD recorders amounted to 10%.

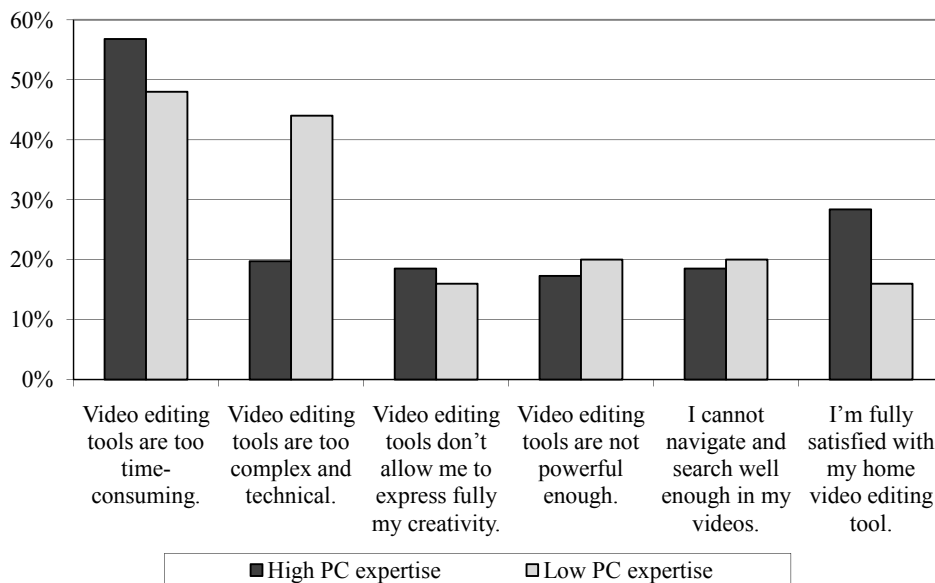


Figure 2.6. Frequencies of video editing problems against PC expertise.

Users who edit videos were asked what are the most evident limits of video editing tools. Figure 2.6 displays the answers and their frequency. Importantly, more than half of users editing videos observe that editing tools are too time-consuming; by far the main complaint. Among users who edit videos, 44% of users with low PC expertise find editing programs too complex and too technical, against 20% of users with high PC skills.

The respondents who edit videos were asked to make spontaneous remarks about unsatisfying aspects of the video editing technology. These remarks concerned the amount of time needed for “uploading, re-encoding and DVD creating” of the video, the limited memory and speed of PCs, the complexity of technical options related to codecs and video filters, and the complexity of the editing tools

that forces some users to re-learn them every time they use them again. Some respondents perceive video editing as a professional activity, while home videos are totally for leisure.

In our survey, participants were invited to express their interest for a number of possible wishes related to home video editing (question 21 of Appendix A). The respondents indicated a clear interest for “combining own videos with own pictures, friends’ content or content from the Internet”, “changing and manipulating a video while watching it”, and “editing videos on a TV”. The owners of long videos expressed significantly more interest for editing their audiovisual memories on a TV than the owners of short videos ($\chi^2 = 11.55$, $df. = 4$, $p = 0.03$).

The respondents of the questionnaire were also invited to freely write down their needs and wishes related to home video editing. Some ideas concerned the possibility of involving others in the editing process. For example, one participant remarked

“It would be nice if I can send them low quality vids, and let them edit it, and i can use the same edit points on my high-res vids and give them the final version.”

Other participants expressed the desire for more user-friendliness, for example by simplifying technical details of encoding operations or by having a wizard or a “coach” to do better video editing. Other wishes concerned better ways of organizing and annotating videos and of compiling one movie out of many small clips.

In the former sections of this chapter, we have seen that many systems for home video editing aim at (semi-)automatically removing the video segments with low video quality, under the assumption made by Lienhart [1999] and others that unedited video is not watched because of its low quality. In our survey, participants were asked on which aspects they choose the home videos to watch. Some respondents described in an optional text field of the questionnaire what influences the frequency of watching home videos. Among these respondents, 45% pointed out that they watch particular events more often than others; the type of event captured on a home video is of great importance when deciding whether to view the video. Another 23% said that some videos are watched more often because they are asked for by other people: children, friends, family members, acquaintances. Other factors that influence whether or not a home video will be watched more than once, are the level of interest (is the video amusing, funny, or boring), the accessibility (whether it is easily accessible on a DVD or in a video file), the duration, the quality, the time users have available. These factors were mentioned less often. Therefore, whether a video is watched often or not does not depend primarily on its quality.

2.3.4 Conclusions

The results from the Internet-based questionnaire clarify many behavioral aspects of home video editing. People author their home videos mainly to keep the clips deemed as interesting or useful, and to cut out the others. A second important reason for editing videos is to enrich them in diverse ways. The quality of the videos does not seem to be the primary concern of our respondents. They appear to be attached to their videos independently of the low video quality, some respondents even remarked that they avoid editing on purpose, in order to leave the home videos in the unedited state that they like. As Kirk et al. [2007] have argued, a final tangible result of the editing, like a DVD, is often seen as an important goal.

We may now look back at the assumptions on home video usage proposed by Lienhart [1999] (p. 17) and accepted by most of the literature. The assumption that home video editing is perceived as too time-consuming is confirmed. However, the assumption that video editing is rarely done is contradicted by our results. Home videos, in fact, are edited quite frequently and also shared quite often. Furthermore, the assumption that the raw footage is not appealing because of its low video quality is also contradicted by our results.

The main user needs clarified by our survey can be summarized in the following points:

1. Users want to shorten and summarize the raw video material.
2. Users want to convey in their videos their own creativity and wishes.
3. Users want video editing to be fast.
4. Users want a user-friendly and simple system.
5. Users want to edit videos on a TV (particularly people older than 36 years).
6. Users want to combine their videos with pictures and third parties' content.
7. Home videos are used mainly in social context and for communicative purposes.

Before concluding this survey and coming to the conclusions mentioned above, we had already developed the first version of a solution for home video editing. The algorithmic part of it is presented in the next chapter, while Chapter 4 presents the interactive part. Section 4.1.3 deals with how our solution answers the user needs elicited here.

3

Algorithms for automatic home video editing

In Chapter 2 several solutions for home video editing proposed in the literature have been analyzed. Furthermore, user needs related to video editing have been collected, through the review of the literature and via our own Internet-based survey. Before the results of the questionnaire were available, we had created a initial concept and a prototype of a system for home video editing. As explained in Section 1.1.4, we think that a system for home video editing should be semi-automatic, implementing a convenient balance between automation and user control. Therefore, our solution has an automatic part (an algorithm that edits the user's video) and an interactive part (an interface that allows the user to refine the automatically edited video). This chapter is dedicated to explain the automatic part, while Chapter 4 presents the interactive part. Figure 3.1 shows a flowchart of our system, explaining the relation between the automatic and the interactive part.

As said before, we assume that a home video editing application should not be fully automatic. In fact, the content of the optimally edited video depends heavily on the user's subjective preferences and wishes, on the semantics of the video and on its final recipient. Since the user defines how an optimally edited video should be composed, a fully automatic algorithm that applies a set of user-independent rules on objective properties of the video will never reach what is optimal for a

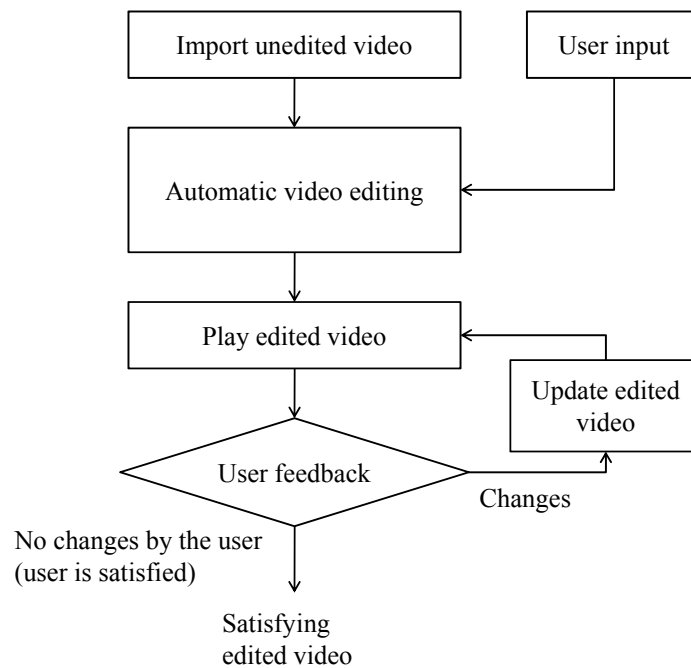


Figure 3.1. Flow diagram showing the relation between the automatic part and the interactive part of our system.

particular person. Nevertheless, we think that automation in home video editing is still useful. An automatic algorithm can effectively perform operations that would be very cumbersome for the user; for example discarding video segments with poor quality or selecting content with particular properties. Even more, an automatic algorithm can produce an edited video that matches some criteria suggested by the user and is therefore close to the optimally edited video.

Because of this, for our application we designed an algorithm for automatic video editing. The video that is produced by the algorithm is refined by the user via an interface. Figure 3.1 shows the relation between the automatic and the interactive components of our system. While the edited video is played, the user can modify it iteratively, until his/her subjective wishes are met.

The present chapter describes how the automatic video editing block in Figure 3.1 works. Many studies in the literature have already presented systems for automatic video editing. We do not claim that our algorithm performs better than these methods. Still, our algorithm is designed to provide the user with an edited version of his/her home video that matches the user's wishes at least in some aspects.

This chapter is structured as follows: the user requirements are first specified (Section 3.1), and the problem of automatic video editing is formally defined (Section 3.2). Then, our solution is described in detail (Sections 3.3, 3.4, 3.5, 3.6).

In order to do automatic home video editing, we employ detection of zoom sequences. To have a fast and precise zoom detection, we have devised a novel algorithm. A benchmark on 37 minutes of home video has shown that our zoom detection algorithm performs better than a state-of-the-art method. Section 3.7 explains the research done on zoom detection in detail.

3.1 User requirements

Before specifying the user requirements, we make some considerations on home videos, from a content analysis perspective. In Section 1.1.1 we have seen that home videos are always subdivided in *camera takes*. A camera take is a continuous portion of home video that begins when the user presses the “record” button on the capturing device and ends when the user presses the “stop recording” button. Nowadays, all capturing devices record *timestamp* information for each camera take. A timestamp contains the date and time at which the camera take was captured. Another observation is that in almost the totality of the capturing devices, the home video is stored in digital format and it can be uploaded to a PC as a video file.

There is another property that is more peculiar to home videos. As said before, home videos are captured by non-professional users; usually home video authors do not capture according to a storyboard and do not think about choosing the best angle or the best illumination. Moreover, home video authors often introduce defects in their videos, like shakiness, jerkiness, blur, “ground shooting”¹, too low or too high luminance, etc. Evidently, most of the defects and most of the unpleasantness of home videos are determined by the lack of professional capture skills, and not by technological limitations. Consequently, the quality of a home video cannot be judged with the same criteria that are applied for commercial content, as [Mei et al., 2005] have argued. Hence, an automatic home video editor should pay attention to the typical flaws present in home videos and eliminate or correct them.

In most cases, home videos are edited in order to select the most important or pleasant video portions and cut out the less interesting ones (see user need 1, p. 35). Unedited home video is very often seen as too long and boring, therefore the most important reason for doing video editing is to have a selection of desired excerpts of the unedited video. We think that an automatic video editing program should be able to select which parts of the video to keep and which to discard according to

¹“Ground shooting” refers to video footage undeliberately captured by a user that forgets to switch off the camera and walks with it while pointing it to the ground.

criteria that approximate as much as possible the user's method of decision. Generally, in a home video each camera take is shot for a specific reason. Therefore, it is not possible to assume that some camera takes are more important than others, unless the user specifies so. Because of this fact, the video segments selected by our algorithm should uniformly represent all the unedited video material.

Because of the considerations made till now, we identified the following *user requirements* for our automatic video editor:

Requirement A: *Duration of the edited video.*

The user should be able to specify the desired duration of the edited video. The user, in fact, should be able to decide how long the selection of clips from the unedited video should be, or how much of the unedited video is going to be kept in the edited video.

Requirement B: *Selection of good-quality video.*

The edited video should contain as much as possible video portions with high visual quality (little shakiness, in focus, etc.), meanwhile parts with low visual quality should be excluded.

Requirement C: *Representativeness of the edited video.*

The edited video should contain video segments that uniformly represent the raw unedited video. This requirement comes from the fact that every camera take should be considered important for the user.

Requirement D: *Content/semantic criteria.*

The edited video should contain video segments that are interesting for their content or for their semantics, such as segments with faces or segments captured after a zoom in operation. Usually, when the user zooms in on a particular, it is because the particular is deemed as important.

Requirement E: *Customizability of requirements.*

Some of the former requirements may clash with each other. Therefore, the user should be able to specify the relative importance of the requirements. For example, a user may be more interested in high-quality content and less in uniformly distributed content. That user may want to give weights of importance to each requirement, these weights should be taken into account by the system and the video should be edited accordingly. It is indeed common practice in automatic video editing tools to allow the user to specify the "style" according to which the video should be composed.

It is possible to think to other user requirements, for example concerning the audio stream associated with the home video. For example, one may require that

the edited video should contain clips selected in a way such that sentences are not abruptly interrupted. However, during the user evaluations performed in the context of this thesis, we have observed that authors of home videos do not deem as very important to preserve the audio of their videos; often they cover it with a music track. Furthermore, we have observed that the home videos of our users contained very few speech segments: home videos are captured primarily to record interesting visual events and not interesting sounds. Also Gatica-Perez et al. [2003] have stated that audio streams of home videos usually contain short speeches interleaved with long segments of ambient background sound. Therefore, to keep our algorithm simple, we do not analyze the audio stream associated with a home video.

Our algorithm will therefore analyze the video stream and create a selection of video segments aiming at satisfying the requirements listed before. For simplicity, we impose the constraints that a certain video clip cannot be chosen twice to be part of the edited video, and that the clips selected to form the edited video maintain the same temporal order they had in the original unedited video. During the user tests done in the context of this thesis, we have noticed that users do not ask to include the same video clip two times. Some users wanted to shuffle the temporal order of their video clips. Still, we think that it is best to automate the task of content selection and to let the user decide about the order of the clips. Selection of good content is in fact an important need in home video editing (see user need 1, p. 35). That is why this automatic algorithm focuses on this need and not on other possible requirements such as adding music, video transitions, video effects, text, etc. Also, our algorithm is designed to produce an edited video that will be successively refined and possibly enriched by the user (Figure 3.1).

3.2 Problem formulation

In this section, the considerations and requirements listed in the previous section are precised in formal language. Also, a formal definition of the problem of automatic home video editing is given. These definitions are used to describe our algorithm in the rest of the chapter. For clarity, a table with all the symbols defined in this chapter is shown at page 74.

We start by defining what an unedited home video is:

Definition 3.1 (Unedited home video). An *unedited home video* \mathcal{V} is a finite sequence of N video frames $\mathcal{V} = (f_1, \dots, f_N)$ that the user has captured with a device. These frames are meant to be displayed one after another at a frame rate R . A home video has also an audio track associated to it, which is not considered in the present analysis. \square

As seen before, home videos always come divided in *camera takes*.

Definition 3.2 (Camera take). In a home video \mathcal{V} , a *camera take* c is defined as a sequence of consecutive video frames

$$c = (f_i, \dots, f_j), \quad 1 \leq i \leq j \leq N.$$

A camera take c is a continuous portion of home video that begins when the user presses the “record” button on the capturing device and ends when the user presses the “stop recording” button. We can define \mathcal{T}_k as the timestamp associated with the first frame of camera take k . \square

Definition 3.3 (Camera take segmentation). The *Camera take segmentation* of the home video \mathcal{V} is a partition $\mathcal{C}(\mathcal{V})$ of the video \mathcal{V} :

$$\mathcal{C}(\mathcal{V}) = \{c_i, 1 \leq i \leq N_C\}.$$

N_C is the total number of camera takes in the video \mathcal{V} . In a home video, the camera takes never overlap: $\forall c_i, c_j \in \mathcal{C}(\mathcal{V}), c_i \cap c_j = \emptyset$. Also, the union of the camera takes is the whole video: $\bigcup_{i=1}^{N_C} c_i = \mathcal{V}$. \square

In order to automatically edit a home video, an approach very often used in the literature consists in dividing the camera takes into even smaller video portions ([Hua et al., 2004] [Yip et al., 2003] [Girgensohn et al., 2000] [Takeuchi and Sugimoto, 2007]). These portions are usually called sub-shots or *video segments*. The algorithm described here also adopts the approach of structuring the home video in video segments. In Section 1.1.1 it has been clarified that a single frame cannot be seen as the basic element of a movie. A small video segment, instead, carries meaningful temporal information and can be used as the basic unit in the editing process. The video segments are created such that each one of them is easily characterizable by some video properties (for example zoom segments, low luminance segments, close-up segments, etc.). After dividing a home video into video segments, the process of automatic video editing can be formalized as selecting a particular set of video segments to compose an edited version.

Definition 3.4 (Video segment). A *video segment* v is a set of consecutive frames of a home video \mathcal{V} that belong to the same camera take.

$$v = (f_i, \dots, f_j), \quad 1 \leq i \leq j \leq N$$

$$\forall v \exists! c \in \mathcal{C}(\mathcal{V}) : f \in v \Rightarrow f \in c.$$

The formula above stresses the fact that each video segment v belongs to one and only one camera take c . \square

Definition 3.5 (Segmentation in video segments). A *segmentation in video segments* of a home video \mathcal{V} is a partition of \mathcal{V} into video segments:

$$\mathcal{S}(\mathcal{V}) = \{v_i, 1 \leq i \leq N_V\}.$$

N_V is the total number of video segments in the segmentation $\mathcal{S}(\mathcal{V})$. As for camera takes, video segments are non-overlapping and cover the whole video \mathcal{V} . \square

Before, we specified that each video segment belongs to only one camera take. Since video segments cannot extend across multiple camera takes, each camera take boundary is also a segment boundary. To have more clarity in the rest of the

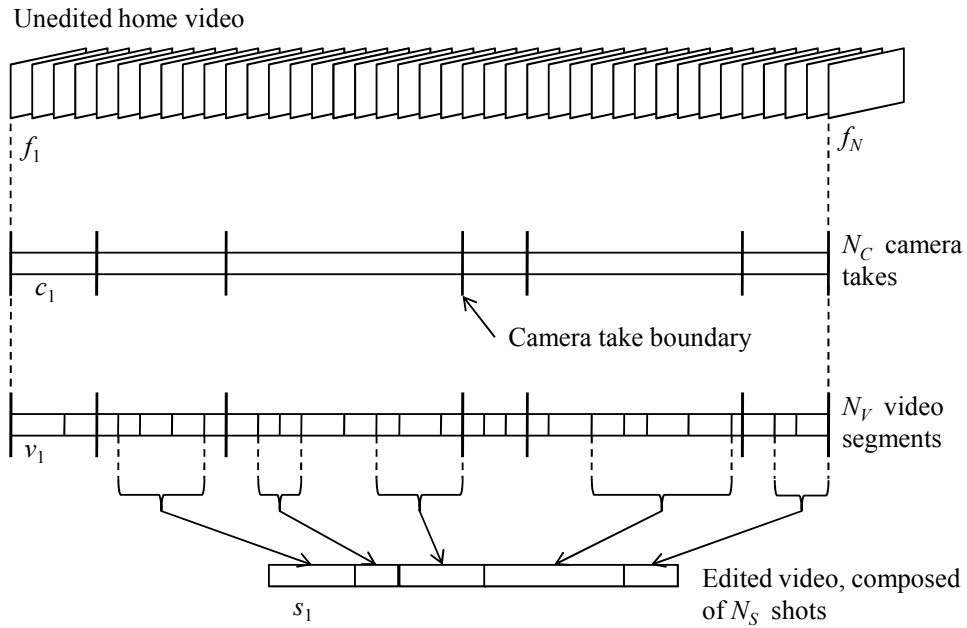


Figure 3.2. Hierarchical organization of a home video. On the top of the figure, the unedited video with its N frames is displayed. Below, the home video is represented as a thin rectangle subdivided into the N_C camera takes. Further below, the home video is divided at a finer level: the N_V video segments are shown.

chapter, we define here a duration function to be applied to any of the defined video items.

Definition 3.6 (Duration of video items). Given a video item x (a video segment or an arbitrary set of video segments), the *duration* $\delta(x)$ of the video item x is defined as the time duration of the video item x , expressed in seconds. \square

Till now, we have shown how an unedited home video is structured in camera takes and video segments. Figure 3.2 visualizes the relationship between camera takes and video segments in a home video. In the figure, the unedited video is partitioned in three levels: camera takes, video segments, shots. The rest of Figure 3.2 is explained in the following paragraphs, after some definitions.

In order to formalize the problem of automatic video editing, we need to define how the final result should look like. The algorithm will compose the edited version of the home video by choosing some of the video segments in which the video has been partitioned. We now define the building blocks of the edited video, the *shots*.

Definition 3.7 (Shot). A *shot* is a set of temporally contiguous video segments belonging to the same camera take.

$$s = (v_i, \dots, v_j) \quad 1 \leq i \leq j \leq N_V,$$

$$\forall s \exists! c \in \mathcal{C}(\mathcal{V}) : v \in s \Rightarrow v \in c \quad \square$$

Definition 3.8 (Edited video). An *edited video* E is a sequence of non-overlapping shots belonging to the same unedited home video \mathcal{V} :

$$E = (s_i, 1 \leq i \leq N_S)$$

$$\forall s_i, s_j \in E, s_i \cap s_j = \emptyset$$

$$\forall s_i, s_j \in E, \forall f_m \in s_i, \forall f_n \in s_j, i < j \Rightarrow m < n. \quad \square$$

In Definition 3.8, the second equation expresses the fact that the edited video E does not contain overlapping shots. The third equation expresses the fact that the shots of the edited video E are ordered according to the temporal order of the video: if the shot s_i comes before the shot s_j in the edited video, then the shot s_i contains frames of the unedited video placed before the frames of the shot s_j .

In principle, a user may wish an edited video E containing two times the same shot, or may want to swap the shots' order. However, for simplicity we focus on the case of edited videos made of non-overlapping, ordered shots. The user can always refine the edited video E that the system calculates.

An edited video is visualized in the bottom part of Figure 3.2. The edited video is composed by shots that have been selected from the video segments of the unedited video.

When the user watches the edited video, he/she cannot be aware of the segment boundaries. In fact, the optical flow of the edited video is interrupted only at shot boundaries, since between different shots there are frames missing or there is a camera take boundary. Consequently, the end user perceives the edited video as being composed of different shots. Also, shots are important because the user can

express the requirements for the edited video in terms of shots (number, average duration).

Given an unedited home video \mathcal{V} , it is possible to generate from it an immense number of edited videos E , as many as the possible subsets of $\mathcal{S}(\mathcal{V})$, that are $2^{N_V} - 1$ if the empty set is not considered. The automatic home video editor has the task to search the edited video E that satisfies the user requirements and constraints as much as possible. Therefore, one way of formalizing automatic home video editing is to map it to an optimization problem. We can define a function of E that is to be maximized in the space of all the possible edited videos. The higher the value of E , the better E satisfies the constraints and the requirements.

Definition 3.9 (Objective function). The objective function $\Omega(E) : \mathcal{P}(\mathcal{S}(\mathcal{V})) \rightarrow \mathfrak{R}$ is defined as follows:

$$\Omega(E) = \mathcal{F}(\varsigma(E), \tau(E), \chi(E)) \quad \square$$

The function $\Omega(E)$ has as domain $\mathcal{P}(\mathcal{S}(\mathcal{V}))$, the set of all the possible subsets of $\mathcal{S}(\mathcal{V})$. In the equation above, $\varsigma(E)$ is the *suitability score* of the edited video E and models the user requirement B defined in Section 3.1. In other words $\varsigma(E)$ calculates the quality of the video segments in E . $\tau(E)$ is the *temporal uniformity score* of the edited video E and models requirement C. $\tau(E)$ measures how uniformly the video segments in E are distributed across the whole unedited video. $\chi(E)$ is the *content score* of the video E and models requirement D. $\chi(E)$ models how well E satisfies the content/semantic criteria.

After defining the objective function $\Omega(E)$, we can formulate our problem as the problem of maximizing $\Omega(E)$ in the space of all the possible edited versions of video \mathcal{V} . In order to do that, the function $\Omega(E)$ and the functions $\varsigma(E)$, $\tau(E)$ and $\chi(E)$ have to be designed in order to model the user requirements. Then, a known optimization algorithm can be used to maximize $\Omega(E)$. Part of the procedure is also the design of algorithms for analyzing the home video and partitioning it in video segments.

3.3 Solution overview

From this section on, the algorithm for automatic video editing is described in depth. This algorithm is subdivided in three main steps, as Figure 3.3 shows.

In the first step, the home video \mathcal{V} is taken as input. We assume that the information about how the video is partitioned in camera takes is also available as an input ($\mathcal{C}(\mathcal{V})$ in the figure). The information about the camera take segmentation is usually available by parsing the timestamp data in the video stream. The first step of the algorithm consists in extracting some low-level features. These are the

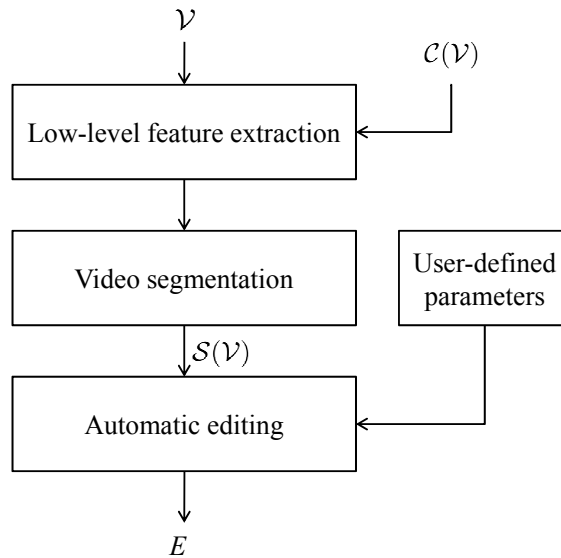


Figure 3.3. Flowchart of the algorithm for automatic home video editing.

camera motion, the image luminance and contrast, and the presence of faces. The process of extracting the low-level features is explained in detail in Section 3.4.

In the second step (“Video segmentation” block in Figure 3.3), a segmentation $\mathcal{S}(\mathcal{V})$ of the video \mathcal{V} is calculated. The video segments are calculated in such a way that the low-level features inside one segment present similar properties. For example, the camera motion inside one video segment should indicate the same type of movement (pan right, or zoom out); also, the luminance and contrast properties should be homogeneous inside the same video segment (good luminance and low contrast, for example). Segment boundaries should correspond to relevant changes of the low-level features and therefore of the properties of the video. After the video (V) is segmented, a *suitability score* for each video segment v is calculated. This score expresses how suitable a particular video segment is to be included in the edited video. The suitability score is computed by looking at the presence of shakiness, low luminance or low contrast in the video segment. Section 3.5 describes in detail how $\mathcal{S}(\mathcal{V})$ and the suitability scores are calculated.

In the third step (“Automatic editing” block in Figure 3.3), the edited video E is calculated by selecting some video segments belonging to $\mathcal{S}(\mathcal{V})$. The computation of E is based on the user requirements illustrated at page 40. To compute E , some user-defined parameters are taken into account. These parameters are the desired duration of the edited video (Requirement A) and the relative importance of the criteria for selecting the video segments (Requirement E). There are three

criteria: the video quality of the segments in E (Requirement B), the temporal uniformity of E (Requirement C) and the presence of faces in E (Requirement D). The calculation of the edited video E is explained in detail in Section 3.6.

3.4 Low-level feature extraction

Our method extracts the following low-level features from a home video \mathcal{V} : the camera motion, the image luminance, the image contrast and the presence of faces. We think that these are the most useful features in order to detect the typical mistakes of amateur movie makers (shakiness, low luminance) and to effectively characterize a home video. These features are also used often in the literature [Mei et al., 2005] [Girgensohn et al., 2000].

A well-know definition of camera motion is given in the MPEG-7 standard [Manjunath et al., 2002]. This standard defines 14 possible camera movements, shown in Figure 3.4. The figure shows 6 translational movements (left drawing) and 6 rotational movements (right drawing). Two camera movements are not shown in the picture: “zoom in” and “zoom out”. Zoom movements are movements of the camera’s focal lenses that enlarge or shrink the view on the captured object.

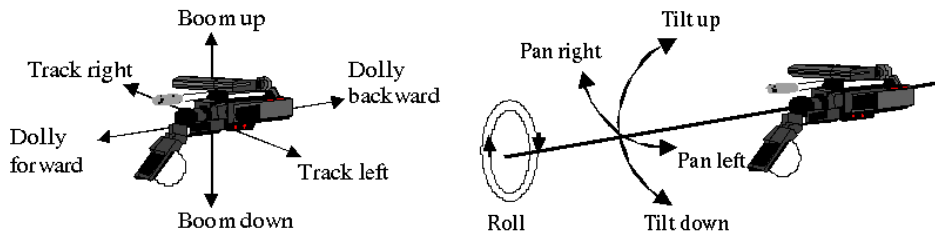


Figure 3.4. Camera motion parameters according to the MPEG7 standard.

We do not take into account all the 14 camera movements, since the estimation of these 14 movements is computationally very expensive and inaccurate. We extract only the *pan*, *tilt* and *zoom* movements, since these are enough to effectively characterize a home video. In order to detect pan and tilt movements, we use the “luminance-projection correlation” method, described in [Uehara et al., 2004]. To estimate zoom movements, we have designed a novel algorithm that is presented in detail in Section 3.7.

Besides the camera motion, the image luminance and contrast are also extracted. For each video frame, we calculate the luminance as the average of the luminance values of the frame. The luminance is the Y value in the YUV color space. Moreover, for each video frame we calculate the contrast as the standard deviation of the luminance values of the frame.

Finally, we perform face detection to every frame of the video \mathcal{V} . We apply a multi-view face detector based on [Viola and Jones, 2001] trained to detect also tilted and partially rotated faces, as in [Kroon, 2006]. We store only the number of faces present in a frame.

In conclusion, for each frame $f_n \in \mathcal{V}$ we extract the following low-level features:

- the amount of camera pan p_n (range $[-187 \ 187]$ screens/minute, the unity of measure “screens/minute” will be explained in Section 3.5.1),
- the amount of camera tilt t_n (range $[-187 \ 187]$ screens/minute),
- the amount of zoom z_n (range $[-0.146 \ 0.146]$, explained in Section 3.7.3),
- the average luminance l_n (range $[0 \ 255]$),
- the average contrast o_n (range $[0 \ 255]$),
- the number of faces present a_n .

The reason for extracting the camera motion is twofold. First, the video can be structured in video segments by making sure that every change in type of camera motion (pan left to pan right, for example) defines a new video segment. Second, the detection of the camera motion enables the estimation of shakiness (frequently changing camera movements) and jerkiness (fast camera movements). Shakiness and jerkiness are aesthetically unpleasant; they are two of the most common defects found in videos captured by non-professional users [Mei et al., 2005]. According to Requirement B, these parts should be removed from the final summary. Other defects typical of home videos are the lack of proper illumination and the lack of informativeness (for example shooting the ground). Luminance and contrast are extracted to detect respectively these defects. It is most likely that users do not want video segments with low luminance and video segments that are not informative (that have low contrast). Also, we keep track of the faces that each frame contains. In fact, the user may be interested in video segments containing faces (Requirement D).

3.5 Home video segmentation

In this section, we explain how the video segmentation $\mathcal{S}(\mathcal{V})$ is calculated based on the values of the low-level features (second block of the flowchart in Figure 3.3).

The purpose of the video segmentation is to structure a home video into short video segments in which the low-level features have approximately constant values. As said in Definition 3.4, a video segment cannot belong to two camera takes. Therefore, the camera take boundaries defined in $\mathcal{C}(\mathcal{V})$ are also video segment boundaries. The home video segmentation algorithm explained here is repeated for each camera take $c \in \mathcal{V}$.

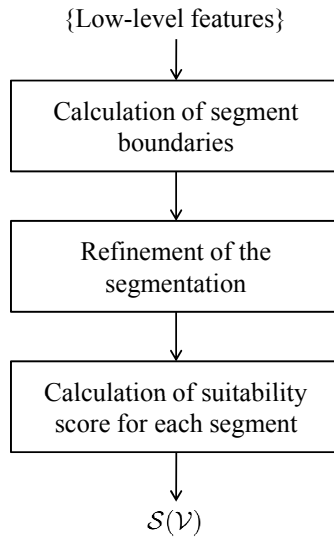


Figure 3.5. Steps of the segmentation procedure.

Our video segmentation algorithm is composed of three steps, as Figure 3.5 illustrates. First, the low-level features are filtered and analyzed to find their main properties. When there is a change in the features' properties, a segment boundary is created. In the second step, the so-obtained segmentation is refined. Video segments that are too short are merged to the adjacent ones and video segments that are too long are split. The minimum duration for a video segment is 0.8 sec and the maximum is 5 sec. The value of 0.8 sec has been chosen to have not too short segments that would follow the noise in the low-level features, this is useful to have a segmentation that reflects the most important changes in properties of the video. The value of 5 sec has been chosen in order not to have too long segments that would make it difficult to divide long camera takes where the low-level features do not significantly change. Once the segmentation is refined, in the third step a suitability score is calculated for each video segment. This score expresses to what extent a video segment is suitable to be included into the edited video. The suitability score depends on quality considerations, more precisely on the possible presence of shakiness, low luminance or low informativeness in the video segment. Finally, the video segmentation $\mathcal{S}(\mathcal{V})$ is saved in XML format. In this way, $\mathcal{S}(\mathcal{V})$ does not need to be recalculated if the user wants to produce many edited versions of the same video according to different specifications.

The three steps visualized in Figure 3.5 will be now illustrated in detail. Section 3.5.1 deals with the calculation of the segment boundaries, while section 3.5.2 explains the refinement of the segmentation. Finally, Section 3.5.3 describes how

the suitability scores are calculated.

3.5.1 Calculation of the segment boundaries

In this section we show how the low-level features are processed to calculate a first version of the segmentation $\mathcal{S}(\mathcal{V})$. Figure 3.6 shows the amount of pan movement

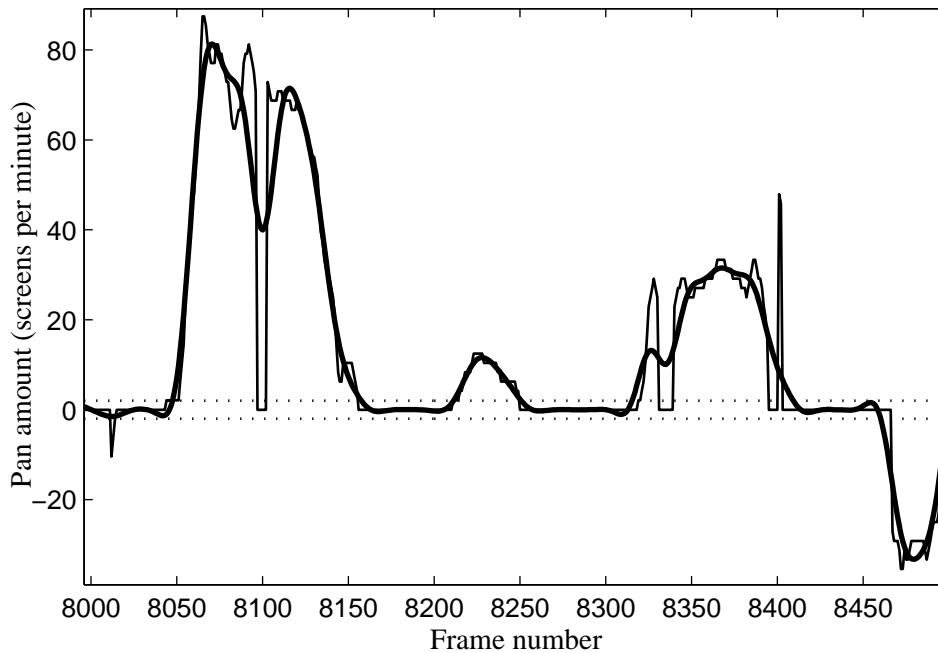


Figure 3.6. Amount of pan movement (thin line) and filtered pan curve (thick line). On the horizontal axis the frame numbers are displayed; on the vertical axis the pan values p_n are represented, measured in screens per minute.

for 20 sec of a home video (thin solid line). In Figure 3.6 the pan movements are measured in “screens per minute”. For example, a “pan right” movement of one screen per minute means that the video background is moving from right to left at a speed such that the leftmost pixel becomes the rightmost in one minute. We use this unity of measure in order to quantify pan and tilt movements independently of the video format. In fact, different video formats can use different screen resolutions and frame rates. This is important, since nowadays home videos come in different formats and we want to apply to all of them the same algorithm for the segmentation.

Positive values of the sequence of the pan values $\{p_n\}$ represent “pan right” movements, negative values correspond to “pan left” movements. The segmentation procedure is based on extracting the sequence of the pan values $\{p_n\}$, filtering

it, and applying thresholds to it to find relevant changes in the pan movement. The video frames are left unchanged.

Because of the shakiness while capturing the video, the thin solid line in Figure 3.6 is very irregular. To obtain a meaningful segmentation, we filter out the shakiness and preserve the main pan movement using a low-pass FIR filter with 1 Hz as cut frequency. For the common frame rate of 25 fps, we use a filter of 60 samples. The result is the thick solid curve, which is smoother and represents the main pan movement.

To calculate the segment boundaries, initially the sequences of pan movements $\{p_n\}$ and tilt movements $\{t_n\}$ are filtered with the low-pass filter mentioned before. After the filtering, the segment boundaries are computed by applying thresholds on the values of the low-level features. This can be explained by looking again at Figure 3.6. In this figure, the thresholds are represented by the two dotted lines at 2 and -2 screens per minute respectively. The segment boundaries are determined as the intersections between the filtered pan curve (thick solid line) and the thresholds (dotted lines). Segments of the thick curve below, between and above the two thresholds correspond to pan left, still and pan right segments respectively. The values of 2 and -2 screens/minute have been determined empirically.

Other segment boundaries are determined by applying thresholds in the same way to the other low-level features. More precisely:

- The sequence of tilt values $\{t_n\}$ is filtered with the same low-pass FIR filter of 60 samples with 1 Hz as cutoff frequency. Segment boundaries are determined by intersecting the filtered tilt curve with two thresholds set at 2 and -2 screens per minute.
- Additional segment boundaries are found by intersecting the sequence of the zoom values $\{z_n\}$ with two thresholds, placed empirically at 0.003 and -0.003 . Zoom values below the thresholds correspond to “zoom in” movements, values in between the thresholds signify absence of zoom and values above the thresholds indicate “zoom out” movements.
- Additional segment boundaries are determined by intersecting the sequence of luminance values $\{l_n\}$ with a threshold $T_L = 70$. The value 70 has been chosen empirically, assuming that the luminance values range from 0 to 255. Frames with average luminance below T_L are considered to have low luminance.
- Additional segment boundaries are determined by intersecting the sequence of contrast values $\{o_n\}$ with a threshold of $T_C = 20$, again determined empirically. Frames with contrast less than T_C are considered to carry a little amount of information.

Till now, we have seen that a particular camera take c is segmented by defining segment boundaries in correspondence to changes in the properties of camera motion, luminance and contrast. This procedure is iterated for every camera take $c \in \mathcal{C}(\mathcal{V})$. All these segment boundaries are used to create the first segmentation $\mathcal{S}(\mathcal{V})$.

3.5.2 Refinement of the segmentation

After the first instance of the segmentation $\mathcal{S}(\mathcal{V})$ has been created, some refinement operations are carried out. First, the segments that are too short are merged with the neighboring ones. We adopt as minimum duration for a video segment 0.8 sec. In order to merge the short video segments, we use the algorithm shown in Figure 3.7. Given a camera take, the algorithm stores the durations of all its video segments into the sorted list \mathcal{D} . This list is sorted from the shortest duration to the longest. Successively, by browsing the sorted list \mathcal{D} the algorithm finds all the segments with durations smaller than 0.8 sec, starting from the shortest and going to the longest. Each one of these short segments is merged with the shortest of the two neighboring segments of the segment list $\mathcal{S}(\mathcal{V})$. After a video segment v_i is merged with another segment v_{i+1} , the duration of v_{i+1} has increased. Therefore the position of the duration of v_{i+1} inside the list \mathcal{D} is updated to maintain the list sorted in ascending order. After \mathcal{D} is updated, the procedure is reiterated again on the shortest video segment. The whole algorithm is reiterated for each camera take of the video.

