VideoTrees: Improving the presentation of video surrogates using hierarchy

Michel Jansen
michel.jansen@cs.utwente.nl

ABSTRACT

As the amount of available video content increases, so does the need for better ways of browsing all this material. Because the nature of video makes it hard to process, the need arises for adequate surrogates for video that can readily be skimmed and browsed.

This paper explores the effects of the use of hierarchy in a pictorial summary of keyframes. A novel type of video surrogate is presented: the VideoTree. To test whether using hierarchy makes for a better user experience and performance, a prototype browser was developed and tested in a preliminary usability study.

Users performed better using the VideoTrees browser than using a regular storyboard-based browser. They also found it more flexible, yet more difficult and confusing to use.

1. INTRODUCTION

The Internet is no longer just for text. With broadband Internet connections becoming more ubiquitous, the amount of rich media like video content available on the Internet is becoming ever larger. With the amount of available video material increasing, so does the need for an effective way of searching, navigating and browsing through this material.

Since video can visually express many concepts that would be hard to capture unambiguously in words, searching by means of a keyword-based query like in search engines for text will not always work. Alternative approaches to searching exist, such as ‘query by example’ [9], where an example image or video provided by the user is used as a search basis. Unfortunately, they too suffer from a *semantic gap*: low-level visual features may not correspond to higher-level semantic visual concepts [9]. As concluded by Lew et al.: “We should focus as much as possible on the user who may want to explore instead of search for media” [16]. This research therefore focuses on navigating and browsing video.

Even if a search query results in only a small number of results, the user is still faced with the task of going through those results to judge each video on relevancy. This can be difficult and time-consuming. Video is temporal and linear in nature and therefore it is inherently difficult to process. This is in contrast with text, which is instead spatial in nature and can therefore easily be skimmed [11]. Since it is obviously not feasible to have to watch several hours of video to find a relevant section, the need arises for an adequate surrogate, which can abstractly represent video in a way that is easier to process, so assessing the relevancy of a video consumes less time and effort.

Extensive research has already been performed to come up with adequate surrogates for video material. There are also already many different ways of automatically generating sets of keyframes for inclusion in a video surrogate [22]. The question of how to effectively present these keyframe sets to a user, however, has remained unanswered.

This research will therefore assume the existence of methods for acquiring keyframe sets and aim to improve user experience and performance when browsing video by providing better visual abstractions as surrogates for the actual video. As concluded by Truong and Venkatesh [22], a lot of work is still to be done in optimising visualisation and representation of keyframe sets. This especially holds for keyframe sets that are potentially very large.

One promising approach to visually presenting such keyframe sets is by introducing hierarchy to control the amount of required screen real estate and consequently the amount of cognitive effort required to process the surrogate. Users can then start exploring video on a high, abstract level and ‘drill down’ to more detail. By testing a prototype that incorporates these principles in a laboratory setting, the following research question was investigated:

What are the effects of hierarchical presentation of keyframes in video surrogates on user performance and satisfaction?

The hypothesis is that by using hierarchy when abstractly representing video, users will be able to find what they are looking for in a video more easily and with a more pleasant experience. In order to be able to test this hypothesis and answer the research question as a whole, the following three sub-questions, which follow from the main question, were investigated.

- What methods of hierarchical visualisation are suitable for presentation of keyframes in video surrogates?
- What are the effects of hierarchical presentation on user performance?
- What are the effects of hierarchical presentation on user satisfaction?

To answer these questions, this paper will start by a short literature review of methods and tools required for creating video surrogates in general, and hierarchical video surrogates specifically.

Then the area of hierarchical data visualisation and hierarchical video browsing is explored. Subsequently, a prototype hierarchical video browser using VideoTrees, a novel video surrogate, is presented. This prototype was tested in a laboratory setting against a flat storyboard browser, where it was evaluated on task performance and user satisfaction.

2. CREATING VIDEO SURROGATES

In order to make video's easier to browse, coming up with an adequate surrogate or summary for that video is essential. Such a surrogate should be less complex than the original, while retaining as much of its informational value as possible. Ideally, any summary should have the following four properties, as defined by He et al [11]:

- *Conciseness* - be as short as possible
- *Coverage* - contain all relevant information
- *Context* - information is selected and presented such that their context is preserved
- *Coherence* - the flow of information should be fluid and natural

There are a number of approaches for creating video summaries and for any of these approaches, the original video has to be segmented in some way to select relevant sections for inclusion in the surrogate. In the rest of this section, the different types of video surrogates will be discussed, followed by different techniques for segmenting the video.

2.1 Types of video surrogates

There are roughly two approaches for creating video surrogates [22]: temporal and spatial summaries. Temporal summaries compress the entire video into a much shorter fragment and are more commonly referred to as *video-skims*. Spatial summaries spread the contents video over a two-dimensional or three-dimensional space and are often called *pictorial summaries*. Both video-skims and pictorial summaries have advantages and disadvantages, which will be shortly discussed for the case of video browsing.

2.1.1 Temporal video-skims

One way for creating an abstract video surrogate, is by literally making a shorter version of the original. Such a summary can provide its viewer with a "fast forward" skim through the original, while taking a lot less time to watch. By selecting the most salient segments of a video and placing those in sequence, a movie-trailer-like result can be obtained [10].

Because a video-skim has the same modality as its original, it can retain many of its properties, including its expressiveness [22]. The temporal order of events can be kept intact, as well as any audio present in the original, so it is easier to preserve context and coherence. This means that video-skims can provide viewers with a good gist of the original video [14]. Also, since the audio information is preserved, video-skims are generally good at summarising material with little informational value in the visual channel, but a lot of information in the audio, such as presentations and lectures [11]. For such cases, automatically generated summaries have shown to perform very close to summaries produced by human experts [11].

