Interface Designs for Pen-Based Mobile Video Browsing

Wolfgang Hürst [1,2]

huerst@cs.uu.nl

[1]Utrecht University
Department of Information and Computing Sciences
PO Box 80.089, 3508TB Utrecht, The Netherlands

Georg Götz [2]

mail@georg-goetz.de

[2]Albert-Ludwigs-University, Freiburg
Institute for Computer Science
Georges-Köhler-Allee, 79114 Freiburg, Germany

ABSTRACT
In this paper, we describe two interface designs for mobile video browsing on pen-based handheld devices. Both feature different interaction types, such as speed- and position-based navigation which enable users to interactively skim a video's content along the timeline at different granularity levels. The feasibility and usefulness of the proposed designs is demonstrated in a comparative user study.

Categories and Subject Descriptors
H.5.2 [**Information Interfaces and Presentation (e.g. HCI)**]: User Interfaces – *Graphical user interfaces (GUI), input devices and strategies, interaction styles, screen design*

General Terms
Design, Experimentation, Human Factors.

Keywords
Handheld devices, pen-based computing, mobile video, interaction modes, video browsing, interactive navigation.

1. INTRODUCTION
Usage of video on handheld devices such as multimedia cellphones and PDAs is usually quite different from watching TV at home. For example, while being "on the move" (e.g. when riding a train or waiting at a bus station) people often do not have much time, can not or do not want to concentrate over a longer period, etc. and therefore sometimes tend to watch rather short clips or only selected parts of longer files. Given the small size of the display, people usually do not gather around a PDA in larger groups to watch, for example, a TV show together. However, that doesn't mean that replaying mobile video is a pure individual activity. For example, in a recent study O'Hara et al. (2007) also identified several "cooperative" scenarios where people used their mobile devices, for example, to present new clips to each other, show their most favored scenes of a particular movie, etc. One general observation with such mobile usages is that people often do not passively watch video like they do at home but consume the content rather interactively, for example, by replaying a short clip several times to show it to different people, by skipping scenes to only watch selected parts of a video because they have limited time, etc.

Despite the different viewing behavior, the interface of common mobile media players is often designed in a traditional way, enabling users to start, stop, and pause, fast forward/backward, and sometimes scroll along the timeline using a slider interface (cf. Fig. 1). Such a design can hardly fulfill all the needs for the interactive scenarios described above. For example, when browsing a file's content by modifying its replay speed (*speed-based browsing*), the optimum speed value depends on various issues such as task (e.g. quick browsing to get an overview vs. slow browsing to closely examine a particular event), data (e.g. slower browsing speeds might be favorable if there is a lot of activity going on), and last but not least individual user perception (e.g. some people might prefer slower speeds whereas others are able to process information faster). Hence, it seems more appropriate to offer some sort of slider-like interface where a user can easily select from a variety of replay speeds instead of just a few buttons with a fixed set of discrete speed values.

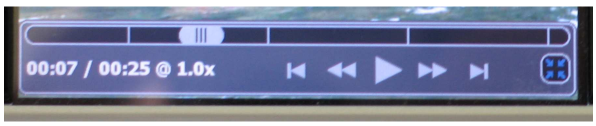

Figure 1. Standard interface of a mobile video player.

A timeline-based slider is another a very powerful interaction element for navigation in a video: If visual feedback is provided in real-time while the user moves the slider's thumb along the timeline, users can, for example, pass larger areas of lower interest quickly, abruptly stop or go back if a part of interest is found, etc. (*position-based navigation*). One problem with sliders is that they do not scale to large documents: Since a slider's length is restricted by size and resolution of the display, not all parts of a file can be accessed directly via the timeline. Common approaches to solve this problem include adding additional sliders with different scales to the interface (e.g. Casares et al. 2002, Richter et al. 1999) or providing some widgets to modify the scale's resolution. However, these solutions are generally not suitable for handheld devices because the number of GUI elements involved would take up too much space of the limited screen size. In addition, small widgets like a slider's thumb are often hard to target with a pen (esp. if the thumb is moving during normal replay).

In this paper, we present two interface designs for pen-based mobile video browsing each featuring two interaction styles: speed-based navigation (i.e. browsing via manipulation of replay speed) and position-based skimming (i.e. browsing via interactive navigation along the timeline using a slider-like interface). Both designs get by without explicit widgets and use very little GUI components thus being well suited for the small screen sizes of common mobile devices. The first interface is a slightly modified version of a design we recently introduced in Hürst et al. 2007a and 2007b, the second one is a completely new design which offers a similar functionality but in different ways (Section 2). After a short description of the actual implementation (Section 3), we present a comparative evaluation of the two interfaces (Section 4) which confirms their usefulness, identifies advantages of the particular designs, and provides directions for future work (Section 5).

2. THE TWO INTERFACE DESIGNS

Before introducing the *ScrollWheel design* for mobile video browsing in Section 2.2, we first summarize our previous work on the *MobileZoomSlider* in Section 2.1 insofar as it is required for the evaluation presented in Section 4. For a detailed description of the interface and the related evaluation we refer to Hürst et al. 2007a.

2.1 The MobileZoomSlider Design

Basic idea. Instead of relying on explicit widgets and GUI elements, with the *MobileZoomSlider*, users can click anywhere on the screen and scroll forwards and backwards along the timeline by moving the pen to the right and left, respectively. The scale of this "virtual", on-screen timeline depends on the vertical pen position: At the bottom, the same scale resolution as with the original slider is used. At the top, the finest scale is used, i.e. each pixel is mapped to a single frame in the video. In between, a linear interpolation is done (cf. Fig. 2). The main advantages of this interface design are that users do not have to target a very small widget (the slider's thumb) and that they can directly access different scales without the need for explicit adjustments.

