Design of an interactive authoring tool for creating branched video

Design av ett interaktivt författarverktyg för att skapa grenade videor

Simon Schmidt & Nils Tyrén

Supervisor : Niklas Carlsson
Examiner : Marcus Bendtsen
Upphovsrätt

Detta dokument hålls tillgängligt på Internet - eller dess framtida ersättare - under 25 år från publiceringsdatum under förutsättning att inga extraordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopian för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämnd som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/.

Copyright

The publishers will keep this document online on the Internet - or its possible replacement - for a period of 25 years starting from the date of publication barring exceptional circumstances.

The online availability of the document implies permanent permission for anyone to read, to download, or to print out single copies for his/her own use and to use it unchanged for non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission.

All other uses of the document are conditional upon the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its www home page: http://www.ep.liu.se/.

© Simon Schmidt & Nils Tyrén
Students in the 5 year Information Technology program complete a semester-long software development project during their sixth semester (third year). The project is completed in mid-sized groups, and the students implement a mobile application intended to be used in a multi-actor setting, currently a search and rescue scenario. In parallel they study several topics relevant to the technical and ethical considerations in the project. The project culminates by demonstrating a working product and a written report documenting the results of the practical development process including requirements elicitation. During the final stage of the semester, students create small groups and specialise in one topic, resulting in a bachelor thesis. The current report represents the results obtained during this specialisation work. Hence, the thesis should be viewed as part of a larger body of work required to pass the semester, including the conditions and requirements for a bachelor thesis.
Abstract

With the release of “Bandersnatch” in 2018, an interactive movie where the viewer makes choices that affects the outcome of the story, we know that successful interactive movies are possible and appreciated. Although this technology already exists the possibilities are seemingly limitless. Perhaps in the future, movies could take certain paths based on a predetermined profile of a viewer or by scanning facial expressions during the film to determine what path that best suits the viewer. Interactive films and videos allows the viewer to interact with the storyline of the video. This technique is interesting from both the user and developer perspective and introduces new challenges. To be able to have an overview of the different possible branches of the video is helpful and needed in development of the media player and the branched video. When different possible paths of the video emerges it can be difficult to keep track of all the different story lines. In this thesis, we make significant improvements to an existing authoring tool for a branched video player. The authoring tool is to be used along side with a media player in order to facilitate the development of a non-linear branched video. We will also be exploring what features of the authoring tool offer the most value to the user.
Acknowledgments

We would like to thank Niklas Carlsson, our supervisor for the support and help we have received during the process of this thesis. We would also like to give a big thanks to Jesper Wrang and Eric Lindskog for the support and always providing help when needed.
Contents

Abstract iv
Acknowledgments vi
Contents vi
List of Figures viii

1 Introduction 1
   1.1 Motivation 1
   1.2 Aim 1
   1.3 Research questions 2
   1.4 Delimitations 2

2 Background 3
   2.1 Interactive video 3
   2.2 Authoring tool 4
   2.3 Software tools 4
   2.4 Usability testing 5
   2.5 Related work 6

3 System design and implementation 7
   3.1 Baseline system overview 7
   3.2 Desired features for utility and usability 8
   3.3 Software structure 9
   3.4 Implementation of new functions 9
   3.5 Prototype used for user studies 13

4 User Study 14
   4.1 Methodology and setup 14
   4.2 Results 14

5 Discussion 17
   5.1 Analysis of user study results 17
   5.2 Necessary functionality 18
   5.3 Desired functionality 18
   5.4 Analysis of methodologies 19
   5.5 Improvements and ideas for future work 20
   5.6 The work in a wider context 21

6 Conclusion 22

7 Further improvements 23
List of Figures

2.1 Tree structure .................................................. 3
3.1 Initial view of authoring tool .................................. 7
3.2 Example of metafile from tree structure consisting of nine nodes ...................................... 8
3.3 Example illustrating video preview ................................ 11
3.4 Example with overlapping branches ............................... 11
3.5 Asymmetric tree structure ...................................... 12
3.6 Symmetric tree structure ...................................... 12
3.7 Current state .................................................. 13
4.1 Mean values of usability scores ................................. 15
4.2 Mean values of effort of use scores ............................. 15
5.1 Re-merging of branches ........................................ 21
1 Introduction

1.1 Motivation
Interactive video is an interesting concept with a lot of potential. The viewer can make decisions that ultimately alters the storyline of the film. This could have the potential of widening the audience of many movies since the viewer can make choices to better fit their personal preference. Although the future of interactive video looks promising, creating branched videos can be difficult for several reasons, one being that there is a lack of user friendly authoring tools specifically designed for the creation of interactive video. Since there are multiple choices resulting in different “branches” of the video and perhaps more choices for each branch, the complexity could increase exponentially when adding more choices to the video. This means that being able to see the structure of the film and the different possible paths is important when creating interactive videos. One of the most tedious tasks when creating branched videos is setting the correct time and length of different branches. In this thesis we present an authoring tool that is to be used interactively with a branched video media player to simplify the creation and understanding of the film and making it easier to adjust length and starting time of branches.