However, in retaining the original video's modality, a video-skim also retains its disadvantages. The resulting surrogate is still temporal and linear in nature and requires a user to watch it to gain information from it. Even if the original video's length is reduced to a fraction of its duration, this may still be quite long. Furthermore, since shortening the original video almost inevitably means throwing away information, increasing the conciseness of a video-skim comes with the trade-off of losing coverage.

Because of these disadvantages, video-skims are not very flexible for browsing or navigating inside a video. Therefore, they are not very suitable as stand-alone surrogates for such a case. Also, spatial surrogates have shown to be preferred by users [14]. Since we are seeking to improve user satisfaction as well as performance, video-skims will not be further explored in this paper.

2.1.2 Spatial pictorial summaries

Another approach for creating an alternative representation of a video is by spatially laying out its content. By extracting keyframes from the original source video and placing them together on the screen, optionally enhanced with additional content, such as textual transcripts, a pictorial summary or storyboard can be created.

Although such a static storyboard does not retain the same amount of expressiveness as its video source, because motion and sound are lost, its spatial rather than temporal nature does allow the viewer to gain an overview of the video 'at a glance', without having to sequentially step through the video [23].

Moreover, the spatial approach allows for the keyframes to be used as an index to the video. Many existing systems, like Boreczky's manga summaries [3] and the CueVideo system [21] allow the keyframes to be clicked to have the video playhead automatically jump to the desired fragment. Because of this property, and because users seem to prefer spatial surrogates over linear surrogates [14], pictorial summaries are often used in video retrieval systems.

However, spatial summaries suffer from drawbacks of their own. Eventually, the amount of space or screen real estate available is a limiting factor for any pictorial summary. If a user is to quickly grasp the contents of the surrogate, the amount of keyframes in the summary has to be kept under control. On the other hand, reducing the number of images also reduces the level of detail [22]. As a consequence, there is always a trade-off between conciseness and coverage.

It goes beyond the scope of this paper to cover all approaches to this problem. Instead we will suffice it to say that there are roughly two ways to improve the coverage of a pictorial summary. One one hand, increasing the salience of the images selected for inclusion in the surrogate by means of a good method of segmentation, also makes that surrogate more representative for the source video [18, 13]. On the other hand, improving the layout of the selected keyframes by adding clues to the relations between and importance of represented segments, increases the information value of the summary [3, 23]. As we will see later, the approach used by VideoTrees is based on a combination of these two factors: a semantic spatial layout and, combined with semantic video segmentation.

2.2 Video segmentation

Whether the summary being created is a video skim or a pictorial summary, a selection of which parts of the video to include in the summary has to be made. For a video-skim, this selection consists of a set of video segments, for a pictorial summary it is a set of keyframes. Again, it would go beyond the scope of this paper to give a complete overview of the field of video segmentation. Instead, only concepts and techniques relevant to this research will be introduced.

At its most basic, fine-grained level, any video consists of a number of *frames* placed in sequence [5]. For the video to appear smooth to the human eye, one second of video generally consists of about 24 to 30 frames per second [8]. This means that decomposing a mere 5 minutes of video would result in a minimum of 7,500 images. For some specific applications like video editing, this may be suitable, but for most applications the amount of frames will be too high [18].

More often, a *shot* is considered the smallest building block of video. A shot can be seen as the sequence of continuous action from the start to end of a single camera operation, as defined by Yeung and Yeo [23]. Segmenting a video into shots can be done automatically using shot boundary detection using a number of algorithms [4]. On the average, most video types were observed to have about 200 shots for 30 minutes of video [23]. Considering the screen space available on most modern computers, this many images can still not be displayed without resizing them to very small dimensions or requiring the user to scroll.

A level of granularity that is yet coarser than the shot level, is that of scenes. In this paper, the cinematographic definition of a scene will be used. By that definition, a scene is *"a subdivision of an act of a play in which the time is continuous and the setting fixed and which does not usually involve a change of characters "* [1]. Automatically detecting scenes is more difficult than detecting shots, because it generally involves the extraction of higher level visual features, such as the background setting [9]. In the video material used during this research, an average of 20 scenes were found in a 25 minute documentary.

For some types of video, an even higher conceptual level exists. Bertino et al. define concept levels as clusters of scenes grouped on semantic similarities [2]. For example, a news programme generally consists of a number of news items, which in turn may be grouped by subject into politics items, sports items and so forth. Depending on the type of video, any number of semantic levels may exist.

Using segments extracted from various levels of granularity, a *concept hierarchy* [2] can be built. This concept hierarchy will be one of the foundations of the VideoTree design.

3. HIERARCHICAL VIDEO BROWSING

In this section, the concept of hierarchical video browsing is introduced. Based on the concept hierarchy mentioned in the previous section, video can be seen as a hierarchical data structure, allowing existing techniques for visualising such structures to be applied. Some examples of past video browsers that include a notion of hierarchy in the past are given, before the VideoTree hierarchical browser is introduced.

3.1 Hierarchical visualisation

Considering the video concept hierarchy as a regular hierarchical data structure, many techniques for representing and navigating general hierarchical data can be equally applied to it. Two major issues in any interactive graph visualisation are layout and navigation [12]. In the rest of this section, these issues are described along with a number of ways to cope with them.

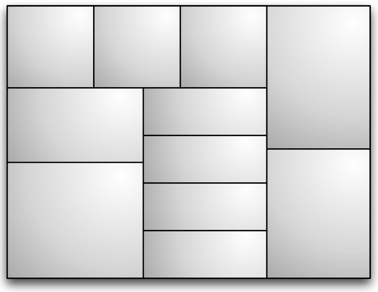


Figure 1: Example of the tree map layout method.