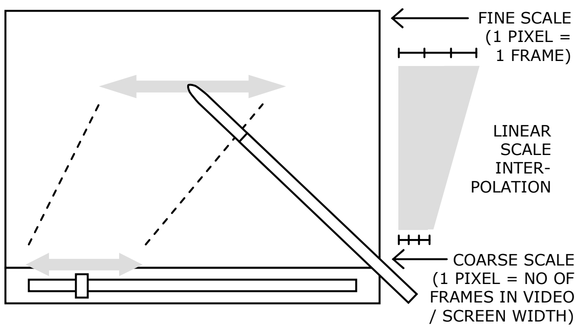

Figure 2. MobileZoomSlider: Position-based navigation.

In addition to this position-based navigation, users can browse a video at different replay speeds by clicking on the right border of the screen. Again, the second dimension is used to modify granularity: The fastest possible replay speed is accessible at the bottom of the screen. Slow motion is realized at the top. In between, replay speed is interpolated (cf. Fig. 3). Again, the main advantage of this design is that users have direct access to a variety of different speed levels without the need to explicitly grab a very small icon which is hard to target. In addition, both interaction styles are realized without explicit widgets, thus leaving more screen space for the actual content, i.e. the presentation of the video.

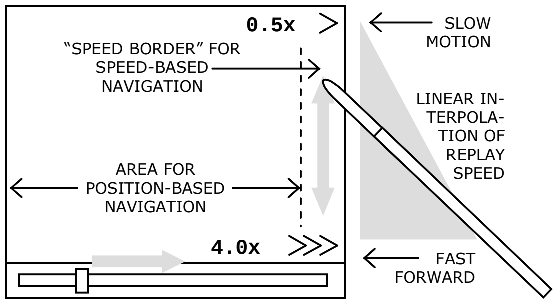

Figure 3. MobileZoomSlider: Speed-based navigation.

Initial implementation. Scale resolution and speed values in the described design are selected to "horizontally" match each other (i.e. starting with a fine scale and slow replay at the top, values continuously increase to coarser resolutions and faster speeds) in order to guarantee a smooth transition between the two interaction modes, i.e. position- and speed-based navigation. However, because of some problems with the initial implementation, there is a significant time delay when entering the right "speed area" of the screen. The same problem prevented us from implementing backwards skimming at the left side of the screen and forced us to restrict maximum replay speed to 4 times normal replay.

Evaluation. In Hürst et al. 2007a we presented an evaluation of an initial implementation of the MobileZoomSlider on a PDA. Participants were asked to evaluate the design based on usability and intuition and had to solve two video browsing tasks in order to verify its usability. Generally, the results have been quite satisfying since users gave very positive feedback and provided useful information for further improvements. As expected, individual users preferred different interaction modes to solve the two tasks, although we could observe a strong preference of position- over speed-based navigation. One reason for this might have been the limitations described above, which is why we re-implemented the internal processing of the player software for the evaluation presented in Section 4.

Revised design used in the evaluation (Sect. 4). Compared to the interface described above we did not make any change on the overall layout and basic design. However, as said before, we re-implemented the internal processing in order to allow faster speedups, avoid time delays when switching between position- and speed-based navigation, and enable backwards scrolling at the left screen border. Figure 4 illustrates the final implementation of the actual interface on a PDA (cf. Section 3). It should be noted that on such a device, speed-manipulation can be done quite easily and comfortable since users just have to move the pen along the salient border of the PDA's body (cf. Fig. 5).

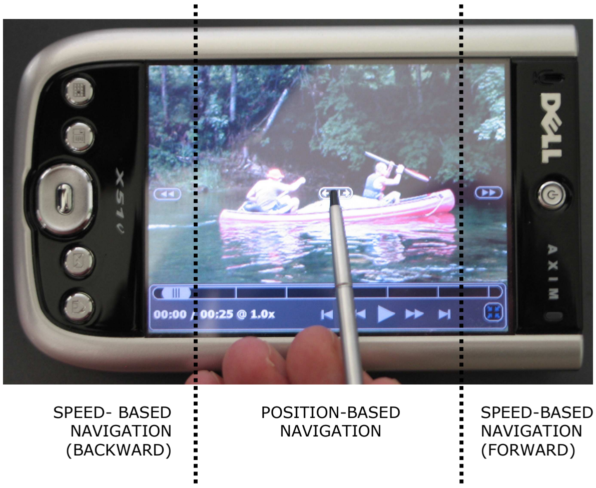

Figure 4. MobileZoomSlider: Implementation on a PDA.

Figure 5. MobileZoomSlider: Modification of replay speed by moving the pen along the salient border of the PDA's body.

2.2 The ScrollWheel Design

In the following we introduce a new interface design which, similarly to the MobileZoomSlider, gets by without any additional widgets by using the whole screen as interaction area.

Basic idea. Position and speed values are manipulated in the MobileZoomSlider by horizontal and vertical pen movements, respectively. In contrast to this, the *ScrollWheel design* uses circular movements for the manipulation of values. For example, similar to a clock, positions of a video can be mapped onto a "wheel" and navigation along the timeline is done by making circle-like movements on the screen. The association between a continuous, time-dependent media type such as video and a clock-shaped interface seems quite natural. In addition, hardware-versions of circular interfaces are common and well-known. For example, the so called *Click Wheel* from the *Apple iPod* (cf. Apple iPod 2007) is not only used for volume control but also to scroll through longer lists of text such as album or song titles. Special, wheel-shaped hardware is often used in video editing to enable better access to single frames. And some VCRs have wheel-shaped controls for manipulation of a tape's replay speed. A software version, as it is realized here, has an additional advantage compared to a hardware installation: By modifying the radius of the circle-shaped movements, users can implicitly manipulate the scale of the timeline along which they are navigating (cf. Fig. 6). The resulting effect is quite similar to the position-based interaction in the MobileZoomSlider: Moving the pen further away from the center of the circle and the original slider, respectively, results in a finer scale. One major advantage of the circle is though, that the resulting "virtual timeline" is not limited by the screen borders but could be infinite. However, for continuous navigation along a very fine timeline scale, a repositioning of the pen is required here as well due to the size of the screen (cf. Fig. 7).