After the application of the algorithm 3.7, $\mathcal{S}(\mathcal{V})$ does not contain any segments shorter than 0.8 sec. At this point, for each video segment v five *properties* are calculated: these are the average amount of pan, the average amount of tilt, the average amount of zoom, the average brightness and the average contrast. Each one of these properties receives also a description based on the same thresholds applied to find the segment boundaries. For example, if the average amount of pan in the segment is less than -2 screens/minute, the segment will be labeled as “pan left”. If the amount of pan is between -2 and 2 screens/minute, the segment is labeled “pan still” and if the pan amount is larger than 2 screens/minute the label will be “pan right”. This labeling of properties is shown in Figure 3.8. In this figure every segment is represented as an XML node with five “Property” subnodes. Each of these subnodes has the name of one of the five aforementioned properties, the amount and the description. This XML format is used for easy visualization and debugging.

After the properties of the video segments in $\mathcal{S}(\mathcal{V})$ have been calculated, neighboring segments that have equal labels in all the five properties are merged together. Finally, all the video segments longer than 5 sec are split. If a video segment v has

```

algorithm MERGE SHORT SEGMENTS
input: minimum segment duration MIN_DURATION = 0.8 sec ;
     $\mathcal{C}(\mathcal{V})$  ;
     $\mathcal{S}(\mathcal{V})$  ;

begin
  for each  $c \in \mathcal{C}(\mathcal{V})$  do
    begin
      for each  $v_i \in c$  do
        begin
           $d_i = \delta(v_i)$  ;
          //The duration  $d_i$  is added at the right place in the sorted list  $\mathcal{D}$ 
          add  $d_i$  to  $\mathcal{D}$  ;
        end
         $d_i = \mathcal{D}(0)$  ;
        while  $d_i < \text{MIN\_DURATION}$  do
          begin
            //  $v_i$  is the video segment corresponding to  $d_i$ 
            if  $\delta(v_{i-1}) < \delta(v_{i+1})$  then
               $v_{i-1} = v_{i-1} \cup v_i$  ;
               $d_{i-1} = d_{i-1} + d_i$  ;
              reorder  $d_{i-1}$  inside  $\mathcal{D}$  ;
            else
               $v_{i+1} = v_i \cup v_{i+1}$  ;
               $d_{i+1} = d_{i+1} + d_i$  ;
              reorder  $d_{i+1}$  inside  $\mathcal{D}$  ;
            end
             $\mathcal{D} = \mathcal{D} \setminus d_i$  ;
             $\mathcal{S}(\mathcal{V}) = \mathcal{S}(\mathcal{V}) \setminus v_i$  ;
             $d_i = \mathcal{D}(0)$  ;
          end
        end
      end
    end
  end

```

Figure 3.7. Pseudocode of the algorithm for merging too short segments.

```

<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<Segmentlist framerate="25" movie="E:\EWW2 use test videos\VideoCalifornia\
California.mov">

  <Segment duration="38" id="0" startFrame="0">
    <Property Name="Pan">
      <Description>Pan still</Description>
      <Value>0.580274</Value>
    </Property>
    <Property Name="Tilt">
      <Description>Tilt down</Description>
      <Value>-3.80085</Value>
    </Property>
    <Property Name="Zoom">
      <Description>Zoom static</Description>
      <Value>2.83854e-005</Value>
    </Property>
    <Property Name="Luminance">
      <Description>Luminance ok quality</Description>
      <Value>79.4899</Value>
    </Property>
    <Property Name="Contrast">
      <Description>Contrast ok quality</Description>
      <Value>41.9575</Value>
    </Property>
    <CameraTake>0</CameraTake>
    <Shakiness>2.53329</Shakiness>
    <FacesPerFrame>0</FacesPerFrame>
    <Suitability>0.873336</Suitability>
  </Segment>

  <Segment ...
</Segmentlist>

```

Figure 3.8. XML representation of a video segment *v* and its properties.

duration $\delta(v) > 5\text{sec}$, then v is split into P equally long parts, where

$$P = \text{ceil} \left(\frac{\delta(v)}{5 \text{ sec}} \right)$$

and $\text{ceil}(r)$ gives the lowest integer number bigger than the rational number r . After these operations are carried out, $\mathcal{S}(\mathcal{V})$ does not contain any video segment shorter than 0.8 sec or longer than 5 sec.

3.5.3 Calculation of the suitability scores

After the refinement of $\mathcal{S}(\mathcal{V})$ a *suitability score* $\rho(v)$ is calculated for each video segment v . As said before, the suitability score depends on quality considerations. It depends on the presence of *shakiness*, *low luminance* or *low informativeness* in the video segment.

Shakiness is a common defect in home videos. It is mostly due to the fact that the users do not always hold the capturing device firmly while shooting a video. Often, the capturing device shakes or trembles, resulting in fast and frequently changing pan and tilt movements in the video. Figure 3.6 shows that the pan curve (thin solid line) presents many irregularities. To detect the amount of shakiness, we calculate the euclidean distance between the pan curve $\{p_n\}$ and the filtered pan curve $\{\tilde{p}_n\}$ (thin solid line and thick solid line in Figure 3.6 respectively). This allows to determine exactly the part of the signal $\{p_n\}$ that has been filtered out by the lowpass FIR filter.

Therefore, the shakiness $S(v)$ of the video segment v is calculated according to the following equation:

$$S(v) = \frac{1}{n_v} \sum_{i=s_v}^{s_v+n_v-1} \sqrt{(p_i - \tilde{p}_i)^2 + (t_i - \tilde{t}_i)^2},$$

where n_v is the duration in frames of segment v , s_v is the start frame of segment v , p_i and t_i are the pan and tilt values for frame i and \tilde{p}_i and \tilde{t}_i are the filtered pan and tilt values for frame i .

Once the amount of shakiness $S(v)$ is computed, a measure of quality related to the shakiness is determined:

$$\rho_S(v) = \frac{S_{\max} - S(v)}{S_{\max}},$$

where S_{\max} is a constant empirically determined representing a large amount of shakiness in a video segment. $\rho_S(v)$ decreases linearly with the increase of $S(v)$: the more shakiness is present in v , the lower the detected motion quality $\rho_S(v)$.

Two other measures of quality, concerning the luminance and the contrast, are

determined. The luminance quality $\rho_l(v)$ is determined as follows:

$$\rho_l(v) = \begin{cases} 1 & \bar{l}_v \geq T_L \\ \bar{l}_v/T_L & \bar{l}_v < T_L \end{cases},$$

where \bar{l}_v is the average luminance value in the video segment v : $\bar{l}_v = \frac{1}{n_v} \sum_{i=s_v}^{s_v+n_v} l_i$. \bar{l}_v can range from 0 (lowest luminance quality) to 1 (highest quality). If the average luminance is above the threshold T_L (the same threshold used for the calculation of the segment boundaries in Section 3.5.1), then the luminance quality gets the maximum value. However, if the average luminance \bar{l}_v is below T_L , then the luminance quality $\rho(v)$ decreases linearly with \bar{l}_v . The reason for the decrease is that generally the lower the luminance, the poorer the quality that the user perceives. There may be sometimes beautiful dark scenes, like shots of famous paintings. The user has anyway the possibility of refining the automatically edited video at a later stage.

The contrast quality $\rho_c(v)$ is determined similarly, as follows:

$$\rho_o(v) = \begin{cases} 1 & \bar{o}_v \geq T_C \\ \bar{o}_v/T_C & \bar{o}_v < T_C \end{cases},$$

where \bar{o}_v is the average contrast value in the video segment v : $\bar{o}_v = \frac{1}{n_v} \sum_{i=s_v}^{s_v+n_v} o_i$.

Finally, the suitability score $\rho(v)$ of the segment v is calculated as follows:

$$\rho(v) = \min [\rho_S(v), \rho_l(v), \rho_c(v)] .$$

The suitability score ranges between 0 and 1. $\rho(v)$ is calculated as the minimum of the motion quality ρ_S , the luminance quality ρ_l and the contrast quality ρ_c . We adopted the minimum and not a linear combination of the three aspects of quality, under the assumption that a defect in one of these three video properties is already sufficient to make the segment not suitable, even if the other properties have acceptable quality. After the computation of $\rho(v)$ for all the video segments, $\mathcal{S}(\mathcal{V})$ is completely determined. To produce many edited versions of the same video \mathcal{V} , it is not needed anymore to segment the video and to calculate the suitability scores.

3.6 Automatic home video editing

This section explains how the edited video E is computed once the input video has been divided into segments (third step of the flowchart in Figure 3.3).

As described in Section 3.2, the principle of the algorithm is to search for an edited video E that satisfies as much as possible the requirements provided by the user. This is formalized as maximizing the objective function $\Omega(E)$. This function is designed to return to what extent the edited video E meets the user requirements. Section 3.6.1 explain how $\Omega(E)$ is designed, meanwhile Section 3.6.2 describes the algorithm that searches for the edited video E that maximizes $\Omega(E)$.

3.6.1 Design of the objective function

Initially the user feeds the following parameters into the system:

- The *total duration of the edited video* Δ_E .
- The *average shot duration* in the edited video Δ_s . Via this parameter, the user can specify what the shot rhythm should be in the edited video. A fast succession of shots conveys a more intense feeling, a slow pace of shot cuts conveys a more relaxed atmosphere [Zettl, 1999].
- The *minimum shot duration* Δ_s^{\min} . It cannot be smaller than the minimum segment duration. The user expresses all these time requirements in seconds.
- The *weight of the suitability* w_ς . This is a number in the range $[0, 1]$ that expresses to what extent the user is interested in having good quality clips (video segments with high suitability score) in his/her edited video.
- The *weight of the temporal uniformity* w_τ . This number also ranges in $[0, 1]$ and expresses to what extent the user is interested in an edited video E that represents uniformly all the parts of the unedited input video.
- The *weight of the content score* w_χ . This number too ranges in $[0, 1]$. The system makes sure that $w_\varsigma + w_\tau + w_\chi = 1$.

To take into account the three user-defined weights, the objective function introduced in Definition 3.9 is designed as follows:

$$\Omega(E) = w_\varsigma \cdot \varsigma(E) + w_\tau \cdot \tau(E) + w_\chi \cdot \chi(E) . \quad (3.1)$$

We will now explain how the functions $\varsigma(E)$, $\tau(E)$ and $\chi(E)$ are designed. The suitability score function $\varsigma(E)$ of the video E is the weighted average of the suitability scores of the video segments included in E .

$$\varsigma(E) = \frac{1}{\delta(E)} \left[\sum_{\forall s \in E} \sum_{\forall v \in s} \rho(v) \cdot \delta(v) \right] ,$$

where $\delta(v)$ is the time duration of the video segment v and $\delta(E)$ is the time duration of the edited video E .

The design of the *temporal uniformity score* $\tau(E)$ of the video E will now be explained. First of all, we introduce the concept of *missing parts* $\{m_j\}$ of the video $\mathcal{S}(\mathcal{V})$. Each edited video E defines a set of video segments that are included in it. The complementary set of video segments is excluded or *missing* from E . These missing video segments can be grouped in temporally contiguous sequences, each one of these sequences is delimited by two successive shots of E .

Definition 3.10 (Missing part). Given a home video \mathcal{V} and an edited version of it E , a missing part m is a sequence of temporally contiguous video segments

$$m = (v_i, \dots, v_j), \quad 1 \leq i \leq j \leq N_V$$

such that

- the first video segment of m is either segment 1 or it follows directly a video segment included in E : $i = 1$ or $v_{i-1} \in E$.
- the last video segment of m is either segment N_V (the last) or it precedes directly a video segment included in E : $j = N_V$ or $v_{j+1} \in E$.
- no video segment of m belongs to E : $\forall v \in m : v \notin E$ □

If E has N_S shots, there can be at most $N_S + 1$ missing parts. We define N_M as the number of the missing parts. In contrast with what happens for video segments and shots, a missing part m can span across multiple camera takes.

To calculate the temporal uniformity $\tau(E)$, we compute the average μ and the standard deviation σ of the duration of missing parts:

$$\mu_m = \frac{1}{N_M} \sum_{i=1}^{N_M} \delta(m_i) ,$$

$$\sigma_m = \sqrt{\frac{\sum_{i=1}^{N_M} (\delta(m_i) - \mu_m)^2}{N_M}} ,$$

where $\delta(m_i)$ is the time duration of the missing part m_i .

Similarly, we calculate the average and standard deviation for the shots of E :

$$\mu_s = \frac{1}{N_S} \sum_{j=1}^{N_S} \delta(s_j) ,$$

$$\sigma_s = \sqrt{\frac{\sum_{j=1}^{N_S} (\delta(s_j) - \mu_s)^2}{N_S}} .$$

Finally, the temporal uniformity score $\tau(E)$ is defined as

$$\tau(E) = 1 - \left(\frac{1}{2} \frac{\sigma_m}{\mu_m} + \frac{1}{2} \frac{\sigma_s}{\mu_s} \right) .$$

If the missing parts have all the same durations, and the shots have all the same duration, then σ_m and σ_s are both zero and $\tau(E)$ assumes the maximum value. However, if the standard deviations of the shots or of the missing parts are large, the shots (or the missing parts) differ widely in duration and therefore they are not uniformly distributed on the unedited video \mathcal{V} .

The *content score* function $\chi(E)$ returns to what extent the edited video E contains faces. We define the binary function $\phi(v)$ that determines whether the video segment v contains faces or not:

$$\phi(v) = \begin{cases} 1 & \frac{1}{n_v} \sum_{i=s_v}^{s_v+n_v} a_i \geq T_F \\ 0 & \frac{1}{n_v} \sum_{i=s_v}^{s_v+n_v} a_i < T_F \end{cases} ,$$

where s_v is the start frame of video segment v , n_v is the number of frames in v , a_i is the number of faces contained in frame i and T_F is a threshold empirically set at 0.08. We also define a similar function for the shots:

$$\phi(s) = \begin{cases} 1 & \exists v \mid v \in s \text{ and } \phi(v) = 1 \\ 0 & \forall v \in s : \phi(v) = 0 \end{cases} .$$

The function $\chi(E)$ is defined as

$$\chi(E) = \frac{\sum_{i=1}^{N_S} \phi s_i}{N_S} .$$

$\chi(E)$ is the fraction of shots of E that contain video segments with at least a minimum amount of faces.

3.6.2 Search of the best edited video

At this point, we have completely defined the objective function $\Omega(E)$. This function is maximized by searching the best edited video with a known local search algorithm: *simulated annealing* [Kirkpatrick et al., 1983]. The pseudo-code of the algorithm is shown in Figure 3.9.

The idea of the algorithm is to start from an edited video E and to increase $\Omega(E)$, by searching in the neighborhood of videos similar to E another video E' with a higher value of objective function. The neighboring video E' is generated by editing one shot of E , according to a procedure that will be explained later. If E' is found to be an improvement of E , then the procedure is reiterated on E' . Otherwise, in the case that E is the best edited video of its neighborhood, then the search procedure has converged to the *locally optimal* solution E . One could object that this search procedure can converge too soon to a locally optimal solution close to the starting point, meanwhile a more extended search could easily find solutions closer to the global optimum. The simulated annealing method is designed to be robust against convergences to poor local optima. It makes use of a control parameter called the *temperature* (T in Figure 3.9). The idea of simulated annealing is that in some cases the search can move to a neighboring solution E' also when $\Omega(E')$ is lower than $\Omega(E)$. The higher the temperature T , the higher the probability of this to happen. A *cooling schedule* is defined so that multiple iterations are run and in each one the value of the temperature T decreases.

In the pseudocode of Figure 3.9, the external “repeat” cycle implements the

algorithm SIMULATED ANNEALING
input: $\mathcal{S}(\mathcal{V})$ **begin** $E^{\text{best}} = \text{generateFirstEditedVideo}();$ $\omega^{\text{best}} = \Omega(E^{\text{best}});$ $T = T_0;$ **repeat** $\omega = \omega^{\text{best}};$ $E = E^{\text{best}};$ **repeat** $E' = \text{neighborEditedVideo}(E);$ $\omega' = \Omega(E');$ **if** $\omega' > \omega$ **or** $\exp\left(\frac{\omega' - \omega}{T}\right) > \text{random}[0, 1)$ **then****begin** $E = E';$ $\omega = \omega';$ **if** $\omega > \omega^{\text{best}}$ **then****begin** $E^{\text{best}} = E;$ $\omega^{\text{best}} = \omega;$ **end****end****until** EQUILIBRIUMCRITERION $T = T \cdot 0.9;$ **until** STOPCRITERION**end**

Figure 3.9. Pseudocode of the algorithm for the optimization of the function $\Omega(E)$. The equilibrium and stop criteria are explained in the text.

cooling schedule: at each iteration, T is decreased. The second “repeat” cycle implements the iterative improvement of the edited video E . This improvements always starts from the best video found up till that moment, E^{best} . At each iteration, given the current solution E , a neighboring video E' is generated and evaluated. If $\Omega(E') > \Omega(E)$, then E' is accepted as the point from which to continue the search.

However, if $\Omega(E') < \Omega(E)$, E' has still a chance to be accepted for the new search step. The probability p to accept E' in this case is given by

$$p = \exp\left(\frac{\Omega(E') - \Omega(E)}{T}\right)$$

and is high if the temperature is high and low otherwise.

During the execution of the algorithm, the edited video E^{best} that has the highest value of the objective function among the evaluated edited videos is memorized. At every search step, if the current video E is better than E^{best} ($\Omega(E) > \Omega(E^{\text{best}})$), then E becomes the best solution. E^{best} is returned as the result when the algorithm ends.

Two details of the pseudocode in Figure 3.9 are now clarified: the functions “generateFirstEditedVideo()” and “neighborEditedVideo(E)”. To generate the first edited video, the following steps are undertaken:

- The number of shots in E is calculated from the requirements of total duration of E and average shot duration: $N_s = \text{round}(\Delta_E / \Delta_s)$.
- Initially, the N_s shots are distributed uniformly in the unedited video \mathcal{V} .
- Then, the shots are moved backwards or frontwards if they contain a camera take boundary. Shots, in fact, cannot extend over multiple camera takes.
- Finally, overlapping shots are corrected.

With the procedure shown above, the first edited video E is generated, from which the search can start. In the function “neighborEditedVideo”(E), E' is constructed with the following procedure:

- A shot $s \in E$ is randomly selected.
- The system checks whether s can be shifted forwards. Here, to shift the shot s frontwards means to increase by one the start and end video segments of s . If the end segment of s is the last segment of the video, or if the segment next to the end segment of s already belongs to another shot, then s cannot be shifted frontwards. If the end segment of s is the end segment of a camera take, then the system checks whether it is possible to create a new shot with minimum duration Δ_s^{min} , at the beginning of the next camera take.
- The system checks whether s can be shifted backwards. All the operations of the former step are repeated in a specular way.
- If no shifting is possible, neither frontwards nor backwards, the system picks another shot and starts over the procedure.
- If it is possible to shift the shot s both backwards and frontwards, the system picks up randomly one of the directions.

- Finally, the shot s is edited according to the selected operation: either shifted forwards or backwards.

Two details of the algorithm still need to be explained: the “equilibrium criterion” that determines the end of the internal “repeat” cycle, and the “stop criterion” that determines the end of the external “repeat” cycle and of the algorithm (Figure 3.9). With regard to the internal cycle, the equilibrium criterion can be met in two ways: either a number N_R of new videos E' have been accepted as improvement of the current search point, or a maximum number of iterations N_I is reached. For high temperature values, it will be very easy for a new video E' to be accepted as improvement of the current. Therefore, the equilibrium criterion will be met in the first way, by going through N_R successful reconfigurations of the video. However, for low values of the temperature, it will become harder for a video E' to be taken as an improvement. Therefore, the equilibrium criterion will be met in the second way, after trying N_I reconfigurations.

The parameters N_R and N_I can be tuned to balance performance and speed in the summarization algorithm. If more iterations are done, the result will be better but also the speed will be lower. To decide how to set these parameters, 36 different combinations of values were tried, generated by combining six values of N_I with six values of N_R . The values of N_I were calculated from the number of video segments in the unedited video N_V . The following values for N_I were tried: N_V , $1.5N_V$, $2N_V$, $5N_V$, $10N_V$, $20N_V$. The values of N_R were calculated as fractions of N_I . N_R was given the following values: $N_I/40$, $N_I/20$, $N_I/10$, $N_I/5$, $N_I/3$, $N_I/2$. Each of the 36 combinations of values was tested on three home videos of duration 12, 30 and 45 minutes respectively. For each combination of parameters and each video, we ran the summarization algorithm 15 times, taking the average of the scores ω^{best} of the 15 results. We also measured the time employed by the algorithm, running on a custom PC. In running these tests, the weights of the algorithm w_ζ , w_τ , w_χ were set all at the same value, and the requested summary duration was set at half of the whole duration. Figure 3.10 shows the results of the summarization algorithm for different values of the parameters.

Looking at the results, we considered only the combinations of parameter values that allowed the algorithm to come to a solution in less than 10 seconds. The procedure, in fact, should be fast enough to allow the user to generate many edited videos out of the same unedited video. We observed also that some combinations of high values of N_I with low values of N_R corresponded to the same execution time as combinations of low values of N_I and high values of N_R . However, keeping low N_I and high N_R gave slightly better performances than the other way around, for the same computational time. The combination of parameters that gave

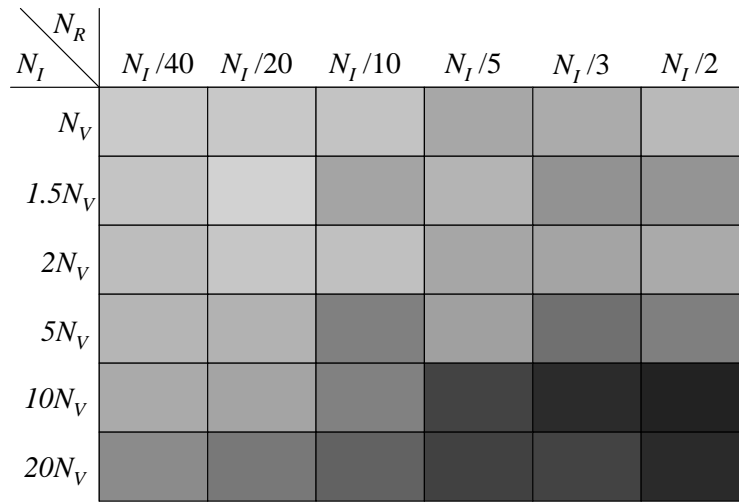


Figure 3.10. Performance of the summarization algorithm plotted against different combinations of values of the parameters N_I and N_R , for a 12 minutes long home video. The darker the color, the higher the value of ω^{best}

best performance, running in less than 10 seconds, is $N_I = N_V$ and $N_R = N_I/3$. With these parameters, the video of 12 minutes was summarized in 0.5 seconds, the video of 30 was summarized in 3 seconds and the video of 45 minutes was summarized in 8.2 seconds.

Once the equilibrium criterion for a given control parameter T is met, the control parameter is lowered. This is repeated until the “stop criterion” is met (external “repeat” cycle in Figure 3.9. In our implementation, the stop criterion is met when the temperature T goes below the value of 10^{-6} . It has been noted in our experiments that there are no significant improvements of the solution when the temperature goes below that value. The stop criterion can be met also in another way. If, for a given value of T , the N_I iterations of the internal repeat cycle are carried out finding very few improvements of the solution, then any lower value of T will probably be even less fruitful and the result will not improve anymore. Therefore, if for a certain temperature T the number of successful reconfigurations is lower than $N_I/200$, the procedure stops.

An algorithm similar to ours has been used in [Barbieri, 2007] to generate short previews of commercial movies. Moreover, in [Hua et al., 2004] an approach similar to the present has been applied to automatic home video editing. Also in [Hua et al., 2004] the home video is segmented and summarized by optimizing an objec-

tive function. However, the search method used is not simulated annealing. Instead, the authors of [Hua et al., 2004] formalize the automatic video editing problem as a nonlinear 0-1 programming problem, to be solved with a genetic algorithm. Still, the objective function they define is the linear combination of functions that evaluate different aspects of the edited video, similarly to our $\Omega(E)$ and to the one in [Barbieri, 2007]. Designing the objective function in this way makes the algorithm easily extendable. In fact, other criteria for selecting the best edited video can be added to the algorithm just by implementing other evaluation functions that can be added to the linear combination that determines $\Omega(E)$ (Equation 3.1, p. 57).

This summarization algorithm has been used by the participants of a user experiment described in Chapter 5. The algorithm was employed to produce edited versions of the users' videos, that the participants were then invited to further edit. After running that experiment, we discovered that the summarization algorithm contained an implementation error. After correcting it, the algorithm's performances measured in terms of the score of the solution $\Omega(E)$, improved by 3% - 4%. Since the confidence interval of $\Omega(E)$ is never bigger than 1%, the improvement is statistically significant. However, we are sure that the error in the algorithm did not influence the results of the experiment of Chapter 5. In fact, all the outcomes of the experiment are related to the editing interface. Also, users edited their videos according to high-level semantic criteria, such as people and events. Therefore, it is very unlikely that an improvement of 4% in an algorithm that summarizes video according to content analysis criteria may have made a significant difference for the users.

3.7 Zoom detection

In this section an algorithm for detecting zoom-in and zoom-out sequences in a video sequence is described. This algorithm is designed to be fast and computationally cheap, while giving reasonably precise results. The idea is to calculate a compact descriptor for each frame, a *projection on the radius*. The projections will be scaled and compared with each other in the same way one would scale and compare two images in order to find out if one is the upscaling or the downscaling of the other. When this zoom detection algorithm is used in combination with the pan and tilt detection algorithm described in [Uehara et al., 2004], the overall camera motion detection performance is higher than the state-of-the-art algorithm presented in [Varekamp, 2004], while the detection speed is the same. This has been revealed by a benchmark test on 37 minutes of home video.

The algorithm described here takes every couple of successive frames in a video and classifies whether these frames belong to a zoom-in sequence, to a zoom-out sequence or to none of these. When a sequence of video frames in which ev-

ery frame is approximately an enlarged (upscaled) version of the former frame is found, the algorithm labels it as a *zoom-in* sequence. In the same way, a sequence of frames in which every frame is approximately a shrunk (downscaled) version of the former frame is labeled as a *zoom-out* sequence.

Therefore, the algorithm takes each frame of a video sequence and compares it with the former frame. Three cases can occur:

1. The current frame is most similar to a linear enlargement of the former frame: in this case the two frames belong to a *zoom-in* sequence.
2. The current frame is most similar to a linear downscaling of the former frame: in this case the two frames belong to a *zoom-out* sequence.
3. The current frame is most similar to the former frame, without any scaling: in this case the two frames do not belong to a zoom sequence.

The algorithm does not scale and compare the whole frame, because this would be very expensive computationally. Instead, the algorithm uses a compact representation of a frame: a *projection*. This projection is then scaled and compared with the projection from another frame. Figure 3.11 shows the flow diagram of the zoom detection algorithm. Each block of the diagram will be now described in detail.

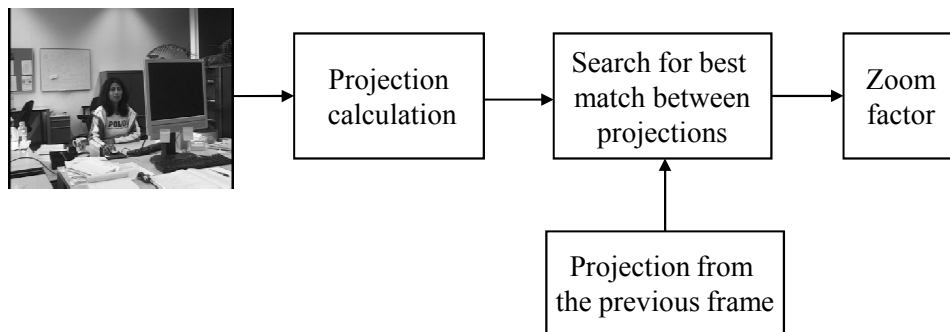


Figure 3.11. Flowchart of the zoom detection algorithm.

3.7.1 Projection calculation

The idea is that each projection should be a compact descriptor of how the image varies along the radial direction. We calculate the *projection on the radius* of a frame as a vector containing the average of the pixel values taken along concentric rectangles having the same aspect ratio as the image. In Figure 3.12 these rectangles are displayed over a video frame.

More precisely, the projection on the radius is calculated as follows. First, the video frame is converted into a gray-scale image (in case of video encoded in the YUV color space, only the Y plane is taken). We can define a gray-scale



Figure 3.12. Concentric rectangles along which pixels are averaged.

image like a bi-dimensional function $I(x, y)$, where $x = 0, \dots, (W - 1)$ and $y = 0, \dots, (H - 1)$ and W and H are the width and the height of the video frame, respectively. The projection of a frame is a vector $P(h)$ of $H/2$ elements:

$$P(h), h = 0, \dots, \frac{H}{2} - 1 .$$

Each value of $P(h)$ is calculated as the average of the pixels laying on a rectangle of height equal to $2(h + 1)$ pixels and width $2k$ pixels, where

$$k = \text{round} \left((h + 1) \frac{W}{H} \right) .$$

The values of the projection $P(h)$ are calculated according to the following equation:

$$P(h) = \text{horSides}(h) + \text{verSides}(h) , \quad (3.2)$$

where

$$\text{horSides}(h) = \sum_{i=-k}^{k-1} \left[I \left(\frac{W}{2} + i, \frac{H}{2} + h \right) + I \left(\frac{W}{2} + i, \frac{H}{2} - h - 1 \right) \right] \quad (3.3)$$

and

$$\text{verSides}(h) = \begin{cases} 0 & h = 0 \\ \sum_{i=-h}^{h-1} \left[I \left(\frac{W}{2} + k - 1, \frac{H}{2} + i \right) + I \left(\frac{W}{2} - k, \frac{H}{2} + i \right) \right] & h \geq 1 . \end{cases} \quad (3.4)$$

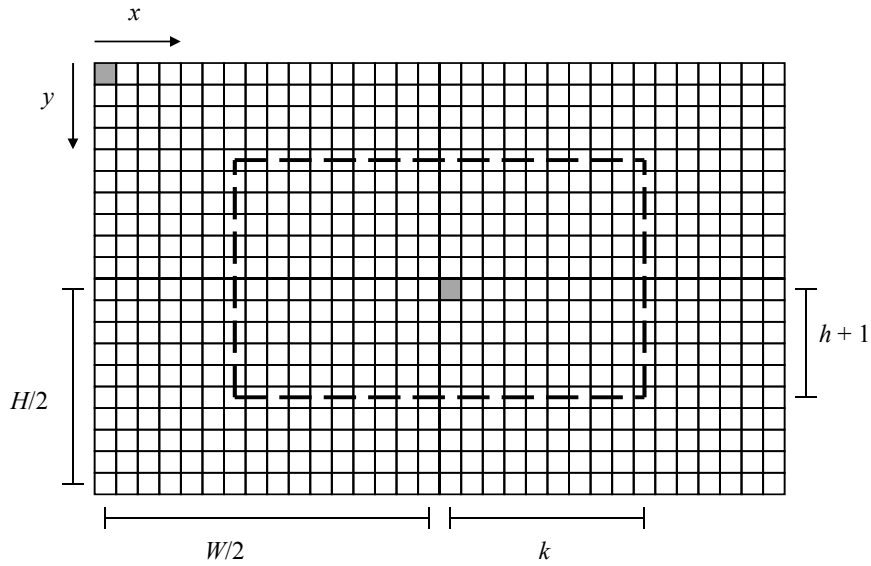


Figure 3.13. Scheme for projection calculation.

Figure 3.13 helps understanding the Equations 3.2, 3.3 and 3.4. In Figure 3.13 the following conventions are used: the image is represented as a grid in which every pixel is represented by a little square, the gray-filled pixel in the top-left corner of the image is the pixel at position $(0, 0)$, the gray-filled pixel at the centre of the image is at position $(\frac{W}{2}, \frac{H}{2})$, the dashed rectangle is the rectangle along which the projection is being calculated. Thus, the four vertices of the dashed rectangle are the pixels at the following positions:

- $(\frac{W}{2} - k, \frac{H}{2} - h - 1)$ for the top-left vertex,
- $(\frac{W}{2} + k - 1, \frac{H}{2} - h - 1)$ for the top-right vertex,
- $(\frac{W}{2} - k, \frac{H}{2} + h)$ for the bottom-left vertex,
- $(\frac{W}{2} + k - 1, \frac{H}{2} + h)$ for the bottom-right vertex.

The variable h is the index of the current value of P and k is the half width of the dashed projection rectangle, h is the half height.

Looking at the equation of the projection calculation, we note that in Equation 3.4 the index i spans from $-h$ to $h - 1$, and not from $-(h + 1)$ to h as expected. This is because in the calculation of the projection we do not want to count the rectangle vertices twice, they are already taken into consideration in Equation 3.3.

To complete the calculation of the projection, the values of $P(h)$ are normal-

ized by the number of pixels involved in each rectangle:

$$P(h) = \frac{P(h)}{4(k+h)}, \quad h = 0, \dots, \left(\frac{H}{2} - 1\right).$$

It is useful to note that scaling an image by a factor s and calculating the projection of it is equivalent to calculating first the projection of the image and scaling the projection by the factor s . It is also obvious that scaling a vector of $H/2$ elements (complexity $O(H)$) is much cheaper than scaling an image of $W \cdot H$ pixels (complexity at least $O(H^2)$).

3.7.2 Search for best match between projections

After the two projections $P_n(h)$ and $P_{n-1}(h)$ are calculated for two contiguous frames n and $n - 1$, these projections are scaled and compared in order to find out whether downscaling or upscaling is present between the two projections.

We make a distinction between *zoom factor* and *scale factor*. Given two images, a base image and a rescaled version of it, the scale factor refers to the ratio between the linear dimensions of one object in the base image and in the rescaled one. For example, if objects become 2 times linearly bigger in the rescaled image, then the image has been scaled (up) by a factor $y = 2$. Instead, if objects become 2 times smaller, the images has been scaled (down) by a factor $y = 0.5$.

We define the zoom factor $z = \log_2(y)$. In this way, scaling up an image by a factor $y = 2$ corresponds to a zoom factor $z = 1$, and scaling down an image by $y = 0.5$ corresponds to $z = -1$. So, the dimensions of an object in a video after two zoom operations with equal duration and opposite coefficients will appear unchanged². Also, $z = 0$ means no scaling up or down ($y = 1$), thus absence of any zoom operation.

In Figure 3.14 the flow diagram of the algorithm that searches for the zoom factor between two consecutive frames is shown. The algorithm tries a number of positive and negative zoom factors. For each zoom factor z , the correspondent scale factor s is calculated with $y = 2^z$. In the flow diagram it is possible to see that, if y is smaller than one, the projection of the *former* frame is upscaled, and if y is bigger than one the projection of the *latter* frame is upscaled. This is done in order to avoid downscaling projections. Downscaling a vector of elements, in fact, results in a vector with fewer elements than the original one, and this is difficult to compare with a vector of more elements.

The function `upscale()` in the flow diagram is defined as:

$$\text{upscale}(F(x), y) = F\left(\frac{x}{y}\right).$$

²Under the assumptions of static rigid objects and no other camera movements

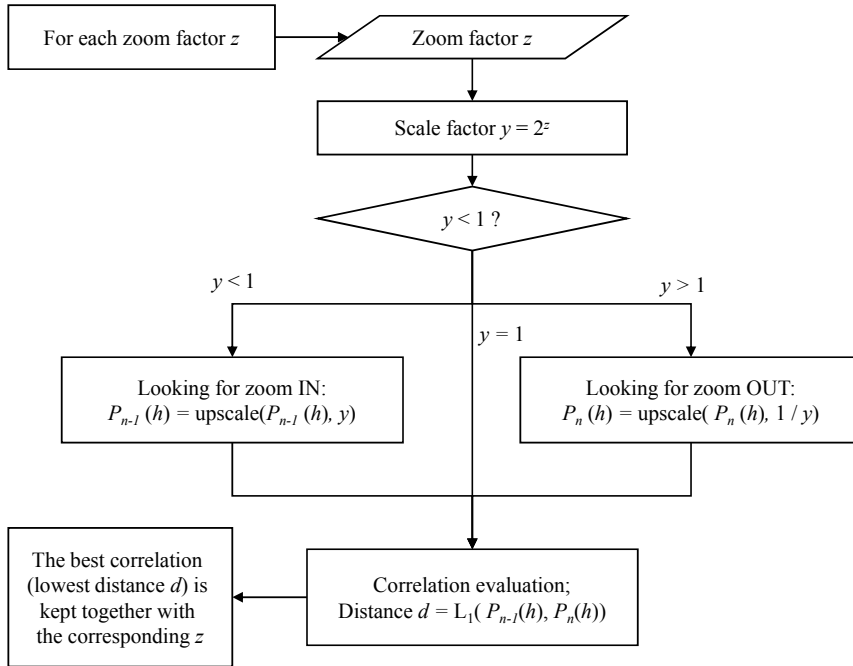


Figure 3.14. Flow diagram of the search algorithm for the zoom factor.

After the upscaling, the projections of the former and the latter frame, P_{n-1} and P_n are compared. To evaluate the correlation between the projections, we calculate the L_1 distance between the two vectors. This distance is defined as

$$L_1(A(x), B(x)) = \sum_x |A(x) - B(x)| .$$

The lower the distance between the projections, the higher the correlation between them. The zoom factor \hat{z}_n corresponding to the minimum distance between projections is considered by the algorithm as the zoom factor detected for the n^{th} pair of frames.

3.7.3 Exploiting the videos' property of continuous optical flow

Till now, the following steps of the zoom detection algorithm have been shown:

- For each frame n of a video sequence, the radial projection $P_n(h)$ is calculated.
- For each pair of contiguous frames, the radial projections $P_{n-1}(h)$ and $P_n(h)$ are scaled according to 540 different scale factors, and compared. Each scale factor corresponds to a possible zoom factor z .

- The zoom factor corresponding to the best match between scaled projections is taken as the zoom factor \hat{z}_n detected for frame n .

The 540 scale factors have been determined considering that the algorithm has to detect very little zoom movements of the camera but also very fast zoom operations. By taking a common camcorder and by measuring the scale factors for the slowest and the fastest zoom movements, the array of 540 scale factors was determined. The detected zoom factor can range from 0.146 (scale factor $s = 1.1$) to -0.146 (scale factor $s = 0.9$).

The algorithm does for each frame in the video sequence 540 operations of projection scaling and comparison, in order to search the zoom factor for each frame. However, in a video sequence every frame is quite similar to the former and to the latter, since objects in a video sequence move according to a continuous optical flow (apart from camera take boundaries, that we assume to know already). Therefore, the zoom factors will resemble each other across successive frames, and the zoom factor \hat{z}_{n+1} can be searched in a small window around the zoom factor of the previous frame \hat{z}_n , instead of trying a window of 540 values for z . Because of this, when the L_1 distance for frame n increases with respect to the one for frame $n - 1$ the algorithm searches the n^{th} zoom factor only among 108 possible values.

3.7.4 Filtering the zoom factors and classifying the frames

We have explained how the algorithm calculates the succession of zoom factors $\{z_n\}$ for a given video sequence. In order to detect only the main zoom movement and to discard the noise, the algorithm filters the succession of zoom coefficients $\{z_n\}$ with a median filter of size 41. The size of the filter and all the other parameters of the algorithm were determined by evaluating zoom detection on a tripod-shot video with clear camera movements. For the median filter, sizes 21, 31 and 51 were also tried, but 41 gave the best results. At this point, the initial problem of distinguishing among the three classes of zoom motion (zooming in, zooming out, no zoom) can be solved.

To each frame n of the video sequence, a class label b_n is assigned according to the following method:

$$b_n = \begin{cases} \text{“zoom out”} & \hat{z}_n \leq -Z_{\min} \\ \text{“no zoom”} & -Z_{\min} < \hat{z}_n < Z_{\min} \\ \text{“zoom in”} & \hat{z}_n \geq Z_{\min} \end{cases}$$

with Z_{\min} empirically set to 0.003.

3.7.5 Refining the zoom detection by exploiting pan and tilt information

One drawback of the zoom detection algorithm lies in the fact that it has been created for the ideal case of a video sequence in which only zoom in or out is

present, and no other type of camera or object motion. Scaling and comparing radial projections is effective only in the case in which successive frames of a video sequence are the linearly upsampled or downsampled versions of each other. In real home videos, however, multiple camera movements can simultaneously occur, for example a pan and a zoom movement can be performed at the same time. In the case of simultaneous pan and zoom, successive frames are not the upsampled or downsampled versions of each other anymore. We have observed that, in the case of multiple co-occurring camera movements, the algorithm tends to over-estimate the zoom coefficients and tends to interpret significant pan and tilt movements as zoom operations.