3.1.1 Layout

The main challenge in drawing any hierarchical graph is viewability [12]. Given a limited amount of space, all the nodes in a graph have to be drawn, while still keeping them discernible from each other and keeping the relations between the nodes intact. Herman et al. distinguish a number of popular graph layout techniques [12], which will be discussed here, along with their advantages and disadvantages.

A classic *Tree Layout* positions each of a node's children nodes below their parent. This way, a classic tree has a clear 'top down' direction, and it is easy to determine any node's position in the hierarchy, but it also suffers from space inefficiency. Because the 'breadth' of the tree grows with each level of the hierarchy, the graph gets exponentially wider or more cramped.

One variation to the tree layout that aims to counter this disadvantage, is the *radial tree*. Instead of positioning children directly below their parent, the nodes are laid out concentrically around their parent. This allows for a more space-efficient result, which has the drawback of being harder to comprehend. For instance, it is not very clear where the root of the tree lies or how deep any node is in the hierarchy.

Cone trees are identical to classical trees, only lifted to a three dimensional view. Using depth to layout the children of each node in a circle below that node wins some space, but also incurs the additional burden of having to view the node in 3D, to ensure no nodes are hidden behind their siblings.

A final visualisation method is that of *tree-maps* (Figure 1). Tree-maps differ from the previous methods in that they do not draw the nodes of a tree connected by edge lines, but instead represent trees as sequences of nested boxes. Because of this, tree-maps are very space-efficient. A drawback is that the structure of the tree is difficult to perceive.

3.1.2 Navigation

In an interactively navigable hierarchical visualisation, it is also important that it is possible to discern which node has focus, and which nodes are related to that node. Using good navigation techniques, even trees that would normally be too large to fit in one view can be made accessible [12].

Zooming and panning are traditionally important tools in graph visualisation [12]. By zooming in on a subsection of a graph, details that would be too small to be visible in the entire graph can be revealed. Panning refers to the action of moving the zoomed-in view on a graph around.

A well-known problem with zooming is that in a zoomed-in view, contextual information is easily lost [12]. Some techniques have been developed to preserve the context while still being able to focus on a subsection of the graph. These techniques have been called *focus+context*.

One very powerful focus+context technique was presented by Lamping et al. is the use of *hyperbolic geometry*. By drawing the graph in hyperbolic space and projecting it onto a circular display region, a distortion resembling a fish-eye effect occurs [15]. Anything near the centre of the circle, the focus, is magnified. The surrounding space is distorted and compressed towards the sides of the circle, yet still visible.

By combining zooming and panning with focus+context techniques, users can navigate a map quickly, while still retaining a sense of the context around focused nodes, at the cost of distorting the graph view.

3.2 Hierarchical video browsers

A number of video browsers already use the concept of hierarchy to improve the usability or informational value of their content. This section gives a short review of related work in the field.

Mills et al. introduced the Hierarchical Video Magnifier [17]. This system allowed the user to select a range on timeline to have a storyboard view pop out and show a graphical storyboard corresponding to the selected timeframe. This storyboard in turn, had a timeline of its own and as such, the magnification step could be repeated numerous times. The keyframes present in each storyboard were linearly sampled from the source video.

The Video Posters by Yeung and Yeo also allow the user to click on a pictorial representation of a video segment to 'drill down' to its contents [23]. At the top level, a video consists of 'stories' which are represented by collages of keyframes. When a user zooms in on a story, a list of keyframes for that story is displayed.

The ClassView system by Fan et al. uses hierarchical video shot classification to generate a segmented view of different levels of semantic concepts [9] for inclusion in a storyboard view. Similarly, the hierarchical movieDNA system clusters segments by semantic concepts [20], but displays them differently. In hierarchical movieDNA, segments are depicted by dots on a strip, like a DNA readout, which can be 'brushed' with the mouse to drill down in a pop-up.

Another example of a video browser that uses a video's conceptual hierarchy for drilling down, is the manga representation by Borezcky et al. [3]. By displaying the contents of a video as an interactive comic book, users found working with the system to be a lot more pleasing.

3.3 The VideoTree Hierarchical Browser

In this section we propose VideoTrees, a novel way for representing the contents of a video along with the VideoTree Hierarchical Browser, which is used to browse a video using VideoTrees.

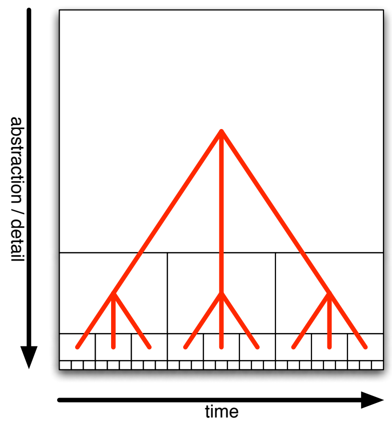


Figure 2: Schematic overview of the VideoTree presentation style. The vertical axis represents 'level of detail', the horizontal axis represents 'time'.

3.3.1 VideoTrees

The concept of VideoTrees is based on visualising the conceptual hierarchy of a video in a pictorial surrogate. By extracting keyframes from each level of the source video, and placing them correspondingly in a tree, a hierarchical representation of the video's contents is created, where each level of the tree contains more keyframes and consequently more detail.

By representing each node in the hierarchy by a keyframe, and placing the images adjacent to their parents and siblings, there is no need for edge lines to indicate relations between nodes. The tree layout used is a combination of the classic tree layout and the tree-map layout: every child is placed below its parent, yet the collective width of the children is restricted to the width of the parent. This results in a composition similar to the one in Figure 2.