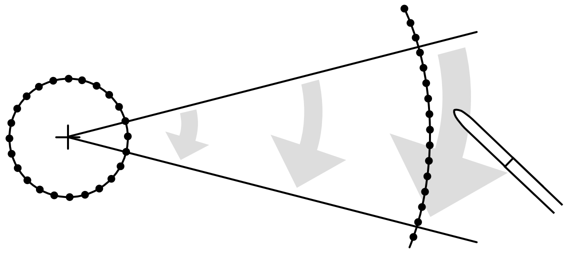

Figure 6. Implicit scale manipulation with circular interfaces.

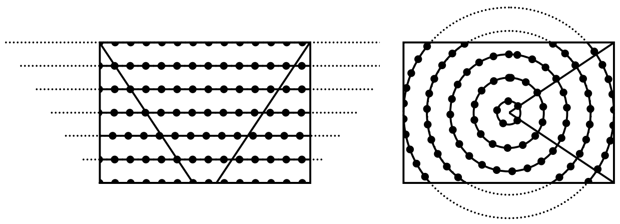

Figure 7. Granularity modification with the MobileZoomSlider and ScrollWheel interface, respectively.

Alternative wheel versions. Different options exist when realizing such wheel-shaped interfaces for video browsing. In the following, we classify them into three categories.

1. Pure position-based approaches (cf. Fig. 8). The timeline of a video is mapped onto a circle as described above. The center of the wheel is in the middle of the display. The initial clicking position on the screen is associated with the current position in the video. Depending on the length of the file, either the whole video or a fixed period of time (e.g. one minute) can be mapped to a full rotation of the circle. In our implementation, we have chosen the latter version in order to be independent from the length of a video. Further options exist; for example, the scale of the timeline can be adapted dynamically based on the speed at which a user is turning the virtual wheel or the overall interaction time. Such an approach is used, for example, for the Apple iPod's Click Wheel if a users is scrolling in long text lists. In our initial implementation, we decided to keep the scale at a fixed value. Experiments with dynamically changing resolutions are part of our agenda for future work (cf. Section 5). The most obvious advantage of such a pure position-based realization seems to be that the wheel has "no end", i.e. we can map any video of any length onto this interface and still provide the same scale resolution. However, in practice, this advantage also has a negative side effect: If you want to skip a larger part of the file, for example in order to go from the beginning to its last quarter, users might need to do a lot of circles with this kind of interface, even if very small ones are made (i.e., a relatively coarse scale is used). Faster navigation for skipping longer parts of the file can normally be done much easier with linear sliders and speed-based navigation.

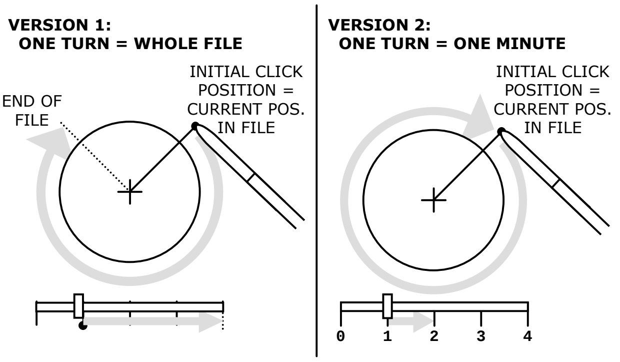

Figure 8. ScrollWheel: Purely position-based version.

2. Pure speed-based approaches (cf. Fig. 9). As said before, some VCRs use wheel-shaped controls for manipulation of replay speed. A software realization of such a control is illustrated in Figure 9. Again, various options exist for such a design. For example, if replay speed is restricted by a maximum and minimum value, the possible speed ranges can be mapped to a full circle turn with different parts for forward and backward scrolling. If no maximum value exists or the speed range is rather large, the implementation can be done in a way that keeps increasing replay speed if the wheel is turned continuously.

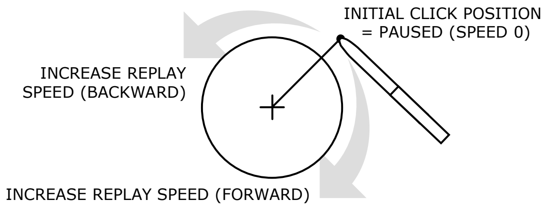

Figure 9. ScrollWheel: Purely speed-based version.

3. Combinations of both approaches (cf. Fig. 10). In order to combine the advantages of position- and speed-based navigation into one interface, we can also integrate both interaction styles into one design. One possibility would be to distinguish between speed- and position-based navigation depending on the distance of the pen from the center. For example, if this distance is smaller than a threshold *d*, a speed-based navigation as illustrated in Figure 9 is used. If it exceeds the threshold *d*, pen movements are interpreted as position-based navigation similar to the one shown in Figure 8. Alternatively, speed-values can be mapped to a particular region of the circle whereas the rest of the wheel is reserved for position-based navigation (cf. Fig. 10). Once a user leaves the speed-area, scrolling can switch back to a position-based navigation. Alternatively, the speed-based navigation region can follow the pen and switching back to position-based navigation is done by releasing the pen and touching the screen again.

