Interface Technologies

Report on Interactive Pen + Touchscreen Project

Jenna Lindh, Sami Lunnamo, Joe Niemi, Janne Timonen

Abstract

This report explains how we executed a group project, the topic of which was to create an electronic pen-like controller that could be used to perform various functions associated with watching and playing videos. One point of interest was to create more sophisticated and intuitive ways of executing these video controlling functions, than what has been done in most software video player GUIs, like VLC player or Windows Media Player. With this philosophy in mind, we had to consider things such as *gestures versus buttons* and how to make controls intuitive enough while simultaneously keeping the on-screen GUI as minimalistic as possible.

Introduction

The goal of this project was to create a prototype of an Arduino-based *pen* that could be used to control a video player. Different functions to be implemented featured a *play/pause* -function, *fast forwarding, rewinding, changing audio volume, frame-by-frame skipping* with *video speed adjusting* and some video library *browsing functions*. All these would be made to work with a multitouch screen. To achieve these functions we selected a bunch sensors and parts, which we connected to an Arduino board, which is an open source electronic prototyping platform. The movie player software itself is developed with Processing, which is a development language and IDE based on Java. The software running on Processing is used to detect and handle input data that is generated by the sensors attached to the Arduino board.

This Arduino prototype is to be used with a touchscreen that recognises the pen and is able to read what is *drawn* with it. This is in essence another abstraction layer in the project. On one hand, we have the sensor readings we get from the Arduino, which control the flow of the program to some extent, and on the other hand, we have events originating from the multitouch screen, that are essentially controlling signals for video playback and other media player functionalities.

In this paper we will first describe the background for this project and similar, previously done work that we encountered. Then we will discuss the design principles we agreed on and then go through the process of implementing our vision. Next, we will raise a few topics of discussion and possible points of future work we noted during the project. Finally we take a look back on what has been done during the whole project.

Background

Previously, some work has been done in which the pen has been outfitted with the ability to give haptic feedback [1]. The physical sensations that can be received by a haptic pen have been seen as an enhancement to the way we interact with digital interfaces, and especially helpful for those with some kind of disabilities. The paper introduces some ways of modeling different types of interactions with the interface, some different kinds of taps and clicks for example.

Another way the pen has been utilized as a digital interface [2]. The gist is that the pen is outfitted with an infrared laser pointer that is used to mark a canvas from a whiteboard, which will from there on reflect whatever is written on a piece of paper with the pen. Whereas in [1] the major point of the pen was to provide an enhanced user experience, [2] aims to innovate a new way of presentation, where a presenter can utilize the freedom of a paper by writing and perhaps drawing illustrations. In neither [1] or [2] have there been any attempts at mapping the movements of the pen to mean different functions, like in the project this team is about to undertake.

Areas and ideas related to our project work are taken into deeper consideration in [3] which examines interaction and controlling video with different kinds of *widgets* that exploit the use of various sensors and their capabilities. The *LEAN* system described in the article recognizes different gestures such as flicking or sweeping right/left, interval rotating and a set of varying tap-functions. Idea is that the VCR-like interface used in most video stream softwares is unnecessary and avoidable and thus the whole interface could be better designed to suit fluid navigation and control of modern digital video (e.g. by the means given by using a pen-tool to perform certain gestures). An informal user test was also carried out during the study described in [3] and it shows that the users were able to grasp these new features after 15 minutes of use.

One way to use an interactive pen is to use it in tabletop games, represented in [4] and [5]. Tabletop games combine both the real and virtual world. The use of physics engine in [4] allows realistic interactions with game components and the virtual environment. In both [4] and [5], user input is done with the use of digital pens. In [5], the pen movement tracking is done by an embedded infrared camera. The players of [4]'s game can draw, reposition or delete virtual dominos with anoto pens on a wireless pen-interface. The wireless pen-interface is a projection table with a special surface that allows simultaneously localizing multiple digital pens. We are also going to use the same type of touchboard in our video browsing project for tracking arduino pen's and fingers movement.

Video image can be interacted with in many different ways, some of which are rather unconventional. In [6] for example, about 30-second fragments of video image are arranged into a three-dimensional blocks by stacking individual frames on top of each other. This way

newer frames offset older ones both in time and distance. The idea of this technology is to enable users to thumb through different frames quickly and effortlessly. Another interaction method presented in [6] is the possibility to make a scrapbook out of video frames while watching the video. The saved frames could then be used, for example, as a notebook.