1.2 Aim
The purpose of this thesis is to make an interactive authoring tool to be used along side with a branched video media player to make the creation of interactive video easier. The following improvements to the current model are created and implemented:

- you should be able to move through the preview using the mouse,
- clicking on a node should give some visual indication of the current position in the film,
- zoom in and zoom out will be implemented,
- handling of branches that are re merged, and
- making larger tree structures possible by relocating nodes when the tree gets bigger.

By conducting a user study, the value of these features will also be assessed in terms of usability and utility.
1.3 Research questions

The two main research questions examined in this thesis are:

- Will the use of an authoring tool help to simplify the process of developing branched videos?
- What features of an authoring tool offers the most value to a user?

1.4 Delimitations

The user study conducted in this thesis had few participants, and all participants were similar in terms of technical background, age and experience working with branched videos. Because of this, results from the user study should be viewed as inspiration and not as factual evidence.
2 Background

2.1 Interactive video

Interactive video (or branched video when referring to videos with a branching storyline) is a video where the viewer can make choices that affect the storyline. As opposed to regular videos, the path is non-linear. This means that an array of new challenges arise. For example, different hardware on different devices of end users, lack of easy to use authoring tools and more complicated forms of navigation [14]. Another problem with non-linear video is being able to construct protocols for near on-demand delivery [3].

A regular progress bar to show how far into the movie the viewer is will not suffice. One of the most intuitive ways of showing the progress could be a tree structure to show all the different paths. This is what the authoring tool in this thesis will make use of.

![Tree structure](image)

**Figure 2.1: Tree structure**

There are several ways of creating branched video. For example, one way of creating interactive branched video would be to stitch together several non-continuous fragments of video to create the non-linear video segment [6]. However, for the authoring tool used in this thesis the branched video is created using one long video with all of the different scenes. Choosing a particular path would result in the player jumping to the time where the correct scene starts. This means that starting time and length of segments will be important to make the viewer experience complete. A branch in this film will simply be a segment of the video, which means it has to be timed correctly. Without a tool to help the creator of the branched video set the correct times for each branch, the development process will take much longer,
since the developer would have to run the program and test the different times in order to see how it would look.

2.2 Authoring tool

Authoring tools are software with pre-programmed elements to simplify the development of multimedia, usually implementing visual aids. There are many different authoring tools available to the public and in an array of different subjects. For example, it can be used in Virtual reality applications to reduce the amount of scripting necessary [15]. One of the most commonly used authoring tools is Power Point, which helps the user to create material for presentations. In this thesis the foundation of the authoring tool is the tree structure that shows the different paths of the branched video. This is meant for the user to keep track of all the different branches of the video by being able to easily add new branches and information connected to that branch. Clicking on a node in the tree gives you information about that particular path on a control panel on the right. For example, each node has a starting time and end time. The information about starting time made it possible to implement a video preview where you arrive at the starting time of that node in the video. This makes a tremendous difference in usability when creating branched video.

An area of focus in this thesis is to improve usability and overall utility of the authoring tool. Because of this, several user studies will be conducted to get more insight into the usefulness of the authoring tool and to receive suggestions for further improvements.

Usability can be defined with four key factors [1]:

- ease of learning - the pace at which a user can learn how the system works,
- ease of use - how the tool feels in terms of usability when using it to perform certain actions,
- intuitiveness - how well the tool explains itself upon usage, and
- enjoyableness - how much a user enjoys using the system.

Although these factors may seem self-evident they are not necessarily easy to improve upon. Knowing how usability is defined does not imply knowledge of what features will be perceived as easy to learn or intuitive. This is why usability testing is useful.

2.3 Software tools

With the use of pre-made software tools the development process was simplified and accelerated. What tools were used is presented in this section.

Programming languages

The authoring tool is a web based program that is built using HTML, CSS and Javascript. HTML is used for the basic structure of program. It determines the positioning of elements and divides elements into different subgroups. The HTML code calls JavaScript functions when certain events occur such as a user clicking on an element. JavaScript is used to make the authoring tool dynamic and connects functionality to the elements. CSS is used for details and appearance such as size or colour of the elements.
2.4 Usability testing

Introducing new features is not enough to improve usability. In some cases the introduction of new features can lead to a decrease in usability [1]. To test what kind of features that should be implemented a number user studies are conducted. This gives a sense of what features that are most appreciated and also what features that could be missing.