In order to solve this problem, a possible approach consists in increasing the threshold for zoom classification Z_{\min} proportionally with the module of pan and tilt motion parameters. Therefore, we exploit also pan and tilt information; more precisely, we implement the algorithm for pan and tilt detection published in [Uehara et al., 2004] because it is robust and fast (pan and tilt parameters are estimated in 1/7 of the zoom estimation time).

We calculate the zoom classification threshold $Z_{\min}(f)$ for every frame f in the following way:

$$Z_{\min}(f) = \begin{cases} 0.004 & \text{Trasl}(f) < 16 \text{ screens/minute} \\ 0.02 & 16 \leq \text{Trasl}(f) < 32 \text{ screens/minute} \\ 0.06 & \text{Trasl}(f) \geq 32 \text{ screens/minute} \end{cases},$$

where $\text{Trasl}(f) = |\text{pan}(f)| + |\text{tilt}(f)|$. The threshold values indicated above have been heuristically found using a tripod-shot video, and have been evaluated to give good performances. After this step, the zoom classification thresholds $\{Z_{\min}(f)\}$ are also filtered with a median filter of size 41 since this improves the classification performances.

3.7.6 Benchmarking our algorithm against a state-of-art method

The effectiveness and efficiency of our algorithm have been compared with the state-of-the-art camera motion detection method described in [Varekamp, 2004]. The benchmark has been performed using four home video sequences, in total 37 min and 5 sec of raw home video material. This material was not shot for this benchmark but was real content shot by a user during his vacation. The video was different from the one used to fine-tune the algorithm's parameters, it was shot using a hand-held consumer camcorder and without the help of a tripod. For convenience, we will call our algorithm *PC* (Projection Correlation) and the method of [Varekamp, 2004] *SoA* (State of Art).

Firstly, the test video sequence has been manually annotated with the camera motion parameters, in order to determine the *ground truth* for the benchmark. More

precisely, for each frame f of the test video sequence we have annotated

- the type of pan movement $G_{\text{pan}}(f) \in \{ \text{“pan left”}, \text{“no pan”}, \text{“pan right”} \}$,
- the type of tilt movement $G_{\text{tilt}}(f) \in \{ \text{“tilt up”}, \text{“no tilt”}, \text{“tilt down”} \}$,
- the type of zoom movement $G_{\text{zoom}}(f) \in \{ \text{“zoom in”}, \text{“no zoom”}, \text{“zoom out”} \}$.

Therefore, each frame of the test sequence has three ground-truth annotations; for pan, tilt and zoom movements, respectively.

We have compared the effectiveness of our algorithm PC with the method SoA by measuring to what extent the motion parameters estimated by the algorithms correspond to the ground truth labels. More precisely, we have calculated the precision of PC , P_{PC} , in the following way:

$$P_{PC} = \frac{1}{N} \sum_{f=1}^N \frac{\epsilon(PC_{\text{pan}}, G_{\text{pan}}) + \epsilon(PC_{\text{tilt}}, G_{\text{tilt}}) + \epsilon(PC_{\text{zoom}}, G_{\text{zoom}})}{3},$$

where

$$\epsilon(A, B) = \begin{cases} 1 & A = B \\ 0 & A \neq B \end{cases}.$$

Note that, in order to have full precision the algorithm has to correctly detect all three motion labels for each one of the frames in the video sequence. If, for a given video, the algorithm detects correctly the pan and tilt movements but fails to detect the zoom, the precision will be $2/3$. In a similar way we have evaluated the precision of the SoA algorithm:

$$P_{SoA} = \frac{1}{N} \sum_{f=1}^N \frac{\epsilon(SoA_{\text{pan}}, G_{\text{pan}}) + \epsilon(SoA_{\text{tilt}}, G_{\text{tilt}}) + \epsilon(SoA_{\text{zoom}}, G_{\text{zoom}})}{3}.$$

Table 3.1 shows the results for P_{PC} and P_{SoA} for the four videos used in the benchmark. The four video sequences used for the benchmark contained zoom

Table 3.1. Results of performance benchmark.

Video sequence	Length (frames)	P_{PC} %	P_{SoA} %
Bombay	11890	78.7	73.6
Engagement	19841	82.8	78.1
Life in Cochin	20701	83.0	78.7
Fort Golconda	3202	80.4	80.3
Total	55634	81.9	77.5

in/out for 15% of the video frames (total 8188 frames). This benchmark shows the superior precision of our zoom detection algorithm combined with [Uehara et al.,

2004] compared to the *SoA* method. P_{PC} results in 81.9% meanwhile P_{SoA} equals 77.5%.

The *SoA* algorithm allows performance tuning via some parameters: the precision of *SoA* can be improved at the cost of significantly decreasing the detection speed. We have tuned the *SoA* algorithm to run with the same efficiency as our method: both algorithms performed the camera motion detection at a rate of 30 frames per second (on a custom PC), each frame having dimensions of 720x576 pixels.

Although the overall precision of our algorithm P_{PC} is higher than P_{SoA} , we have observed that the *SoA* method achieves better performance in detecting the zoom parameters. The precision of zoom detection of *SoA* is 93.2%, meanwhile the zoom detection precision of our algorithm is 86.6%. However, the *SoA* algorithm does not allow the detection of the zoom parameters separately from the detection of the translation parameters pan and tilt. The precision of *SoA* for pan and tilt detection is 69.6%, meanwhile the precision of the method we employ is 79.6%.

We have seen that the *SoA* algorithm can be made more precise at the cost of a slower detection. We wanted to check how much additional time the *SoA* algorithm needs to reach the same precision as our algorithm *PC*. Seven different combinations of parameters were tried on the *SoA* algorithm, to let the speed vary from 30 frames per second to 6 frames per second. Table 3.2 shows the precision and the detection speed for each of the combinations of parameters tried. It is possible to see that the *SoA* algorithm never reaches the precision of *PC*.

Table 3.2: Variation of the precision of the *SoA* algorithm when the parameters are set to spend more time in the detection.

Precision	Speed (frames per second)
77.5%	30.6
77.5%	26.0
77.7%	20.5
78.0%	17.2
77.4%	10.1
78.1%	8.2
78.3%	6.0

Table 3.3: List of all symbols used in the chapter.

<i>Symbol</i>	<i>Description</i>	<i>Page</i>
a_n	Number of faces present in frame n	47
c_i	i -th camera take	42
$\mathcal{C}(\mathcal{V})$	Camera take segmentation of video \mathcal{V}	42
E	Edited video	44
f_n	n -th video frame	41
$G(f)$	Ground-truth annotation of camera motion for frame f	71
H	Height of video frame in pixels	65
$I(x, y)$	Gray-scale image	65
l_n	Average luminance in frame n	47
m_j	j -th missing part in the edited video	58
n_v	Duration in frames of video segment v	55
N	Number of video frames	41
N_C	Number of camera takes	42
N_I	Number of search iterations in simulated annealing	62
N_M	Number of the missing parts	58
N_R	Number of maximum improvements in simulated annealing	62
N_S	Number of shots	44
N_V	Number of video segments	43
o_n	Average contrast in frame n	47
p_n	Amount of camera pan for frame n	47
\tilde{p}_n	Filtered pan for frame n	55
$P(h)$	Radial projection of a video frame	66
P_{PC}	Precision of ‘‘Projection Correlation’’ algorithm	71
P_{SoA}	Precision of ‘‘State of art’’ algorithm	71
R	Video framerate	41
s_i	i -th shot	44
s_v	Start frame of video segment v	55
$S(v)$	Amount of shakiness in video segment v	55
S_{MAX}	Constant for shakiness calculation	55
$\mathcal{S}(\mathcal{V})$	Segmentation in video segments	43
t_n	Amount of camera tilt for frame n	47
\tilde{t}_n	Filtered tilt for frame n	55
T	‘‘Temperature’’ control parameter in simulated annealing	60
T_L	Threshold for luminance classification	50
T_C	Threshold for contrast classification	50
\mathcal{T}_k	Timestamp associated with first frame of camera take c_k	42
v_i	i -th video segment	42

Table 3.3 – Continued

<i>Symbol</i>	<i>Description</i>	<i>Page</i>
\mathcal{V}	Unedited home video	41
w_{ζ}	Weight of the suitability (user requirement)	57
w_{τ}	Weight of the temporal uniformity (user requirement)	57
w_{χ}	Weight of the content score (user requirement)	57
W	Width of video frame in pixels	65
y	Scale factor between one frame and the next	68
z_n	Amount of camera zoom for frame n	47
Z_{\min}	Threshold for zoom classification	70
$\chi(E)$	Content score of video E	45
$\delta(x)$	Time duration, in seconds of the video item x	43
Δ_E	Total duration of the edited video E (user requirement)	57
Δ_s	Average shot duration (user requirement)	57
Δ_s^{\min}	Minimum shot duration (user requirement)	57
$\Omega(E)$	Objective function	45
$\rho(v)$	Suitability score of segment v	55
$\rho_l(v)$	Luminance quality of segment v	55
$\rho_c(v)$	Contrast quality of segment v	55
$\rho_S(v)$	Shakiness quality of segment v	55
$\zeta(E)$	Suitability score of video E	45
$\tau(E)$	Temporal uniformity score of video E	45

4

Edit While Watching I

In Chapter 3 the algorithm for automatically editing the user's unedited video was explained. The edited video produced by the algorithm is then refined by the user via successive iterations through an interface, as is shown in the flowchart in Figure 3.1. This chapter describes this interface, mentioning the underlying motivations, the user requirements, the novelties with respect to the state of the art, and the functioning. Furthermore, a user evaluation of the interface is discussed.

At the time of developing this interface, we were still running the Internet-based survey described in Section 2.3.2. Therefore, we had not yet distilled the user needs presented in Section 2.3.4. Because of this, we started from the assumptions about the user needs as proposed by Lienhart and by the rest of the literature. We focused on the need for ease of use and need for efficiency.

To address the aspect of ease of use, we think that all unnecessary technical details should be hidden from the user. To edit videos, the user should reason about the video's semantics (the wedding scene, my friend swimming, the dog running), not about technical elements of the video (frame numbers, image luminance, etc.). To address the need of quick editing, all editing operations should happen in real time: they should take effect right after the user activates them. If, for example, the user adds content to a wedding scene, the newly added content should be played back to the user right away. This gives users the feeling of editing video while watching it, without having to wait for rendering or encoding processes.

4.1 *Edit While Watching* system

In [Campanella et al., 2007] we have presented our first solution for home video editing, *Edit While Watching (EWW)*, that provides easy interaction and *real-time* editing functions to edit video while watching it on a TV. *Edit While Watching* brings the following novelties with respect to the state of the art:

- *Interacting with EWW requires only a TV and a remote control.* EWW is not PC-based, therefore users do not need particular PC skills to edit videos. Furthermore, users can edit their videos sitting on the living room’s couch, alone or with others. The automatic algorithms of EWW can be implemented on a media center, or a hard disk/DVD recorder or on another suitable device.

- *EWW implements semi-automatic editing functions:* the system helps the user in selecting which video content to keep or to discard. More precisely, the user can just say that he/she wants “more of” the currently viewed content, and the system automatically decides which content to add to the video the user is watching. In the literature only the Silver system [Casares et al., 2002] implements semi-automatic functions for the selection of interesting clips out of a home video; these functions are based on the audio transcript. They work, however, only when speech occurs in the video.

- *All editing happens in real time.* After the user has activated an editing function, the video is immediately played back in correspondance of the edited point, to give immediate feedback on the performed operation.

- *EWW is designed to hide unnecessary technical details* and to obtain the maximum simplicity, while still providing users with a rich set of options for editing their videos. With EWW, the user does not need to browse a timeline to access the video. Knowledge about video frames, codecs or filters is not needed to edit videos.

Edit While Watching aims at finding a convenient balance between fully manual editing tools like Adobe Premiere¹ and fully automatic systems like Muvee². To obtain more simplicity, and to increase the frequency and convenience of editing home videos, video editing is moved from the PC to the TV and remote control in the living room. While striving for simplicity, we also want to provide the user with sufficient options to edit the video according to his or her taste. *Edit While Watching* includes functions for selecting interesting video content and discarding unwanted content, adding music and adding video effects such as sepia or posterization.

¹<http://www.adobe.com/products/premiereel/>, April 2009

²<http://www.muvee.com/en/>, April 2009

4.1.1 Overview of the system

Like most automatic and semi-automatic systems described in literature, EWW automatically analyzes, segments and summarizes the unedited video the user provides. After the initial summary is composed, it is played to the user on the TV screen, and the “editing while watching” can start. Note that users are not aware of video segments: they see only a video divided in shots (see Figure 3.2 on page 43 for the relationship between shots and video segments). Users, in fact, recognize the presence of a shot boundary when the video optical flow is interrupted, but video segments are not as such explicitly shown.

Figure 4.1 shows a screenshot of the EWW user interface. During the playback, some navigation functions are available: in Figure 4.1, two arrow buttons are displayed in the bottom left and bottom right corners of the screen. The user can activate them by pressing the left and right arrows of the remote control. With these arrows, the user goes respectively to the previous and to the next shot.

While the user is playing the video, at any time he or she can press the “pause” button on the remote control and open an editing menu, as shown in Figure 4.2. The menu appears in the bottom part of the screen. Via this menu, users can choose one out of seven editing functions. After the selection of a function, the system displays textual feedback on the operations performed and subsequently plays the edited part of the video. In this way users can immediately notice what the effect of the selected editing operation is, from the change in the video and from the textual feedback. The functions of the menu are now described, from left to right.



Figure 4.1. Screenshot of EWW during playback.

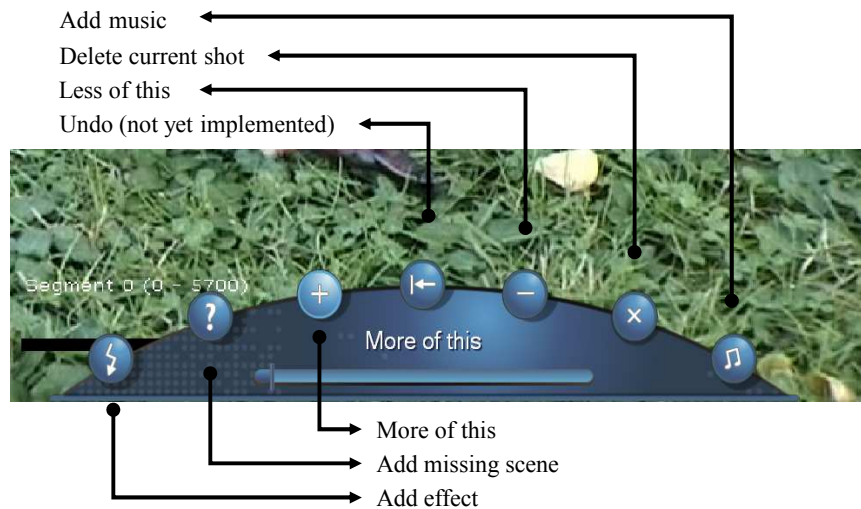


Figure 4.2. The editing menu of EWW in the bottom part of the screen.

- **Add effect.** By pressing this button, a list of possible video effects is presented: black and white, sepia, posterization, etc³. The user can choose one of the effects which is immediately applied to the current shot. The playback resumes by showing the adapted current shot.

- **Add missing scene.** This button shows a menu with thumbnails of the shots that have been *excluded* from the current video summary. Figure 4.3 shows a screenshot of the menu accessible via the “add missing scene” button. The overview of the missing shots is given by a panel with thumbnails; these are browsable with the four arrows of the remote control. While the user is browsing the thumbnails of the missing shots, the one that is currently selected plays one second of the corresponding video. The user can select one of the removed scenes that is then added again to the edited video.

- **More of this.** After pressing this button, the system understands that the content of the current shot is important to the user and adds more of it to the video summary. The system increases the duration of the current shot, at the beginning or end of the shot, by adding the neighboring video segments with the highest suitability score. “More of this” is a *semi-automatic* function, its precise functioning is clarified in Section 4.1.2.

³The full list of video effects available in EWW is: sepia, over exposure, emboss, black and white, twisted, invert colors, wave, pixelate, reduce colors, sunset light.

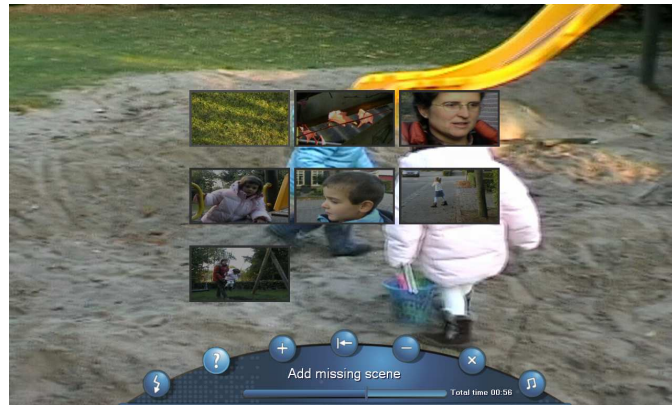


Figure 4.3. Screenshot of the “add missing scene” menu.

- **Undo.** Cancels the last editing operation.
- **Less of this.** This is the operation opposite to “more of this”. By pressing this button, part of the current content is excluded from the video summary. The system shrinks the current shot by removing the video segments with the lowest suitability score, at the beginning or at the end of the shot. “Less of this” is a *semi-automatic* function, its precise functioning is clarified in Section 4.1.2.
- **Delete current shot.** By pressing this button, the current shot is completely excluded from the edited video, and can be added again via the “add missing scene” function.
- **Add music.** Via this button the user can select a music track from a list of available tracks. The selected track is added to the current shot, and the shot is played back to the user to show the change in the audio track right away.

The functions “add music” and “add video effect” allow the user to *enrich* his or her video, meanwhile the buttons “more of this”, “less of this”, “add missing scene” and “delete shot” are needed for including/excluding portions of raw video material to/from the edited video. For evaluation purposes, we have implemented a working demo of the *Edit While Watching* concept, using programming languages convenient for video processing and development of interfaces (C++, C#, DirectX). Only a few buttons on the remote control are needed to completely control the interface: *play*, *ok*, *pause*, and the four arrows.

4.1.2 Semi-automatic functions for video editing

In the former section, the editing functions of *Edit While Watching* have been described (see Figure 4.2). This section explains in detail the two semi-automatic functions “more of this” and “less of this”. The idea behind these two functions is

the following: the system helps the user in selecting the video content to keep or to discard. More precisely, the user can just say that he/she wants “more of” the currently viewed content, and the system automatically decides which content to add to the video the user is watching. In implementing the semi-automatic functions, we have hypothesized that introducing automation in the process of content selection will make the editing process easier and faster for the user.

Figure 4.4 shows, in a flowchart, how the function “more of this” works. First, the system considers the shot s that is currently being played. The system looks whether it is possible to add some of the neighboring video segments to the shot s . Figure 4.5 shows an example: the shot s is formed by the video segments in dark gray; to this shot it is possible to add some of the video segments in light gray. The addition of video segments can be performed either in the back of the shot or in the front. In the flowchart of Figure 4.4, the systems checks whether it is possible to add video segments before the beginning or after the end of the shot s . Looking before the beginning of the shot, for example, the system checks whether there is some excluded video belonging to the same camera take. If there is, the system adds at most three seconds of it (the system adds less if less is available). The added video cannot contain camera take boundaries, otherwise there may be too short shots.

Four cases can occur: no video is found to add to the shot, the shot is extendable only in the front, only in the back, or in both directions. In the latter case, the system selects to add the video segments with the highest suitability score. Then, the shot is extended in the direction the system has decided and a feedback message is given to the user. The user does not have control on the direction the shot is extended and the duration of the added part. On the other hand, the user has an easy way to obtain more of a certain video content.

The function “less of this” works in a similar way. It considers the current shot s and it checks whether it can be shortened in the front, in the back or in both directions. The shortening of the shot should not reduce the shot duration below two seconds. Too short shots, in fact, are considered to be aesthetically unpleasant since the user sees two shot cuts very quickly after each other. If the shortening is possible from both sides, the video with the lowest average suitability score is excluded.

4.1.3 *Edit While Watching* and the elicited user needs

In Section 2.3.4 we have presented the results of our study about user behaviors and needs related to home video editing. These results are summarized in the seven user needs on page 35. Although EWW was designed before the analysis of the data of the questionnaire had been finished, we think that EWW addresses at least five of the seven user needs. EWW automatically analyzes and summarizes the

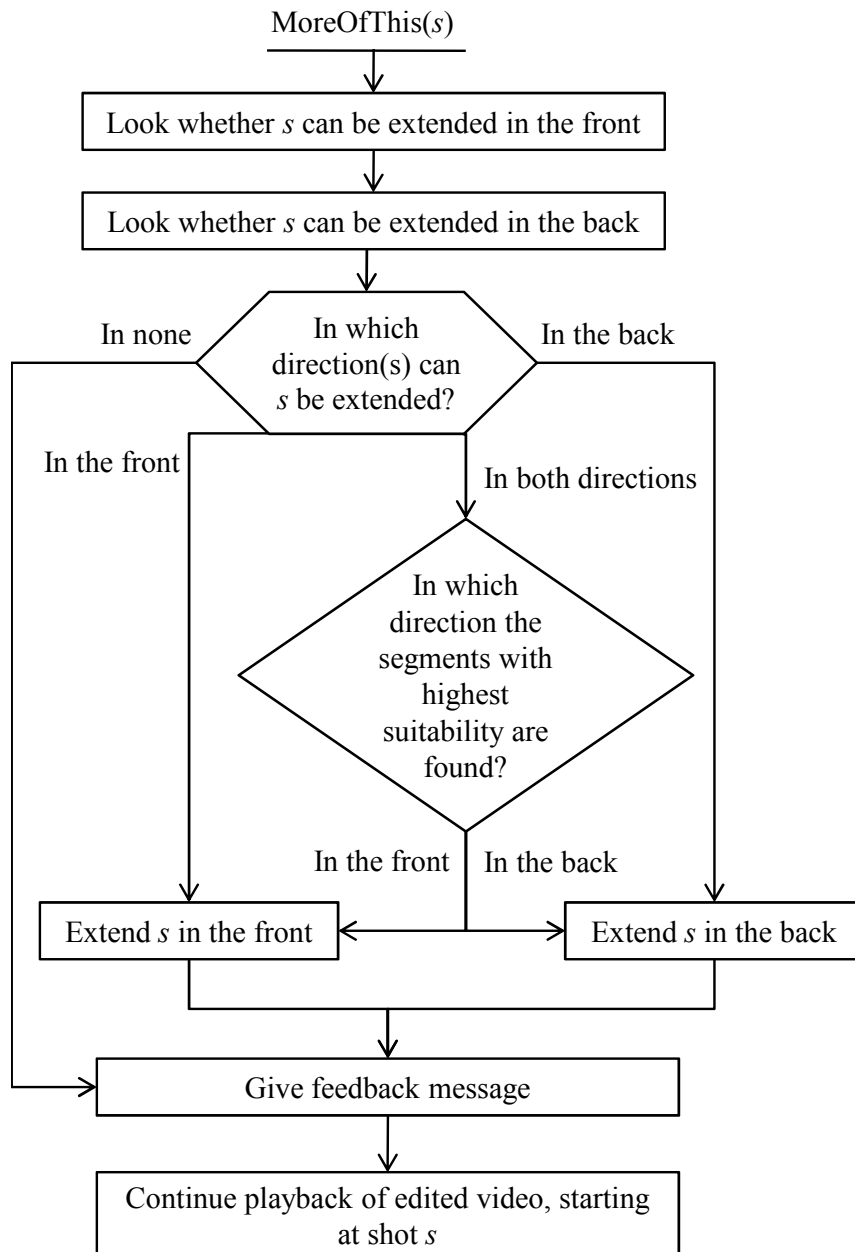


Figure 4.4. Flowchart of the algorithm for the function "more of this".

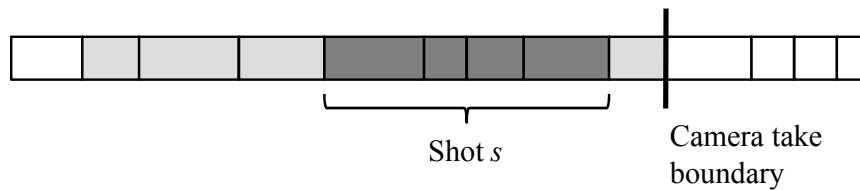


Figure 4.5. Visualization of possible video segments to add to the shot s by the function “more of this”.

input video, addressing need 1 (need for video summarization) and 3 (efficiency). EWW supports the user editing the video according to his or her wishes, addressing need 2. EWW provides video editing on a TV, addressing need 5. Furthermore, in Section 2.3.3 we have shown that people with low acquaintance with computers edit videos much less frequently. With EWW users do not need any computer skills to be able to edit videos. The users can edit their videos sitting comfortably on the couch of the living room, instead of concentrating in front of a PC (need 4). EWW’s semi-automatic editing functions are designed to provide more ease of use and editing speed (needs 4 and 3), at the expense of the control on which video parts are edited. For these reasons, we think that *Edit While Watching* can be an interesting tool for the users.

The user needs 6 (combining video with content of third parties) and 7 (social-communicative role of home video) are not addressed. We think that addressing these needs properly would require several additional research steps (exploring in more depth the relation between home videos and communicative needs, investigating interfaces for collaborative editing, etc.). Therefore, in order not to increase the complexity of the problem, we investigated in the first place whether our system addresses sufficiently the first five needs. In the next sections we discuss a use test we designed and performed to assess the usability of *Edit While Watching*.

4.2 Use test on *Edit While Watching*: objectives

This user experiment has been described in [Weda and Campanella, 2007]. The objective of the use test was to assess the usability [ISO/IEC, 1998] of EWW’s functionalities for video editing. Concerning the usability, we evaluated the learnability of the system, and the user satisfaction with the editing functions and the system’s feedback. We focused on evaluating the ease of use of EWW. We also collected data about the effectiveness of EWW. We wanted to perform a formative evaluation to find the main possible defects of EWW and to know how to improve our design.

4.3 Test design

To answer the mentioned research questions, we decided to set up a test with eight participants. Eight users are enough to quickly find the main flaws in the usability of EWW [Nielsen and Landauer, 1993] and to know how to improve our system. Participants in our test should have at least a basic acquaintance with filming home videos, since this helps them in having a better idea about what they would expect from a video editing system. We recruited participants with at least a basic experience in capturing videos among the students and employees at Philips Research Eindhoven. This could have introduced a bias in the test results, because all participants have a technical education and they could be more open in trying out and learning new systems. On the other hand, all the non-technical people we initially approached had to be excluded due to lack of video capturing experience.

4.3.1 Test setup and structure

Since the EWW system is designed to be used in a home setting, we selected the Philips ExperienceLab as test environment [Aarts and Diederiks, 2006]. The ExperienceLab contains a living room with video cameras that recorded the test for later reference. A remote control and a TV set were used for the interaction. A picture of the setup can be seen in Figure 4.6.



Figure 4.6. A picture of the test setup in the ExperienceLab.

We structured the test in three parts:

- In the first part, the user received a short explanation of the test structure; subsequently he/she was interviewed about the behaviour and experiences with respect to capturing, watching and editing home videos. These initial interviews helped understanding the users' profile.

- In the second part, the participants tried and used *Edit While Watching*. All the participants tried EWW on the same video material: the recording of a visit to a zoo. Each user received a short, standardized explanation of the system and of the editing tasks to perform. The explanation of the system and of the tasks was written on one A4 sheet, reproduced in Appendix B. This paper reported a short explanation for each of the buttons in EWW's menu, and the tasks to be performed by the user. The clarification that the test monitor gave to each participant was very short. The user was told which keys on the remote control were needed to use the system (play, pause, ok and the four arrows), and to learn the system by trying its functions. The user was not told the effect of the arrows left and right (going to the previous and to the next shot respectively). The editing functions were not shown either. After the initial explanation, the A4 paper was given to the participant, so that he/she could refer to it while using the system. The explanation of EWW was on purpose very limited, the users were in fact invited to try out the system by themselves and play around with it until they would consider themselves confident with the functionalities. Each user could spend as much time as considered necessary to get acquainted with EWW. We considered this to be quite realistic, since very few users spend time reading the manual of the new technology they buy, most people prefer to learn the tools by trial and error. Since the beginning of the experiment, participants were invited to think aloud. When the participant felt to have gained enough confidence with EWW, he or she was invited to perform four editing tasks, consisting in including or excluding specific clips from the edited video (for example adding more of the "panthers" or less of the "bears"). After executing these tasks, the user had as much time as he/she desired to edit the video according to his/her wishes, exploiting all the functionalities of the system. If necessary, the user was reminded to think aloud. Moreover, the system logged all the keys that each participant pressed during their experience and all the editing operations performed on the video, for post-test analysis. Due to time restrictions, the "Undo" button had not yet been implemented at the moment of testing.

- In the third and last part of the test, the users were asked to give their feedback on EWW in a semi-structured interview. The users were asked to fill in a questionnaire on the functionality, usability and satisfaction with EWW. The questionnaire was composed of Likert-scale questions and is reported in Appendix B. The users were asked to explain the motivation behind their evaluations expressed

on the Likert scales; they were also invited to make any comment not already expressed in the interview.

4.3.2 Video material

As explained before, we decided to use the same video material, which consisted of a home video of a visit to the zoo, for all the participants. The advantage of the fixed video material is that we eliminate the variability that comes from the participants bringing their own videos, with different durations and different wishes for editing, and that we evaluate the system's usability in carrying out specific editing tasks to perform on a video we know. This allowed us to evaluate the system with respect to particular tasks. The underlying assumption is that we know and understand how the system will typically be used. A disadvantage is that the test could have seemed artificial to the participants: they may not feel involved with the provided material, since it is not their own. As a consequence, we cannot precisely investigate what the user needs are in editing the users' own material. However, from the test we found that the users considered the task scenario realistic.

The unedited home video used for the test lasted 46 minutes, and was divided in 77 camera takes. Our analysis program partitioned it in 1541 video segments. These segments lasted on average 1.8 seconds, range (0.8 - 5 seconds). Concerning pan movements, 431 segments were classified as pan left, 383 as pan right, and 727 as pan still. Concerning tilt movements, 243 segments were labeled as tilt up, 188 as tilt down, and 1110 as tilt still. Regarding zoom movements, 387 segments contained zoom in, 352 zoom out, 802 no zoom. All video segments were classified to contain a sufficient level of luminance, and only one segment was labeled as containing poor contrast. The number of segments with faces was 34. The average suitability score of the video segments was 0.784, the standard deviation 0.055. The suitability scores ranged from 0 to 1.

4.3.3 User profile

Eight students and employees from Philips Research in Eindhoven, with some experience in home video capturing, participated in the test: three females, and five males. None of the participants were working on related topics. Their age varied between 22 and 29, most of them captured video four or five times a year (range 2-100 times a year). When capturing video, six of the eight participants (6/8) used a digital still camera, or occasionally a camcorder (5/8). All of the participants owned a digital still camera with video capturing functionality, half of them owned a cell phone with video capturing functionality, and only two own a camcorder.

The captured video was rarely or never edited (7/8). Adobe premiere was the most frequently used program for editing (2/8). When editing was done, it took a lot of time, one hour to one day for one minute of edited video. Some participants

had no video editing experience at all (3/8). After capturing, the video was mostly watched directly with friends, and only rarely shared or watched later.

4.4 Results

4.4.1 User remarks

The participants were instructed to think aloud during the test. During the experience with the system and the final interview, users also produced a number of free comments and remarks on the advantages and the deficiencies of EWW. Their comments and actions have been written down, and were recorded on video. Based on these records, the following results can be noted.

Many users found the “more” and “less of this” functions inadequate for their needs and difficult to understand. Five over eight users would like to be able to decide whether to shorten or expand a shot from the beginning or from the end; they did not like the system to decide for them. Three out of eight users would like to select by how many seconds to shorten or to grow a shot, again they were not always satisfied with the automatic choices made by the system. Lastly, six users found the effect of “more of this” or “less of this” on the video hard to understand. These comments were made generally, but became most apparent in the practical task where a specific shot had to be shortened. Since after each “less of this” operation, this specific scene shortened only by a few seconds, the participants had to use this function many times to reach an acceptable result.

The system proved to have easy and effective navigation functionalities, all users but one were able to navigate among the shots by using the arrows. One user did not find out that moving between the shots by using the arrows was actually possible. Furthermore, four out of eight users commented that fast-forward and rewind functionalities are missing and would be needed. Three users complained about the lack of an overview of the whole raw footage, for example a key-frame panel or a timeline. They were, in fact, unfamiliar with the raw material and had difficulties to get acquainted to that using only the “add missing scene” functionality.

The “add missing scene” functionality provided an overview too limited to easily find a missing shot. One of the practical tasks consisted of adding a specific shot to the summary that was not yet there. Five users expressed their difficulty in finding this specific shot.

Concerning the video enriching functionalities, seven users found the music aligning function not satisfactory, or not fully satisfactory. They needed a finer scale of editing to add music and effects at a particular point in a shot. Furthermore, the music was inconsistently aligned to the shot: when a specific piece of music was added to shot *A*, the music continued playing in the next shot (*B*). However,

when jumping from shot *A* to shot *B* using the arrow keys, the music stopped playing in shot *B*. This behaviour is clearly inconsistent, as was also noted by most participants. The “add special effect” functionality was well understood and easily activated by every user; however four users were disappointed by the lack of a function for adding video transitions (like fade in/out or dissolve) between different shots.

As general remark, six users were willing to use EWW or an improved version of it to edit their own video material. Everybody agreed that EWW is easy to learn and use, however four users clearly said that the system was too rigid and that they would like to have more control on the parameters of the editing operations and more editing functionalities. Furthermore they would have liked to personalize the final video by adding titles or text. Table 4.1 summarizes the most frequent user remarks.

Table 4.1. Most frequent user remarks.

Remark	N. users
Music alignment with video is weird	7
I do not understand the effect of more/less of this	6
I would like to use EWW or an improved version	6
I cannot choose whether to edit a shot at the beginning or at the end	5
I want fast forward/fast rewind	4
I would like video transitions	4
I would like more control on the editing	4
I want to select by how much to shorten or grow a shot	3
I miss an overview on the video	3

4.4.2 Log files analysis

On average each participant used the system for 31 minutes. The whole test, including interviews, had a maximum duration of 60-90 min. Participants performed 2.1 editing operations per minute; the navigation operations (arrow left, arrow right, play and pause) were not logged. The system crashed 0.9 times on average per test. Although the editing operations were lost when the system crashed, the participants were not very upset by this behaviour. The system crashed more often when users were pressing buttons very fast after each other.

From the log files it emerged that often the users needed to repeat many times consecutively the same operation to achieve the desired result. For the task of shortening the bear sequence, for example, users employed on average almost 9 consecutive “less of this” operations, with a maximum of 18. For lengthening the geese shot, users employed on average more than four consecutive “more of this”, and for lengthening the penguins sequence they used more than three consecutive

“more of this”. On average, users used five consecutive times the same operation for these tasks. This shows that options would be needed to let the users choose by how much they want the shot to be lengthened or shortened. The two most used editing functions were “more of this” and “less of this”. These were used more than twice as often as any other operation (see Figure 4.7).

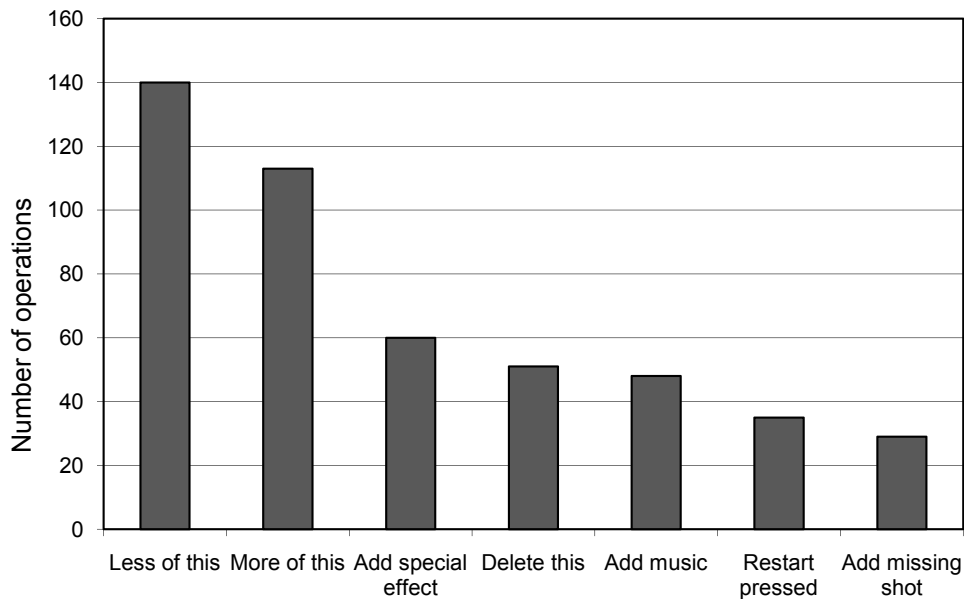


Figure 4.7. All the editing operations performed by the users during the test.

By analyzing the log files, the time users took in carrying out the various parts of the experiment was measured. The users employed on average nine minutes (range 6 - 16 minutes) to get acquainted with the system and to feel ready to start the tasks. These tasks were completed on average in 14 minutes (range 10 - 18 minutes). The average time in which the users edited the video using their own creativity was on average 8 minutes (range 3 - 15 minutes). The time range the users spent on the different parts of the practical test is rather large. We have observed that the participants used a personal style in dealing with the system. Some participants started with carefully exploring the system before beginning the four tasks. Other participants concentrated on doing the tasks quickly and efficiently. Participants behaved very differently also while editing the video according to their personal wishes, as the rather large spread of the free editing time suggests. Some of them tried out extensively all the video effects and music combinations, others focused on few effects on some shots for a shorter time.

Furthermore, the average time dedicated to the creativity was about as long as

the time spent on getting acquainted with the system. This phenomenon, combined with the results from the questionnaire, suggests that the users liked the system and enjoyed trying it out and spending time on it.

4.5 Discussion

We performed a user test study on the *Edit While Watching* system. The participants regarded the use and task scenario as realistic. They judged the system as an easy and fast video editing tool. It took the users a short time to try and learn all the functionalities implemented in the system. With regard to this aspect, the requirements of easiness and simplicity of EWW are met.

However, most users did not feel to be in control of the system. The lengthening and shortening of scenes seemed random to them, and they could not control the start and end of the added music precisely enough. Furthermore they missed an overview of the video. It was regarded difficult to easily see what and where certain parts were included, and what was left out. This could also be caused by the fact that the video was not their own, and consequently they were not familiar with all the content. Still the system was judged as reasonably “fun” to work with. The users liked the system and liked to try it out and spend time on it. This positive feedback from the evaluation has inspired the use of *Edit While Watching* as a paradigm in other testing environments. For example, *Edit While Watching* has recently been adapted and implemented as a PC application. In this form, it was included in the Simplicity Lab⁴ environment of Philips, a web-based usability research facility, as test case to validate online usability research methodologies.

Since the participants had a technical background and were highly educated, there is the danger of bias in the results. Participants with such a profile are typically fast learners and easy adopters. Furthermore they may tend to accept fewer surprises and unpredictability, and would like to have more overview and control. On the other hand, all non-technical people we have initially approached for the test were excluded due to lack of video capturing experience. Still, we think that the main deficiencies found in EWW are so critical (lack of effective overview on the video material, rigidity and slowness of the editing functions) that they would have been found also by non-technical users.

In the design of video editing systems, there is a trade off between the ease of use, and the amount of user control. When the system allows the user to edit the video at frame level by offering the possibility to set parameters per frame, the system becomes difficult to learn and use. While a fully automatic system is easy to use (for example, because only clicking on a button is needed) the user cannot control all details of the editing. With respect to these points, an optimal video

⁴<http://www.simplicitylabs.net>

editing system should try to optimize the balance between ease of use and user control. Thus, in an improved system, the ease of use, and ease of learning of *Edit While Watching* should be carefully preserved, while the user control and overview should be improved. In the next chapter we will explain how we dealt with these issues.

5

Edit While Watching II

In the previous chapter we described a use test on our video editing tool, *Edit While Watching*. This test left us with some issues to solve: how to design a video authoring system with more editing control and a more effective overview on the video material, keeping at the same time the ease of use and learnability of EWW. In this chapter, based on [Campanella et al., 2009], we explain how *Edit While Watching* was improved. A second version of *Edit While Watching* was created, developed and evaluated.