Smart pens can be utilized in various ways when dealing with video streams. In [7] a pen was used to scroll forwards and backwards on the video's timeline. The position of the pen affected the scale of the timeline, so that scrolling on the lower part of the screen used the same resolution as the original slider, whereas scrolling on the upper part of the screen used a much finer scale, resulting in a slower and more accurate browsing.

Another approach mentioned in [7] is to use circular motions to control, for instance, the current position on the timeline. The radius of the circle affects the scale of the action, similar to the positioning of the pen in the previous example. By using a circular motion to control the playback the user never has to take the pen off the screen while going backwards or forwards on the timeline. In addition, both approaches took notice of the pen's speed, so that a higher speed resulted in a faster action, such as playback speed.

Mobile video browsing in [8] is done with using accelerometer sensor and touch screen. In [8]'s mobile devices video browsing accelerometer operations like shake and tilt and touching the screen act as user's inputs. For example with shaking right, users can fast forward the video and after shaking left, the video goes back to normal playing, vice versa. Detecting whether a user is shaking or tilting is done by looking at the speed of gestures.

Design

The initial idea was to create an actual pen with different controllers placed around it. This draft was scrapped, though, when we realised that the Arduino board we used was too big to actually achieve any elegant-sized instrument. Instead we decided to do a crude Arduino breadboard prototype and focus on other features. Figure 1 shows our first ideas about the implementable functionalities and a raw sketch of what we thought the pen would look like.