During the user studies a number of factors are examined. These are some of the critical measures of usability [1]:

- Efficiency - Can the user find the requested information without assistance? Is the authoring tool self-explanatory?
- Effective - Can the user actually create a design for a branched video?
- Engaging - Will the user describe their experience as enjoyable or satisfying?
- Easy to learn - How long does it take for the user to get started?

The user studies are conducted by letting a user try the authoring tool while answering questions about usability. For these studies questionnaires are prepared. There are several types of standard questionnaires such as System Usability Scale (SUS), Single Ease Question (SEQ) and NASA-TLX(Task Load Index) [9]. SUS is one of the most well-known questionnaires for analyzing user experience. This questionnaire is given to the user after the entire testing session is over. It works by letting the user rank how much they agree with different statements about the product being tested. An example of a statement could be “I consider the system to be easy to use” [9].

The SEQ is used after a specific task is completed by the user. The main advantages of using SEQ is the ability to gain an understanding of what particular parts of the system the user finds most problematic or most useful and since the task was completed very recently the user is more likely to remember initial thoughts about the task. The SEQ asks the user to rate the difficulty of a particular task directly after the task was performed [9].

The NASA-TLX is another post-task questionnaire meaning it is answered directly after a task is performed. It tends to be less used in UX - testing but can still be a very useful questionnaire [9]. It works with the user rating a task by several factors such as mental demand or effort. It is considered to be a more complex questionnaire and requires more effort from the user when completing.

For the authoring tool in this thesis the user tests are using questionnaires based on elements from all of these standard models. Since every system is different and has its own features it is important to customize the tests to fit the system best as possible.

Scalable Vector Graphics

The authoring tool uses Scalable Vector Graphics (SVG) for displaying the tree structure. SVG is a language for describing two-dimensional graphics and allows for three different types of graphic objects. These are text, images and vector graphics shapes such as straight lines and curves [4]. Objects in SVG can be interactive and dynamic by connecting “event handlers” to perform a certain tasks when the user clicks on an object or moves the mouse cursor for instance.
2.5 Related work

In previous work by Meixner et al. [14], they implemented an authoring tool called the SIVA Producer and presented an iterative process used to improve the usability of the tool. This tool has been used both in conjunction with a player, and alone, as an authoring tool in order to create interactive non-linear videos [12]. Separately, they have also presented a player to display interactive video on mobile Android devices [13].

Others have focused on the more technical side of improving the playback of interactive videos. These videos require new adaptations of techniques such as prefetching and buffer management to acquire smooth playback. Krishnamoorthi et al. present in their paper a design for HTTP-based adaptive streaming (HAS) in interactive branched video, including prefetching policies and buffer management [6] [8]. In another paper by Krishnamoorthi et al. [7] they consider the problem of providing users with instantaneous playback of alternative videos. This could perhaps be used in branched video by prefetching segments from alternative branches.

A natural consequence of the development of interactive video is the adaptation of these techniques in streaming services. Xiu et al. [18] have explored improvements in interactive multiview video streaming (IMVS) systems as well as optimization for their overall frame structure. Such systems aim to provide clients with multiple views of the same scene, ultimately providing a free viewpoint. These systems also introduce challenges in terms of networking. Toni and Frossard [17] propose in their paper an optimization of multiview data on the server in order to minimize resource requirements. Ren et al. [16] have looked at which coding structure supports view switching as well as what data to replicate on servers.

In another paper by Carlsson et al. [2] they introduce “multi-video stream bundles” that allows users to switch between different camera angles instantly during playback.

Lindskog et al. [11] presents and evaluates a branched video player implemented in dash.js. The player includes a generalized playback bar that visualizes the tree-like video structure and the buffer levels of the different branches.
3 System design and implementation

3.1 Baseline system overview

A view of how the authoring tool looks is shown in Figure 3.1

Figure 3.1: Initial view of authoring tool

The main part of the page is a box where the tree structure is created. When creating a new branch, a node will appear and a line will connect that node to the point where the node was created. The control panel in the right side of the page allows the user to type in the name of the branch, starting time and length, deleting or creating branches and saving and loading the tree structure from files.