5.1 Mockups of new interaction paradigms

To overcome the pitfalls of *Edit While Watching*, we initially came up with three new concepts of video editing interfaces. We built mockups for each one of the concepts, with the purpose of showing the mockups to users and getting feedback on them. We created the following concepts:

(1) The “*scene*” paradigm is shown in the upper half of the screenshot in Figure 5.1(a). The system automatically divides the video material in a convenient number of scenes (eight in the example). A *scene*¹ is a part of the video about the same place, time and event. The video is divided into scenes according to its semantic

¹In Chapter 4 the word “scene” was used as a synonym of “shot” (e.g. “add missing scene”). From now on, the term “scene” will be used to refer to a set of shots.

structure. For example, in a wedding video, the first scene could be the arrival of the bride; the second, the exchange of rings; the third, the reception; etc. The number of scenes is calculated to make sure that the scenes are enough to represent the whole video, but not too many, so that the user's attention is not overloaded. Each scene is represented by a thumbnail inside a blue rectangle. The user has therefore a compact overview of his/her video, organized in a sequence of semantically relevant video scenes. The user can play the video, browse the scenes and edit them with high-level operations like "more of this scene", "less of this scene" or "delete scene". This paradigm is designed to improve the user's overview of the video material.

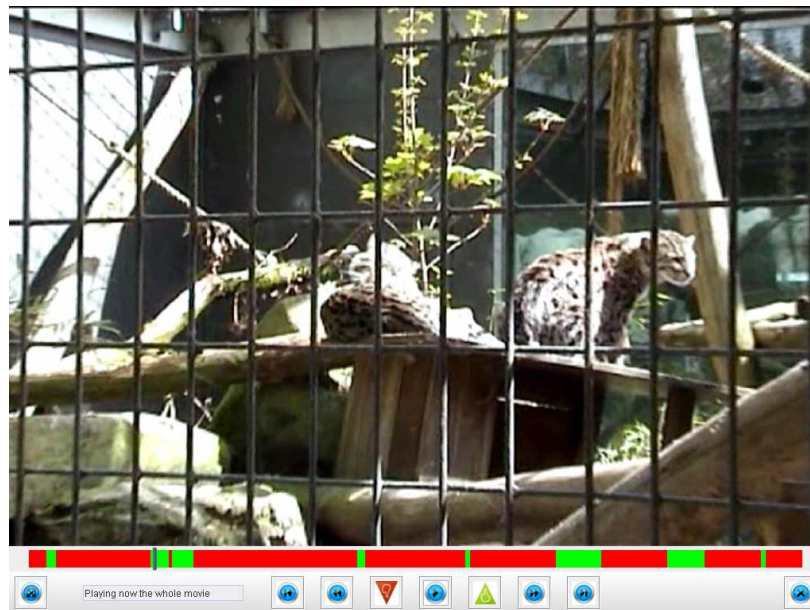
(2) The "*blunt-ended bar*" paradigm is shown in the lower half of the screenshot in Figure 5.1(a). Here the user is provided with a finer type of overview, consisting of small segments of video drawn as blunt-ended rectangles ordered on a time-bar. Each rectangle contains a thumbnail that represents the video segment. These video segments are smaller entities than the scenes: they are short clips of 1-5 seconds belonging to a scene. Video segments included in the edited video are drawn above the others, like the third blunt-ended rectangle from the left in Figure 5.1(a). By moving a time cursor around, the user can do fine editing operations like cutting a video segment in smaller pieces. These pieces can be included or excluded in the edited video. This interaction paradigm can be imagined alone or in conjunction with the scene paradigm. In the latter case the user can browse between two levels of overview: the overview of scenes and the overview of the video segments inside one scene. This would increase the overview's effectiveness. In this paradigm users have full control on the video segments edited: cut points can be decided at frame detail by moving a cursor along the timeline.

(3) The "*cutting while watching*" paradigm is shown in the screenshot in Figure 5.1(b). While the video is played, a timebar is shown in the bottom of the screen. Video segments included in the edited video are represented as green portions of the timebar, while red portions correspond to excluded video segments. The user can play, pause, fast-forward, fast-rewind the video and jump across the green (included) video segments. To edit the video, the user can press the green thumbs-up button or the red thumbs-down button. The editing happens by keeping pressed these buttons on the remote control. If the user keeps pressed, for example, the green button, the video starts fast-forwarding and the portion of timebar scanned by the cursor becomes green (included). When the user wants to stop including video material, he or she releases the green button. The red button has the effect of excluding video, in the same way. With "cutting while watching", the user can quickly include or exclude parts of the video.

We showed these three mockups to seven of our colleagues, with experience



(a) Mockups of “scenes” and “blunt-ended timebar” paradigms.



(b) Mockup of “Cutting While Watching”

Figure 5.1. Three mockups to explore different overview and editing paradigms.

in editing home videos or expertise in human-computer interaction. We got feedback from them via semi-structured interviews. None of the three mockups were clearly preferred by our interviewees for editing their home videos; some people preferred the scene overview in combination with cutting while watching, others the scene overview together with the blunt-ended timebar. Other respondents preferred “cutting while watching”, others the scene paradigm alone. All of the presented paradigms were considered to be appealing for editing home videos. Moreover, from these interviews it was evident that the user requirements that users see as important for home video editing are very diverse and depend on the particular user, on the video content and on the purpose/occasion. This confirms the findings of Kirk et al. [2007] in their study about users’ behavior with home videos: for some users, it is important to make a precise selection of the most interesting video material; for some others it is important to add extra information like text, music or pictures. Therefore, a home video editing system should have a rich set of functionalities in order to address many user wishes, requirements and scenarios.

After these interviews, we decided to try to design a concept that would include the strong points of each one of the three paradigms, and try to overcome all the weak points. The new system had to address the requirements of the old concepts:

- to provide an effective overview,
- to provide high editing control,
- to be easy to use,
- to be efficient,
- to be easily learnable,
- to be easily usable via a TV and a remote control.

Furthermore, the new system had to address new requirements which became evident from the comments on the mockups:

- to support a scalable overview for effective video browsing and navigation,
- to support both basic and fast editing operations and advanced and precise ones, in order to meet a vast range of user requirements.

To address these design requirements, we thought of organizing the navigation and editing functions in different *modes* and having *dedicated functions* to switch between the modes. In this way, the user is able to choose the mode that best serves the particular task he or she wants to carry out. As a result, we created the second version of *Edit While Watching*, which is described in detail in the next section.

5.2 The second version of *Edit While Watching*

5.2.1 Three modes of overview and editing

In the previous section we provided motivations for the development of the second version of *Edit While Watching* (EWW2). In the present section we explain in depth the interaction paradigm devised for EWW2. Most of the concepts and algorithms of the first version of EWW (EWW1) are included in EWW2. In both applications the user uploads his or her video material to the system, the system analyzes it, segments it and produces a first video summary, as described in Chapter 3. Then, the user refines this video summary while watching it.

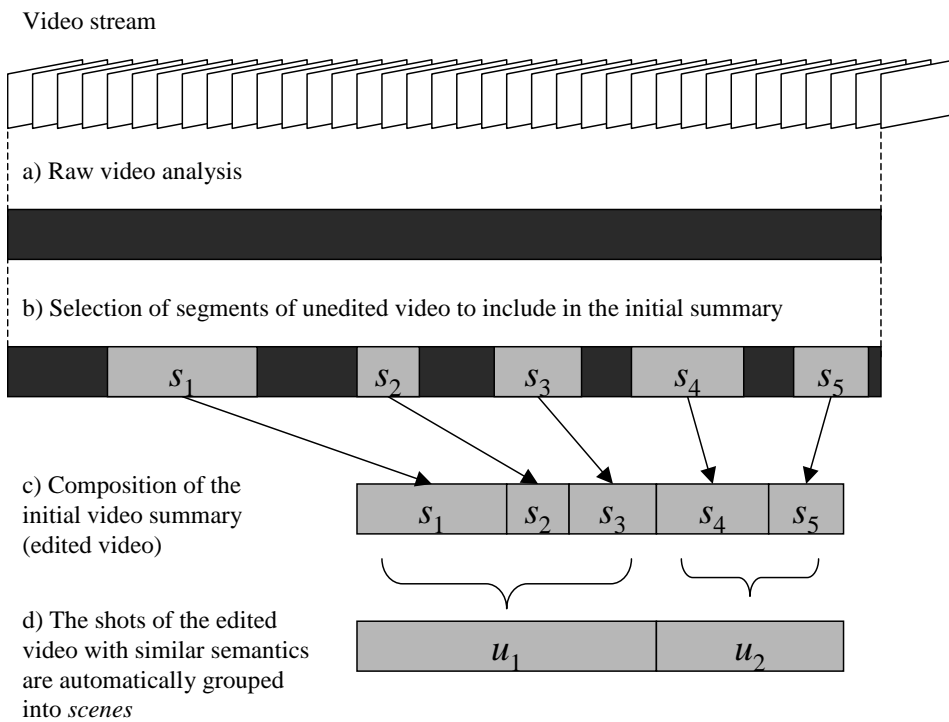


Figure 5.2. Schematic overview of video analysis done by EWW2.

In Figure 5.2 the analysis done by EWW2 is summarized in four steps. In step a) the user uploads his or her video material to the system, in step b) the system chooses which parts of the unedited video are to be included in the initial version of the edited video. In step c) the system composes the edited video: in the example of Figure 5.2 the system aligns five shots after each other: s_1 to s_5 . Finally, in step d) the system automatically groups the shots that contain similar semantics into scenes (u_1 and u_2 in the example). In the literature, a scene is also

called *logical story unit* [Vendrig and Worring, 2002] and is defined as “a series of shots that communicate a unified action with a common locale and time”. Scenes are therefore groups of shots with similar semantics (event, time and place). In summary, EWW2 incorporates the video analysis steps done by EWW1 and adds the organization in scenes performed in step d).

At this point, we shortly explain how the calculation of the scenes performed in step d) works. For clarity, we give a definition of scene.

Definition 5.1 (Scene). In a home video \mathcal{V} , a logical story unit or *scene* u is defined as a sequence of temporally contiguous camera takes. The semantic content of the camera takes belonging to a single scene concerns the same event, time and place.

$$u = (c_i, \dots, c_j), 1 \leq i \leq j \leq N_C.$$

The scenes of a home video \mathcal{V} are disjoint, and their union is the whole video. According to our definition, a scene boundary cannot be in the middle of a camera take. This is because we assume that camera takes are shot in only one place and time. The scenes in a home video are not unequivocally defined: different viewers may segment a video in scenes in different ways, according to what they consider an “event”.

To offer to the user a useful scene overview, the number of scenes should be conveniently balanced. The scenes should not be too few, otherwise navigating through them will not help the user. On the other hand, the scenes should not be too many, otherwise the user’s attention is overburdened. To calculate the convenient number of scenes, the following procedure has been devised:

$$\Delta_u = \Delta_{\mathcal{V}}^{0.66}$$

$$N_U = \text{round}(\Delta_{\mathcal{V}}/\Delta_u) ,$$

where $\Delta_{\mathcal{V}}$ is the total duration of the unedited video in seconds, Δ_u is the target scene duration in seconds, and N_U is the number of scenes to be displayed in the overview. To give a better idea of this procedure, Table 5.1 displays the number of scenes calculated for some typical durations of unedited video.

Table 5.1. Number of scenes for typical home video durations

Unedited home video duration	Number of scenes
1 min	4
5 min	7
10 min	9
30 min	13
60 min	24

Table 5.1 shows that in the case of short videos, the system provides a rea-

sonable number of scenes to browse; while in the case of long videos, the scene number is not too high. Once the number of scenes is calculated, the unedited video is divided into the desired amount of scenes, making sure that a camera take is never split between two different scenes. Therefore, a scene boundary always corresponds to a camera take boundary (see Definition 5.1). In Chapter 6 we present a more sophisticated algorithm to do scene segmentation according to the video semantics (in particular, the timestamp information).

One of the main differences between EWW1 and the second version is that in EWW2 users can interact with their videos through three *modes*. Each mode gives a particular overview of the video content and allows particular editing operations. These three modes are:

- The *scene* mode: allows users to see and edit the scenes of the video.
- The *shot* mode: allows users to see and edit the shots inside one single scene.
- The *fine editing* mode: allows users to have a finer overview of the unedited video and to do low-level editing operations.

In Section 5.1 the reasons that brought us to adopt a scalable overview on the video are presented. We chose to have three modes because we think that three levels of overview are a good compromise between scalability and simplicity. The idea of a scalable overview on the video for an interface for home video editing is not new. In [Girgensohn et al., 2001], the “Hitchcock” semi-automatic system for video editing is described. Hitchcock’s overview is scalable in two levels: the view of the thumbnails of the shots (finer level) and the view of “piles” of similar thumbnails (coarser level). Also, in [Casares et al., 2002] the “Silver” editing system is discussed that supports a storyboard view of the video material and three timeline views with different zoom level. Each of the three modes of EWW2 will now be explained showing screenshots of the application.

5.2.2 The *scene* mode

In Figure 5.3 a screenshot of *Edit While Watching* version 2 is shown. In the main window the video summary generated by the system is played. In the bottom panel an overview of the video scenes is presented. Each scene is represented as a pile of video frames, where each frame represents one of the shots contained in the scene. While the video summary is played, the user can at any time pause the playback. The action of pausing opens a menu with editing functions, which is visible at the centre of Figure 5.3. Via this menu, the user can select editing operations in order to modify the current summary according to his or her wishes. An explanation of the single functions of this menu is given in Figure 5.4. The functions corresponding to the two central buttons in Figure 5.4 are called “less of this scene” and “more of this scene”. These functions respectively remove and add

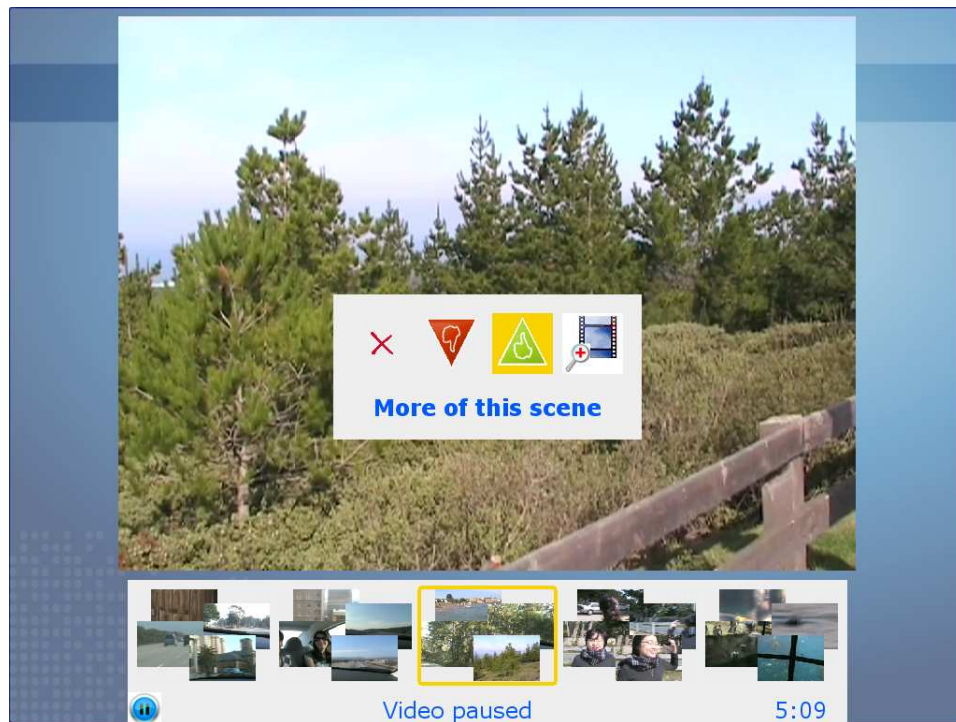


Figure 5.3. Screenshot of *Edit While Watching* version 2, scene mode.

content to the current scene. The video segments that are removed or included are automatically determined according to their suitability score.

In the *scene* mode, the navigation and editing functions refer always to entire scenes of the edited video. With the left and right arrows of the remote control, the user can navigate through the scenes. Every time the user presses left or right, the scenes in the overview panel “slide” to show that the current scene is moving to the left or to the right respectively.

In the scene mode, the user can also do gross editing operations: adding or removing content from a scene, or deleting a scene. In this mode, the user has a broad overview of the edited video and of the main semantic units that constitute it. However, the user does not have all the details about the video content present in one single scene. To gain this level of detail, the user can press the “zoom in” button, the rightmost in Figure 5.4. By doing this, the user “zooms in” inside one single scene and an overview of the shots contained in the scene is displayed: the user has reached the shot mode. When the user goes from the scene to the shot mode, the overview panel shows a “zoom in” graphic animation, the overview

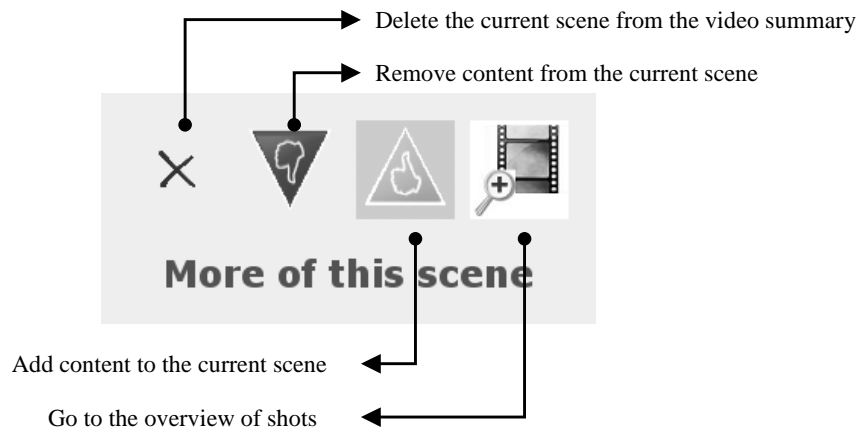


Figure 5.4. The editing menu in scene mode.

zooms inside the current pile of frames and shows single frames.

5.2.3 The *shot* mode

Figure 5.5 shows a screenshot of EWW2 in shot mode. In this mode, the bottom panel of the interface gives an overview of the video shots. Each shot is represented by its middle frame. In the shot mode, the overview of the material, the navigation functions and editing functions refer to the single shots of the video summary. Via the left and right arrows the user can navigate across the shots of the edited video, making smaller jumps than in the scene mode. These jumps are represented in the overview panel by sliding the key-frames of the shots left or right.

Like in the scene mode, at any time the user can pause the video playback and enter a menu with functions for editing the video. This menu is presented in Figure 5.6. The buttons contained in this menu correspond, from the left, to the functions “delete shot”, “less of this shot”, “more of this shot” and “add missing shot”. These functions work in exactly the same way as in EWW1 (Section 4.1).

In the shot mode, the user looks at a smaller temporal window of the video, losing information about the semantic units (scenes) of the video but gaining detail about the single shots. The editing functions involve now smaller portions of the video. From the shot mode, the user can “zoom out” (second button from the right in Figure 5.6) and go back to the broad overview of the scene mode, with a “zoom out” graphic animation. Alternatively, the user can gain even more detail by further “zooming in” in the video material: by pressing the rightmost button in Figure 5.6, the user goes to the fine editing mode. The application shows that the user is entering the fine editing mode by animating a “zoom in” inside a shot.

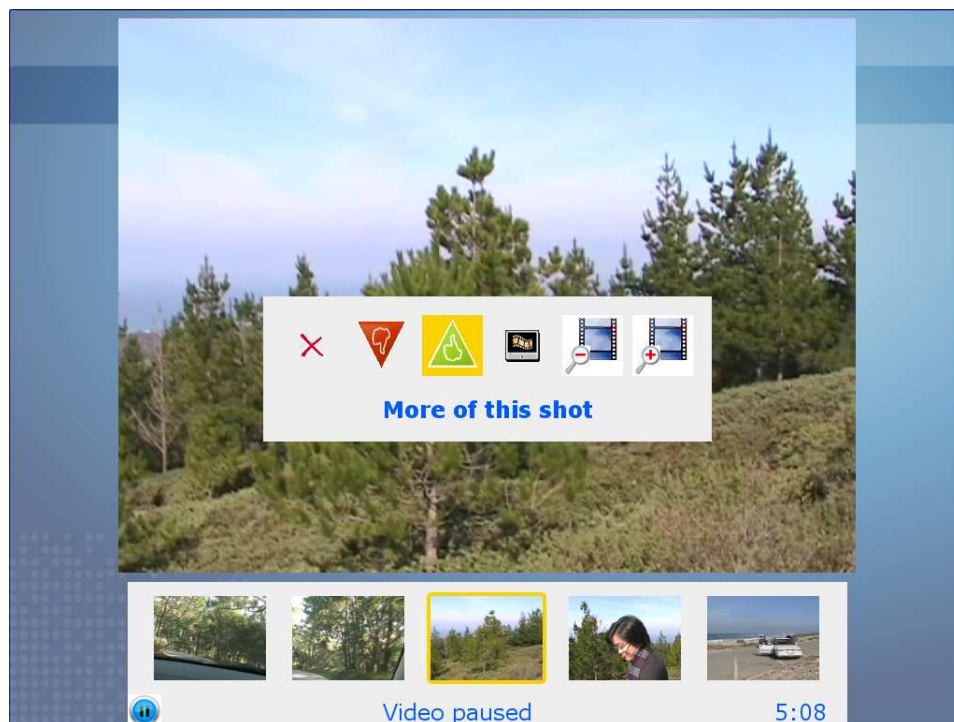


Figure 5.5. Overview of the video shots in the shot mode.

5.2.4 The *fine editing mode*

Figure 5.7 shows a screenshot of EWW2 in fine editing mode. In this mode, a timeline is displayed in the bottom of the interface.

In the fine editing mode, all the unedited video material is played, not only the edited video like in the other two modes. In fact, the purpose of this mode is to allow the user to precisely select which parts of the video to include or exclude from the summary. In the timeline of Figure 5.7, dark-red segments of video correspond to excluded portions of unedited video, meanwhile light-green segments of video correspond to parts of unedited video that are included in the summary.

During the video playback, the timeline translates from right to left in order to be always centred on the current instant. On fast-forward or fast-rewind, the timeline translates quicker, right-to-left or left-to-right.

Like in the other two modes, when the user presses the pause button an editing menu is shown. This menu is displayed in Figure 5.8. By selecting the “thumb down” or “thumb up” button in the menu of Figure 5.8, the user can start to precisely select a portion of the timeline to be excluded or included respectively in

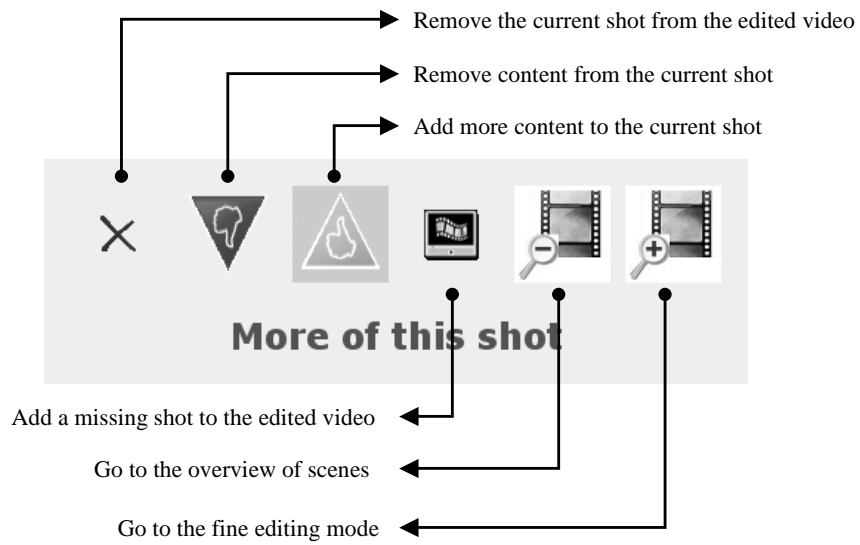


Figure 5.6. Editing menu in shot mode.

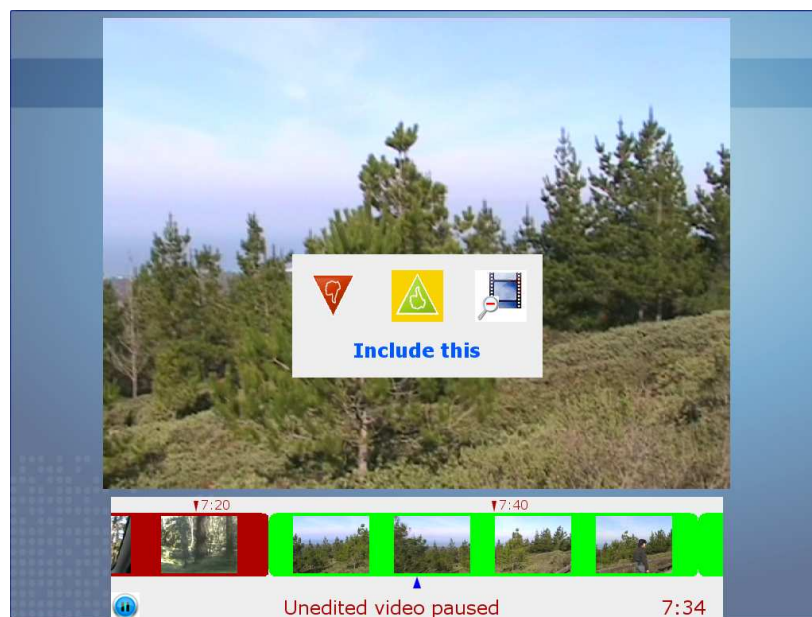


Figure 5.7. Timeline overview in the fine editing mode.

the video summary. To end the selection, the user presses the button “ok” of the

remote control. The part of timeline included between the start and end points of the cut selection is inserted or removed in the edited video, according to the selected function. These functions do not cut the video exactly in the frame that the user has selected. Instead, the cut points are approximated with the closest video segment boundary. This is done because the segment boundaries that *Edit While Watching* has calculated are assumed to be convenient points for a cut, since one of the low-level features present a change. The editing functions in timeline mode give the user control of the video content.

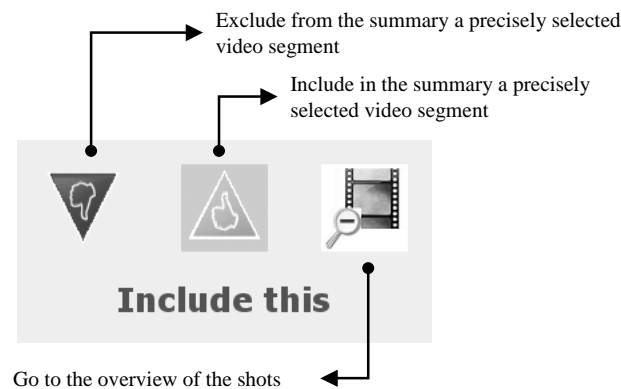


Figure 5.8. Editing menu in the fine editing mode.

5.2.5 Differences with the first version of EWW

Here the novelties of EWW2 with respect to EWW1 are analyzed, pointing out why EWW2 is designed to overcome the limitations of EWW1.

EWW2 allows the user to visualize, access and navigate video material by means of three concepts: the *scenes* of the video, the *shots* of one scene, the *video segments* in and around one shot. On the other hand, EWW1 does not support any kind of overview of the material, except for the overview of the shots not included in the video summary and the video playback. The three-level navigation of EWW2 is designed to address the overview limitations resulted from the use test on EWW1.

Concerning navigation functions, all the modes of EWW2 support *fast-forward* and *fast-rewind* of the video, in addition to the play, pause and skip functionalities of EWW1.

Concerning the editing operations, EWW2 incorporates the shot-based editing functions present in EWW1 and adds the scene-based functions and the functions in the fine editing mode. The editing options in the fine editing mode are designed to provide full control on the edited video, in order to overcome the lack of pre-

dictability of EWW1's functions.

For technical reasons and given the time constraints, it was not possible to implement in EWW2 the functions for music and video effects present in EWW1. Table 5.2 summarizes the differences in editing and navigation functionalities between EWW 1 and 2.

Table 5.2. Differences in editing and navigation functionalities between EWW 1 and 2.

Functions present only in EWW2	Functions present in both versions	Functions present only in EWW1
<i>Editing</i>		
More of this scene Less of this scene Delete scene Include this (fine editing) Exclude this (fine editing)	More of this shot Less of this shot Delete shot Add missing shot	Add music to shot Add video effect to shot (black & white, sepia, etc.)
<i>Navigation</i>		
Fast-forward Fast-rewind Next scene Previous scene Jump ahead 30 seconds Jump back 30 seconds	Next shot Previous shot	

EWW2 requires only 12 keys on the remote control² while EWW1 requires only eight. When the user jumps across the shots, or across the scenes, or when he or she navigates through the three modes, EWW2 gives feedback about what is happening by showing “slide” and “zoom in/out” graphic animations among the elements of the overview panel (shots, scenes, timeline). The “zoom in/out” effects are designed to communicate to the user that the overview of the video is changing from finer to broader or the other way around, in order to make EWW2 easier to understand. We also added a text field on the extreme bottom of the interface. In this text field the application displays its current status and the effect of navigation or editing actions initiated by the user.

A working demo of EWW2 has been implemented using Java and the Quick-Time library for video playback and real-time editing.

²Play, pause, fast forward/rewind, previous/next, four arrows, ok, exit.

5.3 Use test on *Edit While Watching* version 2: objectives

In this section the use test on EWW2 is described. First, the objectives of the test are presented (present section), then the test design is explained (Section 5.4). After that, the results of the test are presented (Section 5.5) and discussed (Section 5.6).

To perform an initial evaluation of our application, we wanted to observe users editing video with EWW2 and to explore how they behave with the system. We wanted to know whether users like the EWW2 concept: editing video on TV with three modes to navigate the video.

Moreover, we were interested in investigating a number of aspects of *usability* of EWW2:

- **Ease of understanding.** We wanted to test whether EWW2's concepts, three-level overview and editing functions are perceived as easy to understand.
- **Ease of use.** EWW2 has many functions for navigation and editing, and each mode has its own functions. We were interested in evaluating whether the functions for browsing and editing are perceived as easy to use or whether there are any difficulties.
- **Satisfaction with the video overview.** Video is a difficult medium to visualize because of its temporal dimension. For this reason we implemented an overview system with three levels of scalability. We wanted to assess whether users think that this helps them to have a clear overview of their video material.
- **Satisfaction with the navigation functions.** We were interested to know whether users think that they can move around in the video quickly and according to their wishes, and whether they can effectively search for particular events.
- **Satisfaction with the editing functions.** EWW2 provides diverse editing possibilities: there are functions for fine selection of video content and also tools to do gross and quick changes to large scenes of the video. We wanted to assess to what extent EWW2's editing tools meet the needs of users, whether users would like more editing functions, whether users think that the editing functions can be improved, whether some functions are more used or more important than others, and whether users feel a lack of control on the editing process.

5.4 Test design

This section describes how we investigated the objectives previously explained and how the experiment was designed and run. For this experiment, we invited the participants to bring their own home video material and to use EWW2 to edit it. We made this choice to increase the ecological validity of the experiment. Although the evaluative setup for EWW1 was judged as realistic, we thought that users may

not be consciously aware of their editing needs and wishes when editing a video that is not their own. By letting users edit their videos, instead, we were sure to observe their authentic editing behavior and wishes. However, allowing the participants to bring and edit their own videos introduced more variables, such as video duration, type of event, video format etc. We recruited nine users, considered sufficient to find the main flaws in the usability of a system [Nielsen and Landauer, 1993].

5.4.1 Recruitment and profile of the participants

We were interested in users with no or only basic experience with home video editing, and without expertise in the fields of multimedia content analysis and management. In fact, EWW2 is designed for these amateur users. The use test on EWW1 was run with users chosen among employees and students of Philips Research. This introduced a potential bias in the results of the test because all participants had a technical background.

To avoid this bias, we looked for users without a background in multimedia content analysis or management, and with diverse backgrounds. Nine users participated in the test, five of them did not have a technical background. Among the other four, one was a civil engineer, another was an electrical engineer, another had a background in user-system interaction, and another had a background in computer science. Most users were highly educated. Six males and three females were recruited. The age varied between 26 and 46 years and the average was 33 years. Three users were married and had children.

Five users capture home videos a few times per year, two capture home videos once a month and the remaining two capture videos twice a month or more often. All users use a digital camcorder, furthermore six of them use also a digital photo camera with video capturing capabilities, and one employs also a mobile phone. Six users capture videos of an average length of 30 minutes or less, the other three capture videos of an average length of 1 hour or more. Two users did not have any previous experience in video editing.

To join the experiment, participants needed to bring at least five minutes of unedited video material. The properties of the videos brought by the participants are shown in Table 5.3. After the test, the participants received a gift certificate worth € 20 as compensation.

5.4.2 Test setup and structure

In order to create a realistic scenario, we ran the test in the living room of the Philips ExperienceLab and we invited the participants to bring their own home videos and to edit them using EWW2, according to their wishes. The test was structured in five parts: a pre-test interview, an introductory demo, three initial editing tasks, a

Table 5.3. Video material brought by the participants for the experiment. N_C stands for number of camera takes, N_V for number of video segments, in the unedited videos.

User	Duration of unedited video (min)	Duration of automatically edited video (min)	Video format	N_C	N_V
1	6	4	Quicktime (15 fps)	12	193
2	40	20	DV AVI	75	1551
3	45	36	DV AVI	31	1151
4	27	15	DV AVI	6	769
5	56	15	DV AVI	89	1807
6	31	15	MPEG-2	80	850
7	56	25	DV AVI	89	1807
8	55	10	DV AVI	73	1057
9	61	30	DV AVI	64	2003

session of editing, and a final interview.

- *Pre-test interview.* This interview addressed the participants' habits in filming and editing home video. We needed to know this to relate the users' behaviour and answers during the test to their experience and usage of home video. This interview was carried out one or two weeks before the actual experiment, in the participants' homes. During the interview, the participant was asked for the video material to be used for the test. Also, the user was asked how long the initial summary of his or her video should be, the summarization and transcoding of the user's video could be then done before the rest of the experiment.

- *Introductory demo.* This part of the experiment and the following took place in the Philips ExperienceLab. The introductory demo concerned only the main aspect of EWW2: the three levels of overview, the navigation among the shots and the scenes, the function "more of this shot", and the editing functions in the fine editing mode. The user was invited to discover autonomously the remaining functions. The demo was given by reading to the user the instructions shown in Appendix C, Figure C.1 and C.2. We could also have decided to let the users try to learn EWW2 by themselves instead of showing them the demo. This would have made the test longer and taken more effort from the users, shortening the editing phase. Since we were primarily interested in observing the users while editing their videos, we decided to give to the users the initial tutorial of EWW2.

- *Initial editing tasks.* After the demo, the user was asked to perform three tasks on his/her own video. The tasks were chosen to reflect real editing activities.

While performing the tasks, the user was asked to think aloud. The three tasks were:

1. Search the x moment in the edited video. The “x moment” was chosen by the test monitor as a particular moment in the edited video, in a way that the task was equally difficult for each participant. This was done by selecting moments in the videos so that the number of key presses needed to navigate to a given moments was approximately the same for all participants.
2. Include more of the y moment in your edited video. The “y moment” was chosen by the test monitor as a particular moment among the video segments excluded from the summary.
3. Make sure that the shot with the y moment in the edited video starts precisely at point z. “z” was decided before by the test monitor. An example of this task was: “Make sure that the scene with you on the speedboat begins exactly when you start smiling at the camera”.

The tasks were part of the initial tutorial on EWW2, to make sure that the participants had the same level of acquaintance with the search and editing functions of EWW2. By defining the tasks, we could also observe which functionalities of EWW2 were used to carry out a specific task, and which were the difficulties in reaching a certain goal.

- *Editing session.* After completing the tasks, the users were invited to freely edit their own videos. During the editing, participants were invited to make remarks about the application and to explain what they were trying to do. Also, every key press was logged by the system. Using these methods, we were able to observe how users interacted with the system and in which ways they edited their video. By analyzing the log files we were able to obtain information about the productivity (how much raw video did users edit) and the error frequency (which functions were used immediately without problems and which needed more attempts).

- *Final interview.* After the editing, we conducted a semi-structured interview with each participant to have their feedback about the editing experience. The interview was designed to investigate the aspects of usability listed in Section 5.3 and to know whether the participants could edit the video as intended. For this interview, a Likert-scale questionnaire was employed. In this way users could rate positively or negatively several aspects of EWW2; they were subsequently interviewed about the motivations leading to their rating.

5.5 Results

The results of the use test have been collected from the analysis of four different sets of data, as described before. The analysis of the log files is first presented

(Section 5.5.1), then the analysis of the users' remarks (Section 5.5.2).

5.5.1 Analysis of logged activities of the users

In this section we describe the results of the analysis of the log files. These contain all the keys that the users pressed during the experiment. Figure 5.9 shows for each user how the test time was distributed across the initial tutorial, the tasks and the free editing.

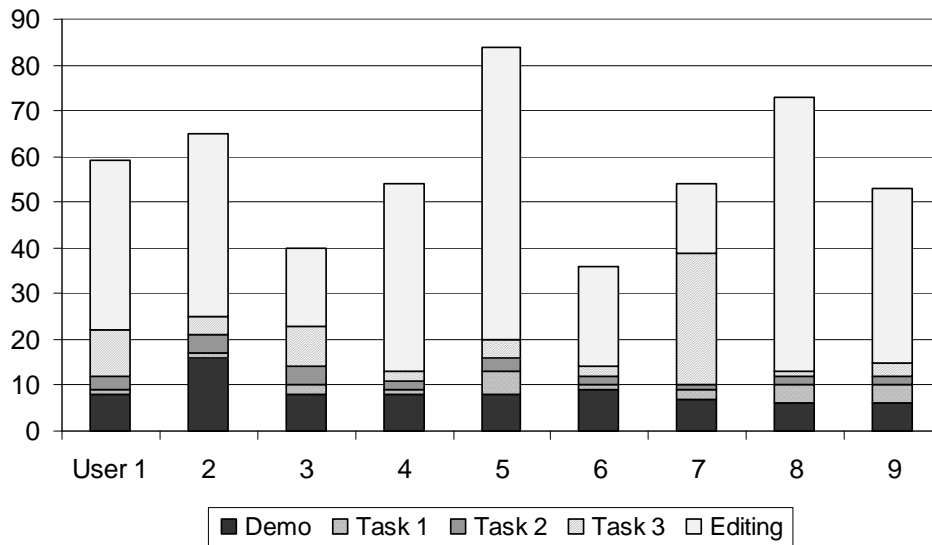


Figure 5.9. Test time (in minutes) across users and test sections.

From Figure 5.9 we notice that most of the time was spent by the users in editing their home videos (average 37 min, range from 15 min to 1h 4 min). 12 min on average were taken to carry out correctly the three tasks and eight min on average were dedicated to the initial demo.

During the editing time, users freely authored their home video exploiting the three modes and different functions of EWW2. Figure 5.10 shows how the time users spent in editing the video distributed across the three modes. We note that users spent most of the editing time in the fine editing mode (average 27 min); 10 min were spent on average in the shot mode and one minute in the scene mode.

We looked at how users spent their editing time, considering the log files together with the users' comments. We were able to distinguish two groups of users, according to their intention. The first group is composed by four users: user four, five, eight and nine (visualized in the four leftmost columns of Figure 5.10). The second group is composed by the remaining five users (five rightmost columns in Figure 5.10). The users of the first group spent their editing time with the intention

of precisely editing their video from the beginning to the end and to come to an end result. The users of the second group were instead focused on trying out the possibilities of the application, navigating and editing only some portions of their videos.

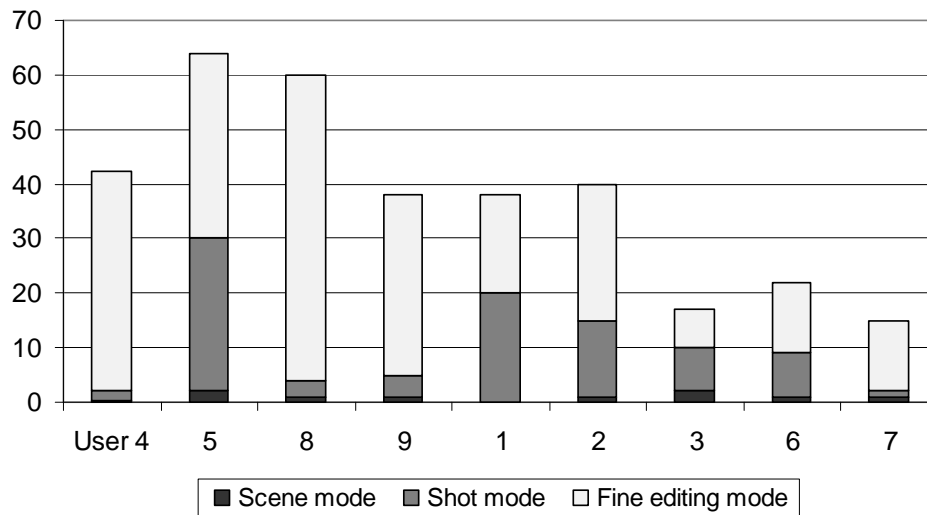


Figure 5.10. Distribution of the editing time (in minutes) across the three modes.