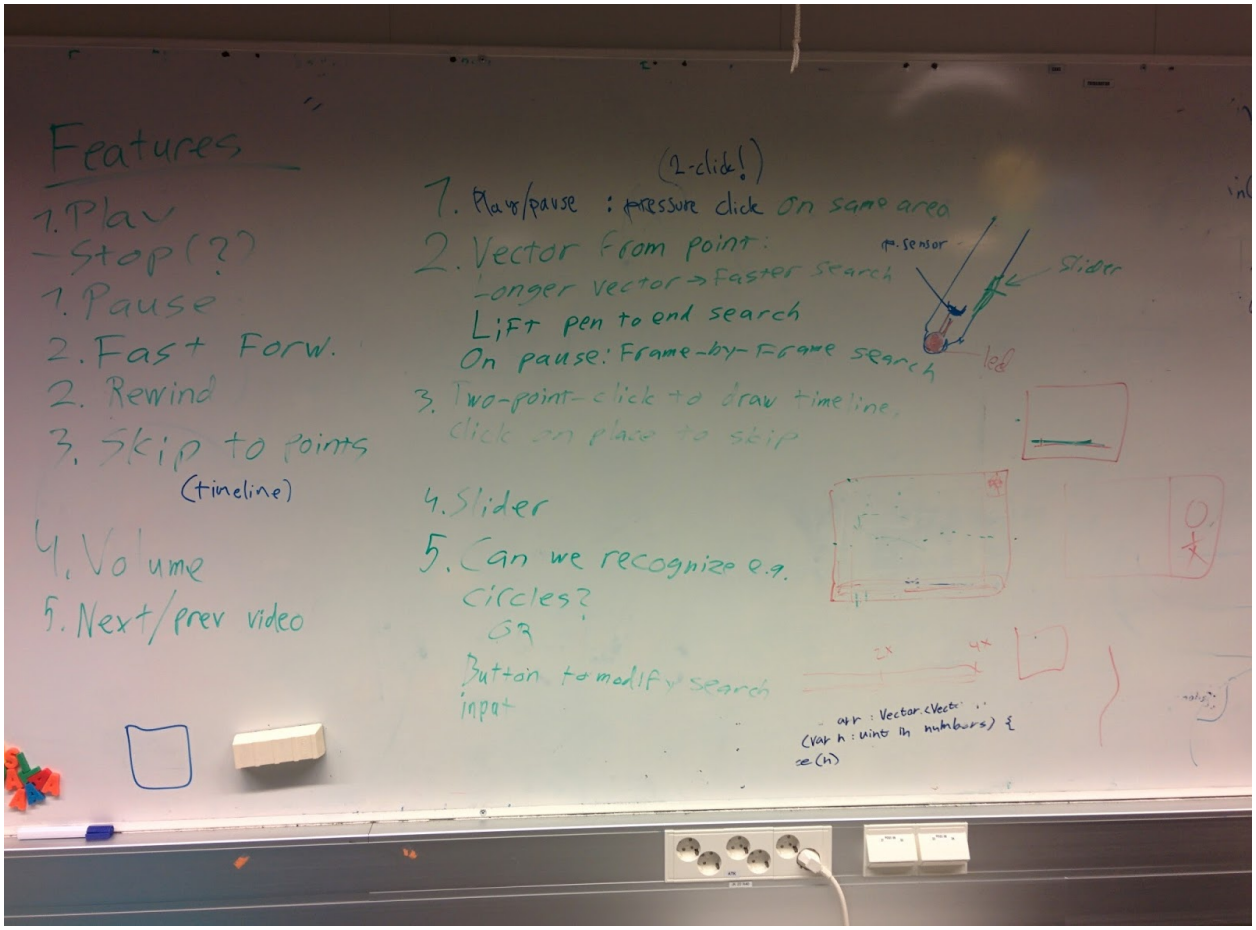

Figure 1. The first steps of the pen project.

Our main design focus was to try and implement as many functionalities we could with as few components as possible, and our tutors agreed on this. The features we wanted to implement were:

1. Playing and pausing the video
2. Stopping the video
3. Fast forwarding and rewinding
4. Skipping to specific points on the video
5. Adjusting the volume while playing
6. Skipping to next/previous video while playing another

It was clear to us from the beginning, in part due to the tutoring we received from our tutors, that the pen generates events on the multitouch screen through an Infrared LED, so it was naturally placed at the tip of our pen design. A pressure sensor was placed behind it so that it would be activated only when sufficient pressure was introduced, ie. the tip was pressed against the multitouch screen. Keeping in mind our goal of simplicity, we decided to distribute a lot of the features behind different kinds of gestures. They are all listed in figure 1, for

example fast fowarding was designed to be enabled when the IR led was being slid over the multitouch screen. We decided that pausing and playing should be keyed under double-clicking, since we all agreed that sometimes people tend to accidentally touch the screen once, without any intent of actually pausing the video. For volume controlling we agreed to add a potentiometer in the form of a slider, which could easily be used to control the volume.

The two features that gave us the most trouble were how to handle jumping to a next or previous video and, within one video, how to jump to a specific point in the video. For jumping to a specific point, we visioned that by clicking on the screen in two different points, we could draw a “timeline” on the screen, representing the full length of the video. The user could then jump to any point in the video by clicking the equivalent point on the timeline.

We decided to withhold the planning of the the main video library feature, the skipping between videos, until we could start actually working with our hardware. Our initial thoughts were about identifying a more complex gesture of the pen, like the user drawing a circle, which we could interpret as skipping to the next or previous based on perhaps whether the circle was drawn clockwise or counter-clockwise. This seemed a little complex, so as a backup we agreed that we could introduce a button to modify the input generated, kind of like a “video library functions” button.

Taking a step back at this point, we were wondering if the design was perhaps too simple, or did not utilize different sensors sufficiently. The number of different hardware-inputs seemed really low, but realizing that the main input-type was going to be through different free gestures through the multitouch screen, and that we actually had two different interfacing points, one through the Arduino and the other through the screen, we were confident that this was going to be enough.

Development

We set out to create our video pen prototype by implementing the functions we had come up with with different sensors attached to Arduino-board [10]. Infrared emitter’s (IRE) use is for the multi-touchscreen to recognise whether the pen is in use or not (i.e. to differentiate the pen from, for instance, fingers). For the play/pause -function we decided to use pressure sensor to detect if the pen is pressed against the screen. Fast forwarding and rewinding are executed by drawing *vectors* of a certain length on the screen to the direction wanted (as we are accustomed to, vector to the left means rewinding and to the right means fast forwarding).

Some amount of time was spent on thinking how to implement features and what to include. One new aspect was that we had to actually build the pen and not just fiddle on the software level. This, of course, was a constraint and we had to for instance take into account how to utilize pressure sensor so that it really works in practice (e.g. if pressure sensor is ON then

the IR-led will turn on also, otherwise nothing would happen). Some sensors like accelerometer and gyroscope were also left behind for we didn't find any use to them or using them would have demanded more time, skill and resources than we would be able to muster.