For simplicity, the aspect ratio of each frame is preserved. This means that all nodes are treated equal, even if their duration differs significantly. In other words, the information about the duration of a node is not encoded in the resulting composition. Keeping the aspect ratio of the frames also has the consequence that if any part of the tree is unbalanced, nodes of the same level may not remain vertically aligned with nodes of a different parent, as visible in Figure 3.

With the four aspects of *conciseness*, *coverage*, *context* and *coherence*, as mentioned in section 2: 'Creating video surrogates', in mind, VideoTrees have many desirable properties for a video surrogate.

First of all, the temporal order of the segments included is preserved. Moving from left to right in the tree, means moving forward in time. This way, the resulting surrogate is very coherent. It keeps a close relation to the video and can be seen as an alternative to the 'slider' control present in most video players.

Also, because a VideoTree is built as a hierarchy of subtrees, the tree as a whole can contain a lot of detail. The level of coverage is very high, while nodes on each level in the tree still contain a limited number of children and can therefore still remain concise. Depending on the branching factor of the tree, the width of an entire VideoTree can easily be contained to a region with the width of the root frame and twice its height. The screen usage is therefore very efficient.

Contrary to the visualisation methods used by the video browsers mentioned before, a VideoTree contains represents the entire video *in a single composition*. A VideoTree contains keyframes for each level of the concept hierarchy, up to as much detail as the shot level. For a user to be able to reach this level of detail, a good method of navigation is crucial to benefit from the properties of VideoTrees. The VideoTree browser, which is described next, is a first attempt at creating such a browser.

3.3.2 Browsing using VideoTrees

The most challenging part of creating a browser based on VideoTrees, is allowing the user to navigate the tree to see the details in the lower branches, without getting lost. This challenge can be reduced to three questions a system should answer for the user at any given time [20]:

1. Where am I?
2. Where can I go?
3. Where is X?

To support the user in navigating through the video surrogate, navigation was implemented by making the VideoTree dynamic. At any given time, the node that has the user's focus is centred in the view. Figure 3 is a screenshot of a typical situation where one node has the focus. The adjacent segment-nodes on the same level are shown to the left and right, the parent node is shown above the focused node, and all children with respective sub-trees are shown below the node.

Clicking any node adjusts the focus by panning and zooming, so the focused node is centred in the view as described earlier. If the newly focused node is on the same level as the previously focused node, the resulting transition will only consist of a panning motion. If the clicked node is on a higher or lower level, the view will zoom out or in as needed. All motions are smooth so it is clear what is happening.

Clicking a node also results in the playhead position of a video player placed on the side to jump to the time index corresponding to the node. The position of a user in the video is always a combination of the focused node, which is centred in the navigator view, and the playhead position, which can be determined from the slider.

To show the user his options during navigation, the mouse cursor changes upon hovering over any node. As can be seen from Figure 3, there are four directions: up, down, left and right. There is always at most one sibling node on each side, so the user can only pan one segment left or right at a time. For going up there is usually also only one option: the parent of the selected frame. However, if the selected node lies on the edge a higher level segment, both its parent and 'uncle' node are clickable. In other words, diagonal movement is also possible. For 'drilling' down, there are always the most options. Not only all of the selected node's children can be clicked for focus, but all the nodes below the one selected.

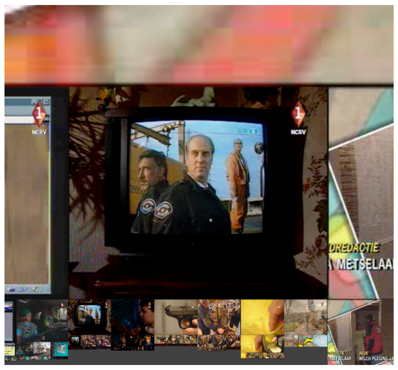


Figure 3: Screenshot of a part of the user interface in a typical situation when browsing video using the prototype: an semantic-level node is selected, showing its neighbours on the same level to the left and right, more detailed scenes, shotclusters and shots below and one level above.

When navigating through a VideoTree, the node which has focus is always clearly visible, as well as all of the nodes below it. The context above and to the left and right of the node is more limited. This may cause a sense of disorientation in the user. In graph visualisation where zooming is required, this is often solved by including a thumbnail 'navigator' view with a smaller rectangle inside to represent the current viewport [12]. Due to time limitations, however, such a 'navigator' was not implemented in the VideoTree browser.

4. USABILITY STUDY

In this study, a prototype of the hierarchical VideoTree browser was compared with a regular storyboard-based video browser, which served as a baseline. Using both a laboratory experiment and a post-experimental questionnaire, the goal was to determine whether the hierarchical browser outperformed the storyboard-based browser in task performance and user satisfaction. Because of the preliminary nature of the study, only a small number of participants was used. Each of the participants was asked to perform the tasks using both prototypes and participate in the survey.

4.1 Methodology

To test the performance and user satisfaction of the VideoTree browser, a within-subject laboratory experiment was conducted. Participants were presented with a number of search tasks, which they were asked to perform using the VideoTree prototype described in section 3.3: 'The VideoTree Hierarchical Browser' and a prototype with an identical layout, yet with a flat 'storyboard' overview instead. User satisfaction was measured using a questionnaire before, during and after the performance tests.

4.1.1 Participants

The usability study was conducted with a group of 15 students, with an average age of 22, with various levels of previous experience with video browsing. None of them had worked with VideoTrees before.

4.1.2 Video material and segmentation used

Both the VideoTree prototype and the storyboard prototype were configured to operate on a different episode of 'Willem Wever', a

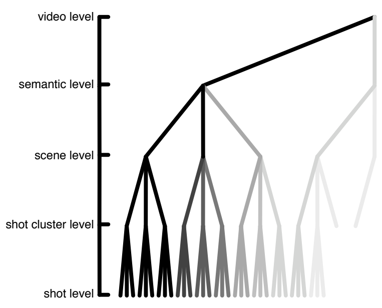


Figure 4: Schematic overview of hierarchical levels present in a typical video.