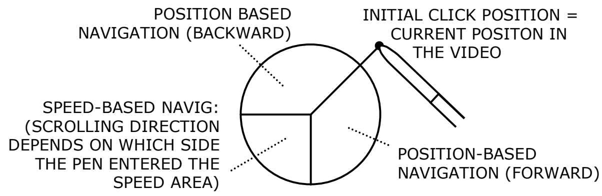

Figure 10. ScrollWheel: Combination of both interaction styles.

Implementation for the initial evaluation. Because we think that both interaction styles are equally important for video browsing, we decided to realize a combined version for our initial implementation of this interface concept. After experimenting with different alternatives on the desktop, we implemented a variant similar to the one illustrated in Figure 10 because it appeared to be the most promising. Speed is mapped to a small area on the opposite side of the initial clicking position and represents replay speeds from 0.5 to 4 times normal replay speed. Once a user enters the speed-area, scrolling switches from position- to speed based navigation. Again, there was a significant time delay here due to initial problems with our implementation (cf. Section 2.1). Replay speed increases if the user keeps turning the wheel. Once the end of the speed area is reached, it follows the pen movements along the wheel. Switching back to position-based navigation is done by releasing the pen and touching the screen again. Speed-based backwards navigation was not implemented in the initial version. If a user reached the speed area from the opposite side (i.e. while scrolling backwards along the timeline in position-based navigation mode), the position of the speed area was adapted, i.e. moved with the wheel in order to guarantee a pure position-based backwards navigation. Mapping of frames from the file to positions on the wheel was done in a way that one full turn of the circle represented one minute in the file. The functionality was visualized via a small, transparent icon which appeared in the middle of the screen (which was also considered as the center of the ScrollWheel).

Initial evaluation (heuristic study). In order to get a first idea of the usefulness and feasibility of the proposed approach, we performed an initial user study in form of a heuristic evaluation (Nielsen and Landauer, 1993). As proposed by Nielsen (2000), we asked five user interface experts (3 males, 2 females, ages from the early to late 20s) to participate in the study. They were asked to evaluate the interface design based on a given set of questions related to functionality, intuition of the involved interactions, and possible problems and drawbacks. In addition, they should comment on the visualization and the particular parameter settings (e.g. accessible speed-ranges, scale-resolution for position-based navigation, etc.).

General feedback was very positive. The participants rated the interface design as intuitive, and easy to understand and operate. For the evaluation, we used a common news show recording with a total length of 15 minutes. Four participants commented that the mapping of one minute to a full circle turn was an appropriate one, whereas one user noted that it might be too coarse if you want to search in scenes with many cuts. Interestingly and a little surprising for us, three users did not figure out by themselves that they are able to implicitly modify the scale by moving the pen away from the wheel's center and hence, were very skeptical if the average user would be able to take advantage of this characteristic of the interface.

Some participants had reserved feelings for the speed-based navigation. Although all of them considered this type of interaction as important, some noted that the current implementation might be a little confusing. These concerns are best described by the comment of one user who said that the "*circle-shape animates you to keep turning*". However, in our implementation, users are actually expected to keep the pen still at the position representing the desired speed value once they are in the speed area. As a consequence, users made lots of alternative comments on how to better integrate speed-based navigation which influenced the final interface design we present below. Two participants also explicitly noticed that it might be confusing for the average user to combine both interaction types into a single interface and that it might be better to keep them strictly separated. As said before, there was a small, but noticeable delay before speed-based navigation started when entering the speed area. This was also considered to be a significant usability problem by most users.

Because of its small size and transparent appearance, all participants considered the visualization as not disturbing although it appeared directly in the middle of the screen. Three users however criticized the visualization of the speed area as not very intuitive and some noted that it might be a little irritating that it appears at different positions of the wheel depending on the initial clicking position of the user.

Revised design used in the final evaluation. The main positive result from the previous evaluation is that the participants approved the overall design of a circle-shaped interface for video browsing on small, pen-based devices. On the negative side, combining speed- and position-based navigation into one interface seems to be critical and might not be useful in practice. Hence, we present a re-design which strictly separates these two kinds of interactions. In addition, we completely re-implemented the internal frame processing because the time delay when switching to speed-based navigation and the lack of speed-based backwards skimming turned out to be more critical in practice than we initially expected.

In the new interface design (cf. Fig. 11), the circle is only used for pure position-based navigation. Once a user clicks on the screen, an icon appears in the center of the display. If no interaction follows this initial click for five seconds, the icon disappears again. Making clockwise or counterclockwise movements around the center results in a position-based navigation along the timeline (forwards and backwards, respectively). The scale resolution of this round, "virtual timeline" is similar to the previous implementation, i.e. one minute from the video is mapped to a full turn of the circle. In addition to making circular pen movements, users can grab the icon and modify replay speed by moving the pen horizontally: Moving the pen to the right increases replay speed. The speed value is proportional to the distance between the pen and the icon, i.e. replay increases the more the pen moves away from the center of the screen. Backward scrolling is realized in a similar way if the pen is moved on the left side of the icon. Once the pen is released, the player switches back to its previous state (i.e. normal replay or pause mode). With this implementation, both navigation styles are strictly separated by different interaction modes (circular vs. horizontal pen movements for position- and speed-based navigation, respectively). Yet, both functionalities are locally close to each other so users can easily switch between the interaction modes. Figure 12 shows the actual implementation on the PDA used for the evaluation presented in Section 4. In this realization, the icon followed the horizontal position of the pen after the user was grabbing it, and the distance to the center was illustrated with a red line.