Metafile

The authoring tool has the ability to both upload and download tree structures as metafiles. This makes it possible to upload pre-made tree structures and make the current structure more accessible from other programs. Metafiles are file formats that can store multiple types of data. The data is stored using JavaScript Object Notation (JSON) where the structure of the file is a list of all the current branches. The list contains the id, start time, length, display
name and the “children” of that branch. Figure 3.2 is an image of the structure of the metafile that is downloaded from the authoring tool with a tree structure that contains nine nodes.

```
3.2. Desired features for utility and usability

With our own ideas and the help of our supervisor and the project members responsible for the system baseline, a number of features were identified and prioritized. Analysis of the usability of these features are conducted with user studies.

Moving around the preview

To improve ease of use, programming blocks or components in software should be able to explain themselves to user best as possible. Drag and drop is a technique that is widely used and often perceived as easy to learn and use. Therefore a feature that was introduced to the authoring tool is being able to drag the tree around the preview using drag and drop. This is especially important when the tree structure is bigger. The user may want to focus on a particular path by zooming and then moving the tree to the coveted position.

Zoom buttons

Being able to zoom in and out is important when the tree grows in size, otherwise parts of the tree structure would fall outside of the screen making it impossible to see the entire structure in at once. It is also important to be able to zoom in to be able to focus on specific parts of the tree in the development process.

Video preview

In order to get a deeper understanding of what the branched video will look like, a video preview was introduced. This means that clicking on a node in the tree will result in a version of the video showing up at the bottom of the control panel on the right side. Once a user clicks on a node, the start time is used to connect the user to the correct location in the video. This

Figure 3.2: Example of metafile from tree structure consisting of nine nodes

```
means that the user can create branches and decide start time based on the video preview. This simplifies the development process by a large margin since the user does not need to open and run the video player in order to test different starting times, it can be done directly from the authoring tool.

Handling of large trees
In earlier versions of the authoring tool, the branches would overlap when the tree gets too big or when too many children are added to one node. This is because the rendering works by drawing new branches on a pre-determined offset from the parent branch. A solution to this problem would be a dynamic offset that takes into account how many nodes the tree consists of or how many children a branch currently has.

Merging paths
The current authoring tool does not support merging of branches. This means that the user can not make separate paths that ultimately re-aligns, this is at least not visually possible. However an option for the user to achieve this would be to make multiple nodes with the same name and starting time. To make this look visual appealing, a visual effect of re-merged branches would have to be introduced.

Thumbnails
Instead of having names on each node, the user may prefer having pictures i.e. thumbnails of the branches on the nodes. This could include snapshots from a specific part of the video. Whether this increases usability or not is examined in the thesis.

3.3 Software structure
The structure of the authoring tools is built on one HTML file, three CSS files and three JavaScript files that each handle different functionality. The HTML file is the main structure of the page that contains the authoring tool and all of the different elements that it consists of. It also contains events such as “onClick” or “onMove” that calls functions of the JavaScript files when the user performs a certain action. The CSS files are responsible for styling and smaller details in appearance such as colour or size of elements.

The JavaScript files handle all of the functionality, one of the files handles functionality connected to the branches and includes functions such as getting a branch from an id and calculating the maximum number of nodes on each level that is used when determining the distance between nodes. Another file handles functionality related to the preview such as drawing the tree structure and receiving feedback from the user in terms of mouse cursor position and movement. Another file handles all of the functionality for the control panel at the right side of the authoring tool.

3.4 Implementation of new functions
The process of implementing new functions to an existing program started with acquiring a familiarity with the authoring tool and the code written in HTML and JavaScript. By making smaller changes to the code and simultaneously testing the authoring tool, a basic understanding of the program was obtained fairly quickly. Knowing approximately what functions to implement made it easier to pinpoint more essential parts of the code. After acquiring a basic understanding of the program it was time to make a preliminary schedule for making the improvements. This is something that can be particularly challenging since there needs to be
3.4. Implementation of new functions

a rough estimation of the difficulty and workload of the improvements. Some improvements may be more difficult than initially anticipated while other may be more simple. There has to be a balance between the importance of the improvement and how much work it requires to implement. It is favorable to start with easier improvements since this builds up a better understanding of the code which will make it easier to implement the future improvements. Although if some of the more difficult improvements are more important there needs to be time to make sure that they will be finished. In this chapter the method of implementation of the different functions will be explained.

Moving the tree with the mouse

This was the first function to be implemented. It works by using three predefined functions in JavaScript - OnMouseMove, OnMouseDown and OnMouseUp. OnMouseMove is a function called when the mouse cursor is moved by the user. OnMouseDown is called when the user clicks or holds the mouse. OnMouseUp is called when the user releases the mouse. By using OnMouseMove together with OnMouseDown we can determine how far the user has moved the mouse while holding it. The tree is moved the same distance as the mouse cursor when the mouse is held down. Then OnMouseUp is used to save the new position when the mouse is released. This position is the new starting position for the tree.