Dutch educational program for children. This material was taken from the TRECVID 2007 set for shot boundary detection and was segmented into shots using the reference shot-boundary indices provided by Christian Petersohn for TRECVID [19]. Any resulting segments that were either the product of a falsely detected or a missed shot boundary, were left alone for realism.

Next, the shot-segments were then manually grouped into scenes, based on changes in the scenery or setting. The resulting scene-segments were then grouped into logical semantic segments. 'Willem Wever' is a program where children can submit questions on a particular subject, which will then be investigated in the program. In each episode, a number of questions are treated and transitions between these questions are indicated by a short cut-scene of the 'Willem Wever' logo and tune. By manually segmenting the video on occurrences of this cut-scene, five semantic segments per video were obtained. Two segments for respectively the intro and the outro of the episode and three segments dealing with the questions.

Finally, because the number of shots per scene turned out to be high compared to the number of scenes per item and the number of items per movie, an extra level of segmentation was added. By grouping the N shots of each scene into $\sqrt{N}$ clusters of size $\sqrt{N}$, a rudimentary shot-cluster level was placed between the shot level and the scene level. The resulting hierarchy of abstraction can be seen in Figure 4.

Since only the VideoTree prototype actually used all of these levels of abstraction, the complete segmentation was only applied to one episode. This resulted in 5 semantic segments, 16 scenes, 58 shot-clusters and 262 shots. The storyboard browser only made use of the 23 scene-level segments extracted from its episode. For each of these segments, the first keyframe of the segment was taken as a representation. The duration of the material used in the VideoTree browser and the storyboard-browser was respectively 25:33 and 24:05.

4.1.3 Test set-up

For use in the usability study, both the VideoTree browser and the storyboard browser were implemented in Flash and loaded in a webbrowser. Figure 5 shows the two resulting prototypes.

The user interface of both prototypes was built up of two basic components: a video player, positioned on the right, and a browser pane, positioned on the left. Since the goal of this study was to test the concept of hierarchical browsing, both prototypes were kept identical except for the layout of the contents of the browser pane.

The browser pane had a size of 640 pixels wide and 720 pixels high in both prototypes. In the case of the VideoTree browser prototype, it contained a navigable VideoTree. In the 'flat' prototype, it contained a static storyboard view. A click on one of the keyframes in either browser pane, resulted in the video player jumping to that position in the video and starting playback from there.

The video player was sized to display a 352 by 264 pixels video, as is common for many on-line video application. It was given a horizontal seek bar or slider as its only method of control, to prevent users from resorting to controls they were already familiar with. Dragging or clicking the slider resulted in the video seeking to the position clicked on or dragged to. Actions on the slider had no influence on the browser pane in either prototype.

For the sake of time registration, a large button labelled 'Register' was included in the interface of both prototypes. This button was used solely for tracking when users started and completed each task. It had no effect on the operation of the browser itself.

4.1.4 Task performance test

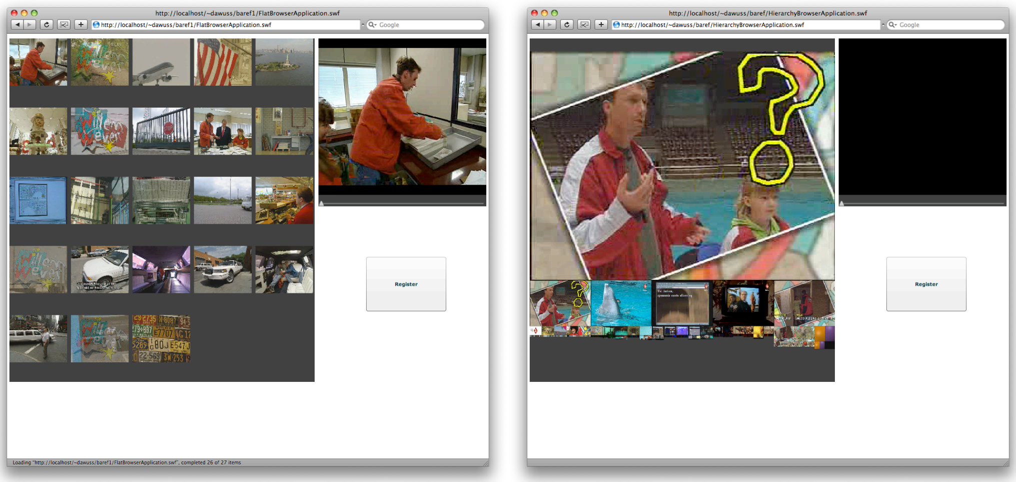
Using the test set-up and material described, a task performance test was performed to measure how fast and efficient users could work with the system. Each participant was asked to perform a number of search tasks with both the VideoTree prototype and the storyboard browser. To prevent order-effect biases, the order of the prototypes was balanced among the participants.

Before working with each prototype, participants were told they would be given four questions, all of which were answered in the video they were going to work with. They were asked to find the fragments in that video where the answers to the questions were given as fast as possible. There were no penalties or rewards involved, and participants were told they could do anything they wanted with the prototype.

Due to time limitations, each prototype had only been prepared to work on one video. This meant that users could not be given any time to explore the prototype in advance, because they might already find the answers to the questions they would be asked.

For each of the two video's used in the task performance test, four questions were developed. For symmetry, the nature of these questions and their answers were chosen to be as similar as possible. For instance, the answers to the first question for both prototypes was located about 20 seconds from the start of the second scene of the second semantic block. The questions used in the task performance test were as follows:

- Prototype 1 (storyboard)
    1. What is the question asked by the first child? (find the fragment where he asks this question)
    2. What is the answer to the question asked by the first child? (find the fragment where the answer is given)
    3. Who was responsible for 'productie New York' for this episode of 'Willem Wever'?
    4. Who wrote the poem on the 100 guilders bill?