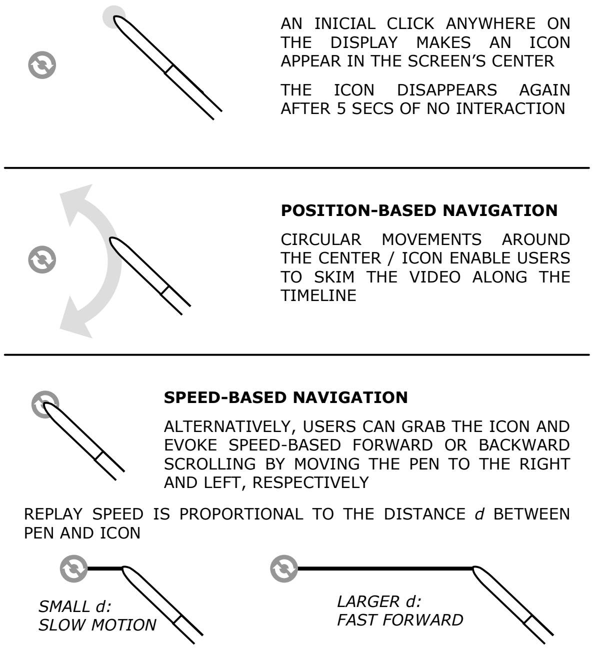

Figure 11. ScrollWheel: Illustration of the final design.

3. IMPLEMENTATION DETAILS

The two proposed interface designs can only be realized if the system features real-time visual feedback, i.e. the respective content of the video is displayed instantly while a user is navigating along the timeline or skimming the file at different replay speeds. Hence, performance is one of the major requirements for the actual implementation of these concepts.

Therefore, we ran a couple of performance evaluations with several player toolkits for mobile video. Available players included Microsoft's Media Player, the VideoLAN VLC Media Player, Kinoma Player 4, as well as The Core Pocket Media Player (TCPMP 2007). We tested several of them, mostly on a PDA using a 15 minutes long video with a resolution of 640x480. The file size of such a video is approximately 300 MB when using MPEG-1 and MP3 for video and audio compression, respectively, and a frame rate of 25 frames per second. Due to its file size, the video was stored on an external storage card of the PDA. MPEG-1 encoding was done with the ffmpeg encoder program.

Figure 12. ScrollWheel: implementation on a PDA. (Top: position-based navigation via circular pen-movements around the screen's center. Bottom: speed-based navigation via horizontal pen-movements after grabbing the icon.)

Based on these tests, we decided to use TCPMP for our implementation because it delivered a very satisfying performance. In addition, TCPMP is an open source project, thus giving us full control over the code and all necessary access and APIs we needed to implement our interfaces on top of the actual player software.

TCPMP is written in plain C and provides many codec libraries. Hence, most audio and video formats can be replayed. It can be compiled for several platforms including multimedia cellphones and Pocket PCs featuring the Windows Mobile platform as well PDAs with Palm OS. So far, we tested our implementation on a Palm Treo smartphone (cf. Fig. 13) and a Dell Axim™ X51v PDA (cf. Fig. 4 and Fig. 12). Both systems are running the Windows Mobile platform. The evaluations presented before and in the next section have been done with the Dell PDA which features an Intel XScale PXA 270 processor with a clock frequency of 624 MHz and 64 Mbytes of SDRAM. The diagonal size of its display is 3.7'' with a resolution of 640x480 pixels and a color depth of 16 bits.

In order to create a new front end for the player, we implemented our own skinnable user interface in C++ which provides standard components such as labels, buttons and sliders. This class library is based on the Win32 API using GDI (Graphics Device Interface) for rendering the graphics. Hence, the source code remains portable among different versions of Windows including mobile devices running Windows Mobile. In addition, connecting our newly implemented interfaces with TCPMP was rather easy because they are both based on the same basic APIs.

Figure 13. Implementation on the Palm Treo smartphone (running Windows Mobile).

4. COMPARATIVE EVALUATION

Many advanced browsing and search tasks can not be solved with standard interfaces such as the one depicted in Figure 1, for example because of the scaling problem discussed in the introduction. In contrast, the two designs described in Section 2 are more powerful and offer a much larger range of functionality than regular interfaces. The critical question is, if this advanced functionality is intuitive and can be handled by the users in an easy, manageable way. Hence, instead of comparing the new design with a common, traditional interface, we decided to set up a comparative user study of the MobileZoomSlider with the ScrollWheel design in order to evaluate their usability and usefulness and identify potential problems and drawbacks.

4.1 Setup and Procedure

Setup. We asked 16 users to participate in our study (9 males at ages from 20 to 35, 7 females at ages from 15 to 31, average age of all users: 26.25 years). Only few of them had any experience with pen-based handheld devices or video browsing. Significant differences between experienced and inexperience users as well as between male and female ones could not be observed. As device, we used the Dell Axim PDA described in the last section. Two interfaces were used: The MobileZoomSlider and the ScrollWheel design described at the end of Subsection 2.1 and 2.2, respectively, and depicted in Figure 4 and 12. One general problem with these designs is that the current position in the file is not clearly indicated. Hence, we believe that it is important to represent the timeline as well. Therefore, the normal interface (cf. Fig. 1) also appeared on the screen during the evaluation. However, users were asked not to use it but solely rely on the newly introduced features for this study.

Tasks. For the evaluation, we took two recordings of a TV news show (one for each interface) and set up three scenarios and related search and browsing tasks that had to be solved by the users with each interface design:

1st task (overview): Skim the content of this news show from the beginning to its end. The goal is to get an overview of the different news messages covered in this show.

Scenario: Assume, for example, the following situation: You are waiting at the bus stop and want to get a quick overview of the different news messages in order to watch the most interesting ones while riding the bus. The bus ride is not long enough to watch all of them, so you have to do an individual pre-selection.

2nd task (scene search): You are at the end of the file now. Try to go back and find the news message about XY, which is located in the first third of this news show.