The four users in the first group spent their editing time with the aim of entirely editing their video: this is visible from their remarks and from the fact that their editing operations are sequentially and uniformly placed along the video stream. We were able to measure the time these users took to edit a certain amount of raw video material; our results are reported in Table 5.4.

Table 5.4 shows that the users had a good productivity, managing to edit their videos in a remarkably short time. The ratio between the editing time and the duration of the edited material is never higher than 1.7. The two users without experience in video editing had a good productivity. User 5 edited her video in a time close to the video's playback time; she was alternating between using the functions for fine editing and the "delete shot" function according to her intentions. Also user 9 managed to edit the video in a time almost equal to the playback time of the video. This was possible because he cleverly used the fast-forward and jump functions in the timeline, navigating the video faster than playback time. He remarked, however, that the video editing program "Virtual Dub"³ allows being even faster, because it permits scrolling the video timeline with a slider to immediately access

³www.virtualdub.org

Table 5.4. Measurements of editing efficiency.

User	Duration of edited video (min.)	Time taken for editing (min.)	Editing time / edited material	Remarks
4	22	36	1.67	No editing experience
5	56	62	1.1	No editing experience
8	32	52	1.6	EWV2 crashes once. After crash, ratio was 1.3. Said that EWV2 is faster than Adobe Premiere
9	26	28	1.06	Said that Virtual Dub is faster than EWV2

any instant of the video. Importantly, the users 4, 5 and 8 remarked that they were satisfied with the end result. User 9 said that he would have liked to do more things like adding music or text.

The users not presented in Table 5.4 either were concerned with trying the application and giving comments about it, or spent an editing time too short to take any significant measure of productivity.

As said before, the users spent most of their editing time in the fine editing mode (Figure 5.10), especially the four users of the first group. This is also confirmed by the fact that the most frequently used editing functions were the functions of the fine editing mode. Figure 5.11 shows the frequency of usage of each editing function.

The “cut in” and “cut out” functions were by far the most used for editing. Users 4, 8 and 9 used only these functions for editing. Figure 5.11 also shows that the semi-automatic editing functions (more/less of this) were used only a few times.

The editing functions in the timeline mode were also the most problematic functions. Many users would have liked to control more precisely the moments of the video in which to start or stop a cut. Often they had to do several attempts before cutting the video as desired. Table 5.5 shows the users who had to try many times the timeline editing functions. For example, user 4 spent three minutes and eight editing operations to edit only 16 seconds of video, because she had to do several trials to obtain the cut she wanted. While trying to apply the desired cut, she remarked that she was always “too late” to cut the timeline at the desired moment, and that she would have liked to move slower in the timeline or to move frame-by-frame.

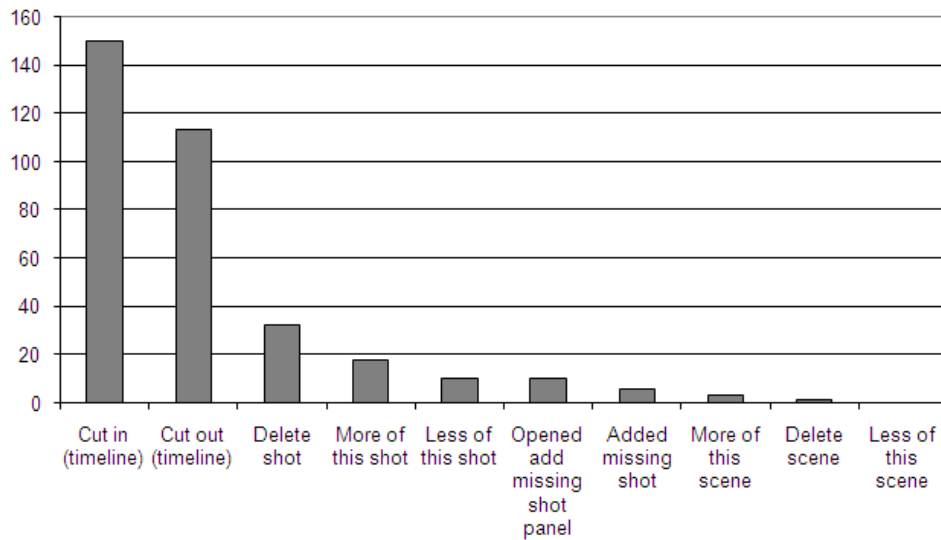


Figure 5.11. Total number of key presses for all participants for each editing function.

Table 5.5. Difficulties with the timeline editing functions.

User	Time spent (min:sec)	Number of cuts	Duration of edited segments (sec)	Users' remarks
1	1:33	4	17	"I would like to select the cut moment more precisely"
2	1:04	3	23	"I have to do trial and error to find the moment when to push the button"
	1:37	3	35	
4	3:00	8	16	"I'm too late every time" "I had problems with cutting, I would like to be frame-accurate"
6	2:38	7	40	"Software doesn't cut exactly where I want"

The fine editing mode was used more often than the shot mode, and the scene mode was hardly ever used (Figure 5.10). Therefore, one could argue whether the scene mode is useful or whether it should be removed. However, the analysis of the

tasks shows that the scene mode is used very often for searching some particular moment in the video. Five users out of nine went in the scene mode to solve the first task (to search for a particular shot) and seven participants used it for the second task (to search for another shot and include more of it). Also, the log files and the users' remarks show that the scene mode is used to search for particular scenes of the video to edit.

5.5.2 Analysis of the users' remarks

We were interested in assessing the *ease of understanding* of EWW2's concepts and features. The users understood the effect and the feedback of all the functions they tried: we had only four remarks concerning some initial difficulties of the users understanding the feedback of more/less of this shot. The most difficult aspect to understand was in which modes the *edited* video is played and in which the *unedited*. Three users said in the final interview that they had significant problems in understanding this aspect.

We collected remarks about EWW2's *ease of use*. In general, EWW2 was judged as an easy to use tool. However, the users thought that moving around in the timeline was difficult. They would have liked different fast-forward speeds. Five users remarked that it was difficult to move to the precise point in which they desired to cut the video. We have seen before (Table 5.5) that the users had to do several attempts to make a cut in the timeline as desired.

We asked about the users' *satisfaction with EWW2's video overview*. The users found the navigation across the three levels of the overview clear and predictable. There were some negative remarks only about using the overview for searching something. Some users remarked that it can be difficult to find a particular shot while browsing the scenes of the video, because it is not easy to see in which scene the shot is contained.

We wanted to know the users' *satisfaction with EWW2's navigation functions*. The users were overall satisfied with the navigation functions present in EWW2, however they strongly remarked that there is a need for more functions for moving around in the video. Table 5.6 shows the extra navigation functions that users would like to have.

We interviewed the users about their *satisfaction with EWW2's editing functions*. In general they did not think that the current editing functions should be improved. Nevertheless, the users expressed a strong need for more editing functionalities. They found the current editing possibilities too limited and rigid and they asked for more ways to author their personal videos. Table 5.7 presents the extra editing functions that the users requested.

We also asked the users with video editing experience to compare EWW2 with

Table 5.6. Number of extra navigation functions requested by the users

Function	N. users
Slowing down the playback to make more precise cuts	6
Having different speeds for fast forwarding (2x, 4x, 8x, etc.)	4
Jumping camera take boundaries and shot boundaries in the timeline	2
Browsing my video in a key-frame storyboard	1
Going immediately to start and end of movie	1
Jumping different time lengths (5 s, 10 s, 30 s, etc.) in the timeline	1
Playing frame by frame	1
Having a “safe exit button” to go back to the initial state	1

Table 5.7. Number of extra editing functions requested by the users

Function	N. users
Video transitions	4
Addition of music to the video	4
Addition of text to the video	3
Shuffling of shots’ temporal order	3
Addition of slides with titles	3
Undo button	2
Video effects	1
Possibility of importing pictures	1
Addition of objects and shapes	1
Mixing music volume with video volume	1
Addition of initial menu with DVD chapters	1

their own editing tool. They considered EWW2 as the easiest and most efficient tool, and their own application as the most powerful. One user found his own tool (Pinnacle Studio) to be more efficient than EWW2, but just because he is more used to it. He said that if he would practice some more with EWW2, our tool would be the most efficient. Another user said that Virtual Dub allows quicker editing than EWW2, as discussed before. A user found that EWW2 is easier because it is less powerful and therefore less complex. Another user thought that EWW2 is easier than PC-based tools because it requires only a TV and a remote control.

As conclusive remarks, generally users said that they would like to use EWW2 for editing their home videos. Three users would like to have EWW2 as it is, meanwhile five users would like to have EWW2 if the issues with the timeline functions are solved and if more editing functionalities are added. User 6 (the one with the highest experience in video editing) explicitly said that he would not like to have

EWV2. He found EWW2 simple and fast, but he considers functions for music, video transitions and video effect as a “must have” for an editing application. If these functions are added to EWW2, he would consider our tool as interesting to use.

5.6 Discussion

The use test on EWW2 showed that most of the concepts and features of EWW2 were easy to understand and to use. Furthermore, the participants seemed to feel confident with the application and managed to edit large portions of their videos according to their intentions, on their first try with EWW2. Table 5.4 shows that even the users without previous experience in video editing were able to select which video segments to keep and which to discard in a time comparable to the video’s playback time. This is a good result since it is known that video editing can be much more time-consuming⁴. It is important that users can quickly select which parts of the video to keep. In fact we know from our Internet-based survey that 89% of users who edit their videos do it primarily with this intention.

Also, the use test on EWW2 has exposed some shortcomings of our application. For example, it was not intuitive to the users in which modes the edited video is played (the scene and the shot mode) and in which the unedited video is played. This type of errors, where the user does in one mode what is appropriate in another mode, are called *mode errors* in the literature [Norman, 1983] [Sarter and Woods, 1995]. One way to prevent mode errors is to make sure that the modes are distinctively marked [Norman 1983, p. 255]. In EWW2, the overview panel changes considerably its configuration from one mode to the other (Figures 5.3, 5.5 and 5.7). However, the playback window maintains the same look in all modes. Therefore, different visual cues (for example different colours at the border, or symbols of a video at different stages) should be applied to the playback window to distinguish when the edited and the unedited video are played.

The navigation in the timeline should also be improved, to make sure that users can control more easily where to cut the video. In fact, we showed how several users had to try many times before being able to cut the video as desired (Table 5.5). It is possible to solve this problem by, for example, modifying the behavior of the buttons fast-forward/fast-rewind. These buttons could allow the users to switch between different playback rates: 0.25x, 0.5x, 1x, 2x, 4x, 8x, etc. The interface should always remind the user about the current playback rate.

In this experiment, we were interested in observing whether the participants would use the semi-automatic editing function “more/less of this” to edit their

⁴60% of the respondents to our survey said that they take a time for editing five times longer than the edited video, 40% need a time for editing ten times longer.

videos. These functions are designed such that the system automatically selects which video segments to add or to subtract to the current shot, so we wanted to see whether this type of automation is useful for the editing process and adds value to EWW2. Figure 5.11 shows that the users employed much more frequently the fine editing functions than the semi-automatic ones, that were rarely accessed. In the interview this aspect was not discussed in depth, because the log files were analyzed only after the completion of all the tests. The very low usage of the semi-automatic editing functions could be caused by the fact that people want to control more precisely which portions of video they include or exclude. We think that better semi-automatic functions can be designed: these should help the users by automatizing some operations, on the other hand they should give to the user enough decisional control about which material is edited. If such functions were designed, the users would not need to always use the full-control functions of the fine editing mode. We plan to explore this research direction further.

We were particularly interested in knowing whether EWW2 overcomes the main flaws of EWW1. In EWW2 no user said that he or she lacked control on the video selection process, in contrast with what the users of EWW1 expressed. This could be partly due to the fact that all the users of EWW1 had a technical education, so they could be more used to control a system in detail. However, four users of EWW2 had a technical background, and the complaint about the lack of control was not heard at all. We can infer that the limitations of EWW1 regarding control in editing are solved, by adding the fine editing mode. Also, with EWW2 no user requested a better overview on the video, contrarily to EWW1. Therefore, EWW2 overcomes the lack of overview that was observed by the users of EWW1. In conclusion, our experiments showed that EWW2 presents fewer flaws in usability than EWW1.

Also Hitchcock, one of the semi-automatic home video editing systems we reviewed from the literature [Girgensohn et al., 2001], presented problems with the perceived control: users wanted to have more control on shot duration and boundaries. This adds more evidence to the viewpoint that controlling precisely which content is selected or discarded is an important requirement for a video editing system. This study shows that EWW2 provides to the users enough control on the video selection process.

The most frequent negative comments on EWW2 concerned the request for extra functions. Several users thought that the system was too limited since it did not allow enrichment operations like adding music, video transitions, text or video effects. Therefore, we should ask ourselves how EWW2 can be extended with more functions without compromising its simplicity and usability. EWW1 had functions for music and video effects. These functions were designed so that the user would see their effect immediately after activating the proper button. All

users of EWW1 understood this design, and the enriching functions were used frequently⁵. Hence, we think that the enriching function should be designed in the same way as in EWW1. There should be a separate menu, called “enriching” or “special effects”, containing all the extra functions. There could be a “safe return to start” button: in case of difficulty the users could press it to return to the EWW2 interface whose simplicity has been validated. More features could be added to EWW2 to increase its ease of use: for example an “undo” button or a help screen. With these extensions, we think that EWW2 would be a more powerful tool without compromising its simplicity.

Looking at our experiment from a general perspective, it appears that some users liked EWW2 as it is and enjoyed using it for editing long portions of home video (Table 5.4). However, other users found EWW2 not complete, “too simple” or not fully-featured, they tried it out for a shorter time without aiming to edit the video. Therefore, it appears that different users understand the video editing experience in different ways. Some users only ask their editing application to be simple and to have good usability. Other users see the possibility of doing a lot of things with their video editing tool as very important. These users desire high editing power in order to express their creativity, besides simplicity. A tool which is easy to use but with restricted functionalities will leave them unsatisfied. This is shown also by the answers to our Internet-based survey: when asked about limitations of current video editing applications, some people complain about the excessive complexity (23%), others about the lack of editing possibilities (17%). One respondent said that he likes “the ease of the user-friendly tools but also the power of the professional ones”. Another commented that “video editing tools are either too simple or too advanced”. Also in [Kirk et al., 2007] (p. 67) we read that sometimes tools like Windows Movie Maker can appear too simple and therefore limited. This brings us to think that the functionalities of a video editing application should be easy to use and intuitive, but at the same time rich enough to allow the users to edit and enrich videos in a great variety of ways.

⁵The EWW1 function “Add video effect” was used 7.5 times per user, “add music” was used 5 times per user. The users tried these functions for 8 min on average.

6

Evaluating the role of intelligent chaptering and the level of user control in video editing

6.1 Introduction

In the previous chapter we presented the second version of *Edit While Watching*. A user evaluation has shown that users can edit their videos with EWW2 with good efficiency; furthermore, our participants judged EWW2 as an easy to use and appealing application. In this chapter we want to investigate further which are the elements of EWW2 that contributed to these results, in particular, we want to take a close look on the automation present in EWW2. To what extent does automation contribute to the effectiveness and to the simplicity of EWW2? Which user requirements are easier and fulfilled more effectively by automating parts of the system? Also: what happens to the effectiveness and simplicity of EWW2 if automation is added to or removed from its functionalities? Can EWW2 be further improved by finding a better balance between automation and user control in some of its functionalities?

EWW2 implements diverse types of automation. The video material that the user uploads to the system is automatically analyzed, segmented and summarized. Automation is applied to structure the video in small segments that are then used

as building blocks for the initial video summary and as “indivisible units” during the editing process (the user cannot cut inside a video segment, not even in the timeline). Automation is employed to organize the video shots into scenes and to display the scenes overview (Figure 5.3). Automation is also applied in the editing functions *More of this* and *Less of this* to help the user in selecting or discarding video content. These types of automation were introduced in EWW1 and 2 to assist the user in the most technically difficult and time-consuming tasks for home video editing. Does this automation really add value to EWW2? Or should the balance between automation and user control be revised and improved?

The problems related to balancing the level of automation in interactive systems have been discussed in a number of studies in the literature. Parasuraman et al. [2000] have defined automation as the full or partial replacement of a function previously done by the human. Automation therefore is not all or nothing, but can vary across a continuum of levels, from fully automatic to fully manual. In order to select the proper level of automation, Parasuraman et al. [2000] propose to look not so much at the consequence on the technology but at the consequence on the interaction with the human. In particular, they propose two main criteria for selecting the proper level of automation: the associated consequences on human performances and the reliability of the automation. Concerning human performance, they argue that, although well-designed automation improves the system performance and reduces the human effort, there is also evidence of highly automated functions that can compromise the results and worsen the system performances. It can happen, for example, that when automation is difficult to understand and to put in action, the mental workload of the user is increased instead of lowered. It can also happen that high automation generates complacency, an “over-trust” effect where the user does not monitor the actions of the automated system, which may make mistakes without the user being aware. Concerning the reliability, Parasuraman et al. [2000] argue that the automation should be implemented and presented to the user so that he or she can rely on the automated functionalities. Unreliability causes mistrust that may motivate the users to disable the automatic functionalities.

We can see a relation between these different levels of automation and a similar idea used in computer-supported learning and gaming environments. Reflecting on these environments, Gentner [1992] has classified them according to their *locus of control*. Learning systems in which the student can fully determine the sequence of lessons and the learning procedure are said to have *internal* (to the user) locus of control. Learning situations when the student has to follow passively a sequence of lessons fixed by a teacher or by a system are said to have *external* control. Gentner has observed that video games instead have a *mixed* locus of control, since both the user and the system can autonomously take initiatives. The mixed control present in games gives the user both surprise, challenge and a degree of control. This may

be the reason why games are so motivating and engaging, despite the considerable amount of learning that they require. Because of these observations, Gentner has hypothesized that learning systems with a mixed locus of control may be more motivating than situations with external control (because they take into account the user's initiative) and more productive than systems with internal control (because they challenge and direct the user). In [Masthoff, 2002] evidence is reported that indeed in computer-based learning systems a mixed locus of control is preferable to either the student or the system exclusively taking the initiative and selecting lessons. We may therefore hypothesize that also in home video editing a design based on mixed locus of control can have beneficial effects on the users' motivation and performances.

The review of the aforementioned literature suggests that, to study where and to what extent automation adds value to EWW2, we could vary the level of automation of EWW2's functionalities and observe the effects on the user's objective performance, on the mental effort, on the user's trust of the system and on the perceived ease of use. To further sharpen the research questions of our study, we have developed a scheme of the user requirements of a video editing system. The scheme is shown in Figure 6.1. Picturing the user requirements for home video editing is useful to study the balance between automation and user control for each requirement. Furthermore, it is also useful to study whether some requirements are needed by others (whether, for example, doing a precise cut in a timeline requires precise navigation in the timeline). The set of user requirements represented in the figure has been obtained by considering the requirements of EWW2, the requirements desired by the participants of our user tests, the requirements considered most important in literature and the requirements of the most used commercial applications (distilled by trying out the functionalities of these applications).

In Figure 6.1, navigation requirements are represented by solid-line boxes, and editing requirements by dashed-line boxes. Arrows between requirements express the fact that one requirement needs another one. For example, to do a fine cut in the video, one needs functionality for frame-by-frame browsing; or to search for a particular scene in the video, a navigation system (storyboard-based or timeline-based) is needed.

To check whether some user requirements are frequently needed by other requirements, we can look at the requirements from which a high number of arrows depart. It appears that navigating the video, both in a high-level and in a fine way, is very often requested for doing editing actions. Therefore, if the navigation requirements are addressed more efficiently and effectively, also editing operations might be carried out with more efficiency and effectiveness. EWW2 has a navigation system with three levels of overview. Automation is used to group the shots into a number of scenes (Section 5.2.1) and to display the scenes in the highest level

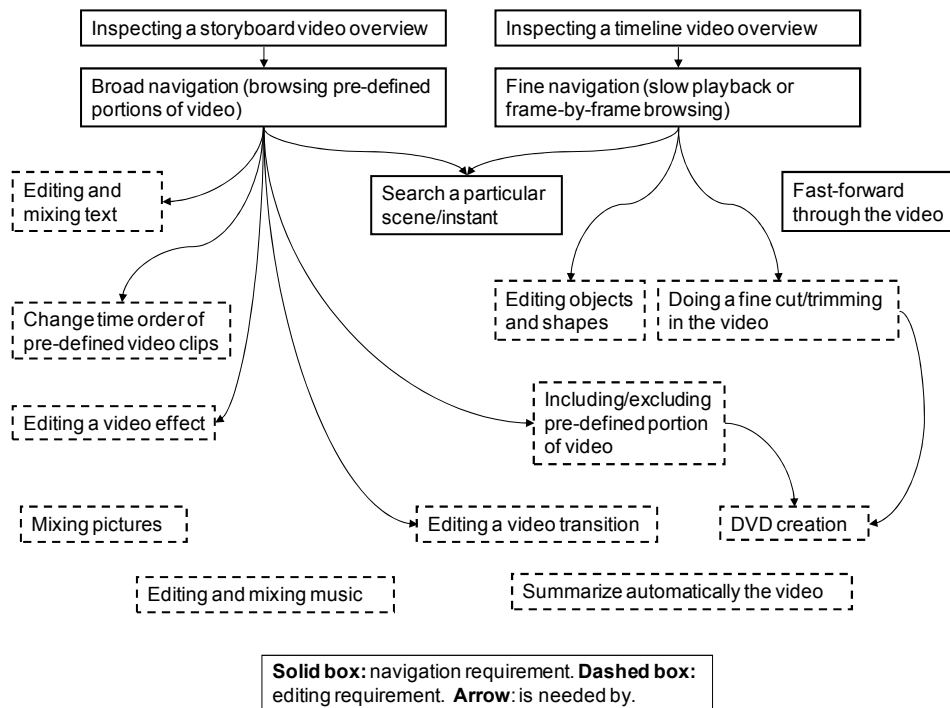


Figure 6.1. Relation between editing and navigation requirements for home video editing.

of overview. It is therefore interesting to study whether a better automatic scene segmentation algorithm leads to more effectiveness and better user satisfaction in some of the requirements correlated with video browsing. If, for example, a user is faster in searching for particular moments in the video material, he or she will probably feel more in control. In Section 6.2 and following an experiment designed to assess the effects of an improved automatic scene detection is described.

In Section 2.3.3 it was clarified that users do video editing primarily for selecting valuable parts of the video and removing the others. To provide the user with means to perform this selection, EWW2 implements the semi-automatic editing functions *More of this* and *Less of this*, and the full-control functions of the fine editing mode *Include this* and *Exclude this*. The semi-automatic editing functions were implemented with the assumption that they could aid the user in the selection of video material. However, they were hardly used during the test on EWW2, while the fine editing functions were used much more often (Figure 5.11 p. 113). It is interesting to investigate why the semi-automatic editing functions were rarely

used and how the balance between automation and user control can be modified to obtain better semi-automatic functions. An experiment designed to investigate this issue will be described in Section 6.6 and following.

6.2 Experiment 1: Effect of intelligent chaptering on navigation tasks

In Section 5.2.1 we have explained that EWW2 automatically segments the unedited video material into scenes, and that the number of scenes is chosen in a way that the overview displayed to the user is useful yet not too complex. The process of partitioning a video into scenes is called in the literature “scene segmentation”. In EWW2, the scene segmentation is performed without taking the semantics of the video into account. Therefore, camera takes related to different events or to different places can fall in the same scene; this may cause surprise and confusion in the user when browsing the scenes of the video. Also, a bad scene overview may hamper the user understanding the structure of the video, browsing it and searching for a particular instant in the video. Problems in browsing and searching may also cause problems in editing particular portions of the video, since the general overview will be less clear.

The algorithm that calculates the scenes can be improved by considering semantic information or metadata. A simple and easily available kind of metadata is constituted by the timestamps associated with each camera take of a home video. Therefore, we may hypothesize that, if the scenes are calculated taking into account timestamp information, the resulting scene overview will serve better the user in at least some requirements. Here we consider two requirements: video browsing (navigating a video to have an understanding of its general structure and storyline) and video retrieval (searching for a particular clip in a video). The following hypotheses are formulated:

Hypothesis 6.1 (*intelligent chaptering improves video browsing*): Scenes calculated exploiting timestamp information help the user in browsing the video better than scenes calculated without exploiting timestamp information.

Hypothesis 6.2 (*intelligent chaptering improves video retrieval*): Scenes calculated exploiting timestamp information help the user in video retrieval better than scenes calculated without exploiting timestamp information.

Before explaining how to test these two hypotheses, we describe in detail a scene segmentation algorithm that takes into account timestamp information. The algorithm’s pseudocode is presented in Figure 6.2. The objective of the algorithm is to calculate the scene segmentation $\mathcal{U}(\mathcal{V}) = \{u_i, i = 1 \dots N_U\}$. At the beginning of the pseudocode, the number of scenes N_U is calculated as described in Section

algorithm **TIMESTAMP-BASED SCENE SEGMENTATION**
inputs: camera takes segmentation $\mathcal{C}(\mathcal{V})$, total duration of unedited video $\Delta_{\mathcal{V}}$;
output: scene segmentation $\mathcal{U}(\mathcal{V})$;

begin
 $\Delta_u = \Delta_{\mathcal{V}}^{0.66}$;
 $N_U = \text{round}(\Delta_{\mathcal{V}}/\Delta_u)$;
// Starting with trivial scene segmentation
for each c_i **in** $\mathcal{C}(\mathcal{V})$ **do**
begin
 $u_i = \{c_i\}$;
add u_i to $\mathcal{U}(\mathcal{V})$;
end
// Calculating time distances between camera takes
for $k = 1$ **to** $N_C - 1$ **do**
begin
// \mathcal{T}_k is the timestamp associated with the beginning of c_k
 $d_k = \mathcal{T}_{k+1} - \mathcal{T}_k - \delta(c_k)$;
// The distance d_k is added at the right place in the sorted list \mathcal{D}
add d_k to \mathcal{D} ;
end
// Iteratively merging the couple of scenes closest to each other
while $|\mathcal{U}(\mathcal{V})| > N_U$ **do**
begin
 $d_k = \text{first}(\mathcal{D})$;
// c_k is the camera take corresponding to d_k
 $u_i = \text{get_scene_containing}(c_k)$;
// The scenes u_i and u_{i+1} are merged together
 $u_i = u_i \cup u_{i+1}$;
 $\mathcal{U}(\mathcal{V}) = \mathcal{U}(\mathcal{V}) \setminus u_{i+1}$;
 $\mathcal{D} = \mathcal{D} \setminus d_k$;
end
return $\mathcal{U}(\mathcal{V})$;
end

Figure 6.2. Algorithm for timestamp-based scene segmentation.

5.2.1. Successively, three main steps are performed:

1. A trivial scene segmentation is calculated, where each scene contains one camera take. At this point, there are N_C scenes, as many as the camera takes.
2. According to the timestamp information, time distances are calculated between each couple of successive camera takes. These time distances are stored in a sorted list (\mathcal{D} in the pseudocode).
3. The two successive scenes closest together in time are merged into one single scene. This procedure is iterated until there are N_U scenes.

An experiment was designed to compare the performances of the timestamp-based scene segmentation algorithm with the scene segmentation performed without timestamp information. The experiment aimed at testing the Hypotheses 6.1 and 6.2, and its design is explained in the next section.

6.3 Design of Experiment 1

To study the effect of timestamp-based scene segmentation, two versions of EWW2 were prepared: one that exploits timestamp information to group into scenes the camera takes that are close in time (EWW2^T) and another one that does not exploit timestamps (EWW2^{NT}). In both these versions, all editing functions are disabled. In both versions, the user can only access the shot mode and the scene mode, without the timeline mode. All the functions in EWW2's editing menus are disabled, except the button for zooming out from the shot mode to the scene mode and the button for zooming in from the scene to the shot mode. In summary, only the navigation functions are active (play, pause, fast forward, fast rewind, go to next shot/scene, go to the previous shot/scene, go from the shots to the scenes, go from the scenes to the shots). The only difference between EWW2^T and EWW2^{NT} is in the way the scenes are calculated.

To compare the two algorithms for building the scene overview, we decided to invite users to perform browsing and retrieval tasks on the same set of home videos. A between-subjects design was adopted. The independent variable was the scene segmentation algorithm, this variable had two possible levels: EWW2^T and EWW2^{NT}. The dependent variables were the efficiency of the users in accomplishing browsing tasks and the efficiency in carrying out search and retrieval tasks.

To measure the user efficiency in browsing and searching through the video material, two lists of tasks were developed. One list of tasks consisted in answering questions about the storyline of a video, the other list of tasks consisted in answering questions about precise moments in another video. We used two home

videos: one for each list of tasks. The browsing tasks were designed on a video about a holiday trip in California, the search tasks were prepared on a video about a visit to a zoo. In order to test Hypothesis 6.1 about browsing and Hypothesis 6.2 about searching, we wanted to invite the users to carry out the two lists of tasks by browsing the video with $EW2^T$ or $EW2^{NT}$ and to measure the time taken by the user and the percentage of correct answers. The two lists of tasks are reported in Appendix D.1 p. 189.

Since a user knows how to answer a question after having answered it for the first time, it was not possible to adopt a within-subjects design. This brought us to develop a between-subjects design. To clarify the difference between $EW2^T$ and $EW2^{NT}$, in Figure 6.3(a) a screenshot is shown of the scene overview for the holiday video displayed by $EW2^T$, while Figure 6.3(b) shows the scene overview of the same video displayed by $EW2^{NT}$.

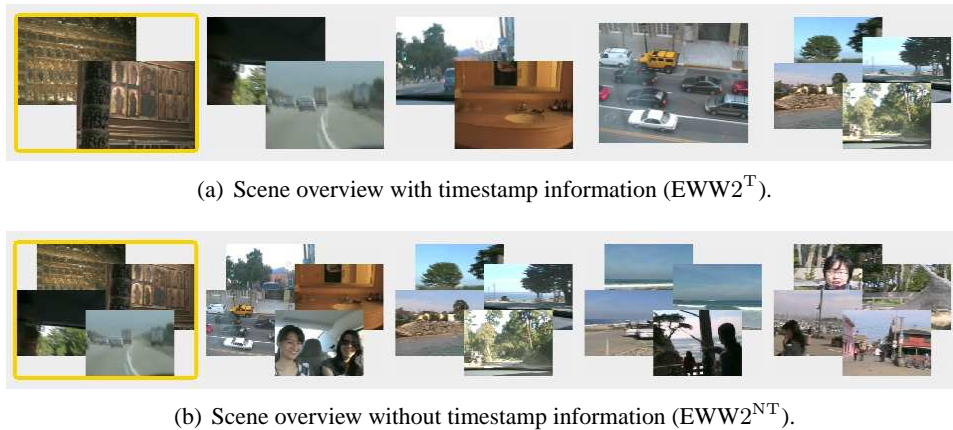


Figure 6.3. Two screenshots of scene overviews, the upper calculated exploiting the timestamp information, the lower calculated without using timestamps. Scenes are represented as piles of key-frames. By looking for example at the first two scenes in both panels, it is possible to note that the scenes in the upper panel are composed of shots with similar semantics. This is less true for the lower panel.

The structure of the test will be now explained.

- *Initial demo.* At the beginning, the user was informed about the test and was given a demo of the navigation functionality of $EW2$. The explanation of the test and of the system was contained in one A4 paper that was read to each participant. The video used for giving the demo was different from the two videos employed for the two lists of tasks. The user was also left as much time as he/she needed to get acquainted with the functionalities. During the experiment, all the users sat at the same distance from the TV screen (different distances would have advantaged

the users who sat closer to the screen). Figure 6.4 shows a picture of the test setup. When the users would feel acquainted enough with the system, they could start with the next step of the test.

- *Execution of browsing tasks on the holiday video.* After the initial demo and acquaintance, the user was invited to perform a set of browsing tasks on the video about the holiday in California. The user had to answer 11 questions about the structure and the storyline of the holiday video. The questions were appearing one by one in a screen placed at the right side of the user (see Figure 6.4). It was possible to see only one question at a time, and it was not possible to go back to a question already answered. All questions were presented with a multiple choice answer set. The user was invited to select the desired answer with a click of the mouse. The tasks related to browsing the video about the holiday in California are reported in Appendix D.1.1 p. 189.

- *Execution of search tasks on the zoo video.* After the completion of the browsing tasks about the holiday video, the user was invited to answer another list of 10 questions about the zoo video. Each question concerned a specific moment of the video. To find the answer, the user had to search the video to retrieve that particular moment. The tasks related to searching the zoo video are reported in Appendix D.1.2.



Figure 6.4. Setup for the experiment related to video browsing and searching.

If the questions had been prepared by using EWW2, they could refer to moments in the video easily reachable with the navigation functions of EWW2. In

order to avoid this bias, the browsing and searching tasks were prepared without using EWW2. The video overview provided by Windows Movie Maker was used to create the questions about the holiday video and the zoo video. During the test, the user answered the questions using a PC. The use of a PC for the questions eased the measurement of the time taken to find the answers. The PC logged a timestamp every time the user answered a question. The home video about the holiday in California was 27 min and 40 sec long, and the scene overview was composed by 12 scenes. The video about the zoo was 14 min and 55 sec long and the scene overview had 11 scenes.

In the beginning of the experiment, the user was explicitly informed that the time taken for answering the questions was measured. The user was invited to take as much time as needed to find the correct answers, and he/she was asked also to avoid pauses during the completion of the tasks.

We recruited 40 users from the High Tech Campus in Eindhoven, 22 of them were male. We divided them in two groups of 20 users each, with the same ratio of male and female users in the groups. One group did the experiment using EWW2^T, the other did the experiment with EWW2^{NT}. The whole experiment lasted on average 40 minutes per participant.

6.4 Results of Experiment 1

In this section the results of the experiment are presented. Table 6.1 shows the average time taken to complete the browsing and searching tasks with the two versions of EWW2. In the table, the set of tasks related to the holiday video is called T_b (browsing), while the list of tasks related to the zoo video is called T_s (searching).

Table 6.1. Completion times of the experiment.

	EWW2 ^T (min:sec)	EWW2 ^{NT} (min:sec)
Average time for completing T_b	12:20	14:26
Std. dev. time for completing T_b	3:45	3:22
Errors in the answers to T_b	23	23
Average time for completing T_s	10:25	11:46
Std. dev. time for completing T_s	2:45	2:29
Errors in the answers to T_s	6	6

Two one-tailed t -tests were performed to assess whether the lists of tasks T_b and T_s were answered significantly quicker by the participants using EWW2^T. The t -tests were run with $\alpha = 0.05$. Both t -tests showed that users with the timestamp-based system were significantly quicker (T_b : $t = -1.866$, $df = 38$, $p = 0.035$; T_s : $t = -1.720$, $df = 38$, $p = 0.047$). Figures 6.5(a) and 6.5(b) show the boxplots of

the time taken by the users of the two groups to answer the questions of T_b and T_s respectively.

In Figure 6.5(b) Subject 15 is displayed as an outlier. Subject 15 did T_s in 18 min and 11 sec; moreover, this subject took also the highest time in answering T_b (20 min 47 sec). If Subject 15 is excluded from the dataset and the t -tests are performed again, the null hypotheses can be rejected with more confidence. The probability of the null hypothesis becomes $p = 0.011$ for T_b and $p = 0.007$ for T_s (T_b : $t = -2.398$, $df = 37$, $p = 0.011$; T_s : $t = -2.563$, $df = 37$, $p = 0.007$).

Looking at Table 6.1, it appears that the participants made many more mistakes in answering T_b than in answering T_s . The list of tasks T_b appeared to be therefore more difficult than T_s for the users. The questions 4 and 9 of T_b turned out to be the most difficult to answer correctly. They were answered incorrectly 10 and 8 times, respectively, by the 20 participants.

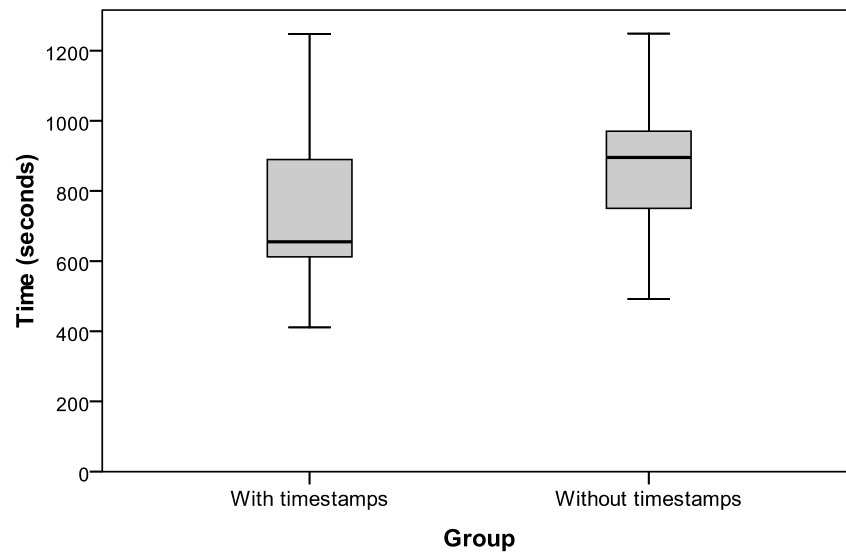
During the experiment, the system logged all the keys that users were pressing to browse the videos. We repeated the analysis also on the number of keys needed to carry out the tasks, to find out whether the participants needed significantly fewer key presses to perform the tasks with EWW2^T than with EWW2^{NT}. Table 6.2 shows the results related to the number of keys pressed.

Table 6.2. Average number of key presses and standard deviation.

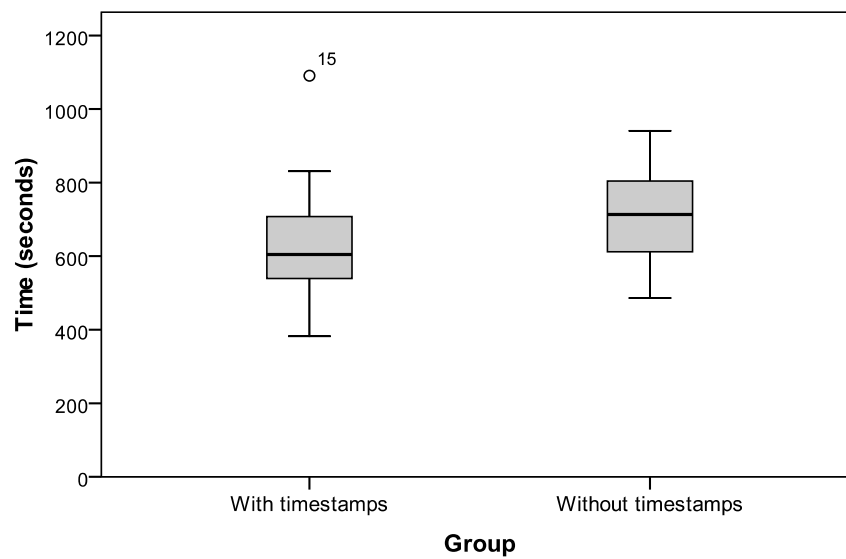
	EWW2 ^T	EWW2 ^{NT}
Average key presses for completing T_b	279	344
Std. dev. key presses for completing T_b	92	158
Average key presses for completing T_s	290	333
Std. dev. key presses for completing T_s	88	129

Two one-tailed t -tests revealed that the number of keys pressed does not significantly decrease in EWW2^T (T_b : $t = -1.597$, $df = 38$, $p = 0.059$; T_s : $t = -1.210$, $df = 38$, $p = 0.117$).

During the experiments, about half of the users made an observation about the buttons of the remote control. In particular, the “arrow up” button had the effect of zooming out from the shot overview to the scene overview, and the “arrow down” button had the opposite effect of zooming in from scenes to shots. The participants found the effects of the “arrow up” and “arrow down” button counter-intuitive. They advised to swap the functions of the buttons: in their opinion, it would be more intuitive to have the “arrow up” to zoom in inside a scene, probably because of the animation that gave the users the impression to move forward inside a scene to see the single shots.



(a)



(b)

Figure 6.5. Boxplots of time taken to answer T_b and T_s by users of the two groups. The top panel shows the result for the browsing tasks, T_b , the bottom panel the results for the search tasks, T_s .