(a) The 'flat' storyboard prototype

(b) The 'hierarchical' VideoTree prototype

Figure 5: Screenshots of the two prototypes used in the usability study.

- Prototype 2 (VideoTree)

  1. What is the question asked by the first child? (find the fragment where she asks this question)
  2. For what purpose is a 'target' used when training dolphins? (find the fragment where the answer is given)
  3. Who was responsible for 'mixage' for this episode of 'Willem Wever'?
  4. What does a bullet look like after it has gone through six layers of steel?

These questions were presented in order and one at a time. Only after a participant had completed answering one question, he was allowed to continue on the next question.

During the task performance tests, the following performance variables were measured:

- Total time taken to complete all tasks
- Percentage of video watched

Additionally, a number of variables that are indicative of how users interact with the system were recorded. These were:

- Number of tracking-actions using the video player's seek bar
- Keyframe node click actions

Although these variables cannot be used as an absolute measure of performance, they may be used to explain why one system outperforms the other or what aspects may attribute to the user's experience of the system.

4.1.5 User satisfaction

Another important sub-question to the research question, is whether hierarchical presentation influences user satisfaction. To answer this question, a survey was conducted among the participants of the task performance test. The survey consisted of four short questionnaires. The first was presented before the tests, asking some general demographic information. Next, two identical questionnaires on user satisfaction were presented, one after the participant worked with each prototype. Finally, a questionnaire asking users to express their preference between the two prototypes was presented after the test.

After working with each prototype, participants were asked to rate both the 'flat' and the hierarchical prototype on a number of factors in a questionnaire on paper. Since we are testing a novel approach to video browsing, it is important to know if users are willing to accept this new kind of video browser. The Technology Acceptance Model (TAM) [7] defines two variables that are of influence on the acceptance of any new technology: *Perceived Usefulness* and *Perceived Ease of Use*. Also, since the task of browsing is essentially a way of controlling what information is displayed, the *Perceived Level of Control* is also of influence on a user's satisfaction with a given video browser and was treated separately. For each of these three variables, a number of subjective terms, based on those used in the Questionnaire for User Interface Satisfaction (QUIS) [6], were included in the questionnaire:

- Perceived Ease of Use
  - Difficult vs. Easy
  - Frustrating vs. Satisfying
  - Very clear vs. Confusing
- Perceived Usefulness
  - Inefficient vs. Efficient
  - Useless vs. Useful

Table 1: Results of the performance test and user satisfaction questionnaire.

	Storyboard		VideoTree	
Question	Mean	S.D.	Mean	S.D.
Rigid vs. Flexible	2.71	0.61	3.64	1.22
Difficult vs. Easy	3.79	0.89	3.00	1.18
Inefficient vs. Efficient	3.14	0.86	3.36	1.45
Ineffective vs. Powerful	3.07	0.73	3.50	1.09
Frustrating vs. Satisfying	3.57	1.02	3.29	1.14
Useless vs. Useful	3.86	0.77	3.71	0.99
Confusing vs. Very clear	4.00	0.96	2.50	1.16
Using the system gave me greater control in performing the tasks	3.36	0.93	3.57	1.16
Using the system made it easier to complete the tasks	3.57	1.02	3.79	1.19
The overview (on the left) gave me a clear idea of what the video was about	3.50	1.23	3.14	1.29
Time spent	553.53	130.14	487.40	136.75
Percentage Watched	33.73	6.85	26.33	8.09

- Perceived Level of Control
  - Ineffective vs. Powerful
  - Rigid vs. Flexible

Each of these terms were measured using a semantic differential in the form of a question containing a five point scale with the negative and the positive adjectives anchored on either side. To prevent 'response sets', the position of the positive and negative labels of some questions were reversed.

To gather some additional information more specific to the task of video browsing, participants were asked for their opinion on the following three statements:

- Using the system gave me greater control in performing the tasks
- Using the system made it easier to complete the tasks
- The overview (on the left) gave me a clear idea of what the video was about

The first two questions were adapted from Davis' item pools for perceived usefulness [7] to measure respectively the Perceived Level of Control and Perceived Usefulness. The third question was included to determine the perceived quality of the surrogate as a whole. Each of these questions were stated using a five-point Likert-scale, which was laid out in the same way as the semantic differentials, with labels on each end of the scale.

Finally, participants were encouraged to 'spill their mind' at the end of the survey, where a comments field asking them for any remarks regarding their experience working with the prototype. The last questionnaire also asked users to indicate which prototype they would prefer over the other.

4.2 Results

All participants completed the search tasks for both prototypes in less time than the duration of the original video. One participant did not complete the questionnaires for both prototypes, so his answers were not included in the comparison of the user satisfaction scores. The results for both search task performance and user satisfaction are shown in Table 1.

To compare the answers given by each participant for the different prototypes, a Wilcoxon Signed Rank Test was performed on the equal pairs of questions. The four variables for which significant differences were found between the two prototypes are shown in Table 2.

Table 2: Significant results from the Wilcoxon Signed Rank Test.

	Storyboard	Tie	VideoTree	Z	p
Flexible	3	1	10	-1.998	0.046
Easy	8	4	2	-2.066	0.039
Clear	10	3	1	-2.563	0.01
% Watched	11	0	4	-2.017	0.044

A significant effect (p = 0.046) was found for flexibility. Of the fourteen participants, ten participants rated the VideoTree prototype higher on flexibility than they rated the Storyboard prototype. Across all participants, the mean score of the VideoTree prototype was 3.64, versus 2.71 for the Storyboard prototype (Table 1).