Scenario: Assume you are on the bus now and this is the message you want to watch first because it is the most interesting one. (Note: XY was the second message in each show and a still image of its initial scene was presented to the participants during the test. Users were not required to position the file exactly at the beginning of this scene but to roughly find a position from where they would begin watching in such a situation.)

3rd task (exact positioning): From the position that you found in task 2, go as quickly as possible to the weather forecast at the end of the show and find any frame showing a map with the temperatures of the next day.

Scenario: You are close to your destination and before you leave the bus you want to have a quick look at the weather forecast. Since you do not have much time left, you just want to have a look at the expected temperatures. (Note: A still image of the respective map was shown to the users during the test.)

Data. The files used in the evaluation were two different recordings of the most popular evening news show in Germany. Each file had a length of 15:40 and 15:45 minutes, respectively. The scene described in the second task began at 2:37 minutes in the first file and at 2:47 minutes in the second on. The scenes' lengths were 2:04 and 2:11 minutes, respectively. Frames featuring the temperature map requested in the third task were shown for 4 seconds (= 100 frames) starting at 14:57 minutes in the first file and for 8 seconds (= 200 frames) starting at 14:58 minutes in the second one. It should be noted that it would have been very hard to solve this last task with a regular timeline slider interface (cf. Fig. 1) due to the short duration of these scenes.

Procedure. Since there were two designs and two different files, we split the participants into four equally sized groups where each group tested the interfaces in a different order and interface-file association. At the beginning of the evaluation, users were given a short introduction into the first interface and its characteristics. Then the device was handed to them to try it out themselves for two minutes. After this time, they had to solve the tasks for this interface in the given order. Following that, they were asked to give ratings about their overall impression, the interface's intuitiveness and the offered functionality. The last two questions were split into two separate ratings – one for the speed- and one for the position-based navigation. Then the same procedure was repeated with the second interface. When making the ratings for this second design, users were allowed to look at and modify their former ratings for the first interface in order to guarantee that the ratings given for both designs are comparable. The evaluation closed with an informal interview where users were asked some questions related to general issues, particular observations during the tests, etc. The overall duration was about 40 minutes per participant.

During the study, another person sat next to the participant and took notes about what the user was doing, interesting observations, comments given during the tests, etc. Originally, we wanted to implement a logging mechanism to be able to re-track all interactions. However, since this would have resulted in performance problems during video browsing, we gave up this idea. For the same reason, the time to solve the second and third task was not logged but taken with a stopwatch by the interviewer.

4.2 Evaluation Results

Both tested interface designs provide a similar functionality. In both cases, position-based navigation is offered at various granularity levels (cf. Fig. 7). Similarly, different speed levels can be picked in both interfaces by moving the pen between the center of the screen and its left and right side or along the left and right border, respectively. In case of the MobileZoomSlider, single speed values can be addressed directly whereas in case of the ScrollWheel users always have to "speed up" in order to get to any particular value. However, since our implementation reacts instantly to any speed change, we did not expect this to be very critical in practice. The similar functionalities provided by both systems are evoked by different user interactions: Horizontal and vertical pen movements for position- and speed-based navigation, in case of the MobileZoomSlider vs. circular and horizontal pen movements for position- and speed-based navigation, in case of the ScrollWheel design.

Task 1. Since the first task was the easiest one, we expected that most users would use just one interaction mode (i.e. either speed- or position-based navigation). The two entries labeled with "TASK 1" in the two diagrams in Figure 14 show how many participants used each interaction with the respective interfaces. It was interesting for us to observe that with the ScrollWheel, most users preferred position- over speed-based navigation. In contrast to this, with the MobileZoomSlider an almost identical amount of participants used one of the two options. One possible reason is that people might have figured the implementation in the latter case to be more intuitive. A further issue might be the fact that speed modification is done at the border of the player window, thus being the interaction mode which interferes the least with the actual video content. However, this characteristic apparently seemed to have limited impact: Only two users noted that you might be blocking your view when doing large circles with the ScrollWheel, but no one mentioned a similar negative concern in relation to the implementation of the ScrollWheel's speed-based manipulation feature. Another interesting observation is that with the MobileZoomSlider, five people used both modes to solve this task, whereas in case of the ScrollWheel, only one person took advantage of both navigation styles. A possible reason might be that switching between horizontal and vertical pen movements was considered easier than between circular and horizontal ones as required by the ScrollWheel. In addition, with the ScrollWheel, both interactions are centered around the middle of the screen. We thought that this local closeness would actually motivate people to switch more often between different interaction modes. However, it might also be the case that having two different interaction modes so close to each other confused the users and actually prevented some participants from using them interchangeably. Since the task here was just to "get an overview of the content", success was hard to quantify, which is why we did not measure the time to solve this task.

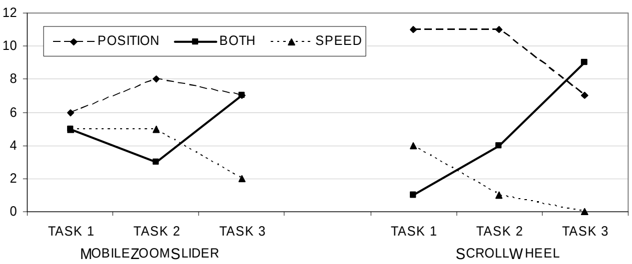

Figure 14. Interaction modes used for the different tasks.