6.5 Discussion of Experiment 1

The results described in the previous section show that the timestamp-based scene overview helps users in video browsing and video retrieval tasks. This is observed in the fact that tasks are carried out faster; however, it is not observed in a significant decrease of the key presses needed to carry out the tasks. This could be due to the fact that a worse scene overview makes users spend more time watching portions of the video to understand it and search in it, without pressing keys.

We also critically review the design of the experiment. It was observed that the list of tasks T_b appeared to be more difficult to carry out than the tasks in T_s . Furthermore, the standard deviation of the time taken for completing T_b is bigger than the standard deviation of the time for T_s . This can also be observed by looking at the boxplots in Figure 6.5: the time results related to T_b are more spread than the ones related to T_s . This could be explained by the fact that, since T_b was more difficult, it involved more of the users' skills and attention, therefore we observe more differences between subjects. We learn from this that particular care should be put in designing tasks with similar levels of difficulty. This may contribute to a smaller variation around the group means and therefore to measure more precisely the difference between the group means. Some of the variance may also be due to the confusion users experienced between the keys "arrow up" and "arrow down" on the remote control related to the "zoom in/zoom out" functions.

Also, a within-subjects design would have served better the purpose of measuring differences in time performances. We were not able to find tasks suitable for a within-subjects design. Since this experiment was designed with a between-subjects approach and analyzed between subjects, we do not know to what extent the variance observed in the data is explained by different scene overviews and to what extent by difference between subjects. In conclusion, this experiment has proven that the Hypotheses 6.1 and 6.2 (p. 123) are correct.

6.6 Experiment 2: Balancing automation and control in editing functions

In the following sections, another experiment is described that aims at finding a convenient balance between automation and user control in the editing functions of EWW2. In Chapter 5 it was explained that EWW2 implements several functions to include or exclude video content from the final result of the editing. In particular, EWW2 has two kinds of functions to address the same user requirement, the selection of particular video content. The two types of functions differ in the balance between automation and user control. The first type of functions exploits a full-control approach: the system displays the whole unedited video to the user via a timeline representation, the user can browse the timeline and select two cut

points as start and end of the video segment that should be included or excluded from the edited video. The second type of functions implements a semi-automatic approach: the user just tells the system that he/she wants “more” (or less) of a certain shot or scene, and the system automatically selects the content that is most suitable to be added to the edited video. Therefore, the user loses control on the selection of video content but gains more simplicity and editing speed.

The semi-automatic functions were implemented under the hypothesis that they could aid the user in at least some editing tasks. However, during the use test on EWW2, they were rarely used (see Figure 5.11 p. 113), while the timeline functions, which provide full control, were used about ten times more often. We do not know why the semi-automatic options were accessed so few times. From the remarks the users gave in the experiments on EWW1 and EWW2, it appears that participants felt a lack of control while using the semi-automatic functions. For example, some users did not like that the function *More of this* chose automatically whether to extend the shot in the beginning or in the end and by how much. Other users found the effect of *More of this* and *Less of this* hard to understand, perhaps because of lack of a clear feedback. Another hypothesis to explain the low usage of the semi-automatic options is that they are perceived as not predictable: the user cannot know beforehand what will happen after activating the functions.

To get a better understanding of the possible reasons for the observed little usage of semi-automatic functions, it is useful to look into the literature on perceived control in the HCI context. According to Hinds [1998], control is

“The perception that ones behavior significantly alters outcomes by producing desired and preventing undesired events”.

There are many variables that may influence someone’s perceived control on a system. Averill [1973] has proposed to see control as a three-dimensional construct. The three dimensions are:

1. *Cognitive control*: it addresses the interpretation of an event into a cognitive model or plan.
2. *Behavioral control*: it addresses the existence of means to exert influence over an event.
3. *Decisional control*: it addresses the ability to choose among different courses of action.

To have control, a user has to have some knowledge or understanding of the functioning of a system, either directly or through analogy. More knowledge (or more cognitive control) will generally increase the overall perceived control. Also, if the user has means to effectively influence the environment, then he/she has a high behavioral control. Finally, the more options and choices the user has to affect the environment, the higher the decisional control and the overall perceived control.

The relationship between predictability and perceived control is seen by authors in slightly different ways. Morris and Marshall [2004] have described predictability as part of the knowledge that the user needs to work with a system, and have argued that knowledge is a pre-requisite to control, since the user must have sufficient knowledge of the system to make the system respond in a predictable manner. They stress the fact that without some knowledge of the system, it is not possible to exercise any control on it and on the environment. Rosenbaum and Smira [1986], Langer [1975] and Hinds [1998] have proposed a softer and more dynamic relationship between knowledge and control: they stress the fact that the more users are knowledgeable (and knowledge includes also predictability) the more they will feel in control. In summary, according to both these points of view, predictability is described as directly related to control.

After having surveyed the definitions, terminology and viewpoints that the literature offers about perceived control, we go back to our investigation about reasons for the low popularity of the semi-automatic functions. Using the constructs defined in the literature, we make hypotheses about why participants did not use the functions *More of this* and *Less of this*. One reason could be the lack of cognitive control, more precisely the lack of predictability and of proper feedback: the users cannot know beforehand the effect of the semi-automatic operations, and the feedback does not sufficiently clarify the editing performed by the operations. Another reason could be the lack of decisional control: the users have no options for how to edit a certain shot, they can only passively accept the decision that the system makes for them. Another reason could be the lack of effectiveness of the semi-automatic functions: during the use test on EWW1, users had to use the functions several times before including or excluding all the content they desired. Another reason could be the lack of credibility: perhaps the users do not trust the automatic algorithm to make the right choice. Finally, the timeline functions could have been preferred over the semi-automatic ones because users are in general more familiar with a timeline.

Below, six different hypotheses about the low usage of the semi-automatic functions as implemented and tested in EWW2 are listed.

Hypothesis 6.3 (*insufficient predictability about how the shot will be edited*): if the functions M/L¹ are modified in a way that the system shows to the user the effect of the function before actuating it, then M/L will be used more frequently and/or the related user satisfaction will increase.

Hypothesis 6.4 (*insufficient decisional control about duration and position of the edited material*): if the functions M/L are modified in a way that the system allows

¹M/L stands for *More of this* and *Less of this*

the user to choose in which direction and by how much to lengthen/shorten the shot, then M/L will be used more frequently and/or the related user satisfaction will increase.

Hypothesis 6.5 (*inadequate feedback*): if more (textual) feedback is added to the functions M/L, then M/L will be used more frequently and/or the related user satisfaction will increase.

Hypothesis 6.6 (*low effectiveness*): if the functions M/L are designed to include or exclude longer video segments (maybe using a different automatic algorithm), then M/L will be used more frequently and/or the related user satisfaction will increase.

Hypothesis 6.7 (*low credibility*): if the functions M/L are presented as extremely good in selecting the content to include or to exclude, then M/L will be used more frequently and/or the related user satisfaction will increase.

Hypothesis 6.8 (*familiarity with timeline*): the low usage of the functions M/L compared to the timeline functions is due to the fact that users are more familiar with a timeline representation for video browsing and editing. If the familiarity with M/L and the familiarity with the timeline functions is the same, then M/L will be used more frequently and/or the related user satisfaction will increase.

By looking at the user behavior during the tests on EWW1 and EWW2 (uncertainties of users with the functions, remarks), we conclude that the Hypothesis 6.4 about lack of decisional control might best explain why the semi-automatic functions were not used often. Therefore, we decided to design an experiment to test Hypothesis 6.4. In order to do that, we wanted to create different experimental conditions in which the participants would use functions with different levels of decisional control, but with the same level of predictability. We decided that there should be three different levels of decisional control: the level of *More/less of this*, the level of the timeline functions of EWW2 and an intermediate level. We wanted to measure what is the influence of changes in the level of decisional control on the users' perceived control, on the perceived ease of use, on the mental workload, on the objective editing efficiency and effectiveness, and on the objective usage rate. Table 6.3 shows the independent and dependent variables of the experiment. In the next section, the design of the experiment on editing functions will be explained.

Table 6.3. Independent and dependent variables of the experiment on editing functions.

Independent variables	Dependent variables
Decisional control (varied among three levels)	Perceived control
Predictability (kept constant)	Perceived ease of use
	Mental workload
	Editing effectiveness
	Usage rate

6.7 Design of Experiment 2

The main idea of the design of this experiment consists in creating three different conditions among which only the decisional control of the editing functions changes, and in measuring the differences among the conditions. To do that, three different pairs of editing functions have been prepared (see Figure 6.6 and 6.7):

(1) *More/Less of this shot*. These semi-automatic functions are the same as the ones used in EWW1 and EWW2. The interaction with these functions was modified to increase their predictability. We want in fact the three kinds of functions to have the same predictability and to communicate their effects on the video with the same visual clues to the user. The functions *More/Less of this* are accessible via the two leftmost buttons in the menu of Figure 6.6 (thumbs-up and thumbs-down button). In this version, after the user presses the *More of this* button, the system opens a dialog window in which the content that is going to be added to the current shot is shown in a timeline overview (see Figure 6.8). At this point, the user can either accept (ok button of the RC) or reject (exit button) the choice of the system. *Less of this* works in the same way.

(2) *Lengthen/Shorten this shot*. These editing functions represent an intermediate level of automation between *More/less of this* and the timeline functions. They have been designed from scratch for this experiment and are accessible via the third and fourth buttons from the left in the menu of Figure 6.6. After pressing *Lengthen this*, the system opens a window and presents to the user a timeline overview of the unedited material in the neighborhood of the current shot (Figure 6.9). With the arrows left/right of the RC, the user can browse through several cut points that the system proposes, indicated as blue triangles below the timeline. By browsing through the cut points with the arrows left/right, the user extends or shrinks a green selection tray below the timeline. The selection tray extends from the beginning of the current shot towards the back, or from the end frontwards. The user can confirm or cancel the cut via the buttons “ok” or “exit”.

(3) *Insert/Remove this*. These are the timeline functions already implemented in EWW2. They are accessible via the two “scissor” buttons in the menu of Figure

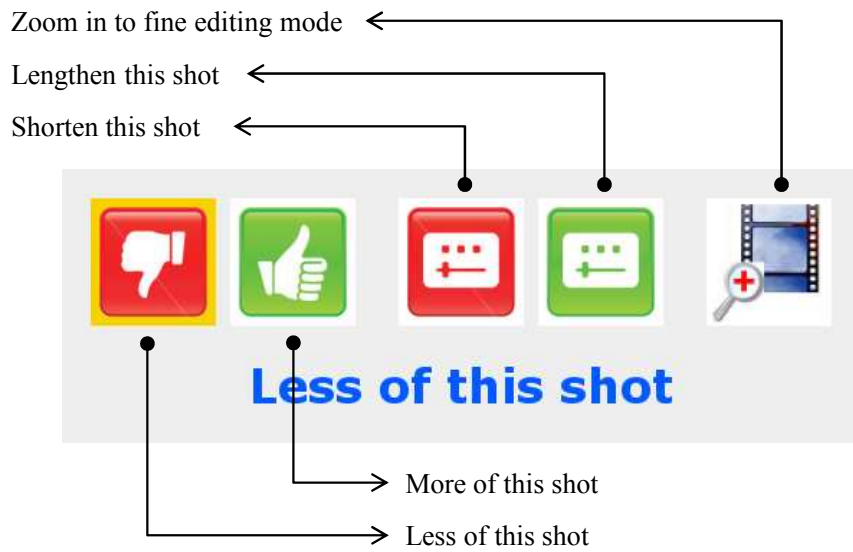


Figure 6.6. Editing menu in shot mode. From this menu, the functions *More/Less of this* and *Lengthen/Shorten this* are accessible.

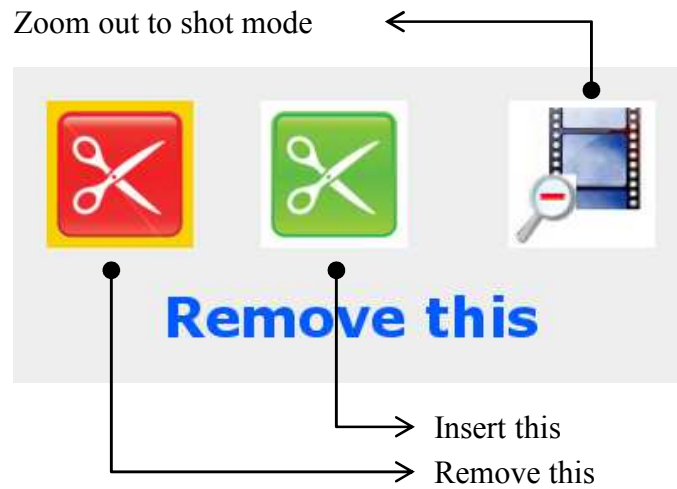


Figure 6.7. Editing menu in fine editing mode. From this menu, the functions *Insert/Remove this* are accessible.

6.7. With these functions, the user has full control on the video content to insert or remove from the edited video. Figure 6.10 shows a screenshot of the function *insert this*. After activating it, the system starts drawing a green “tray” below the timeline, from the instant in which the function was activated (green cut point) to the moment in the video which is currently played back (blue cut point). The user can shrink or extend the green selection tray with the navigation functions (play, pause, fast-forward, fast-rewind, jump ahead 15 sec, jump back 15 sec). When the user is satisfied, he/she presses the “ok” button of the RC and the cut is executed. The cut portion of the video is inserted into the edited video in the case of the function *Insert this*, and is removed from the edited video in the case of *Remove this*. The user can also cancel the cut operation by pressing “exit” on the RC.



Figure 6.8. Screenshot of the *More of this* function. After pressing *More of this*, the system opens a dialog window showing which content will be included, by means of a timeline overview. The user can accept or reject the system’s choice.

In the screenshots of Figures 6.8, 6.9 and 6.10, the three different ways of interaction for selecting and discarding video content are shown. By looking at these screenshots it should be evident that the same visual clues are used in all the functions to communicate to the user what the effect of an editing operation is, before performing it. In all the functions the user can see the content that is going to be inserted or removed both in the timeline and in the main playback window, and can accept or reject the change via the “ok” or “exit” buttons on the RC.

The difference between the three types of functions is in the balance between



Figure 6.9. Screenshot of the *Lengthen this* function. After pressing *Lengthen this*, the system displays a timeline overview with several cut points, indicated as blue triangles. By browsing through the cut points, the user can select which content to include in the edited video.

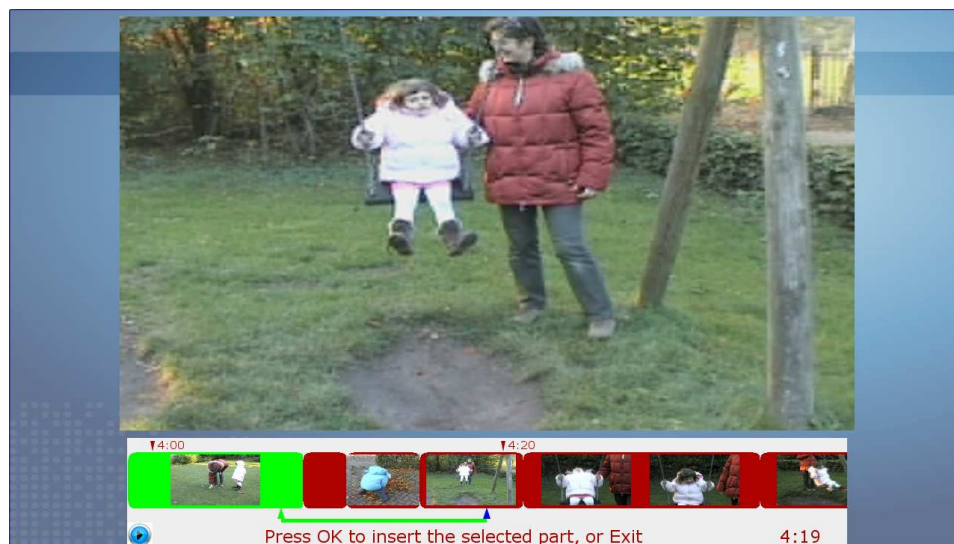


Figure 6.10. Screenshot of the *Insert this* function.

the level of automation employed and the decisional control left to the user. In the functions *More/Less of this*, the system automatically decides whether to edit the beginning or the end of the shot, and how much content to include or exclude. The user can decide between only two options: accept or reject the system's decision. In the functions *Lengthen/shorten this*, the balance is moved one step towards user control. The system automatically calculates a set of convenient cut points, these are proposed to the user. The set of possible decisions the user can take is increased: he or she can choose whether to edit the shot in the beginning or in the end, and can choose the duration of the edited part among a set of proposed options. In the functions *Insert/remove this*, the balance is moved all the way towards user control. The user can now browse the timeline-representation of the unedited video and select any point as start or end of the cut. The system helps the user only with graphical clues, playback of the video and aligning the user's cut points to the segment boundaries calculated at analysis time. For more details, see the description of the functions in Section 5.2.4 p. 102.

To perform this experiment, a special version of EWW2 was created. This version has no scene mode, only the shot and fine editing modes are present. Furthermore, there are no editing functions other than the six functions belonging to the three types described. In this version, the editing menu in the shot mode is modified as shown in Figure 6.6 and the editing menu in fine editing mode is shown in Figure 6.7. No shortcut buttons are available to switch between shot and scene mode, neither to start the timeline editing functions.

We chose a within-subjects design for this experiment: we created a set of editing tasks to be performed by each participant three times, each time with a different type of editing functions. The test was structured in four steps, described below:

Step (1). *Introduction to EWW2 and getting acquainted.* Initially, the participant was given an introduction to EWW2 and to the editing functions. The participant received a sheet with instructions to read and directly try out in order to learn the system. These instructions concerned mostly the editing functions and were in a format very similar to the instructions used for the test in the previous chapter, reported in Figure C.3 of Appendix C. After reading the instructions, the participant had as much time as desired to get acquainted with the system.

Step (2). *Editing tasks with forced choice of functions.* After the introduction to EWW2, the user was given a sequence of nine editing tasks to be performed three times: once using only *More/Less of this*, once using only *Lengthen/Shorten this*, and once using only the timeline functions *Insert/Remove this*. The tasks were performed on a home video related to a visit to a zoo. Since each participant did the same tasks three times, the acquaintance with the tasks increased at each iteration,

which could lead to a gradually quicker execution of the tasks. To compensate for this learning effect, the nine tasks were split in three triplets and the order of execution of each triplet with each function was organized as shown in Figure 6.11. The nine tasks to be performed with pre-determined choice of functions are reported in Appendix D.2.1 p. 193.

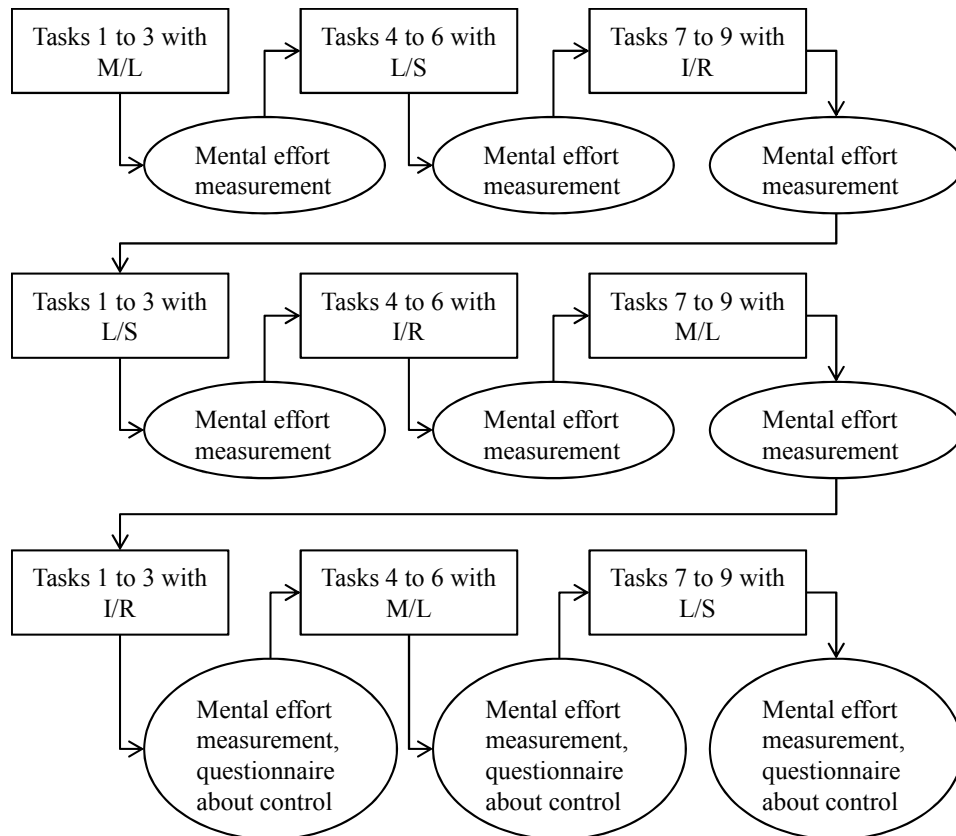


Figure 6.11. Organization of the editing tasks for balancing of the learning effect. M/L stands for *More/Less of this*, L/S stands for *Lengthen/Shorten this*, I/R stands for *Insert/Remove this*.

In this figure, the tasks done with each of the three function types are equally split among the three repetitions. Therefore, the learning effect cannot cause any of the three function types to have an extra benefit over the others. In Figure 6.11, the first three triplets of tasks are carried out with *More/Less of this*, *Lengthen/Shorten this* and *Insert/Remove this* respectively. To balance also the order in which the participants used the functions, we considered the six possible permutations of the order of the three function types and wrote down five extra schemes besides the

one in Figure 6.11. We rotated the participants across these six schemes. Between one triplet of tasks and the other, the user was asked to rate the mental effort taken to execute the triplet by using an appropriate scale [Zijlstra and Doorn, 1985], reported in Appendix D, Figure D.1. Also, after completing each one of the last three triplets, the user was asked to fill in a questionnaire to rate the perceived control, ease of use and perceived self-efficacy in relation with each one of the three function types. The questionnaire has been taken from [Hinds, 1998] and was adapted to our context regarding editing functions; it is reported in Appendix D, Figure D.2. During the execution of the tasks, the test leader made sure that the participant could only use one function type for each task triplet, following the pre-defined rotation of the functions.

Step (3). *Editing tasks with free choice of functions.* After the execution of the tasks on the zoo video, the participant was invited to perform another sequence of eight tasks on another home video, with playing children. This time, the user was free to choose the function type that he/she thought appropriate to carry out each task. In this way, we could measure the objective usage rate of the functions. The eight tasks to be carried out with free choice of functions are reported in Appendix D.2.2 p. 194.

Step (4). *Final semi-structured interview.* Finally, the user was interviewed in a semi-structured way about his/her possible preference for a function type and about any comment the participant may have had. The interview protocol used is reported in Appendix D, Figure D.3.

For this experiment, 25 users were recruited, 17 from the High Tech Campus, Eindhoven, and 8 from outside the Campus, 13 male and 12 female. We recruited people with at least a basic experience in capturing home videos with any device. We thought that this basic experience was needed to be sure that the user had an idea about home videos and about which problems home video editing may bring. We did not require our participants to have any previous experience in video editing. Also, we did not want to have people with a lot of video experience. Therefore, we excluded from the test people who edit video once every two weeks or more often. Moreover, we avoided people with technical knowledge about videos, or about multimedia. We also avoided programmers or IT professionals. The test lasted on average one hour and a half per participant.

6.8 Results of Experiment 2

6.8.1 Perceived ease of use and perceived control

During the execution of the tasks with pre-determined choice of functions, the users were invited to answer a questionnaire about ease of use and control. Each

user filled in the questionnaire three times, once for each function type. The questionnaire was composed of Likert-scale statements concerning the ease of use, the perceived control and the perceived self-efficacy related to a given function type (see Figure D.2 p. 196). Figure 6.12 shows the average answers to the questions related to the ease of use, for the three function types. Figure 6.13 shows the average answers related to the questions about perceived control, and Figure 6.14 shows the average answers for the questions concerning the perceived self-efficacy.

Looking at the answers, it is immediately evident that there is a great difference between the functions *More/Less of this* and the other two function types, which instead appear to get very similar scores. Since each user answered each question three times, once for each function type, the answers for each question were analyzed by performing a within-subjects (repeated measures) ANOVA, with $\alpha = 0.05$. For each question the ANOVA test was significant with a high value of F ($p < 0.001$). Post-hoc tests done after each ANOVA consistently lead to a similar result: for all the 12 questions, the average answer for *More/Less of this* is significantly different from the answers for the other two function types; there is no significant difference between the other two function types. It can therefore be concluded that the functions *More/Less of this* are perceived as significantly more difficult to use than the other two function types. Also, *More/Less of this* appear to provide the users with less control than the other two kinds of functions.

6.8.2 Mental workload

In this section, another set of answers the users gave while performing the tasks is analyzed. Besides rating the ease of use and the control, the participants were also asked to rate the mental effort needed to perform the tasks. Figure 6.11 shows how the moments of measurement of the mental effort were distributed during the execution of the tasks. The users executed the same tasks three times, during each round of execution all three function types were used. The mental effort was measured nine times per user, once for each combination of round/function type. Figure 6.15 shows the average mental effort scores for the three function types. The mental effort is plotted averaged across all three rounds and for each one of the rounds separately. In the mental effort scale proposed to the users, values around 60 correspond to “rather much effort”, while values around 30 correspond to “a little effort” (see Figure D.1 p. 195). It appears from Figure 6.15 that *More/Less of this* took on average approximately two times the mental effort required from the other function types. One would expect that the difference between the mental effort required by *More/Less of this* and by the other functions is statistically significant. Indeed, this was the result of a within-subjects ANOVA to check the effect of the function types on the mental effort scores averaged across all rounds ($\alpha = .05$, $F = 80.614$, $df = 2$, $p < .001$). A post-hoc test confirmed that *More/Less of this*

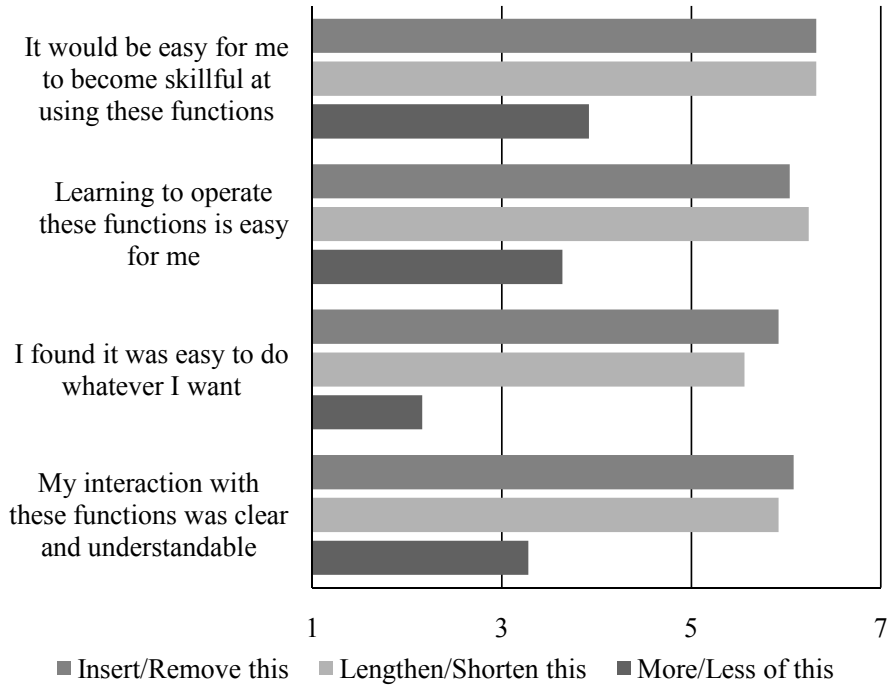


Figure 6.12. Average ratings to the Likert-scale questions about perceived ease of use, plotted for the three function types.

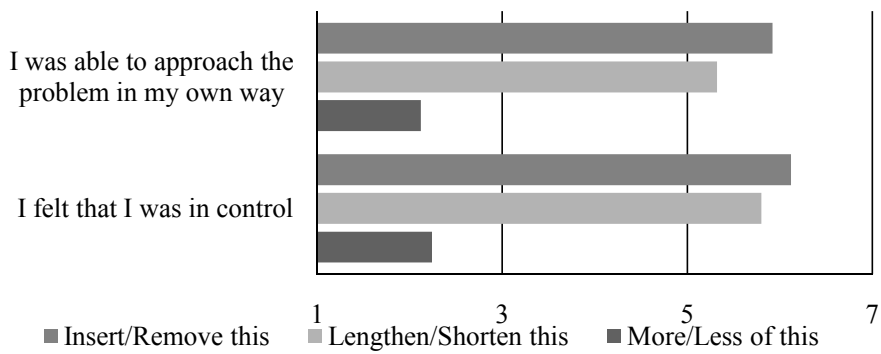


Figure 6.13. Average ratings to the Likert-scale questions about perceived control, plotted for the three function types.

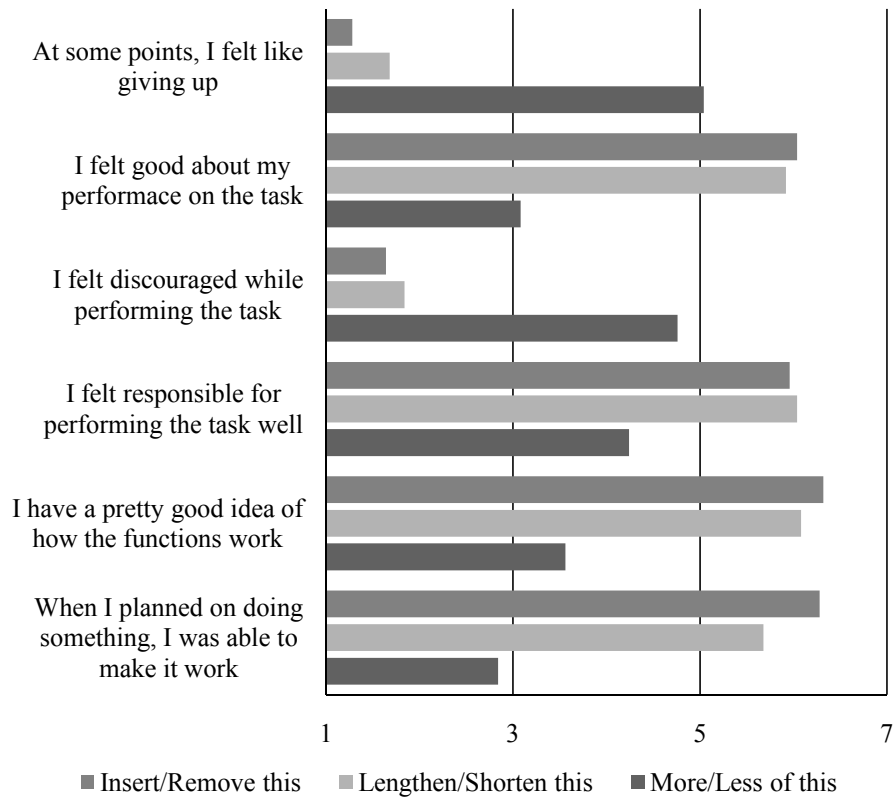


Figure 6.14. Average answers to the Likert-scale questions about perceived self-efficacy, plotted for the three function types.

are significantly more cumbersome than the other two function types, which did not significantly differ in mental workload.

We were interested in checking whether the mental effort required to use a specific function type decreases significantly with the repetitions of usage. Such a decrease would indicate that users can easily learn the functions. Three within-subjects ANOVAs were performed to check the effect of the usage rounds on the mental effort required by each function type. In the case of *More/Less of this*, the mental effort did not significantly change across the three rounds ($\alpha = .05$, $F = .914$, $df = 2$, $p = .408$). Also Figure 6.15 shows that the mental effort required by *More/Less of this* remains approximately constant across the three rounds. In the case of *Lengthen/Shorten this*, the usage rounds had an effect on the mental workload ($\alpha = .05$, $F = 4.759$, $df = 2$, $p = .013$). More precisely, a post-hoc

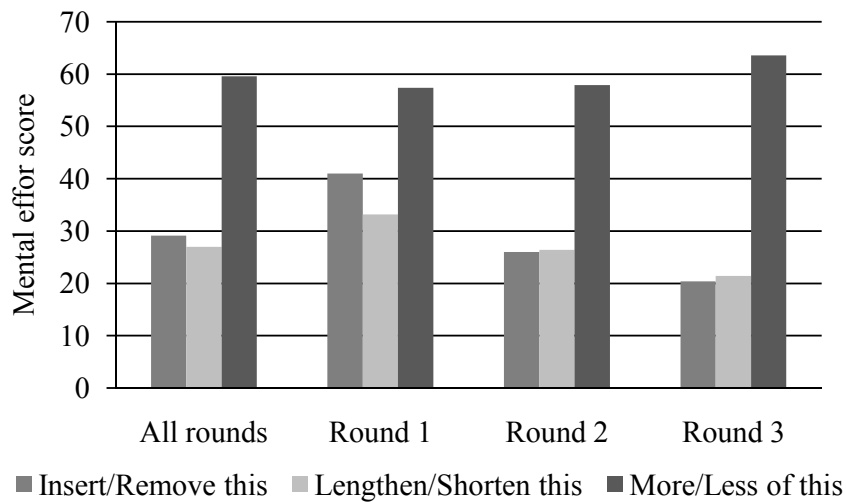


Figure 6.15. Mental effort taken by the users to complete the tasks in all rounds and separately in each round, for each one of the three function types.

test showed that the first round required significantly more mental effort than the third, while the second was not significantly different from the other two. Also in the case of *Insert/Remove this* the usage rounds had an effect on the mental effort ($\alpha = .05$, $F = 13.461$, $df = 2$, $p = .000$). The first round was significantly more cumbersome than the second and the third, which did not differ among each other.

To summarize, the functions *More/Less of this* were a lot more demanding than *Lengthen/Shorten this* and *Insert/Remove this*. The effort taken to use *More/Less of this* did not decrease with more usage. In contrast, the mental workload for the other functions decreased after one or two rounds of usage, probably because of a learning effect.

6.8.3 User satisfaction with the functions

After the experiment, each participant was asked about his or her satisfaction with the editing function and the preference he or she may have for one of the function types. This was done in a semi-structured interview, following the questions reported in Figure D.3 p. 197. Starting from the questions reported in the appendix, a discussion with the participants about the functions was started. In these discussions users made many remarks on their opinions about the functions and the reasons for their opinions.

In accordance with the results presented already, the functions *More/Less of this* were not preferred by any user. Instead, the users were divided more or less

equally among preferring *Lengthen/Shorten this* and preferring *Insert/Remove this*. Table 6.4 shows in detail the remarks the users made when asked about their preference for any function type over the others.

Table 6.4. Remarks of the users about preference for any function type. I/R refers to *Insert/Remove this*, L/S refers to *Lengthen/Shorten this*, M/L refers to *More/Less of this*.

Remark	N. users
I prefer I/R, then L/S, then M/L	11
I prefer L/S, then I/R, then M/L	13
I equally like L/S and I/R, I don't like M/L	1
I would use mostly or only the combination of I/R and L/S	13
I would use mostly or only I/R	7
I would use mostly or only L/S	3

The users gave also reasons for preferring one function type over another. The users that preferred *Lengthen/Shorten this* over *Insert/Remove this* said that *Lengthen/Shorten this* were easier and quicker in completing the tasks. The participants that preferred *Insert/Remove this* over *Lengthen/Shorten this* said that *Insert/Remove this* provide a better overview, since the editing happens on a timeline and it is possible to see the whole video, the excluded clips and the included ones (Figure 6.10 provides a screenshot of the function *Insert this*). The functions *More/Less of this* collected mostly negative remarks. Table 6.5 shows the most frequent user remarks about the three function types.

6.8.4 Editing effectiveness and efficiency

Till now, we have reported results related to subjective measurements. In this and in the next section, we present results related to objective measurements about the effectiveness and efficiency of the three function types.

We start by looking at the effectiveness in performing the editing tasks. In Section 6.7 it was explained that the participants were invited to perform nine video editing tasks three times, once for each function type. The users could not choose which type of functions to use, since the choice was pre-determined by the test leader by enabling only the buttons of one function type at a time in the interface. The users were explicitly informed by the test leader that they could skip a task if they experienced the task as very difficult or impossible to do. Table 6.6 shows the percentage of tasks that were correctly accomplished when the choice of functions was forced.

Looking at the percentages, it appears that 17% of the tasks were not accomplished when using *More/Less of this*. In fact, the users explicitly gave up carrying out these tasks, considering them not (easily) doable. This was mainly because the

Table 6.5. Users' remarks about their satisfaction with the three function types.

Remark	Num. users
<i>More/Less of this</i>	
I couldn't go in the direction I wanted to	10
I don't have control	9
I could not do what I wanted	5
I didn't understand how the video was selected	4
I would drop these functions, take them out	4
I would like to control which side to edit (beginning or end of the shot)	4
I don't like these functions	3
It is difficult to use these functions	3
<i>Lengthen/Shorten this</i>	
They are easy	9
They are fast	6
They give control	4
They are useful	4
If I want to go to the other side of the shot, I shouldn't be scrolling all the way	4
I would like to change the position of the cut points	3
I cannot cut something in the middle	3
<i>Include/Exclude this</i>	
They are easy	8
They give control	5
They give a nice overview of the video you are editing	4
I would use them only for fine editing	4

Table 6.6. Percentage of tasks correctly accomplished with pre-determined choice of functions.

Function type	Correct tasks (%)
<i>More/Less of this</i>	83
<i>Lengthen/Shorten this</i>	99
<i>Include/Exclude this</i>	100

function *More of this* did not allow the users to choose at which side to extend the shot, at the beginning or at the end. There were tasks where it was asked to extend the shot after its end, for example to show the continuation of an event, but the

function *More of this* gave to the users only the options to extend the shot in the beginning or to cancel the editing. Many users tried several times before giving up the task while expressing their frustration. Even if some tasks were considered as impossible by the participants, during the design of experiment it was made sure that each task had at least a solution with each kind of functions. In the cases seen as difficult, the solution consisted in using the functions *More of this* and *Less of this* several times in succession.

After doing the tasks with pre-determined choice of function type, the users were invited to perform other eight tasks on a different home video, this time with free choice of function type. The users were asked to select the function type that they would find most appropriate for solving each task. During this part of the experiment, almost all the tasks were completed correctly (96.5%). Few tasks appear to have been skipped by the users (2%), since the log files do not report any user attempt to solve them. It is possible that the users skipped by mistake these tasks without the test leader realizing it. In fact, this part of the experiment required less interaction with the test leader than the tasks with forced choice of functions. Finally, 1.5% of the tasks were carried out incorrectly. The next section explains which function types were used most frequently during the tasks with free choice of function type.

After looking at the editing effectiveness, we analyze how the editing efficiency changes among the function types. The time taken for fulfilling the tasks with pre-determined choice of functions was measured. To carry out the same nine tasks, the users took on average 15 min and 25 sec with *More/Less of this*, 5 min and 43 sec with *Lengthen/Shorten this* and 8 min and 02 sec with *Insert/Remove this*. *More/Less of this* revealed to be considerably slower than the other two functions. This could be because the users performed many attempts before being able to complete some tasks with *More/Less of this*. A within-subjects ANOVA revealed that indeed the function type has a significant effect on the time performance in completing the tasks ($F = 80.148$, $df = 2$, $p = .000$). A post-hoc analysis revealed that the differences in means among the function types are all statistically significant. Therefore, not only *More/Less of this* is significantly slower than the other two, but also *Lengthen/Shorten this* is significantly faster than *Insert/Remove this*. This is consistent with the users' perception explained in Section 6.8.3 that *Lengthen/Shorten this* are quicker than *Insert/Remove this* in doing the tasks, while *Insert/Remove this* provide a better overview.

Further analysis on the editing efficiency was performed, to see whether there is evidence of a learning effect from the efficiency. Each participant did the same sequence of editing tasks three times, as shown in Figure 6.11. Initially, we checked whether the users were significantly faster in doing the tasks after one or two repetitions of the task sequence. Figure 6.16 shows the time taken for completing the

tasks averaged per user and per task, for each round of execution of the tasks and for each function type. Looking at the figure, it appears that the first round was slower than the other two. Indeed, a within-subjects ANOVA shows that the repetition of the tasks had an effect on the efficiency ($F = 28.929$, $df = 2$, $p = .000$). According to the post-hoc analysis, the first round was significantly slower than the other two, which did not differ significantly from each other.