Zoom function

The second function that was added was the zoom function. This is an essential function especially when dealing with larger trees if the user wants the ability to see the entire tree at once. This was implemented using the scale parameter in HTML. As shown in Listing 3.1 this is done by simply adding a constant to the scale ("zoomFactor" in this case). These lines of code will stay the same but the constant "zoomFactor" will be increased or decreased depending on the input from the user. This means that the user will not really zoom in or out but rather decrease or increase the size of the tree structure.

```html
boxesString += '  
    <g transform = "scale(${'zoomFactor'})"
        onclick="selectBranch('${branch.id}')">
    <rect x="{$x}" y="{$y}"
        class="${branch.id == selectedBranch ? "selected" : ""}">
    </rect>
    <text x="{$x + 120 / 2}" y="{$y + 5 + 60 / 2}">
        ${branch.displayName}
    </text>
    </g>

Listing 3.1: How scale is used for zoom
```

Video preview

Implementing the video preview was actually simpler than initially anticipated. The video element was added to the HTML code and the starting time of the video was set from the code in JavaScript. A function for selecting a branch already existed and the starting time of each branch is used to set the current time of the video preview. When clicking on a branch the current time of the video will be set to the starting time of the selected branch. As shown in Figure 3.3 the video preview will appear at the bottom right corner. In this case the node "rabbit hole" was just clicked on and the video preview shows that part part of the video.
3.4. Implementation of new functions

Handling of large trees

The problem in the previous version of the authoring tool was that when the tree got bigger, the nodes started overlapping with each other. When rendering the tree structure, the root node has a set starting position. Every time a node is added it would be drawn at a constant offset in the x-direction and a calculated offset in the y-direction. The bigger the tree or the more children the user adds to a node, the offset in the y-direction has to be changed to avoid overlap. The initial solution to solve the problem with overlapping nodes was to calculate the maximum number of nodes in one level and use that to determine the correct offset. An example of how the overlap can appear is shown in Figure 3.4.

The maximum number of nodes in a particular level would in this case be nine nodes, at the bottom level. If the offset was adjusted to work for this level it would work for all levels since no other level has more nodes. However this did not work in practice. The nodes would start overlapping as soon as a node has more than one child at the bottom level. This is because
the offset would not increase until the bottom level has the most amount of nodes. The solution that would ultimately solve the problem was to think of the tree as symmetric.

A tree structure could initially look like in Figure 3.5.

![Asymmetric tree structure](image)

Figure 3.5: Asymmetric tree structure

The tree in Figure 3.5 is not symmetric which means that the maximum number of nodes is in third level and not the bottom level. If we would change the tree to be symmetric it would look like Figure 3.7.

![Symmetric tree structure](image)

Figure 3.6: Symmetric tree structure

Now, the maximum number of nodes will always be at the bottom and even by adding just two child nodes to one branch would result in an increase in the maximum number of nodes.
3.5 Prototype used for user studies

This is the version of the authoring tool that was used in the user studies. Figure 3.7 shows the default mode of that version. In this mode the user can interact with the tree structure on the left hand side and edit the data that makes up the tree on the right hand side.

![Diagram of tree structure]

Figure 3.7: Current state

The implemented functions, relative to the baseline product, are zooming, drag & drop handling of large trees, video preview and immediately responding to keyboard input when a user changes starting time. In this state the authoring tool is mainly to be used to create the tree structure intended for the branched video and finding the correct length and starting time of branches. The implemented functions are intended to provide a notably user friendly environment and does not have any functions considered to be redundant.
4 User Study

4.1 Methodology and setup

The user studies were performed with the intent of being interactive where the user used the tool and answered post task questions. In the first user study, the user was given a questionnaire and instructions. The user wrote the answers to the questions on the questionnaire. To make the studies more fluent and to get more information from the users the studies where changed so that the user would only answer the questions verbally while one of the instructors wrote down the answers. This approach seemed to have the desired results which led to the rest of the user studies being performed in this manner.

A total of eight user studies, with one test subject at a time, were conducted at Linköping University. The reason for this low number of studies is a combination of time constraints and that this project to a large degree consists of implementation. The user studies started of with a short explanation of what the tool was meant for. This explanation was customized depending on the users previous knowledge of branched videos. After this, the user was instructed to try out the tool and use the different functions. The user was then asked to answer a couple of questions regarding the overall ease of use of the tool as well as initial thoughts about basic functionality. Following this, the user was asked to rate specific functions one at a time related to usability and how much effort the function required to use. The rating was done on a scale from one to five. The user was also asked to suggest improvements to the function if possible. The third and last part of the user study consisted of answering questions about our own suggested improvements. For each suggested improvement the user was asked to answer questions regarding necessity, usability and utility.

4.2 Results

The mean value for each of the questions where the user would answer with a score from one to five are presented in this section, as well as new ideas for improvements that were extracted from the user studies and input from the subjects when it comes to suggested improvements.

For simplicity, questions regarding usability and effort are presented separately here. Figure
4.2. Results

4.1 summarizes the average score regarding usability questions and Figure 4.2 summarizes the average score regarding user efforts. Both Figures also show a calculated confidence interval. We can with 95% confidence say that the "true average" lies within this interval.