When asked to rate the prototypes between Easy and Difficult, participants rated the VideoTree prototype significantly often as more difficult than the Storyboard prototype (p = 0.039). The mean scores of 3.79 for the Storyboard prototype, compared to 3.00 for the VideoTree prototype indicate that that this difference is not as large as for flexibility, but still notable.

The VideoTree prototype was also rated significantly often as being less clear than the Storyboard alternative (p = 0.01). As can be seen from Table 1, the difference for this question is rather large: the mean score for Confusing vs. Very clear was 2.50 for the VideoTree browser compared to 4.00 for the Storyboard prototype.

On the task performance tests, the VideoTree prototype performed slightly better on both the total amount of time spent as the percentage of video watched. Only the scores for the percentage of video watched were found to be significant (p = 0.044). Out of fifteen participants who completed the tests, eleven completed the task having watched a smaller percentage of the video with the VideoTree browser than with the Storyboard browser. For this task, Table 1 shows a notable difference of over 7 percent points.

5. DISCUSSION

In the previous sections the VideoTree surrogate and browser were introduced and subjected to a usability study. The performance test and the user satisfaction surveys from that study had a number of interesting results. This section discusses those results, and possible explanations.

First of all, while users found the VideoTree prototype more flexible and they were able to complete the tasks in less time and watching a smaller percentage of the video footage, they also found it more difficult to use. This was supported by users' comments during and after the performance tests. Multiple participants stated that once they figured out how to work with the VideoTree browser's navigation, they could work with the system a lot faster.

Because users were not given any introduction or exploration time beforehand, some users never figured out the interactive navigation at all. This was the case for both the Storyboard prototype and the VideoTree prototype. Users who had this problem did all the navigation using the very limited slider control of the video player in the top right corner.

Since the length of the entire video, which was over 24 minutes, was represented by the small slider, users struggled with it's accuracy. While this was not an expected result, it does attribute to an overall lower appreciation of both video browsers. This was aggravated by a bug present in both prototypes, which caused the slider to lag when seeking backwards. Because the slider control was the same in both prototypes, it is does not affect the comparison between the prototypes.

Another result from the user satisfaction survey shows that the VideoTree prototype was found to be more confusing than the Storyboard prototype. It was expected that adding more detail and requiring navigational through that detail using zooming and panning would attribute to the perceived complexity of the browser. As can be seen in Table 1, the Storyboard prototype simply scored really high on clarity. Still, there are a number of things that may have attributed to participants finding the VideoTree browser confusing.

One aspect of the current VideoTree implementation that was found confusing, is that it does not visualise duration. Two nodes are given the same size, even if they significantly differ in lenth. This makes it hard for uses to estimate the duration of a segment. A number of users were observed getting stuck in the introduction sequence, which had a lot of shot switches in a very short time. Because there was no visual cue telling users that this segment was actually a lot shorter than the next segments, it took them longer to figure this out. Adjusting the width of the nodes to encode their relative duration may provide such a cue, at the cost of losing the original aspect ratio.

The choice of preserving the aspect ratio of each keyframe also resulted in some nodes ending up on a different height, even though they were on the same level. At least one user was bothered by this, stating he was confused by the nodes not being aligned.

Another issue the the VideoTree prototype suffered from, was a lack of progress indication. Users found it hard to relate the slider to the VideoTree navigator view, to see what part of the video was playing. They often made jumps backwards from where they were playing, only because they did not realise they had already passed that segment. Having a visible progress indicator in the VideoTree display would also aid in determining the length of different segments.

For shots of a very short duration, like the ones in the introduction, the shot-level segments turned out to be not very useful. For any segment shorter than a few seconds, users found it easier to just watch the fragment than to navigate inside it in the VideoTree. This effect was especially apparent in conversations, which take up a lot of nodes in the tree, while their total duration is short. This suggests that nodes of a duration below a certain threshold might better be merged or removed.

Also lacking, was an indication of how far one is zoomed in. The thumbnail navigator view that was originally planned might have helped, but for a future version of a browser based on VideoTrees, more advanced focus+context techniques should also be considered.

After users found out how the hierarchical browsing worked, most agreed that this was an interesting way of browsing video, but they needed some practice before they could work with it. This suggests that the learning curve on VideoTree browsing makes it more suitable for situations where the detail it makes accessible is needed.

6. CONCLUSION

In this paper, the benefits of hierarchical presentation of keyframes in video browsing were explored. It was expected that by using hierarchy when abstractly representing video, users will be able to find what they are looking for in a video more easily and with a more pleasant experience. A literature study on methods for hierarchical visualisation of keyframes lead to the design of a novel type of video surrogate: VideoTrees.

VideoTrees are capable of coherently and concisely representing video in detail in a hierarchical form. The VideoTree browser was developed as a way of navigating VideoTrees for browsing video.

A prototype of this VideoTree browser was compared with a storyboard browser on task performance and user satisfaction in a preliminary usability study. Users found the VideoTree prototype more flexible, but also less easy to use. While they were able to complete the tasks faster having to watch less of the video, they also found the prototype more confusing. These problems may be attributed to a number of issues, such as the lack of progress indication, the lack of visual cues to the duration of segments in the VideoTree and the absence of a clear sense of how far the user is 'zoomed in' on the VideoTree. For a future browser to address this problems, a number of suggestions were given.

All in all, hierarchical presentation as it was applied in the VideoTree concept, is definitely of influence on user performance and satisfaction. It improves performance and perceived flexibility at the cost of adding more complexity to the video browser's user interface. This makes a hierarchical browser less easy to use. A number of factors that made the VideoTree browser more confusing may be easily addressed. Future research should determine ways for making this kind of hierarchical browser less confusing. Until then, the learning curve present in the VideoTree browser makes it only useful for situations where a high level of flexibility and detail is desirable over simplicity.