Task 2. When looking at the used interaction modes (Fig. 14, entries labeled "TASK 2"), again, we can observe a clear tendency towards position-based navigation in case of the ScrollWheel. However, more people used both interaction modes here, which is not much surprising given the higher complexity of the task. For position-based navigation with the ScrollWheel, we could clearly observe in five cases that the participants were making smaller circles at the beginning (in order to quickly get to the scene) and larger ones at the end (in order to find the beginning of this particular scene). This is particularly important given the skeptical remarks of the experts in the initial heuristic evaluation presented in Section 2.2. The average time to solve this task with the ScrollWheel was 19.4 sec (max. 46 sec, min. 9 sec, standard deviation 9.8) and 25.3 sec for the MobileZoomSlider (max. 45 sec, min. 12 sec, standard deviation 11.3). Despite the relatively large difference of the average search times, this result did not turn out to be significant (t-test, $p = 0.13$). An interesting observation is that users could recover easily from simple mistakes such as accidentally skipping the scene or going too far backwards, etc. In general, they did not consider such things as erroneous input but as natural part of their search process.

Task 3. This was the most complex task resulting in higher average search times of 33.0 and 34.7 sec for the ScrollWheel and the MobileZoomSlider, respectively. Again, this result did not turn out to be significant (t-test, $p = 0.8$). In four cases we could observe that people used smaller circles for faster navigation and larger circles for fine tuning in case of the ScrollWheel. With the MobileZoomSlider, it was clearly observable that people also took advantage of the different granularity levels offered by the system by modifying the vertical position of their pen movements. For example, for the people who purely used position-based interaction, one frequent interaction pattern was to first use a coarse scale to get to the weather forecast and then continue navigating in the upper part of the screen in order to find the temperature map with a finer scale. Considering the interaction type (cf. Fig. 14, "TASK 3"), it is noteworthy that more participants have used both modes to solve this task. This might be due to its high complexity requiring people to quickly skim the file (in order to get to the scene at the end) and then do a detailed search and exact positioning to one of the required frames. Generally, we would have expected people to use speed-based navigation first and then switch to position-based navigation for fine tuning and exact positioning. However, a clear tendency in the order of used interaction modes could not be observed in any of the three tasks.

If we compare all usages presented in Figure 14, we can observe a general trend that with increasing complexity of the task, pure speed-based navigation is used less often and more people tend to use a mixture of interaction modes. We interpret the fact that more participants took advantages of the various functionalities offered by the user interface as a good argument for the intuitiveness and good usability of the proposed design and its integration of different interaction modes.

Users' ratings. After solving the tasks, users were asked to give ratings for the interface on a scale from "-2" (worst) untill "2" (best) for different categories. When asked about their general impression, there was a significant difference (t-test, $p < 0.1$) in the average ratings for the MobileZoomSlider (avg. 0.9) and the ScrollWheel (avg. 1.4). However, both interfaces are rated rather high, supporting the claim that users were able to handle both of them equally well. The detailed results are shown in Table 1 which illustrates the number of users who picked the particular rating. Looking at the individual users, nine people gave a better rating for the ScrollWheel (one user "1" vs. "-1", one user "1" vs. "0", seven users "2" vs. "1"), two rated the MobileZoomSlider higher (both "1" vs. "0"), and five users gave equal ratings (three users "1", two users "2"). The participant who rated the MobileZoomSlider with "-1" claimed that he found "*turning to be more intuitive*" than the position-based scrolling in the MobileZoomSlider. One of the two users who gave the lowest ratings ("0") for the ScrollWheel argued in the opposite way, i.e. that she did not like the circular input mode required here. The second user actually preferred this interaction type, but motivated her low rating with the different implementation of the speed-based navigation.

Table 1. Ratings of the overall impression (no. of users).

Overall impression	(significant, p<0.1)				
Ratings	-2	-1	0	1	2
MobileZoomSlider	0	1	1	12	2
ScrollWheel	0	0	2	5	9

This illustrates, that it is also important to look at the individual features, which is why we asked the participants to rate the speed- and position-based navigation mode of each interface based on their intuitiveness (from "-2" = not intuitive to "2" = very intuitive) as well as offered functionality (from "-2" = insufficient to "2" = completely sufficient). Whereas the results for the speed-based navigation turned out not to be significant (t-test, $p > 0.1$), there was a significant difference in the ratings for the position-based approaches. The results are presented in Table 2 and 3. For speed-based navigation, the few low ratings for the intuitiveness ("-1" for the MobileZoomSlider and "-2" for the ScrollWheel) and the functionality ("-1" of the ScrollWheel) were motivated by the general dislike of this kind of interaction by the respective users rather than being directly related to the actual implementation. The same is true for the two negative ratings ("-1") for the intuitiveness of position-based navigation with the MobileZoomSlider, i.e. these two users generally prefer speed-over position-based navigation. Even if we ignore these users with individual preferences, the ratings for position-based navigation with the ScrollWheel tend to be generally higher. For the intuitiveness, we somehow expected this result, because of the association between a clock and the "round timeline" defined by the ScrollWheel. However, we were surprised by the ratings for the functionality, since both interfaces are quite comparable to some degree (cf. Fig. 7). Possible reasons for this are that users either found the involved interaction mode easier to handle or made subjective judgments which have been influenced by their assessment of the intuitiveness.

Table 2. Ratings for intuitiveness (no. of users).

Intuitiveness:	**Position-based navig.** (significant, p<0.1)				
Ratings	-2	-1	0	1	2
MobileZoomSlider	0	2	5	5	4
ScrollWheel	0	0	1	6	9

Intuitiveness:	**Speed-based navig.** (p > 0.1)				
Ratings	-2	-1	0	1	2
MobileZoomSlider	0	1	2	6	7
ScrollWheel	1	0	3	7	5

Table 3. Ratings for functionality (no. of users).