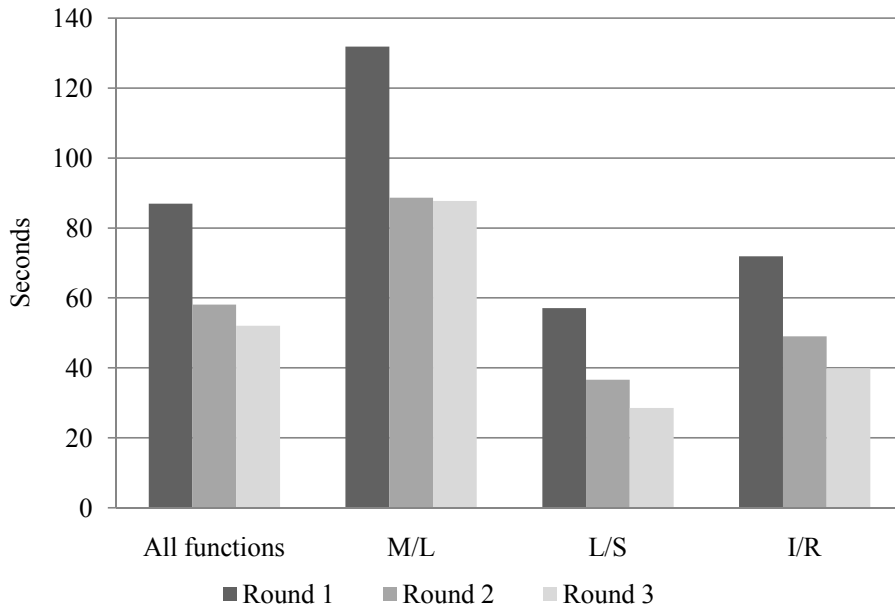


Figure 6.16. Time taken for completing the tasks, averaged per user and per task, displayed for each function type and for each round.

We already observed a learning effect while analyzing how the mental effort required by the users to perform the tasks is influenced by the repetitions of the tasks (Section 6.8.2). The fact that the users were faster in doing the tasks after the first iteration indicates the same learning effect. To get a clearer picture, we analyzed whether this learning effect was more or less intense depending on the type of functions used. Were some functions easier to learn than others? Or was the learning effect the same across the three function types?

To answer these questions, we compared the performance in doing tasks with the same function type and by the same user for different repetitions. The participants, in fact, used each function type in all three repetitions of the task sequence (Figure 6.11). It should be noted that the users did not do the same tasks with the same functions during the different repetitions. Some tasks may have been easier than others, which could have influenced the performance across different

rounds. However, the order of execution of the tasks was rotated across the users to compensate for the effect of tasks that may have been more difficult than others. Therefore, we used a within-subjects ANOVA to study how the efficiency in using a single function type changed across different rounds of execution. In the case of *More/Less of this*, ANOVA revealed an influence of the rounds on the time measurements ($F = 7.690$, $df = 2$, $p = .001$). A post-hoc analysis revealed that users were significantly slower in using *More/Less of this* in the first round compared to the other two, which did not differ from each other. Also in the case of *Lengthen/Shorten this* the effect of rounds on time was significant ($F = 14.412$, $df = 2$, $p = .000$). According to the post-hoc test, the first round was significantly slower than the other two, which do not differ significantly from each other. As for the other two function types, for *Insert/Remove this* the rounds had an influence on the time performances ($F = 14.332$, $df = 2$, $p = .000$). A post-hoc test shows that the first round was slower than the other two, which did not differ from each other. Figure 6.16 summarizes how the time performances were influenced by the rounds, for each function type.

6.8.5 Frequency of usage of the functions

In the previous section, we focused especially on the analysis of the tasks done with a pre-determined choice of functions, inferring conclusions about the editing effectiveness and efficiency of the three different kinds of functions. In this section, we take a closer look at the tasks done with free choice of functions, studying which functions were most frequently used and which less.

Figure 6.17 shows the total number of times each function was used during the execution of the tasks with free choice. For each function, two columns are reported: one corresponding to the number of times that an editing operation was started, the other displays the number of times that an operations was carried out. In some cases, in fact, operations were started and then canceled by the user.

We performed a within-subjects ANOVA to check whether some function types were used significantly more often than others. For this analysis, we calculated for each function type the times the functions were started. The results show a significant difference in usage frequency among the function types ($F = 8.408$, $df = 2$, $p = .001$). The post-hoc analysis revealed that *More/Less of this* were activated significantly less frequently than the other two types, which were not significantly different.

Looking at Figure 6.17, it can be noticed that there is a relatively big gap between the number of times the functions *Lengthen/Shorten this* were started and the number of times they were completed. This means that it happened often that *Lengthen/Shorten this* were started and then canceled. We performed a detailed analysis of the log files to reconstruct why the users would open these functions

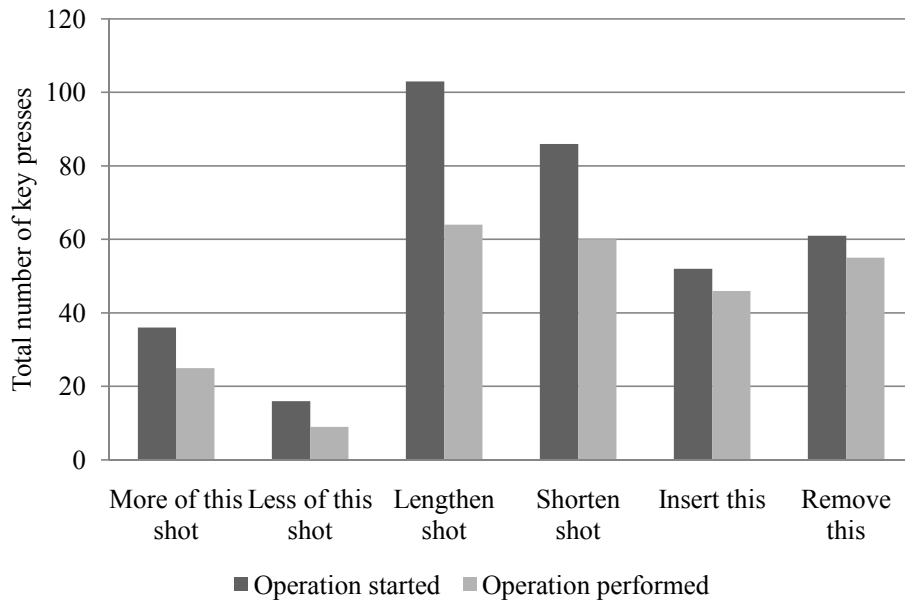


Figure 6.17. Histogram of usage frequency for each function, during the executions of the editing tasks with free choice of function type.

and then cancel them. Table 6.7 shows the patterns that were found analyzing the log files after a start-cancel sequence of *Lengthen/Shorten this*.

Table 6.7. Number of operations done after canceling *Lengthen/Shorten this*.

Situation in the log files	N. of occurrences
Canceled and gone to another task	22
Canceled and opened the opposite function in the couple <i>Lengthen/Shorten this</i> , remained in same task	10
Canceled, reopened with the same function and done, same task	8
Canceled and gone to fine editing, same task	8
Canceled and gone to <i>More/less of this</i> , same task	6
Canceled when out of any task	6
Canceled, reopened with the same function and canceled again, same task	3
Crash after cancellation	1
Shutdown after cancellation	1
Total	65

The most frequent pattern found consists in a cancellation and then an editing

operation on another task (22 occurrences). One could think that the users would leave the task unsolved and go to another. By doing a more in-depth analysis on the log files, it appears that, out of these 22 cases of abandoning a task, 13 happened after the user had completed the task, 8 happened before the user had finished the task, and one happened without the user ever finishing the task. One possible explanation of these frequent cancellations is that these functions were opened and canceled on purpose, to have a better overview of the unedited video. These functions, in fact, are available in shot mode, where the user only has the overview of the shots that belong to the edited video. To edit a shot, a user may feel the need to know also what is in the unedited video material positioned before and after the shot. To gain this overview, a user may open the *Lengthen/Shorten this* functions just to display and browse the timeline with the cut points around the current shot.

Continuing to read Table 6.7, we find that for 10 times the users seem to have confused *Lengthen* shot with *Shorten* shot. It is possible that the two functions were confused with each other by mistake, also because the users remarked that it was a bit difficult to remember all the functions and to distinguish among them.

The following line in Table 6.7 shows that the users canceled a function to reopen it at the same shot and carrying out the task. This may be also explained by the need for more overview on the unedited video, given by the timeline in the dialog window of *Lengthen/Shorten this*.

Reading more in Table 6.7, we find that sometimes the users canceled a *Lengthen/Shorten this* operation to perform the same task with *Insert/Remove this* (8 occurrences) or *More/Less of this* (6 occurrences). It is possible that in some cases the users felt the need to perform a more precise cut than what is possible with *Lengthen/Shorten this*, therefore they went to use the timeline functions.

Finally, the fact that for three times the functions *Lengthen/Shorten this* were canceled, reopened at the same shot and canceled again is probably caused by a defect of the remote control used for the experiment. In fact, if the user keeps the button “ok” of the remote control pressed too long, the system may understand two “ok” events after each other. If the user is pressing “ok” to enter the *Lengthen/Shorten this* functions, the system interprets the first “ok” event as the command to open the dialog window and the second “ok” as the command to perform a cut. Since no cut point is selected, the system understands that the user does not want to cut anything and cancels the operation. Therefore, pressing the “ok” button too long causes a *Lengthen/Shorten this* operation to be quickly started and canceled.

6.9 Discussion of Experiment 2

In this section, some lines of interpretation of the results presented in the previous section are given. First of all, a great amount of evidence has been collected about the performance problems of *More/Less of this* with respect to the other functions and about their inadequacy for home video editing. According to the users' perception, *More/Less of this* were the hardest to learn and use, the most uncontrollable, the most frustrating. Objectively speaking, *More/Less of this* were the most unproductive, the slowest, the least frequently used.

It is very unlikely that the poor performance of *More/Less of this* with respect to the other functions is caused by the lack of predictability, since all three function types were implemented to show to the user a preview of the effect of the editing with the same visual clues. The poor performance of *More/Less of this* must be mainly related to the lack of decisional control. As the participants remarked, these functions do not allow the user to choose at which position the shot should be extended or shortened, at the beginning or at the end. The automation fails to "guess" which decision the user wants to take; this is mainly due to the inability of the system to understand the semantics of the video and to decide accordingly which semantic event is more suitable to be included or excluded. Since *More/Less of this* do not allow the user to select the point in which to edit the shot, the user may feel frustrated while using a function that simply will not listen to him. The options that *More/Less of this* offer are just too few, and the users do not like at all to give the system full control of which point of the shot will be edited. The functions *Lengthen/Shorten this* provide users with control about the position of the shot to edit, and are subjectively and objectively better than *More/Less of this*. These evidences show that Hypothesis 6.4 (p. 133) is correct: if *More/Less of this* are modified to have more decisional control, the related user satisfaction will increase.

While *More/Less of this* collected negative remarks, the other two kinds of functions were positively evaluated by the users. In general, *Lengthen/shorten this* were judged to be as good as *Insert/Remove this*. We saw that some users preferred *Lengthen/Shorten this* over *Insert/Remove this* because the first type was perceived as faster than the second in editing tasks (and it was indeed significantly faster). Other users preferred *Insert/Remove this* over *Lengthen/Shorten this* because of the better overview of the unedited video that they could get with the timeline functions. The reason for preferring *Insert/Remove this* over *Lengthen/Shorten this* was not the increased decisional control. In fact, no user mentioned the control as a reason for preferring *Insert/Remove this*. This is an evidence of the fact that users do not mind losing the possibility of precisely selecting the start and end points of a cut. They like the system choosing the cut points for them and proposing them

a set of possible alternatives, like in *Lengthen/Shorten this*. Since the introduction of automatic cut-point selection did not affect the user satisfaction and increased the editing efficiency, this level of automation can and should be considered as an alternative to the many timeline-based video editing interfaces of PC programs that the users find so time-consuming and complicated.

The analysis of the mental effort and of the time needed for the editing shows a learning effect while using the functions. Especially after the first round of usage, the interaction with the functions was less cumbersome and quicker. This was evident in particular for *Lengthen/Shorten this* and *Insert/Remove this*. In the case of *More/Less of this*, the time performances decreased less with the repetitions of usage, and the mental effort in using the functions did not decrease at all. This could be explained by the fact that the users were getting more and more acquainted with the tasks and with the video material and therefore were going faster in performing the tasks, but *More/Less of this* remained difficult despite the repetitions in usage. Figuring out how to make these latter functions work the way the users wanted, remained mentally demanding even if the users had multiple attempts to learn the functions.

Possible improvements to the editing functions can also be proposed. We have seen that *More/Less of this* offer too few options to the users. The number of editing options in *More/Less of this* can be increased, for example by letting the user select whether he or she wants to edit the shot at the beginning or at the end. Otherwise, or in addition, the system could *learn* from the user which video segments are preferred and which are not. For example, if the user activates *More of this*, the system proposes to add a video segment and the user cancels the operation, the system may learn that this particular video segment is perceived as not important in that particular context. The next time the user will activate *More of this*, the system will therefore propose to add another segment to the shot. This would perhaps increase the perceived behavioral control of *More/Less of this*, because the user could perceive that he/she is able to influence the automatic reasoning of the system. Perhaps, with these changes, *More/Less of this* would challenge the other two function types with more success. Concerning *Lengthen/Shorten this*, it would be an interesting idea to let users refine the position of the chosen cut point, like some participants proposed. Also, the fact that users perceive the overview provided by *Insert/Remove this* as better than *Lengthen/Shorten this* could be addressed. In shot mode, a system could be studied to quickly show the excluded clips available in the neighborhood of one shot. Otherwise, *Lengthen/Shorten this* could be made available also in timeline mode. Finally, it is possible to think about editing functions that adapt the level of automation to the user: the editing interface could switch from *More/Less of this* to *Lengthen/Shorten this* if the user desires more cut options, or from *Lengthen/Shorten this* to *Insert/Remove this* if the user asks for more

precision in the editing.

To compare the three function types, in this experiment objective and subjective variables have been measured. We note that the objective and the subjective evaluations agree with each other to a large extent. The functions *More/Less of this* perform much worse than the other two function types in the perceived control, in the perceived ease of use but also in the objective editing efficiency. Furthermore, the functions *Lengthen/Shorten this* are perceived as faster than *Insert/Remove this*, and objective measurements show that they really are. Since many objective and subjective variables have been independently measured, and since these measures are consistent with each other, we can be particularly confident about our results. We have shown how, in technology for home entertainment, reducing too much the decisional control in favor of a not well-designed automation can cause dissatisfaction and frustration in the users and very poor performances.

To run this experiment, we recruited users without a great experience in video editing and without technical knowledge in video or in PCs. It is possible that the preference users showed for the function type has to do with their level of experience in video editing. Perhaps users with more technical skills or with more experience in video editing would have preferred the timeline, full-control functions over the other two types of functions. Skilled users, in fact, may prefer interfaces that provide more control at the cost of more difficulty and technical detail.

This experiment consisted in a direct comparison between three different designs of semi-automatic functions for home video editing. Evidence has been provided that shows how the typical timeline-based video editing interfaces can be improved by implementing functions that automatically calculate a set of cut-points among which the users can choose. On the other hand, the choice of the position to be edited in the video should be left to the user. Having the system selecting which position to edit in the video generates a great decay in performances and confusion and frustrations in the users. These results were obtained by inviting users to perform pre-defined editing tasks on a home video the users were not familiar with. This setup may be not completely realistic. Therefore, to strengthen our findings, the comparison between the three function types could be repeated by letting users edit their own videos. Other types of automation remain to be tried, such as “learning” editing functions or functions that adapt the automation level to the characteristics and demands of the particular user.

7

Conclusions and suggestions for future work

In this chapter the conclusions of the present research work are drawn. The experimental methods used are critically reviewed, and design guidelines for a home video editing application are given. Moreover, some suggestions for further research are discussed.

7.1 Summary of experimental findings

The goal of this research consisted in finding an easy to use, effective and efficient solution for home video editing. To accomplish it, a semi-automatic application for editing videos has been developed, following a user-centered approach.

Home video editing is nowadays perceived by amateur users as very time-consuming and often difficult. This probably comes from the fact that tools for home video editing are too much modeled on professional video editing systems, instead of being designed for amateurs. In Chapter 2 we have reviewed the literature relevant to home video editing. We have seen that, although many ideas have been proposed to make home video editing easier and faster, very little research has been done to understand what users of home videos deem as easy and useful. To gain more insights in behavioral aspects and needs related to home video editing,

we have surveyed 181 users of home videos by means of a questionnaire on the Internet. By analyzing the answers collected through the questionnaire, we have found that editing of home videos is done often, even though users still perceive it as a very time-consuming activity. A disadvantage of common video editing tools is that they are all PC-based. As a consequence, users that are not very acquainted with PCs tend to edit videos less frequently than users with high PC expertise, and to perceive video editing as very difficult. Many respondents would like to edit videos on a TV and to combine their content with friends' videos or pictures.

While collecting the results of the survey, we developed a semi-automatic solution for home video editing, called *Edit While Watching*. We have focused on designing an easy to use and efficient system. To do that, we have studied a suitable combination of automatic and interactive functionalities. In Chapter 3 the algorithmic part of *Edit While Watching* has been described. We have shown that the user can input the unedited video to *Edit While Watching* and get an edited version of it, automatically calculated. The system edits the video by structuring it in small video segments and subsequently selecting suitable video segments according to three criteria. These are the quality of the segments, the extent to which these segments uniformly represent the unedited video, and the presence of faces. The unedited video is structured by using also camera motion information. This information includes the detection of “zoom in” and “zoom out” sequences in the video. To detect these sequences, a novel algorithm has been devised that outperforms the state of the art.

In Chapter 4 the interface of *Edit While Watching* has been described. In fact, the video automatically edited is just a preliminary version of the final result. To reach a result that satisfies the author of the video in his or her subjective wishes, the user refines the edited video via an interface that works with a TV and a few keys on a remote control. This interface is designed to avoid any unnecessary technical detail and to execute the editing operations the user does in real time, without re-encoding, in order to be intuitive and efficient. Moreover, the interface implements two *semi-automatic* editing functions, designed to help the user in the process of content selection. With these functions, the user simply says that more content should be added to a certain shot, and the system automatically selects which video content is included in the shot. Therefore, the user does not need to manually select content by for example performing a cut in a timeline overview.

The usability of *Edit While Watching* has been evaluated in a formative test with eight users. The participants judged the interface to be simple and easy to learn and use. However, they remarked that the system was too rigid and that they did not feel in control of the editing process. Also, they wanted to have more overview of the video. To overcome these shortcomings, a second version of the *Edit While Watching* interface was created, which has been described in Chapter 5. In this

version, the user can access the video through an overview structured in three levels of detail. In the first level, the user browses the *scenes* of the video, represented by “piles” of keyframes in a visualization panel. In the second level, the user browses the *shots* inside one single scene, represented as keyframes sequentially displayed. In the third level, the user browses a *timeline* representing in detail the unedited video and the clips that are currently included or excluded from the edited version. In each of these three levels, editing functions are also available; these functions become also more and more detailed as the user goes towards the timeline mode, the most detailed one. In timeline mode, the user is provided with all the detail to edit the video. While browsing the timeline, he or she can cut a portion of it by selecting start and end points. The selected portion of video is then inserted or removed from the edited video. The semi-automatic editing functions of the first version of *Edit While Watching* were also present in the second version.

A user evaluation of the second version of our system was run with nine users. They were invited to bring their own home videos and to edit them according to their own creativity, using *Edit While Watching*. The test revealed that the shortcomings of the first version were indeed overcome in the second version. Moreover, the participants were able to edit their videos in a short time: the editing time was 1.1 to 1.7 times longer than the duration of the edited video material. From the Internet-based questionnaire, it appears that video editing can take much longer. However, the semi-automatic editing functions were used very rarely.

Although the overall evaluation of the usability of *Edit While Watching* was positive, we wanted to investigate why the users liked it, and which aspects of the system contributed most to its positive evaluation. In particular, research was done to assess which kind of automation adds value to *Edit While Watching*. This research has been described in Chapter 6 and consisted of two experiments.

The first experiment was aimed at assessing to what extent a timestamp-based algorithm for creating the overview of the video scenes helps the users in searching and browsing tasks. Two versions of *Edit While Watching* were created, with two different scene overviews. The first version featured a scene overview calculated exploiting timestamp information. The second version of the scene overview was obtained by trivially dividing the unedited material into a number of equally long scenes. The experiment was run with 40 users: one group of 20 performed browsing and search tasks with the timestamp-based overview, the other group of 20 users carried out the same tasks using the other scene overview. The results have shown that exploiting timestamp information for calculating the scenes gives a measurable improvement in the performance of the users for browsing and searching.

The second experiment focused on the semi-automatic editing functions. The test on the second version of *Edit While Watching* featured the functions “More

of this” and “Less of this”, that used automation for helping the user in selecting interesting content or removing unwanted content. However, these semi-automatic functions were rarely used, while the full-control timeline functions were used about ten times more often. To explain this observation, we developed the hypothesis that the semi-automatic functions were discarded because they give too few editing options or, in other words, because they give too little decisional control. To test this hypothesis, three function types with different levels of balance between automation and decisional control were devised. In the first type of functions, the system takes all the decisions about which content to edit and how, and the user can only confirm or reject the system’s proposal. In the second function type, the system gives the user a set of cut options automatically calculated, and the user selects one of the options. The third type of functions corresponds to the full-control timeline editing: the user takes all the decisions about a cut: start and end points, insertion or removal. We wanted to compare these three function types with respect to the perceived ease of use, the perceived control, the mental effort, and the objective editing performance. We invited 25 users to perform editing tasks with each one of the three function types. Results showed that the type of function with the highest level of automation performed significantly worse than the other two function types, in all the tested aspects. The other two function types were, in general, equally liked by the users. However, the function type with intermediate level of automation allowed the users to be significantly faster than the full control functions in performing the editing tasks. We conclude therefore that editing functions which automatically select a set of cut options can be considered as a valid alternative to the many timeline-based full-control editing interfaces available on the market.

7.2 Some reflections on the experimental methods

The results obtained in this research work are based on four experiments. The first two are the formative tests on EWW1 and EWW2, described in Chapters 4 and 5. These experiments were run to find most of the usability flaws of the EWW prototypes. The other two experiments are described in Chapter 6. They aim at comparing performances and behavioral aspects of EWW under different conditions. The first experiment of Chapter 6 compares two algorithms for partitioning a video in scenes. The second compares three levels of automation versus user control in the editing functions.

We make now some considerations about the number of participants we chose in the four experiments. In the first two formative tests, we recruited eight and nine users respectively. A well-known guideline for usability experiments [Nielsen and Landauer, 1993] states that at least five users are needed to find 80% of a system’s

defects. However, recently it has been shown that, in order to expose 80% of the problems of a system, a usability test with 15-20 participants may be needed [Schmettow, 2008]. The optimal number, in fact, depends on the variance of the probabilities that the defects of a system are found in one usage session. Still, it is accepted as true that a test with eight users will reveal most of the usability flaws in a system. We are therefore confident of the significance of the results of our formative tests.

In the two comparative experiments, 40 and 25 users were recruited, respectively. The number of participants is higher than in the formative tests, since obtaining a statistically significant comparison of two systems is a more demanding goal than finding most of the usability flaws. The first comparative experiment featured a between-subjects design, while the second was set up as a within-subjects design. In a between-subjects design, since two or more group averages are compared, the variance within the single groups can make the comparison more difficult and therefore decrease the test power. A within-subjects design does not have this problem. Therefore, we recruited a higher number of users for the between-subjects design (40 instead of 25) to increase the test power.

Finally, some comments are made about the choice of video material in the four experiments. In the two comparative experiments, the same video material was adopted for all the participants. This is in fact the only way to implement a fixed set of tasks for all the participants, on which different interfaces or functions can be compared. Coming to the two formative tests, the video material was fixed in the experiment on EWW1, and was changing with each subject in the experiment on EWW2. We chose to fix the video material in the first formative test, in order not to introduce extra variables such as type of event in the video, video duration, video format, etc., that could have made the test more difficult to assess. The participants may have felt not involved in the video material, since it was not their own. However, they judged the test scenario as realistic. After the experience of the first formative test, we increased the variables in the second, by inviting the participants to bring along their videos to edit. In this way, the test was more interesting for the participants, and they engaged in a longer editing time. Furthermore, we had a more realistic evaluation of EWW's usability, since a video editing system should be able to support videos of many different durations and formats.

7.3 Design guidelines

This thesis has described the creation, design and iterative improvement of an application for easy and fast home video editing. Here some design guidelines for good video editing systems for amateur users are given, based on the experience and on the information collected.

First, a home video editing system should *automatically structure* the input video in small video segments, and optionally in shots and/or scenes. This is of great help for the user for browsing and editing the video; furthermore, it saves the user from knowing many technical details that are typical of professional video editing systems, such as detailed timelines, frame-by-frame browsing and frame numbers. The user can browse the structural elements of the video which are automatically built by the systems (video segments, scenes, etc.). Video editing becomes therefore more similar to browsing a collection of pictures and selecting or discarding the wanted or unwanted ones.

A second design guideline consists in *eliminating details about single frames*, or keeping it as an extra option for experts. Users simply do not need it. In Chapter 6 it has been shown how editing functions based on small video segments (Lengthen/Shorten this, see Figure 6.9 p. 138) can allow more efficiency than editing functions based on frame-level cuts (Insert/Remove this, see Figure 6.10 p. 138). We are also convinced that users are not interested in interacting with frames. They want to edit videos in terms of semantic elements, like people or happenings. This guideline can be also extended to eliminating any technical detail that is not strictly needed, such as parameters of the color filters or of the video transitions, parameters of codecs, etc.

A third design guideline consists in *implementing an effective and scalable overview of the video*. Users can be helped considerably if the overview on the video is expanded with respect to what common editing programs offer (a timeline view on the video and a storyboard representing the camera takes). EWW2 implements an overview system which is scalable in three levels: the scenes, the shots inside one scene, and the timeline with the video segments inside and around one shot (see Section 5.2). This scalability is important, since users perform tasks that require a broad overview (like browsing and searching) but also a detailed overview (like cutting out a short video segment).

Another design guideline consists in *implementing a rich set of editing functionalities*. Users are not only interested in editing speed and simplicity, as we have observed during the experiment on EWW2 in Chapter 5. Most users see video editing also as a fun activity, nice in itself. They want to have at their command many tools and options to enrich the video and to express their creativity: video effects, transitions, music, combination with pictures, with 3D shapes, with text, etc. These functions should be easy to use but pleasant and entertaining. The accent should be not on the technical precision but on the fun. A function valued by many users is the easy creation of a final result like a DVD or a file to share on the Internet.

The last design guideline consists in *moving from a PC-based to a TV-based design*. This opens the possibility of editing video to users without a great expertise with PCs. During our experiments, many users remarked that they would edit

videos more often if they could do it on a TV. Furthermore, the remote control of a TV is usually an easier interface than the keyboard and mouse of a PC, having fewer buttons and no pointing systems. Some aspects of EWW2's interface can be taken as examples of TV-based design for video editing: the main central window with the video playing, the text feedback for each key press, the graphic animations that effectively convey to the user feedback about the navigation in the video (overview elements that slide back and forth, or that move apart from each other when the user zooms into a finer level of overview).

7.4 Suggestions for future work

In this section some suggestions about how the work in this thesis could be continued are given. These suggestions concern possible improvements on *Edit While Watching*, and new directions in the research on automatic video editing and on the user needs related to home videos.

7.4.1 Evaluating and improving *Edit While Watching*

To further improve *Edit While Watching*, a very interesting experiment consists in performing a field trial. *Edit While Watching* could be installed in the living rooms of ten or twenty families, who then could use it for one month or more. The field trial would allow to detect more flaws in the usability of *Edit While Watching*, and it would provide useful information about which usage patterns are more frequent, how the user satisfaction changes over time, how frequently home videos are edited, which type of content is edited more frequently, in which occasions and why, etc. Testing *Edit While Watching* for months in people's living room could also reveal whether people prefer to sit alone to edit videos or whether the family members or friends sit together to decide about the edited version of the video. Perhaps the experiences of watching home videos and of editing home videos would blend together into one experience: while the author of the video shows the holidays to his or her family, some family members could come with suggestions about applying a music track or text to a shot or about mixing the video with clips and pictures from the former year; these suggestions could be implemented right away.

The *Edit While Watching* prototype developed in the context of this thesis is mature enough to be installed in people's homes for field trials without too much implementation effort. To make a full-featured product out of *Edit While Watching*, functions for enriching the video with transition, music and effects would be needed. Also, functions for saving the edited video in various formats, including the creation of a DVD, should be implemented.

In Chapter 6, the balance between automation and decisional control in *Edit While Watching*'s semi-automatic functions has been studied. Further research

could also address the *behavioral* control of the semi-automatic functions, or the effectiveness with which the editing functions allow the user to author the video as they wish. Perhaps, if the design of the semi-automatic functions is improved (in the algorithmic or in the interactive aspect), then the perceived behavioral control will increase (the perception that users have means to author their videos as they wish) and/or the editing effectiveness will increase. To give an example, the semi-automatic editing functions of *Edit While Watching* could be improved by implementing relevance feedback, or by letting the system *learn* the user preferences for video content while the user is editing the video. For example, if a user removes a video segment that was included, the system could learn that the user is not interested in that segment. Or, if the user does a lot of fine editing around one camera take or one shot, the system could learn that the user considers that portion of content as very important. The system could take into account that the importance of the video clips may change from one editing session to another. The system could check whether the clips often perceived as important by the user share common color features, or texture features, or camera motion patterns. If this is the case, then the system can propose to the user to include in the edited video content with similar features.

Furthermore, if *Edit While Watching* is used in the living room of a house by the whole family, then the system could learn which video content is preferred by each member of the family. Moreover, if *Edit While Watching* is used to edit the video collection of the family over one or more years, a summary of the most interesting and important shots of the last year or of the last five or ten years could be automatically calculated and proposed to the users by *Edit While Watching*.

In Chapter 6, the semi-automatic editing functions “Lengthen this” and “Shorten this” have been introduced. They are based on allowing the user to select a cut point among a set of possible options that the system preselects as suitable. The user evaluation showed that these functions are liked as much as the full-control, timeline-based editing functions; also, “Lengthen” and “Shorten this” allowed to perform editing tasks faster than the full-control editing functions. This result suggests that automation can add value to the timeline-based editing interface used by many commercially available applications. More research would be needed to strengthen this result and to gain more insights about it. For example: is this result still true when people edit their own videos according to their own wishes? This result, in fact, was obtained from users performing a predefined set of editing tasks, that may not be fully realistic (see Section 6.9). Also: how can the interaction provided with the functions “Lengthen/Shorten this” be improved? The functions “Lengthen/Shorten this” employ automation to select a set of possible cut points. Can the automation behind the cut point selection be improved?

7.4.2 Improving the automatic home video editing algorithm

More research could be done also for the evaluation and the improvement of the automatic video editing algorithm of *Edit While Watching*. A user evaluation could be performed to assess to what extent the algorithm selects the clips that the users really prefer. Moreover, additional features and metadata could be used to calculate the edited video. Timestamp information could be used to calculate clusters of camera takes which were captured at very close moments. Then, the requirement of temporal uniformity could be expressed by saying that each cluster of camera takes with neighboring timestamps should be represented equally.

Also, audio analysis could be exploited to improve the automatic video editing. The audio could be classified and segmented into speech segments, music segments, noise segments or silence segments. This would generate new segment boundaries that would be taken into account into the segmentation of the unedited home video. Also, an additional evaluation criterion could be added to the evaluation function $\Omega(E)$: if the edited video E contains shots that abruptly interrupt speech, then E could be penalized by receiving a low score.

Automatic home video editing could be improved also by involving motion continuity rules [Zettl, 1999]. These rules can provide an additional set of criteria to be used to evaluate an edited video E . Two consecutive shots of E should contain camera movements consistent with each other. If, for example, a shot of E ends with a pan left and the next shot begins with a pan right, the transition between the two shots is unpleasant to see, therefore it should not be selected. Motion continuity could also be used to automatically insert video transitions between two shots. An unpleasant situation, like a shot ending with a pan left followed by one beginning with a pan right, could be automatically repaired by terminating the first shot with a fade-to-black transition and beginning the second with a fade-from-black.

7.4.3 Deepening the insights in user needs for home video editing

In Section 2.3.3, our findings about the user needs related to home video editing have been reported. The respondents of our survey indicated a clear interest for “combining their own videos with own pictures, friends’ content or content from the Internet” (p. 34). This observation could be clarified and deepened with further research. Which requirements are considered by the users as important: mixing videos with pictures? Or mixing videos with videos of friends? Or adding pictures and videos from the Internet to their personal videos? And which kind of Internet content: user-generated content or professional content? Interviews with users could be run to answer these questions.

Additionally, mock-ups or prototypes of systems for collaborative editing could be created and shown to authors of home videos. This would be useful for under-

standing to what extent collaborative editing is of interest for authors of home videos, and which user requirements and functionalities are more important than others. As an example of such mockups, a prototype of a web service could be created where a group of users uploads home videos. When a user uploads a video, the system automatically finds video content uploaded by other members with similar metadata; for example, videos shot in the same place or at the same time. The system may then propose to the user to mash up his or her video with videos of other members, or it could even automatically generate a mashup. If the videos to mash up were shot at around the same time and at the same event, the mashup system described in [Shrestha et al., 2006] and [Shrestha et al., 2007] may be employed. This system exploits video synchronization based on flashes and on audio to automatically mash up multiple videos. The system could use the same interaction metaphors of *Edit While Watching*, exploiting TV and remote control.

Bibliography

- AARTS, E., AND E. DIEDERIKS [2006], *Ambient Lifestyle, from concept to experience*, BIS Publishers, Amsterdam, the Netherlands.
- ACHANTA, R. S. V., W.-Q. YAN, AND M. S. KANKANHALLI [2006], Modeling intent for home video repurposing, *IEEE Multimedia* **13**, 46–55.
- ADAMS, B., AND S. VENKATESH [2005], Situated event bootstrapping and capture guidance for automated home movie authoring, *Proceedings of the 13th annual ACM International Conference on Multimedia (MULTIMEDIA '05)*, Singapore, 754–763.
- ADAMS, B., S. VENKATESH, AND R. JAIN [2005], Imce: Integrated media creation environment, *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)* **1**, 211–247.
- ANER-WOLF, A., AND L. WOLF [2002], Video de-abstraction or how to save money on your wedding video, *Proceedings of the sixth IEEE Workshop on Applications of Computer Vision, (WACV 2002)*, Orlando, Florida, USA, 264–268.
- AVERILL, J. R. [1973], Personal control over aversive stimuli and its relationship to stress, *Psychological Bulletin* **80**, 286–303.
- BARBIERI, M. [2007], *Automatic Summarization of Narrative Video*, Ph.D. thesis, Eindhoven University of Technology, November 2007.
- CAMPANELLA, M., AND J. HOONHOUT [2008], Understanding behaviors and needs for home videos, *Proceedings of the “British HCI” Conference in Liverpool, United Kingdom (Volume 2)*, 23–26.
- CAMPANELLA, M., H. WEDA, AND M. BARBIERI [2007], Edit while watching: home video editing made easy, *Proceedings of the IS&T/SPIE Conference on Multimedia Content Access: Algorithms and Systems, San Jose, California, USA. SPIE 6506, 65060L (2007)*, IS&T/SPIE.
- CAMPANELLA, M., H. WEDA, M. BARBIERI, A. KOHLRAUSCH, AND J. HOONHOUT [2009], Easy home video editing following a user-centered approach, *Submitted to ACM Transactions on Computer-Human Interaction (TOCHI)*.
- CASARES, J., C. LONG, B. MEYERS, R. BHATNAGAR, S. M. STEVENS, L. DABBISH, D. YOCUM, AND A. CORBETT [2002], Simplifying video

- editing using metadata, *Proceedings of the 4th Conference on Designing Interactive Systems: Processes, Practices, Methods and Techniques, London, England*, 157–166.
- CHALFEN, R. [1987], *Snapshot Versions of Life*, Bowling Green State University Popular Press, Bowling Green, Ohio, USA.
- DAVIS, M. [2003], Editing out video editing, *IEEE Multimedia* **10**, 54–64.
- FROHLICH, D. M., A. KUCHINSKY, C. PERING, A. DON, AND S. ARISS [2002], Requirements for photoware, *Proceedings of the ACM Conference on Computer-Supported Cooperative Work, New Orleans, Louisiana, USA*, 166–175.
- GATICA-PEREZ, D., A. LOUI, AND M.-T. SUN [2003], Finding structure in home videos by probabilistic hierarchical clustering, *IEEE Transactions on Circuits and Systems for Video Technology* **13(6)**, 539–548.
- GENTNER, D. [1992], Interfaces for learning: Motivation and the locus of control, In F. L. Engel, D. G. Bouwhuis, T. Bösser, & G. d Ydewalle (Eds.), *Cognitive modeling and interactive environments in language learning*. Berlin: Springer.
- GIRGENSOHN, AN., S. BLY, F. SHIPMAN, J. BORECZKY, AND L. WILCOX [2001], Home video editing made easy - balancing automation and user control, *Proceedings of the International Conference on Human Computer Interaction (INTERACT '01), Tokyo, Japan*, 464–471.
- GIRGENSOHN, A., J. BORECZKY, P. CHIU, J. DOHERTY, J. FOOTE, G. GOLOVCHINSKY, S. UCHIHASHI, AND L. WILCOX [2000], A semi-automatic approach to home video editing, *Proceedings of the 13th annual ACM Symposium on User Interface Software and Technology (UIST '00), San Diego, California, United States*, 81–89.
- HINDS, P. J. [1998], *User Control and Its Many Facets: A Study of Perceived Control in Human-Computer Interaction*, Technical report HPL-98-154, Hewlett Packard.
- HUA, X.-S., AND S. LI [2006], Interactive video authoring and sharing based on two-layer templates, *Proceedings of the 1st ACM International Workshop on Human-Centered Multimedia (HCM '06), Santa Barbara, USA*, 65–74.
- HUA, X.-S., L. LU, AND H.-J. ZHANG [2004], Optimization-based automated home video editing system, *IEEE Transactions on Circuits and Systems for Video Technology* **14**, 572–583.
- HUA, X.-S., AND H.-J. ZHANG [2004], Content and transformation effect matching for automated home video editing, *Proceedings of the International Conference on Image Processing, (ICIP '04), 3, Singapore*, 1613–1616.
- ISO/IEC [1998], *Ergonomic requirements for office work with visual display terminals (VDT)s - Part 14 Menu dialogues, ISO/IEC 9241-14*.