Usability

1. How would you rate the overall ease of use on a scale of one to five? Mean value: 4
2. How would you rate the usability of the zoom function? Mean value: 4.75
3. How would you rate the usability of the click & drag function? Mean value: 4.875
4. How would you rate the usability of the video preview? Mean value: 4.875

Usage effort

1. How much effort did the zoom function require to use? Mean value: 1.375
2. How much effort did the click & drag function require to use? Mean value: 1.125
3. How much effort did the video preview require to use? Mean value: 1.5
4.2. Results

Several new functions to improve usability and utility was presented during the user studies. Below is a list of improvements suggested by user study subjects in an order from most requested to least.

- Zooming with the touch pad - several users thought being able to zoom with the touch pad rather than using the zoom buttons would be a valuable addition when it comes to usability
- Editor in the video preview - users would appreciate being able to set start time and length directly in the video preview instead of writing in the provided text fields
- Video jumping to correct location directly after input - Instead of having to enter a start time and then clicking on the node to see that position in the video, users would prefer being able to see that position in the video directly after writing start time and then clicking on the enter key
- Enlarging the video preview - several users would appreciate being able to enlarge the video preview
- Info pop-up - to give a sense of what functions exist and how to use them, users want a toast or a pop-up window with information when first opening the authoring tool
- Video stopping when branch is over - users want the video preview to stop playing when a branch reaches its end time
- Colour coding - some users would appreciate a more clear way of connecting nodes to a specific branch or storyline perhaps by being able to change the colour of nodes
- Moving nodes with the mouse cursor - at the moment, the position of new nodes are calculated and put in a fixed position, some users want the ability to change the position of nodes using drag & drop
- Undo and redo - some users suggested implementing and option for redo and undo preferably by using standardized key board commands.

As for the suggested improvements the one that the users agreed with the most was re-merging of branches. 88% of users from the user studies think that when the tree structure is bigger and when the storyline is more complex, being able to make branches re-align is an essential function. Secondly 63% of users would appreciate snapshots from the video sequence when hovering over a branch with the mouse cursor, however the rest of the users do not agree and think that this addition would negatively affect the the cleanliness of the authoring tool and make it more cluttered. Lastly, 50% of users do not think that implementing thumbnails instead of text would positively affect the usability or utility of the authoring tool. However, 40% think it would be be a good addition to the program if the user was able switch between text and thumbnails and 10% think it would be a good addition regardless.
5 Discussion

5.1 Analysis of user study results

Even though the subjects of the user studies were not particularly familiar with branched videos interesting and useful suggestions for improvement were extracted from the studies. We had in total eight user studies where two of the test subjects had done previous work with branched videos. To include users without much knowledge of branched video helped when it came to receiving suggestions previously unthought of. Usually these kind of suggestions related more to overall usability rather than usability related to creating branched videos. A function that five test subjects mentioned would be to able to zoom using the touch pad. This is something that none of the project members had even considered which indicates the usefulness of the user studies. However, having users with no background in the subject has disadvantages as well, for example in the beginning of the user studies, several users did not understand the concept of the tool and what it was design for, despite an introduction in the beginning. The most difficult concept to understand was the structure of the video file. As previously mentioned in Chapter 2 this authoring tool is designed to work with video files where the video is one long sequence where different paths or branches exist in different times in the video. Although, after the subject understood this concept, the quality of the study increased significantly.

Overall subjects of the user studies considered the authoring tool to be fairly user friendly and that this was one of the biggest strengths of the tool. Even though it was made for an arguably complex task, having a tool that is user friendly can demonstrably be of great advantage. Users seemed to find all of the existing functions to be useful and required a fairly small amount of effort to use.

Out of the suggested functions “re-merging” of branches was the one that was considered to be the most important improvement. This was quite expected since in order for the tool to be complete the user has to be able to make storylines that meet. As previously mentioned in chapter 2 it is possible to make storylines that meet by creating two nodes and setting them to the same starting time although as the user studies suggest this would not be an intuitive way of solving the problem. When it comes to being able to see snapshots from the video by hovering over a branch with the mouse cursor the users were conflicted.
Some users thought that it would be a suitable addition to the program while others thought it would clutter the design. Several users appreciated the simplistic design and think that this addition could be unnecessary functionality that would decrease the simplicity. When it comes to implementing thumbnails users seemed to prefer text over images to represent the nodes. The explanation would often be that an image is not necessarily a good representation of a scene in a movie, at least when it is not chosen specifically. If the thumbnails were to be the first frame of the branch there is a possibility that the resulting image would not represent that video sequence well. Perhaps if the image was chosen by the developer, it could be a better indicator. However, despite users preferring text over images, several users thought that having the choice between images and text could be helpful. Perhaps a switch in the tool that allows users to choose between text and images is an appropriate alternative.