Software development proceeded in stages. We started with Arduino IDE and made ourselves comfortable with Arduino and its basic features. This stage included a lot of mock-ups and different setups with different sensors and components just to familiarize ourselves with the functions of the Arduino and what kinds of things we could achieve with the technology. At the same time, this served as a sanity-check for our initial designs, since at this point we could easily find out things that were more efficient at achieving some of our planned features than some other ideas we had had.

This stage also saw our first problems, since none of us were that familiar with electric wiring, so we had to proceed slowly in fear of breaking some of our components, or perhaps even the Arduino itself. Additionally, we found out that we could not use the Department's computers for any work, since we could not procure sufficient rights to properly install the Arduino IDE and later would not have had enough rights for the upper abstraction IDE we had to use to interface the touch screen, the Arduino and the video library functions. The solution was to bring in our own laptops, which meant that we had 2 Linux-based systems, 1 Windows and 1 Mac OS X as development machines on an open source project, which led to some problems with stability with the IDE softwares and runtime executions.

The initial plan was to use OpenFrameworks to achieve the wanted features, but once we got acquainted with the multitouch screen we changed our plans and eventually ended up using Processing language with TUIO framework [9] that handles touch events on the screen. The reasoning behind this was two-fold; firstly, our tutors had found out that the OpenFrameworks community was not as active as the Processing one, and secondly our group as a whole was a lot more familiar with Java, the programming language used by Processing, while OpenFrameworks would've required us to write the program in C++. Finding and selecting technologies one by one during the project slowed down the development process somewhat but on the other hand we were able to gain experience steadily about the technology at hand at any point (i.e. for instance a lot of programming was done with Processing before starting to implement TUIO functions).

Initially we did not encounter any problems with the Processing, partially due to our pretty clear design phase, and partially due to that we did not use the real components we planned, because we were still waiting for them to arrive from the store. Instead we tested out the Processing Video library functions and possibilities with several components we knew we were not going to use in the real prototype. Before moving on to getting input from the touch screen we had already prototyped a solution for controlling the video playback using just the Arduino through Processing. Unfortunately, this also meant that we had used some time to build up functionalities we knew we were not going to keep in the final product, but this was unavoidable and at least let us get more familiar with the working environment.

Getting to work with the multitouch screen took a considerable amount of time, but we luckily found out that interfacing with it was pretty easy. At this stage we forked development, some of us working on the codebase, integrating the touchscreen events in, while others were working with getting our real components hooked up to the Arduino. The latter team eventually came out with the final prototype pen, shown in figure 2. A Q-tip is used as a tip to press against the screen to trigger the pressure sensor inside the tube, which would then turn on the infrared emitter, because we feared that the material of the IR led could scratch the surface of the screen.