Functionality:	**Position-based navig.** (significant, p<0.1)				
Ratings	-2	-1	0	1	2
MobileZoomSlider	0	1	6	7	2
ScrollWheel	0	0	1	4	11

Functionality:	**Speed-based navig.** (p > 0.1)				
Ratings	-2	-1	0	1	2
MobileZoomSlider	0	0	2	9	5
ScrollWheel	0	1	3	4	8

General remarks. Overall, feedback was very positive for both interfaces. The observations reflected in the data presented above were generally confirmed by the additional comments made by the users. All extremely negative remarks were motivated by general dislikes of certain interaction modes (i.e. speed- or position-based navigation) but not by the interface. Preferences for particular implementations existed but in general both interface versions received a positive feedback. However, users also made some critical comments which gave hints for future developments. One user noted that it would be very useful to have audio feedback during scrolling. Another one said that the interfaces lack the possibility to jump directly to a particular position on the timeline. We consider this as a critical issue as well and therefore argue that a final implementation should feature both: one of the new interfaces discussed here as well as the original slider (cf. Fig. 1) for direct access of the timeline. Another user noted that she liked it that she could just move the pen along the displays borders to modify speed in the MobileZoomSlider (cf. Fig. 5).

4.3 Discussion and Conclusion

The most important result is that both designs could be evaluated successfully in terms of usability and user acceptance. Although they had just two minutes to make themselves familiar with the interfaces, all participants had no problems in handling them, were able to solve the tasks in a reasonable amount of time, and generally gave very positive feedback. Comparing both designs with each other, there is a small trend towards the ScrollWheel for position based navigation and the "speed border" of the MobileZoomSlider for speed-based scrolling. Two users actually proposed such a combination after the evaluation. However, the difference between the varying interaction types is rather small and there are inconsistent, conflicting user statements. For example, one user said that she thinks it is hard to pick an appropriate scale with the MobileZoomSlider whereas doing it with the ScrollWheel comes in very natural (*With the ScrollWheel "I don't have to think about granularity first" but just start turning*). In contrast to this, another one mentioned that "turning" is rather unfamiliar for her and therefore *"it's hard to estimate how fast I have to turn"*. Obviously, there is no optimum solution here, because of the significant influence of individual preferences. Such personal biases are also an important argument why common interfaces should support different interaction styles: In the evaluation, there were several users who generally preferred one interaction mode over the other. In addition, a large amount of users liked to switch between different types of interaction, especially when working on a more complicated task.

5. SUMMARY AND FUTURE WORK

We presented two interface designs for video browsing on pen-based, mobile devices and evaluated them in a comparative user study. Although no "best" approach could be identified, the study gave valuable information, for example, about the usefulness of different interaction styles. The most important observation is that both interfaces can be handled quite well and are generally well received by the users. Despite the small screen size, they offer much more browsing functionalities than common mobile video player interfaces while at the same time still being intuitive and easy to handle. In addition, they nicely integrate into the overall interface design of the player software. Users who feel less attracted to any particular interaction mode can just ignore it and are not limited or constricted in their normal replay or browsing behavior in any way.

Based on the current data, we would recommend a circular interface for position-based navigation together with the "speed border" featured in the MobileZoomSlider as final implementation. However, further evaluations are needed. One important aspect is that so far, all of our evaluations have been done in a laboratory setting where users were sitting on a desk while testing the systems. Evaluations in a mobile scenario as proposed by Kjeldskov and Stage (2004) are part of our agenda for future research. As said before, dynamic modification of the mapping between timeline and ScrollWheel is a promising feature we want to investigate in the nearer future as well. In addition, we are planning to add audio feedback to video browsing using some pitch-preserving signal processing approach and evaluate its relevance for mobile video browsing. Finally, we want to investigate how automatic scene segmentation and the resulting structure information can be incorporated into our user interface in order to improve the overall browsing performance.

6. REFERENCES

Apple iPod (2007) http://www.apple.com/ipod/ipod.html

Casares, J., Myers, B.A., Long, C.A., Bhatnagar, R., Stevens, S.M., Dabbish, L., Yocum, D., Corbett, A. (2002) Simplifying Video Editing Using Metadata. *Proceedings of Designing Interactive Systems* (DIS 2002), London, UK.

Hürst, W., Götz, G., Welte, M. (2007a) Interactive Video Browsing on Mobile Devices. *Proceedings of the 15th International Conference on Multimedia* (ACM MM 2007), Augsburg, Germany, 247-256.

Hürst, W., Götz, G., Welte, M. (2007b) A New Interface for Video Browsing on PDAs. *Proceedings of the 9th International Conference on Human Computer Interaction with Mobile Devices and Services* (MobileHCI 2007), Singapore.

Kjeldskov, J. and Stage, J. (2004) New Techniques for Usability Evaluation of Mobile Systems. *International Journal of Human-Computer Studies* 60, 599-620.

Nielsen, J., Landauer, T.K. (1993) A mathematical model of the finding of usability problems. *Proceedings of the Conference on Human Factors in Computing Systems* (INTERCHI'93), Amsterdam, The Netherlands.

Nielsen, J. (2000) *Jakob Nielsen's Alertbox*, March 19, 2000: Why You Only Need to Test With 5 Users, http://www.useit.com/alertbox/20000319.html

O'Hara, K., Mitchell, A.S., Vorbau, A. (2007) Consuming video on mobile devices. *Proceedings of the SIGCHI conference on Human factors in computing systems*, (CHI 2007), San Jose, CA, USA, 857-866

Richter, H., Brotherton, J., Abowd, G.D., Truong, K. (1999) A Multi-Scale Timeline Slider for Stream Visualization and Control. *GVU Technical Report GIT-GVU-99-20*, Georgia Institute of Technology, Georgia, USA.

TCPMP (2007) *The Core Pocket Media Player*: see http://tcpmp.corecodec.org