In summary, the user studies produced interesting and valuable results, possibly more so than initially expected.

5.2 Necessary functionality

As previously mentioned in chapter 4, the suggestions for improvements were a part of the user studies. The users were asked if they deemed these improvements to be necessary or not. The one suggestion that the users thought to be the most necessary was "re-merging" of branches. The user studies also gave rise to new ideas for functions that was previously unthought of. Some of these function are also necessary in order for the authoring tool to be complete. One of those functions are being able to type in starting time in the text field in the control-panel and directly after this change reaching that point in the video preview. This is necessary because one of the main purposes of this tool is being able to find the correct starting times of different branches. Without this function, the user would have to set the starting time, then click on another node and then click on the node again to reach that time in the video.

5.3 Desired functionality

Many of the suggested improvements from the user studies are relevant but perhaps not necessary in order for the authoring tool to function as intended. These improvements would rather improve the overall experience of working with the tool but may not be explicitly needed. One of the functions that both the project group and the users consider to be desirable is being able to use the touch pad in order to zoom in and out. Zooming with this technique has become such a common function that many users expect it to be implemented in most web applications. This function would make the authoring tool more user friendly and would probably not take long to implement. There are already handlers to detect this kind of input from the user.

Another function that is widely used and often expected to be implemented is undo and redo, most commonly performed using “Control + Z” and “Control + Shift + Z”. This is especially useful in this tool since it is easy to make mistakes when creating the tree structures.

Being able to enlarge the video preview is also a function that several users suggested and the project group considers to be desirable. At the moment, the preview is of fixed size and covers about a twentieth of the screen. When the user wants to see the progress of the video on the preview, it would be reasonable to able to enlarge the video so it could be watched similar to other regular videos.

Several users suggested that there should be some kind of information about how the
tool works and what different functions are available. One suggestion was to implement a toast or a pop-up window that would open the first time a user starts the program. This information would probably include a list of what functions that are available, simple instructions on how to build the tree structure and how to set starting time and length on different nodes.

A function that many users suggested but one that the project group does not particularly agree with, is being able to set the starting time and length of segments directly in the video preview. The reason that the project group does not agree with this change is because there are already text fields specified for this purpose and using the small preview to set starting time and end time would in our opinion be less intuitive and more cluttered. Perhaps an editing mode could be entered when enlarging the video but it would also be something that the user would have to learn and get used to. Also a function like this would require additional research and could take long to implement.

5.4 Analysis of methodologies

The two methods used in this paper to achieve the goal of a more interactive authoring tool was system design and implementation and a user study. To begin with, we designed the improvements to the system by our own imagination together with the help of previous work done by Lindskog and Wrang (the developers of the baseline system). With these implementations came new ideas for improvements. Both the implemented improvements and the suggested improvements were then tested in a user study. The overall assessment of the methodology is positive. The work flow has been seemingly smooth and has yielded pleasant results.

System design and implementation

In terms of system design the focus was to improve upon functionality rather than aesthetics. As the product, even today, is not a finished product we felt like working on how the tool looks is something that could be handled in later stages of development. Instead we devoted our time into both designing and implementing new functionality as well as improving on existing functionality. Looking back on the process of development this is probably something that we would not want to change. The design and implementation of functionality is something that we feel has improved the tool’s usability and utility.

One thing that is worth discussing is the fact that we did most of our design and implementation previous to the user study. Although there is benefit to this too, since we then get to receive feedback on our work, there is an argument to be made for doing it the other way around. If the user study would have been made in earlier stages of our work we could have strengthened the argument for implementing each functionality with the result of our user study.

User Study

The user study was generally well received and produced a result which was both interesting and desired. As previously mentioned in Chapter we had first planned to have the subjects of the user study fill out the questionnaire themselves. This turned out to be a not so effective way of conducting the study because the form took a lot of attention away from the tool. Instead, we decided to let the subjects speak freely while orienting the tool. While doing so, one of us asked the questions from the questionnaire and one of us took note of their opinions. This way, we managed to get more information from the subjects as well as removing some workload off their shoulders while taking part of the study.
In some of the questions in the questionnaire the users were asked to rate something on a scale from one to five. In other questions they were asked to give their overall feedback or thoughts on the matter at hand. The way the questions were phrased yielded very different results. The questions that allow participants to speak freely tend to give a lot of information and the questions that ask participants to rate something yield less information but is more specific. For example, we received a lot of suggested improvements from the study but we have no way of weighing them against each other. Perhaps these improvements should be put to the test in another user study and get rated as the functions we had already implemented was. For now though, we are happy with the result. Even though the suggested functions are not rated and weighed against each other, some of them have obvious perks and should be considered in further development.