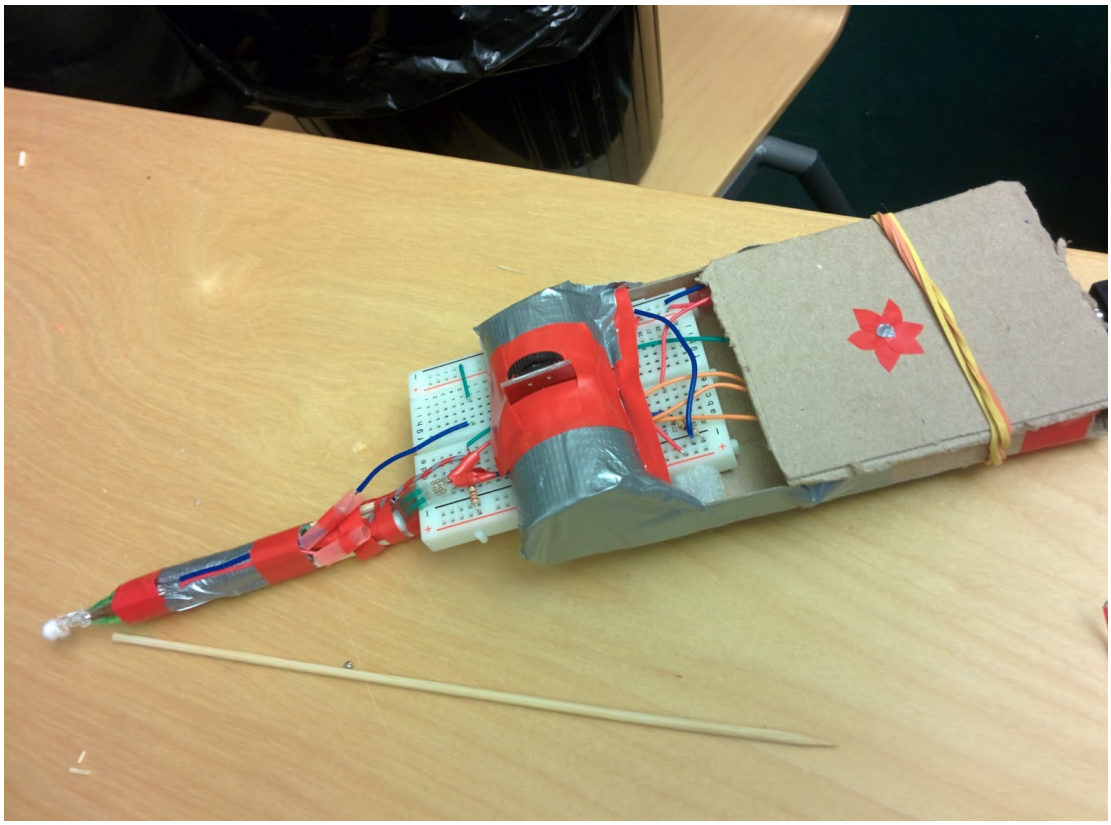

Figure 2. Final stages of pen building. Present now in the design are pen form and stylised cover over Arduino board and different sensors attached to breadboard.

During the design phase we had planned to use some sort of specialized gesture to implement the skipping from one video to the next, but when developing the touch screen functions, we started thinking about the start of the program, how to select the initial video to be played etc. Our tutors had provided us with a three-way button, which could be slid up or down and additionally pressed like a normal button. We decided to use this to handle the volume control, and additionally that button press could be used to implement the stopping feature. The program would start to a video selection screen and the stop button would

always just take us back to this same screen, which would take care of the video skipping features as well, though we did recognize that this could be seen as cutting some corners.

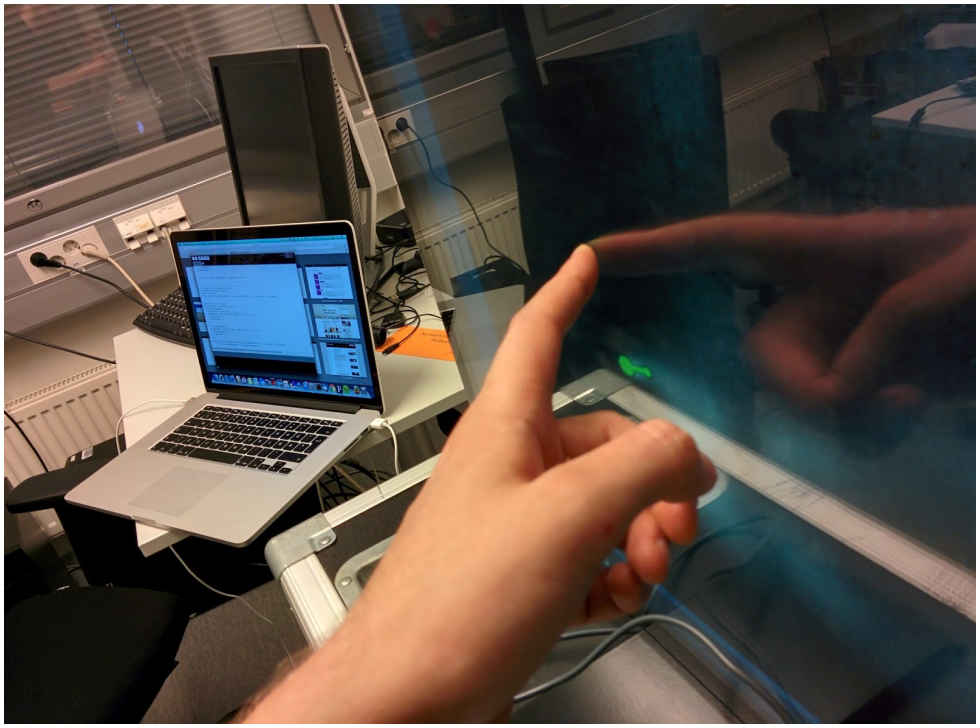

Figure 3. Testing Multitact screen and TUIO-functions in Interaction Lab.