- KIRK, D., A. SELLEN, R. HARPER, AND K. WOOD [2007], Understanding videowork, *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, San Jose, California, USA*, 61–70.
- KIRKPATRICK, S., C. D. GELATT, AND M. P. VECCHI [1983], Optimization by simulated annealing, *Science* **220**, 671–680.
- KROON, B. [2006], Pose estimation and face detection in video material, *Proceedings of the MultimediaN Intern Day, Amsterdam, The Netherlands*.
- LANGER, E. J. [1975], The illusion of control, *Journal of Personality and Social Psychology* **32**, 311–328.
- LIENHART, R. [1999], Abstracting home video automatically, *Proceedings of the seventh ACM International Conference on Multimedia (Part 2) (MULTIMEDIA '99), Orlando, Florida, United States*, 37–40.
- LOCKERD, A., AND F. MUELLER [2002], Lafcam: leveraging affective feedback camcorder, *CHI '02 extended abstracts on Human factors in computing systems*, 574–575.
- MANJUNATH, B. S., P. SALEMBIER, AND T. SIKORA [2002], *Introduction to MPEG-7 - Multimedia Content Description Interface*, John Wiley & Sons, New York, NY, USA.
- MASTHOFF, J. [2002], Design and evaluation of a navigation agent with a mixed locus of control, In S. A. Cerri, G. Gouardres, F. Paraguau (Eds.), *Intelligent Tutoring Systems: Proceedings ITS 2002*. Berlin: Springer Verlag.
- MEI, T., C.-Z. ZHU, H.-Q. ZHOU, AND X.-S. HUA [2005], Spatio-temporal quality assessment for home videos, *Proceedings of the 13th annual ACM International Conference on Multimedia (MULTIMEDIA '05), Singapore*, 439–442.
- MORRIS, S. A., AND T. E. MARSHALL [2004], Perceived control in information systems, *Journal of Organizational and End User Computing* **16**, 38–56.
- NIELSEN, D., AND T. K. LANDAUER [1993], A mathematical model of the finding of usability problems, *Proceedings of the ACM INTERCHI 93 Conference, Amsterdam, The Netherlands*, 206–213.
- NORMAN, D. A. [1983], Design rules based on analyses of human errors, *Communications of the ACM* **26(4)**, 254–258.
- NORMAN, D. A. [1993], *Things that make us smart: defending human attributes in the age of machines*, Basic Books, New York, NY, USA.
- OAMI, R., A. B. BENITEZ, SHIH-FU CHANG, AND N. DIMITROVA [2004], Understanding and modeling user interests in consumer videos, *Proceedings of the IEEE International Conference on Multimedia and Expo, 2004 (ICME '04), 2, Taipei, Taiwan*, 1475–1478.
- PARASURAMAN, R., T. B. SHERIDAN, AND C. D. WICKENS [2000], A model for types and levels of human interaction with automation, *IEEE Transac-*

- tions on Systems, Man and Cybernetics, Part A.* **30**, 286–297.
- ROSENBAUM, M., AND K. BEN-ARI SMIRA [1986], Cognitive and personality factors in the delay of gratification of hemodialysis patients, *Journal of Personality and Social Psychology* **51(2)**, 357–364.
- SARTER, N. B., AND D. D. WOODS [1995], How in the world did we ever get into that mode? mode error and awareness in supervisory control, *Human Factor* **37(1)**, 5–19.
- SCHMETTOW, M. [2008], Heterogeneity in the usability evaluation process, *Proceedings of the British Human-Computer Interaction (HCI) Conference (vol. 1)*, Liverpool, UK, 89–98.
- SHRESTHA, P., H. WEDA, AND M. BARBIERI [2007], Synchronization of multi-camera video recordings based on audio, *Proceedings of the 15th annual ACM International Conference on Multimedia, Augsburg, Germany*, 545–548.
- SHRESTHA, P., H. WEDA, M. BARBIERI, AND D. SEKULOVSKI [2006], Synchronization of multiple videos using still camera flashes, *Proceedings of the 14th ACM International Conference on Multimedia, Santa Barbara, USA*, 137–140.
- SIEGEL, S., AND N.J. CASTELLAN [1988], *Nonparametric Statistics for the Behavioral Sciences*, McGraw-Hill, New York, NY.
- TAKEUCHI, Y., AND M. SUGIMOTO [2007], User-adaptive home video summarization using personal photo libraries, *Proceedings of the 6th ACM International Conference on Image and Video Retrieval (CIVR '07), Amsterdam, the Netherlands*, 472–479.
- UEHARA, K., M. AMANO, Y. ARIKI, AND M. KUMANO [2004], Video shooting navigation system by real-time useful shot discrimination based on video grammar, *Proceedings of the International Conference on Multimedia & Expo (ICME 2004), Taipei, Taiwan*, 583–586.
- VAREKAMP, C. [2004], *Efficient estimation of scaling and rotation in video*, Internal technical note PR-TN-2004/00707, Philips Research Europe.
- VENDRIG, J., AND M. WORRING [2002], Systematic evaluation of logical story unit segmentation, *IEEE Transactions on Multimedia* **4(4)**, 492–499.
- VIOLA, P., AND M. JONES [2001], Rapid object detection using a boosted cascade of simple features, *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR 2001), Kauai, USA*, 511–518.
- WANG, P., T. WANG, J. LI, AND Y. ZHANG [2007], Information-theoretic content selection for automated home video editing, *Proceedings of the IEEE International Conference on Image Processing, (ICIP '07), 4, San Antonio, Texas, USA*, 537–540.

- WEDA, H., AND M. CAMPANELLA [2007], Use study on a home video editing system, *Proceedings of the 21st Annual Conference of the British Human Computer Interaction Group (HCI '07), Lancaster, UK*, 123–126.
- YIP, S., E. LEU, AND H. HOWE [2003], The automatic video editor, *Proceedings of the 11th ACM International Conference on Multimedia (MULTIMEDIA '03), Berkeley, California, USA*, 596–597.
- ZETTL, H. [1999], *Sight, Sound, Motion: Applied Media Aesthetics*, Wadsworth Publishing Company, Belmont, California, USA.
- ZIJLSTRA, F.R.H., AND L. VAN DOORN [1985], *The construction of a scale to measure perceived effort*, Technical Report ISN 6105/6107, NABS N10, Delft University of Technology, the Netherlands.

A

Questionnaire about use of home videos

In this appendix the Internet-based questionnaire developed for the research described in Section 2.3.2 is shown. The questionnaire was put online for one year using an Amsterdam-based web service for hosting of surveys (www.formdesk.com). Below, the text and the questions that were put on Internet are presented.

Beginning of the questionnaire:

Nowadays more and more people like to keep memories of their lives by shooting pictures and videos. Pictures are mostly used, but people increasingly capture personal videos, using devices like mobile phones, digital still cameras, and webcams.

My name is Marco Campanella. I am an employee of Philips Research Eindhoven, and a Ph.D candidate. I am researching semi-automatic home video editing and I am collecting information on the place that home videos have in people's life. I am interested in knowing why people use home videos, how often video is captured, whether and how it is edited, how much it is shared.

If you have even a basic experience in shooting or manipulating home videos, you can help me by filling this questionnaire. Completing the questionnaire will take maximum 15 minutes. Moreover, at any moment you can stop filling the questionnaire, save your answers and finish it another time.

Thank you very much, enjoy!

Marco Campanella
marco.campanella@philips.com

Capturing home videos

1. How often do you shoot home videos?

- Once a week or more often.
- One to three times a month.
- Six to eleven times a year.
- One to five times a year.
- Once a year or less often.

2. What do you mostly shoot? (you can choose multiple answers)

- Parties or experiences with friends.
- Life with my family.
- Weddings.
- Holidays.
- Artistic events (e.g. concerts, live performances, etc.).
- Sport events.
- Other (please specify): . . .

3. How long usually are the videos you shoot? Please indicate the duration for each option that you selected in the former question.

	Less than 5 min	5-15 min	15-45 min	45-90 min 90 min	More than
Parties	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Family life	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Weddings	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Holidays	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Artistic events	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Sport events	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Others	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

4. Which devices do you use for shooting your home videos? (you can choose multiple answers)

- Digital camcorder.
- Digital still camera.
- Mobile phone.
- Webcam.
- Other (please specify) : ...

5. What do you like to do with home videos? (you can choose multiple answers)

- I like to keep memories of my life.
- I like to share my experiences with my friends, by for example posting a video on Internet or giving DVDs to friends.
- I use my home videos for professional reasons, like shooting videos of important situations or meetings at work.
- I use my home videos for artistic purposes.
- I like to do other things with my home videos (please specify) : ...

6. Please indicate how much do you agree to the following sentences:

	Totally disagree	Slightly disagree	Neutral	Slightly agree	Fully agree
Home videos convey experiences better than photographs	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Home are more enjoyable than collections of photographs	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I like to have the videos I captured to watch on TV	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Home videos are in general not appealing, because of their low quality	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Capturing good home videos is easy	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Keeping my home videos organized and safely stored somewhere is difficult	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

	Totally disagree	Slightly disagree	Neutral	Slightly agree	Fully agree
I find it easy to retrieve and watch my favorite home videos	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I like collections of photographs better than home videos	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Watching home videos

7. Once you shoot a home video, how often do you watch it later?

- Never.
- Once.
- 2-5 times.
- 6-10 times.
- More than 10 times.

Please specify if it depends on the type of video, for example if you watch particular videos or particular events more often than others: ...

8. *If you answered “never” or “once” to the former question, please answer also to the question below; otherwise skip to question 9.*

What are the most important reasons for rarely watching your home videos? (You can choose multiple answers)

- Unedited home videos lack in quality.
- Unedited home videos lack in structure.
- Once seen a video, it’s not interesting anymore.
- It’s not easy to retrieve particular moments in an unedited home video.
- Other reasons (please specify): ...

9. With whom do you mostly watch your home videos?

- Alone.
- With friends.
- With the family.

10. On which device do you usually watch your home videos? (you can choose multiple answers)

- On a TV.
- On a PC.
- On my video capturing device (camcorder, digital still camera, mobile phone, etc.).
- Other (please specify) : ...

Sharing home videos

11. Nowadays many people share their videos with others by using Internet, DVDs, VideoCDs, etc. How often do you share your home videos with somebody else?

- Never.
- For less than half of the videos I shoot.
- For more than half of the videos I shoot.
- Always.

If you never share your home videos, please skip to question 14.

12. With whom do you share your home videos? (you can select multiple answers)

- With my family.
- With my friends.
- With everybody (for example by publishing them on the Internet).

13. Which media do you use to share your home videos? (you can select multiple answers)

- YouTube.
- A videoblog (vlog).
- I send video files via email.
- Other Internet-based means.
- I put videos on DVDs or Video CDs and I give them away.
- I share my videos using USB sticks or external hard disks.
- I share my videos with other means (please specify): ...

After question 13, you can go directly to question 15.

14. What are the most important reasons for not sharing your home videos? (you can select multiple answers)

- I never thought about sharing home videos.
- My home videos are too private to share.
- It is too difficult to send big video files.
- It is too difficult to make DVDs out of my videos.
- It is too difficult to manipulate videos.
- It takes too much time.
- Other reasons (please specify): . . .

Editing home videos

15. How often do you edit the home videos you shoot?

- Never.
- I edit less than 25% of my videos.
- I edit more than 25% and less than 50% of my videos.
- I edit more than 50% of my videos.

If you answered “never” or “I edit less than 25% of my videos”, please answer to the following question; otherwise skip to question 17.

16. Why do you never or seldom do home video editing? (you can select multiple answers)

- Home video editing requires too much time.
- Home video editing is too difficult.
- My videos don’t need to be edited.
- Other reasons (please specify): . . .

If you never edit your home videos, please go to question 21; otherwise, please continue.

17. What are the most important reasons for editing your home videos? (Please select multiple answers if they apply)

- I want to cut away some parts of a video and keep only the interesting content: the raw content is normally too long.
- I want to make my videos more pleasant to watch with music and special effects.
- I want to edit my videos to keep them for myself.

- I want to have different versions of my videos to give to different people.
 - I like to add my personal touch in my videos.
 - Other reasons (please specify): . . .
18. How long does it take for you to edit 1 minute your video?
- Less than 1 minute.
 - 1 - 10 minutes.
 - 10 minutes - 1 hour.
 - 1 - 10 hours.
 - More than 10 hours.
19. Which software package or tool do you use to edit your home videos?
(Please select multiple answers if they apply)
- Adobe Premiere, Pinnacle Studio or similar programs.
 - Muvee AutoProducer or similar programs.
 - Windows Movie Maker or similar programs.
 - Video editing web services like Jumpcut, Video Egg, Grouper, etc.
 - Tools embedded on hard disk or DVD recorders.
 - Other (please specify): . . .
20. Are there some aspects of the tools you use that leave you unsatisfied? (you can select multiple answers)
- I'm fully satisfied with my home video editing tool.
 - Video editing tools are too complex and technical.
 - Video editing tools are too time-consuming.
 - Video editing tools don't allow me to express fully my creativity.
 - Video editing tools are not powerful enough.
 - I cannot navigate and search well enough in my videos.
 - Other (Please specify): . . .
21. Below there is a list of actions that one could do with his home videos. Please indicate your level of interest in each one of these (1 = not interesting, 2 = slightly interesting, 3 = interesting, 4 = very interesting, 5 = extremely interesting).

B

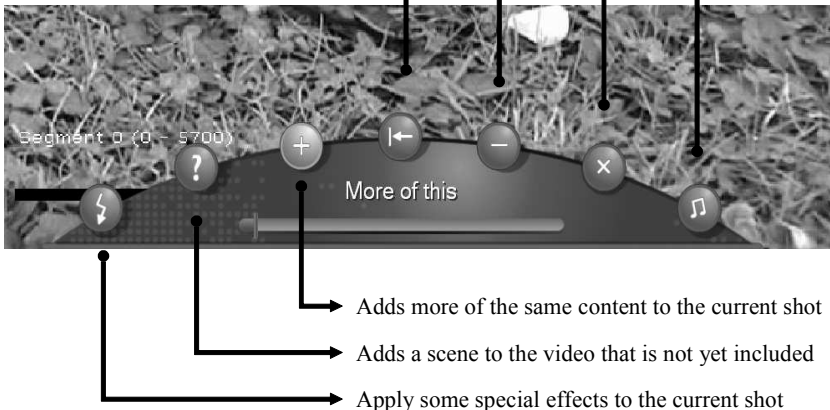
Material for the user study on Edit While Watching

This appendix contains the material used for the user study on the first version of Edit While Watching. Figure B.1 shows the A4 paper that was given to the user as short explanation of the system and of the tasks to perform. Figure B.2 shows the Likert-scale questionnaire that was used for the semi-structured interview at the end of the test.

User Instructions Edit While Watching

Edit While Watching
 The Edit While Watching system is a method to easily edit your home video using just a TV set and a remote control.
 The system automatically generates a summary of the home video. When the video is playing, you can pause the video using the remote control. A menu will be shown, from which you can select one of the editing effects:

- Adds music to the current shot
- Deletes the current shot
- Removes content from the current shot
- Undo (not yet implemented)



The diagram shows a video player interface with a menu overlay. The menu includes icons for adding music, deleting a shot, removing content, undo, and more of this. Arrows point from these icons to their respective descriptions in the text above and below the interface.

- Adds more of the same content to the current shot
- Adds a scene to the video that is not yet included
- Apply some special effects to the current shot

Scenario
 You have been to the Zoo with some of your relatives, and some home video has been captured during that event. Your task is to produce a video that is nice to watch for these relatives. You have uploaded the video into the system, and now you have to start editing. You have to fulfil:

- Your relatives found the “dancing” bear quite sad and shocking, you should shorten this scene.
- Your relatives like the geese a lot. Make sure to include a lot of this content.
- Your relatives found the panthers really boring, since they are only sleeping. They should be removed from the final version.
- Your relatives liked the feeding of the penguins a lot: make sure to include that part.
- Now you have to finish the summary according to your wishes to show it to your relatives and to keep it for yourself, you can use all the features of the editing system.

Please think aloud

Figure B.1. User instructions for the user study on EWW1

	False				True
I was able to reach a final result that satisfies me	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
EWV enables fast editing	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I was able to navigate easily in the material	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I was able to align nicely the music with the video, according to my wishes	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I felt the need for more editing functions	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I find it easy to use the editing functions	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
The editing functions need to be improved	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I managed to include into the final summary the scenes that I wanted	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I would like to use EWV as it is for editing my home videos	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I like using EWV to edit video, it is fun!	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I find the task scenario very realistic	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I would like to have this application	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figure B.2. Questionnaire used for the final semi-structured interview with the users.

C

Material for the user study on Edit While Watching 2

This appendix contains the material used for the use test on Edit While Watching 2. Figures C.1 and C.2 show the instructions that were read and shown to the user to get him or her acquainted with EWW2. Figure C.3 shows the questionnaire that was used for the final semi-structured interview with the user.

Edit While Watching 2 test instructions

Welcome to the HomeLab and to this use test about home video editing. During this test you will use the “Edit While Watching” application, which is a research prototype designed for editing personal videos using a TV set and a remote control.

The goal of this test is to evaluate to which extent this application meets your needs in home video editing. Your feedback will be used to improve the application and maybe, one day, make a product out of it.

Here you are given a quick introduction to the application, after that you will be invited to try it by yourself.

Shot editing mode

Our application has already generated a first edited version of your video material, which is now played on the TV. An overview of the video shots is shown on the bottom of the screen.

You can fast forward using the *fast forward* button and you can fast rewind by pressing the *fast rewind* button.

You can also jump forward to the *next shot* and jump backwards to the *previous shot*.

During the playback, you can pause the video by pressing the *pause* button on the remote control. You can play it again by pressing the *play* button on the remote control.

Please try to use these keys by yourself.

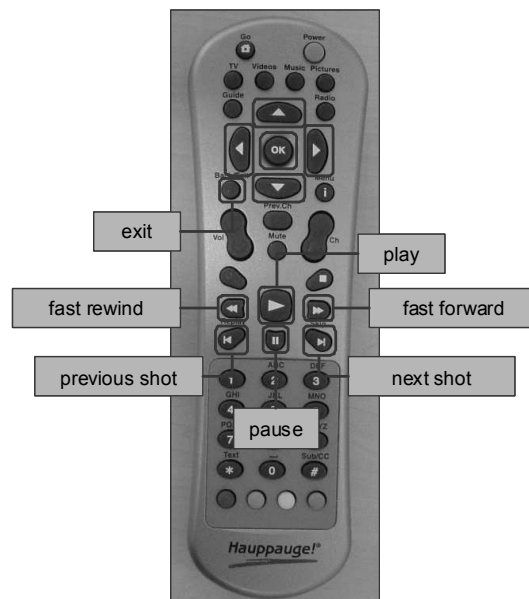
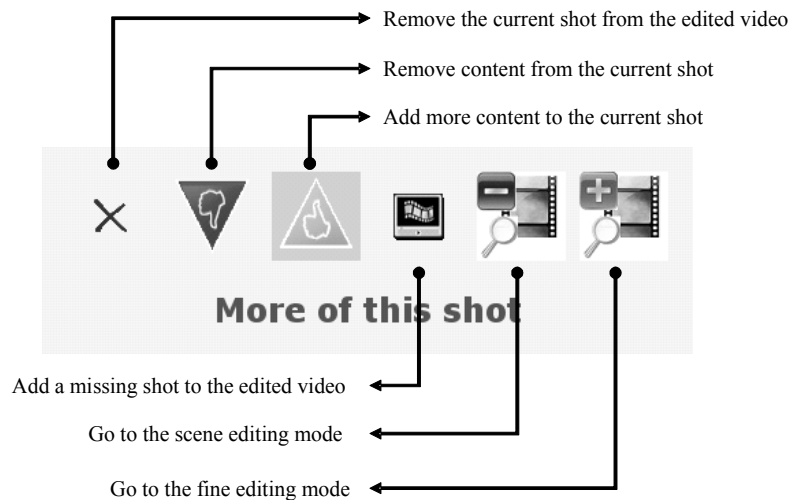


Figure C.1. Instructions read to the participant during the test on EWW2 (first page).

Whenever you press the *pause* button, a menu appears. You can select one of the options by pressing the *left* and *right* keys on the remote control and the button *OK*. If from this menu you press *pause* again, the menu disappears and no option is selected. The functions of the various options are explained in the figure here below:



Please try to pop-up the menu, use its functions and play the video again.

Scene editing mode

By selecting the “*scene editing mode*” option we zoom out, and an overview of the video scenes is displayed and you can browse through the scenes included in the edited video.

If you press the *pause* button now, a new menu is shown. By selecting “*shot editing mode*” we zoom in and the shots inside the current scene are displayed. Try to zoom out and zoom in by yourself, you can also use the arrows up/down.

Fine editing mode

By pressing again the *pause* button and selecting the “*fine editing mode*” option, a timeline is displayed representing all the video material close to the current shot.

The video segments included in the edited version are displayed in green; the segments not included in the edited version are displayed in red.

In this mode, the original unedited video is played and you can precisely select which segments of video to include or exclude in the edited version. This can be done by pressing the *pause* button and choosing the “*include this*” or “*exclude this*” options.

After pressing “*include this*” you can select the desired segment of video and include it in the edited version by pressing the “*ok*” button.

While playing the video in the fine editing mode, you have two shortcut keys to the functions for including and excluding content. You can start selecting content to include or exclude simply with the arrows *up* and *down*.

Figure C.2. Instructions read to the participant during the test on EWW2 (second page).

	False				True
I understood easily when I was playing the edited video and when the unedited	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
The three levels of overview were easy to understand	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
The effects of the editing options were clear	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
The application allows going to a wanted point in the video in a quick way	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
The application allows to move around easily in the video	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
The three levels of overview give a good impression of the structure of my video	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
The meaning of the “add missing shot” panel was clear	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
The final edited video satisfies me	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Using Edit While Watching would save me time when editing video	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
To use Edit While Watching, one needs to remember too many things	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I felt the need for more editing functions	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
The editing functions were easy to use	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
The editing functions need to be improved	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I needed more functions to move around in the video	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I would like to have Edit While Watching for editing my home videos	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I like using Edit While Watching to edit video	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figure C.3. Questionnaire used for the final semi-structured interview with the users.

D

Material used for the experiments described in Chapter 6

D.1 Material used for experiment about timestamp-based scene overview

This section contains the material used for the experiment on the effect of timestamp-based scene detection on browsing and searching tasks. The design of this experiment is described in Section 6.3. Section D.1.1 contains the list of browsing tasks prepared on the video about the trip to California, while Section D.1.2 reports the list of searching tasks related to the video about the visit to the zoo.

D.1.1 Tasks related to browsing the video of the holiday in California

1. Does the video contain images of the Golden Gate bridge?
 - (a) Yes (**true**)
 - (b) No
2. Does the video contain Asian-looking girls?
 - (a) Yes (**true**)
 - (b) No

3. Are there penguins in the video?
 - (a) Yes (**true**)
 - (b) No
4. Which color is the car that the people in the video are driving?
 - (a) Yellow
 - (b) Purple
 - (c) White (**true**)
 - (d) Red
5. Which color is the telephone box on the Golden Gate bridge?
 - (a) Yellow (**true**)
 - (b) Purple
 - (c) White
 - (d) Red
6. Is the visit to the aquarium before the trip with the Asian-looking girls?
 - (a) Yes
 - (b) No (**true**)
7. Does the visit to the red wood (sequoia forest) happen before the visit to the Golden Gate bridge?
 - (a) Yes
 - (b) No (**true**)
8. Is a hotel room shown in the video?
 - (a) Yes (**true**)
 - (b) No
9. In which lane are the protagonists of the video driving on the Golden Gate bridge?
 - (a) The leftmost lane (**true**)
 - (b) The central lane
 - (c) The rightmost lane
10. Which of the following order of events is correct?
 - (a) Visit to the Golden Gate, arrival at the hotel, visiting with Asian-looking girls, visit to the sequoia forest

- (b) Arrival at the hotel, visit to the sequoia forest, visiting with Asian-looking girls, visit to the Golden Gate
 - (c) Visiting with Asian-looking girls, arrival at the hotel, visit to the Golden Gate, visit to the sequoia forest
 - (d) Arrival at the hotel, visiting with Asian-looking girls, visit to the Golden Gate, visit to the sequoia forest (**true**)
11. How old was the fallen sequoia when it crashed down?
- (a) Two years
 - (b) 10 years
 - (c) 20 years
 - (d) 50 years
 - (e) 200 years
 - (f) 1000 years
 - (g) 2000 years (**true**)

D.1.2 Tasks related to searching the video of the visit to the zoo

1. In the zoo, some white polar bears are observed. Does the cage of the polar bears contain a pond with water?
 - (a) Yes (**true**)
 - (b) No
2. Are there some birds standing in the giraffes cage?
 - (a) Yes (**true**)
 - (b) No
3. In the zoo, some seals are swimming in a pond. In which direction are they mostly swimming?
 - (a) From left to right (**true**)
 - (b) From right to left
4. There is a brown bear climbing on some branches of a tree. The bear is giving:
 - (a) The back to the camera (**true**)
 - (b) The front to the camera
5. Some geese are shown staring at a man. The geese are:

- (a) On the left of the screen (**true**)
 - (b) On the right of the screen
6. A spider is shown. The spider is:
- (a) Black
 - (b) Light brown
 - (c) Black and yellow (**true**)
7. Some fishes are shown swimming in a pond. How many are they?
- (a) 3
 - (b) 5
 - (c) 6 (**true**)
 - (d) 7
8. A panther sleeping on a net is shown in a video shot. Which animals are shown in the shot immediately before?
- (a) Bears
 - (b) Giraffes
 - (c) Monkeys (**true**)
 - (d) Seals
9. Two toucans are shown together. Which animal is shown immediately before?
- (a) A spider
 - (b) A zebra
 - (c) A tiger (**true**)
 - (d) An orangutan
10. Which animals are shown after the penguins?
- (a) Zebras
 - (b) Monkeys (**true**)
 - (c) Geese
 - (d) Panthers

D.2 Material used for experiment about semi-automatic editing functions

This section contains the material used for the experiment on the balance between automation and user control in the semi-automatic editing functions. The design of this experiment is described in Section 6.7. Section D.2.1 reports the editing tasks on the zoo video, to be performed with pre-determined choice of functions. Section D.2.2 shows the editing tasks on the video with playing children, to be performed with free choice of functions. Figure D.1 shows the scale used to measure the mental effort taken by the users for doing the editing tasks with each one of the functions. Figure D.2 shows the scale used to measure the perceived ease of use, the perceived control and the perceived self-efficacy during the execution of the editing tasks with each one of the editing functions. Finally, Figure D.3 shows the questions used for the conclusive interview with the participants.

D.2.1 Editing tasks on zoo video with pre-determined choice of functions

1. In the beginning of the video, a panther sleeping on a net is found. The panther doesn't do anything and is not very interesting. You may want to shorten this shot a bit.
2. In the next shot, two toucans are found. Make sure that the shot includes also the lady with the white coat.
3. In the next shot, some geese quacking to a man are found. The geese are fun; you may want to include more content about them.
(change functions)
4. The next shot presents a brown bear climbing on the branch of a tree. Make sure that the shot ends before the bear is on the ground again.
5. The next shot presents a panel with written information on the bear. Since it is quite a boring shot, make sure that it is deleted or shortened to the minimum.
6. The next shot contains a close-up of a white polar bear sleeping. Make sure that the other polar bear walking by the pond is also included.
(change functions)
7. The next shot presents a girl taking out a surprise from a yellow and red machine. Make sure to include in the shot also the object she got as surprise.
8. In the next shot, two orangutans seem to be whispering something in the ears of each other. Include more material related to this "whispering" couple of monkeys.
9. The next shot contains some giraffes. They are quite interesting animals. Make sure to include the close-up of the face of the chewing giraffe.

D.2.2 Editing tasks on the video with playing children, with free choice of functions

1. In the beginning of the video, a black door is filmed. You may want to include the moment in which the door opens.
2. In the next shot, a boy and a girl are playing with some leaves. Include more content of them.
3. The next shot contains only grass and it is not very exciting. Shorten it as much as possible.
4. In the next shot, the boy in blue coat is walking with the mother. Make sure that the shot ends when the mother goes out of the scene.
5. In the next shot, the boy is holding a pole in his left hand. Include also the moment in which he loses the grip of the pole!
6. In the next shot, the mother collects a dry fruit from the ground. You want the shot to start when the mother has already the fruit in the hand.
7. In the next shot, the boy sits on the top of the yellow slider. Why can't he slide down?
8. In the next shot, at some point the boy yawns. Perhaps you want to exclude that part.

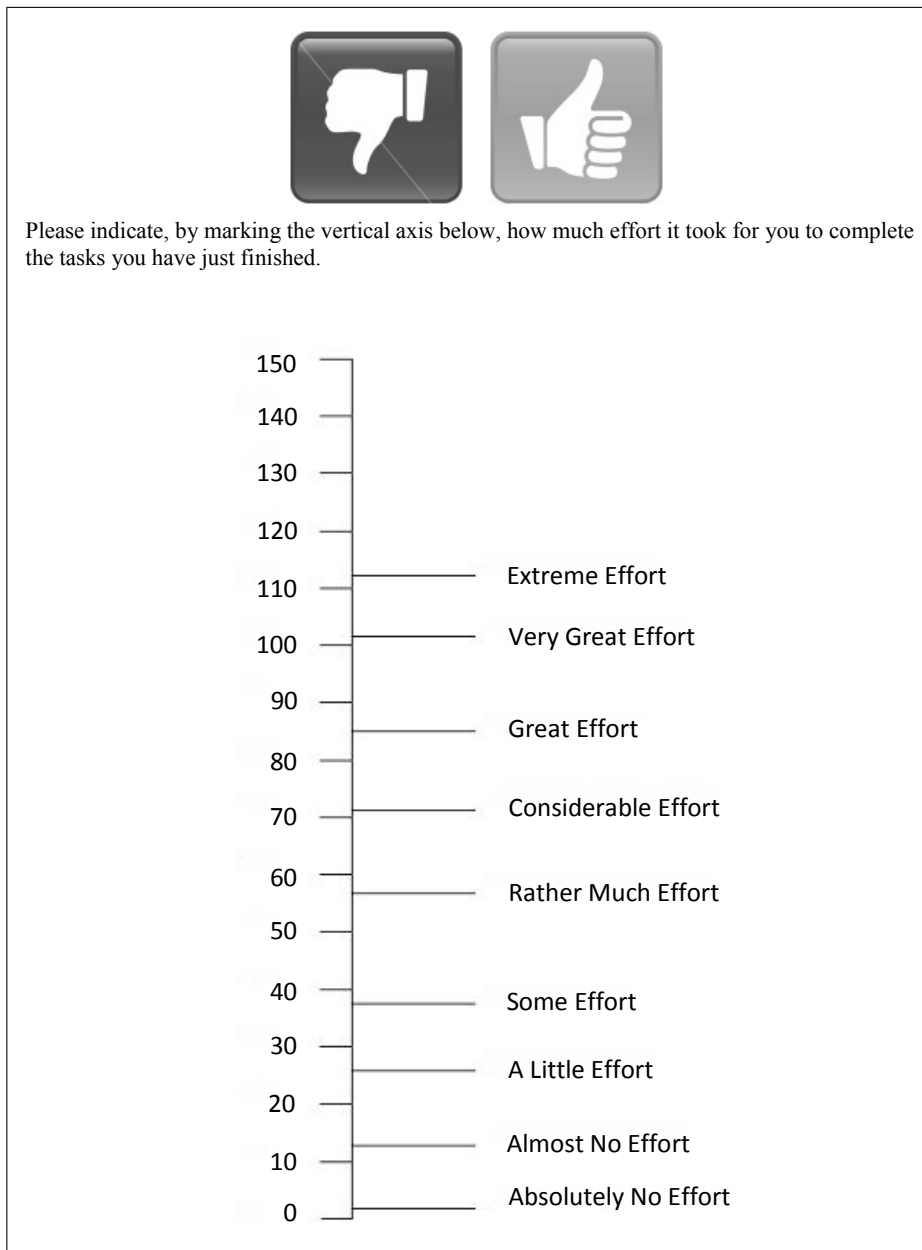




Figure D.1. Scale used to measure the mental effort for doing the editing tasks. The icons on the top of this scale refer to the functions “more of this” and “less of this”. For the other functions, other icons were printed on the scale.

Questionnaire
Please answer the following questions related to the functions “More of this shot” and “Less of this shot”

(1 = totally disagree, 7 = totally agree)

My interaction with these functions was clear and understandable	1	2	3	4	5	6	7
I found it was easy to do whatever I want	1	2	3	4	5	6	7
Learning to operate these functions is easy for me	1	2	3	4	5	6	7
It would be easy for me to become skillful at using these functions	1	2	3	4	5	6	7
I felt that I was in control	1	2	3	4	5	6	7
I was able to approach the problem in my own way	1	2	3	4	5	6	7
When I planned on doing something, I was able to make it work.	1	2	3	4	5	6	7
I have a pretty good idea of how the functions work.	1	2	3	4	5	6	7
I felt responsible for performing the task well	1	2	3	4	5	6	7
I felt discouraged while performing the task	1	2	3	4	5	6	7
I felt good about my performance on the task	1	2	3	4	5	6	7
At some points, I felt like giving up	1	2	3	4	5	6	7

Figure D.2. Scale used to measure the perceived ease of use and perceived control during the usage of the editing functions. The icons on the top of this scale refer to the functions “more of this” and “less of this”. For the other functions, other icons were printed on the scale.

Final interview

1. In this test, you have been using three different kinds of functions: “More” and “Less of this shot”, “Lengthen” and “Shorten this shot” and the cutting functions “insert this” and “remove this”. Do you have functions that you prefer over others? If yes, why?

2. Are there functions that appear to you less useful than others? If yes, why?

3. Did you have any particular problems with any of the functions?

4. Do you think that any of these functions would be particularly useful to you? Why?

5. Is there something that you would like to change in any functions?

6. Did you find the tasks realistic or artificial? Why?

7. Please add any further comments you may have.

Figure D.3. Questions of the semi-structured interview done with the participants at the end of the experiment.

Balancing Automation and User Control in a Home Video Editing System

Summary

The context of this PhD project is the area of multimedia content management, in particular interaction with home videos. Nowadays, more and more home videos are produced, shared and edited. Home videos are captured by amateur users, mainly to document their lives. People frequently edit home videos, to select and keep the best parts of their visual memories and to add to them a touch of personal creativity.

However, most users find the current products for video editing time-consuming and sometimes too technical and difficult. One reason of the large amount of time required for editing is the slow accessibility caused by the temporal dimension of videos: a video needs to be played back in order to be watched or edited. Another reason of the limitation of current video editing tools is that they are modelled too much on professional video editing systems, including technical details like frame-by-frame browsing. This thesis aims at making home video editing more efficient and easier for the non-technical, amateur user. To accomplish this goal, an approach was taken characterized by two main guidelines. We designed a *semi-automatic* tool, and we adopted a *user-centered* approach.

To gain insights on user behaviours and needs related to home video editing, we designed an Internet-based survey, which was answered by 180 home video users. The results of the survey revealed the facts that video editing is done frequently and is seen as a very time-consuming activity. We also found that users with low experience with PCs often consider video editing programs too complex. Although nearly all commercial editing tools are designed for a PC, many of our respondents said to be interested in doing video editing on a TV.

We created a novel concept, *Edit While Watching*, designed to be user-friendly. It requires only a TV set and a remote control, instead of a PC. The video that the user inputs to the system is automatically analyzed and structured in small video segments. The editing operations happen on the basis of these video segments:

the user is not aware anymore of the single video frames. After the input video has been analyzed and structured, a first edited version is automatically prepared. Successively, *Edit While Watching* allows the user to modify and enrich the automatically edited video while watching it. When the user is satisfied, the video can be saved to a DVD or to another storage medium.

We performed two iterations of system implementation and use testing to refine our concept. After the first iteration, we discovered that two requirements were insufficiently addressed: to have an overview of the video and to precisely control which video content to keep or to discard. The second version of *Edit While Watching* was designed to address these points. It allows the user to visualize the video at three levels of detail: the different chapters (or scenes) of the video, the shots inside one chapter, and the timeline representation of a single shot. Also, the second version allows the users to edit the video at different levels of automation. For example, the user can choose an event in the video (e.g. a child playing with a toy) and just ask the system to automatically include more content related to it. Alternatively, if the user wants more control, he or she can precisely select which content to add to the video.

We evaluated the second version of our tool by inviting nine users to edit their own home videos with it. The users judged *Edit While Watching* as an easy to use and fast application. However, some of them missed the possibility of enriching the video with transitions, music, text and pictures. Our test showed that the requirements of overview on the video and control in the selection of the edited material are better addressed than in the first version. Moreover, the participants were able to select which video portions to keep or to discard in a time close to the playback time of the video.

The second version of *Edit While Watching* exploits different levels of automation. In some editing functions the user only gives an indication about editing a clip, and the system automatically decides the start and end points of the part of the video to be cut. However, there are also editing functions in which the user has complete control on the start and end points of a cut. We wanted to investigate how to balance automation and user control to optimize the perceived ease of use, the perceived control, the objective editing efficiency and the mental effort. To this aim, we implemented three types of editing functions, each type representing a different balance between automation and user control. To compare these three levels, we invited 25 users to perform pre-defined tasks with the three function types. The results showed that the type of functions with the highest level of automation performed worse than the two other types, according to both subjective and objective measurements. The other two types of functions were equally liked. However, some users clearly preferred the functions that allowed faster editing while others preferred the functions that gave full control and a more complete overview.

In conclusion, on the basis of this research some design guidelines can be offered for building an easy and efficient video editing application. Such application should automatically structure the video, eliminate the detail about single frames, support a scalable video overview, implement a rich set of editing functionalities, and should be preferably TV-based.

Acknowledgments

During these four years, many people have helped me in reaching the level of Ph.D. Firstly, I want to thank my promotor Armin Kohlrausch, especially for the patience he had in teaching me how to have a scientific approach to my work, and in taking care that all the steps necessary for the Ph.D were executed according to the plans. Also my daily supervisors, Mauro Barbieri and Hans Weda, have significantly contributed to my maturation as a scientist. They have been trainers for me, pointing out defects in my work with great sharpness and constance and suggesting directions of improvements.

I want to thank also the other members of the reading committee: Don Bouwhuis, Anton Nijholt and Huib de Ridder, for reviewing my work. I thank in particular Don Bouwhuis for the very detailed and useful review of my thesis, which significantly improved its quality.

Many people helped me out with my research, in the past four years. I start thanking Ingmar van Dijk and the other software engineers of SES, who helped me to develop the software of *Edit While Watching*. Thanks to Henk van der Weij and the other people of his company, BigCamp, for the development of the GUI in *Edit While Watching* I. Bart Kroon should also be thanked for the face detector that he provided for *Edit While Watching*. Dragan Sekulovski was very helpful in teaching me the science about color temperature and for providing me with algorithms to implement the color temperature effect in my tool. Thanks to Francesco Bonarrigo for refining the zoom detection algorithm of my tool, and for having dared to be my student. I want to thank also Adolf Proidl for providing a home video of his family life for many of my experiments, and Sander Kouwenberg for the zoo video and the initial brainstorming on *Edit While Watching*.

When I took my master degree in engineering, I did not have competences in usability and user-system interaction. During these four years I had to build these competences, and I met many helpful people that significantly contributed to that. I want to especially thank Jettie Hoonhout, she really put an extraordinary effort in helping me understanding the science of usability and of creation of questionnaires. Thanks also to Wijnand IJsselsteijn for the useful feedback on the methodology of the user experiments and the general comments on the research. I also thank Nele van den Ende, (Shalu) Privender Saini, Joica Lacroix, Will Green and all the other

people that shared with me their competences in usability. Thanks also to Jan Engel for the help in the statistical analysis.

A special thanks to all the participants of my user experiments! They made it possible for me to come to the results of my research. They contributed generously, even if sometimes my lack of experience made the experiments a bit difficult. I want to thank all the colleagues that I met in these four years at Philips, in particular Mevrouw Shrestha, who despite spending four years in the same office with me is still a nice friend of mine, Mevrouw Tsoneva, Lu Wang and Gary Caballero Garcia. Thanks to the other three Marie Curie fellows: Greg, Marco Tiemann, and Alberto (beeedduuuuu!), for encouraging each other during the Ph.D. Thanks to Hans van Gageldonk, my group leader, to Philips Research and to all the colleagues that offered me the best tools and organization to perform my research .

It would be really not fair to not mention here the invaluable friends I have met during these years in Eindhoven. They have been really a family to me, providing me an irreplaceable support in the difficult moments. Thanks to Honest Enzo and Noëlie, Gianluca (Sambuca), Tomasso, María and Francis, Stefan and Isa, Met Scalori and all the friends of the football, Fernandinho, Ruben the paparazzo, and all the others. I am honored to give a special thanks to the friends of the Church community: Paula, Prakash and Sowmia, Stefan Schevers, Ania and Asia, Lucie, Sri, Irwan, my twin brother the M. H. Rodrigue, Hervé, Yannick, Discus and Lorraine, Sabrina, Chibuzo, Alina, Caecilia, Tanuja, Greg, Karlijn, Santa Agnese, the rev. Claudius Maria, the rev. Javier and Valeria, Mathieu, Anna and Puszek, Ola, Father Ad, Father Chima, Father Emma.

The last special thanks goes to my family: my dad Ignazio, my mum Nella and my brother Lorenzo.

Curriculum vitae

Marco Campanella was born in Brescia, Italy, on October 13th 1979. From 1993 to 1998, he studied scientific subjects and general culture in the high school “Liceo Scientifico A. Calini”, in Brescia. At the end of these studies he earned the High School Diploma of “Scientific Maturity” with the final mark of 60/60. After that, he enrolled in the faculty of electronic engineering in the University of Brescia. During the academic year 2002/03, he spent eleven months in Barcelona, Spain, supported by an Erasmus grant. In Barcelona, he successfully attended ten courses given by the Polytechnic University of Catalonia on subjects related to telecommunications and software engineering. These courses were given and evaluated in the Spanish and Catalan languages. In February 2005, he earned the Master Degree in Electronic Engineering, with specialization in telecommunication and computer science and with final mark 109/110. His master thesis concerned an experimental project about visualization of low-level features for browsing and annotation of audiovisual documents. From May 2005 to May 2009, he worked at the Philips Research Laboratories in Eindhoven, on the topic of home video editing. During this period, he gained experience in usability testing and in doing and communicating science. This PhD thesis is the result of these four years of work. Since August 10th 2009, he works with a spin-off company of the University of Brescia, Italy. His responsibility is to develop a hi-tech project concerning laser scanning and 3D reconstruction of large objects.