In terms of reliability, the conducted user study should be taken lightly. Because of the small sample size of eight participants, a similar study in a different setting might have yielded a different result. Furthermore, the participants of the study was somewhat of a homogeneous group as most of them were IT-students of similar age. For the purpose of this paper this we consider this fine because the main goal was to provide perspective and nuance.

5.5 Improvements and ideas for future work

Both the results from the user study and working on improving the tool has provided us with ideas for future work. The ideas presented in this section is intended to be viewed as suggestions for future work on either this tool or authoring tools for branched video in general.

Video jumping to correct location directly after input

This is something that was frequently suggested in the user study. It also very distinctly increases the utility of the tool since with the introduction of this function the user has the option to make small changes in start time and directly see the outcome. When editing interactive branched videos it seems reasonable that users would want to make small changes often to make smooth transitions in the story, which is also something that is supported by our user study.

Enlarging the video preview

Another feature that was frequently suggested by subjects of the user studies was being able to enlarge the video preview. In order to keep focus on the editing part of the tool, the tree structure, the video preview is fairly small. Because of this, it might be favourable to include the option of increasing the size of the video preview. Alternatively, the video preview could have another mode in which it is centered and larger. Either way, it seems advantageous to implement an option of enlarging the video preview.

Command functions

Some basic command functions would increase the usability of the tool, such as "Control + Z" to undo change and "Control + Y" to redo change. These functions would improve the effectiveness of the tool substantially.
5.6. The work in a wider context

(a) Current state of merged endings

(b) Desired visual effect of re-merged branches

Figure 5.1: Re-merging of branches

Re-merging of branches

Although the functionality of this is already possible as the tool stands today, the implementation should be improved. If you want two branches to have the same ending you would, as it stands, have to have two separate branches with the same start time. However, the desired visual effect would be that the branches re-merge which would both decrease the amount of branches in total and make more intuitive sense.

5.6 The work in a wider context

Authoring tools are used regularly in many different services and have an abundance of different applications. They allow users that are less technologically knowledgeable to create content that is more complex and impressive [5]. The emergence of authoring tools in branched video will help more people create interactive films with interesting structures and storylines. This increase in user friendly authoring tools could perhaps help push interactive branched video into the mainstream. Although this may seem exclusively positive there could be negative consequences.

Allowing people that are less experienced in the field of streaming may not have the knowledge to implement functionality that is necessary to increase viewer experience. In this day and age people expect streaming to be of good quality and with virtually no lag. This means that techniques such as prefetching and buffer management is essential in order for people to be satisfied with the experience. There may be more interactive videos available but the quality of these videos may be decreased. This could lead to frustration from the viewers potentially leaving branched video with a bad mark. Although the necessary functionality could possibly be added directly from the authoring tool however, this would mean that the users would have to entirely trust the providers of the authoring tools to handle back-end functionality correctly.

Another aspect is the amount of storage and processing power needed to enable interactive video. Letting more people create interactive content would most likely lead to more of this content being available to the public. With branched video being more hardware demanding than regular linear videos, this could require an increase in storage and processing power in different applications.
The aim of this paper was to make an authoring tool made to design interactive branched videos. Initially, we set out to implement a set of functions and designs that aimed to make the tool easy to use and effective. In addition, we asked two research questions:

- Will the use of an authoring tool help to simplify the process of developing branched videos?
- What features of the authoring tool offers value to the user?

Finally, a user study was conducted to help answer these questions.

At this point in time, the functions and designs that we aimed to implement have all been implemented but one (re-merging of branches). From the user study we can conclude that users are overall positively tuned to the tool. It seems the purpose of the tool is clear and users generally find it useful in the context of designing interactive branched videos. The features that were most positively received were generally the features that provided some sort of utility to the tool. For example, users seem divided when it comes to implementation of visual effects such as thumbnails or snapshots. However, the features that offer some sort of functionality such as zoom or click and drag are generally appreciated. This suggests that developers of authoring tools should wary not to clutter their product with too many features unless the purpose is well defined and acknowledged.

For suggestions regarding future work we refer the interested reader to section 5.5.
Further improvements

After taking note of the results from the user study we decided to implement two of the functions that were frequently suggested by the participants. Those implementations are presented here.

- Zooming with the touch pad

This was the most frequently suggested improvement to the tool from the user study. The tool now responds to zoom by either using the buttons, or by using two fingers on the touch pad. By moving two fingers closer to each other the tool will zoom out and by moving them away from each other the tool will zoom in.

- Video jumping to correct location directly after input

A fairly easy to implement function, but with great improvements to usability. The tool now effectively changes the time in the video preview upon changing the start-time in the control panel, allowing users to make small changes to acquire smooth transitions.
Bibliography