7. ACKNOWLEDGMENTS

There are a lot of people to whom I owe thanks for helping me complete and improve this paper. Specifically, I would like to thank the following people: Betsy van Dijk and Willemijn Heeren for their excellent guidance during the course of this research. Roeland Ordelman, for helping me with the video material. Hester Bruikman, for helping me on the questionnaires and data analysis. Also, thanks to everyone who provided me with the feedback necessary for writing this paper and to everyone who provided me with tips on implementing the prototype [1].

REFERENCES

[1] *The New Oxford American Dictionary*. Oxford University Press, 2nd edition, March 2005.

[2] E. Bertino, J. Fan, E. Ferrari, M.-S. Hacid, A.K. Elmagarmid, and X. Zhu. A hierarchical access control model for video database systems. *ACM Trans. Inf. Syst.*, 21(2):155–191, 2003.

[3] John Boreczky, Andreas Girgensohn, Gene Golovchinsky, and Shingo Uchihashi. An interactive comic book presentation for exploring video. In *CHI '00: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 185–192, New York, 2000. ACM.

[4] John S. Boreczky and Lawrence A. Rowe. Comparison of video shot boundary detection techniques. In Sethi Ishwar K. and Jain Ramesh C., editors, *Storage and Retrieval for Still Image and Video Databases IV*, volume 2670, pages 170–179, San Jose, CA, USA, 1996. Society of Photo-Optical Instrumentation Engineers.

[5] W.-G. Cheng and D. Xu. Content-based video retrieval using the shot cluster tree. In *2003 International Conference on Machine Learning and Cybernetics*, volume 5, pages 2901–2906, Xi'an, 2003.

[6] John P. Chin, Virginia A. Diehl, and Kent L. Norman. Development of an instrument measuring user satisfaction of the human-computer interface. In *CHI '88: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 213–218, New York, 1988. ACM.

[7] Fred D. Davis. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly*, 13(3):319–340, September 1989.

[8] Michael Demtschyna. Pal vs ntsc. http://www.michaeldvd.com.au/Articles/PALvsNTSC/PALvsNTSC.asp, December 2007.

[9] J. Fan, A.K. Elmagarmid, X. Zhu, W.G. Aref, and L. Wu. Classview: Hierarchical video shot classification, indexing, and accessing. *IEEE Trans Multimedia*, 6(1):70–86, 2004.

[10] Y. Geng, D. Xu, and S. Feng. Hierarchical video summarization based on video structure and highlight. In *Joint IAPR International Workshops on Structural, Syntactic, and Statistical Pattern Recognition, SSPR 2006 and SPR 2006*, volume 4109 LNCS, pages 226–234, Hong Kong, 2006.

[1] Available from http://micheljansen.org/VideoTree

[11] Liwei He, Elizabeth Sanocki, Anoop Gupta, and Jonathan Grudin. Auto-summarization of audio-video presentations. In *MULTIMEDIA '99: Proceedings of the seventh ACM international conference on Multimedia (Part 1)*, pages 489–498, New York, NY, USA, 1999. ACM Press.

[12] I. Herman, G. Melancon, and M.S. Marshall. Graph visualization and navigation in information visualization: a survey. *IEEE Trans Visual Comput Graphics*, 6(1):24–43, 2000.

[13] H. Iyer and C.D. Lewis. Prioritization strategies for video storyboard keyframes. *J. Am. Soc. Inf. Sci. Technol.*, 58(5):629–644, 2007.

[14] Anita Komlodi and Gary Marchionini. Key frame preview techniques for video browsing. In *DL '98: Proceedings of the third ACM conference on Digital libraries*, pages 118–125, New York, NY, USA, 1998. ACM Press.

[15] John Lamping, Ramana Rao, and Peter Pirolli. A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 401–408, New York, 1995. ACM Press/Addison-Wesley Publishing Co.

[16] M.S. Lew, N. Sebe, C. Djeraba, and R. Jain. Content-based multimedia information retrieval: State of the art and challenges. *ACM Trans. Multimedia Comput. Commun. Appl.*, 2(1):1–19, 2006.

[17] Michael Mills, Jonathan Cohen, and Yin Yin Wong. A magnifier tool for video data. In *CHI '92: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 93–98, New York, NY, USA, 1992. ACM.

[18] P. Mundur, Y. Rao, and Y. Yesha. Keyframe-based video summarization using delaunay clustering. *Int. J. Digital Libr.*, 6(2):219–232, 2006.

[19] Christian Petersohn. Fraunhofer hhi at trecvid 2004: Shot boundary detection system. TREC Video Retrieval Evaluation Online Proceedings, TRECVID, 2004, http://www-nlpir.nist.gov/projects/tvpubs/tvpapers04/fraunhofer.pdf.

[20] D. Ponceleon and A. Dieberger. Hierarchical brushing in a collection of video data. In *System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on*, pages 8 pp.–, 2001.

[21] S. Srinivasan, D. Ponceleon, A. Amir, and D. Petkovic. "what is in that video anyway?": in search of better browsing. In *Multimedia Computing and Systems, 1999. IEEE International Conference on*, volume 1, pages 388–393 vol.1, 1999.

[22] Ba Tu Truong and Svetha Venkatesh. Video abstraction: A systematic review and classification. *ACM Trans. Multimedia Comput. Commun. Appl.*, 3(1):3, 2007.

[23] M.M. Yeung and Boon-Lock Yeo. Video visualization for compact presentation and fast browsing of pictorial content. *Circuits and Systems for Video Technology, IEEE Transactions on*, 7(5):771–785, 1997.