One of the overarching problems we had through the whole project was the fact that we did not properly plan the use of a version control system, which kind of tied in with the fact that the Processing and Arduino IDE strongly direct the developers to use just one file for all code. We decided to use github as our repository, but did not agree on any proper architecture or branching, which led to everyone committing to the master branch all the time. Just like in any software project, this led to pretty frequent conflicts and sometimes even new bugs. Sadly, we did not decide to take time off the actual developing to plan and implement a proper VCS-scheme, so to the last days of development this had been an issue.

Discussion

In our opinion, we did well at this Interactive Pen Project even though we encountered some problems throughout the whole project. Our first problem was that we tried to use Open Frameworks, but it turned out to be an insufficient tool for building this project. After that we started over with Processing, which, of course, took its own time.

Also, time for building this project was quite limited. We found it difficult to come up with times at which we all could be present. Additionally, it was sometimes challenging to get access to the touchscreen, because it depended also on our tutors' timetables. For the first weeks our

work was limited to testing the Arduino, partly because we had to wait for the real components to arrive, and this understandably put us back in terms of actual development and progress on the main project. When looking back at our setbacks though, the only thing that we had to completely scrap as an idea was the use of Open Frameworks, otherwise every difficulty we faced seemed to be just timetable-related, a blocker like having to wait for some components or getting access to the multitouch screen, or an instability problem with the software we had to use, for, as we unfortunately discovered, Processing suffers currently from several annoying bugs.

Overall, though, we could say that what did work, worked pretty much as we expected. We did not have to radically change our plans at any time, the libraries we got to use had pretty clear APIs and were easy to use, and even though we did not have a clear process model for our teamwork, frequent contact through IRC made us pretty flexible in terms of time management. Our tutors were also readily available through e-mail, even at non-office hours, and we extended them the invitation to participate in the IRC channel as well.

If we had to start this project again now, we would undoubtedly have a much clearer picture of the roles of the different components in the end-product, meaning how the Arduino, Processing, multitouch screen and the actual pen tie themselves together, and at which point and so on. Additionally, our knowledge of a collaborative project like this would probably lead us to more carefully consider the use of version control, how to actually maintain a stable codebase and so forth. Lastly, we would prioritize the start of the actual development process much higher, in this project we spent a lot of time understanding the Arduino and its different components, and later found out that we would hardly be using the Arduino IDE at all. But this ties back in with the understanding of the whole project architecture.

Figure 4. Working in Otaniemi when the screen in Exactum's Interaction Lab was unavailable.

Conclusion

In this paper we have retraced our project's steps in developing an interfacing pen for controlling video playback and some limited video library functions. We outline the background of the industry before moving on to our own project concept and initial designs. After introducing the initial design, some of the key points during the development process are laid out, and finally we discuss some of the key points during the whole project, in good and in bad.

In conclusion, we, as a group, feel that the project went well altogether. Even though the project had some minor problems throughout, the group stayed motivated, and nobody felt left alone or unfairly treated in regard to the amount of work. In the end we could produce a working prototype of the pen that could be used in conjunction with a multitouch screen to control the playback of a video, which was the goal of the project. Because of this, we treat the project overall as a success. Further work, if this project were to be continued, should focus on the stability of the program and expanding the video library functionalities beyond just moving to and from the movie select screen.

References

[1] http://dl.acm.org/citation.cfm?id=1029682 (Haptic pen: a tactile feedback stylus for touch screens)
[2] http://dl.acm.org/citation.cfm?id=2380309 (Point and share: From paper to whiteboard)
http://dl.acm.org/citation.cfm?id=1358936 (Twend: twisting and bending as new interaction gesture in mobile devices)

[3] http://dl.acm.org/citation.cfm?id=964708 (Fluid interaction techniques for the control and annotation of digital video)
[4] http://mi-lab.org/files/2008/02/comino_haller_koeffel_leitner.pdf (Bridging the gap between real and virtual objects for tabletop games)
[5] http://dl.acm.org/citation.cfm?id=1658880 (Physical interfaces for tabletop games)
[6] http://dl.acm.org/citation.cfm?id=260044 (Video streamer)
[7] http://dl.acm.org/citation.cfm?id=1394488 (Interface designs for pen-based mobile video browsing)
[8] http://dl.acm.org/citation.cfm?id=2382381 (Accelerometer-based single-handed video browsing on mobile devices: design and user studies)

[9] http://www.multitaction.com
[10] http://en.wikipedia.org/wiki/Arduino