Feedback is… Late: Measuring Multimodal Delays in Mobile Device Touchscreen Interaction

Topi Kaaresoja
Nokia Research Center
Helsinki
Finland
topi.kaaresoja@nokia.com

Stephen Brewster
Glasgow Interactive Systems Group
School of Computing Science
University of Glasgow, Glasgow, UK
stephen@dcs.gla.ac.uk

ABSTRACT
Multimodal interaction is becoming common in many kinds of devices, particularly mobile phones. If care is not taken in design and implementation, there may be latencies in the timing of feedback in the different modalities may have unintended effects on users. This paper introduces an easy to implement multimodal latency measurement tool for touchscreen interaction. It uses off-the-shelf components and free software and is capable of measuring latencies accurately between different interaction events in different modalities. The tool uses a high-speed camera, a mirror, a microphone and an accelerometer to measure the touch, visual, audio and tactile feedback events that occur in touchscreen interaction. The microphone and the accelerometer are both interfaced with a standard PC soundcard that makes the measurement and analysis simple. The latencies are obtained by hand and eye using a slow-motion video player and an audio editor. To validate the tool, we measured four commercial mobile phones. Our results show that there are significant differences in latencies, not only between the devices, but also between different applications and modalities within one device. In this paper the focus is on mobile touchscreen devices, but with minor modifications our tool could be also used in other domains.

Categories and Subject Descriptors
H.5.2 [**Information Interfaces and Presentation**]: User Interfaces – a*uditory (non-speech) feedback, benchmarking, evaluation/methodology, graphical user interfaces (GUI), haptic I/O, input devices and strategies*

General Terms
Measurement, Human Factors

Keywords
Latency, measurement, visual feedback, audio feedback, tactile feedback, touchscreen

1. INTRODUCTION
Multimodal interaction is becoming common in many different product domains, but particularly for touchscreen mobile phones which now commonly give visual, auditory and haptic responses to user inputs. Although computers are becoming faster, operating systems and applications are more complex. This causes latency in interaction, and can disrupt the usage and the user experience.

It has been stated that the system end-to-end latency is one of the most important problems limiting the quality, interactivity and effectiveness of virtual and augmented reality, as well as head mounted display systems [1-3]. Latency issues may even result in physical problems for users, for example 'simulator sickness'. Wright *et al*. [4] claim that several milliseconds of latency and jitter can make the difference between a responsive, expressive, satisfying real-time computer music instrument. Measuring and understanding latency is therefore important for multimodal interaction design and development.

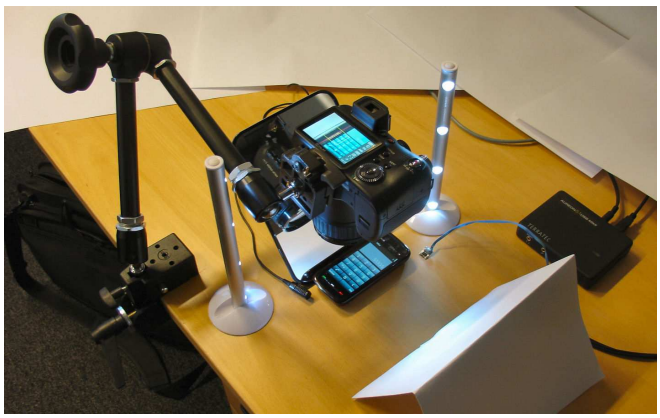

Figure 1. Overall setup of the multimodal latency measurement tool.

Based on the earlier work done in latency we believe the latency also has an effect in touchscreen interaction. In order to understand if latency is an issue or not, or take some corrective actions, we have first to be able to measure the latencies between user action and device response in the visual, audio and haptic modalities. There are several research prototypes [2, 3, 5] and commercial products for latency measurements in different contexts. However, a single tool is missing for measuring the latency simultaneously in the visual, auditory and haptic modalities in touchscreen interaction.

This paper introduces an easy to implement multimodal latency measurement tool for touchscreen interaction (see Figure 1 for overall setup) . The use of the tool requires no changes or modifications to the device being measured. Although it is for laboratory usage, it is portable and easy to move if needed. It uses off-the-

ICMI-MLMI'10, November 8-10, 2010, Beijing, China.

shelf components and free software as much as possible and is capable of measuring latencies accurately between different events in different modalities. The tool uses a high-speed camera, a mirror, a microphone and an accelerometer to measure the touch, visual, audio and tactile feedback events that occur in touchscreen interaction. The microphone and the accelerometer are both interfaced with a standard PC soundcard that makes the measurement and analysis simple. The latencies are acquired by hand and eye using a slow-motion video player and an audio editor. We believe that this tool is especially useful for researchers doing perceptual research with multiple modalities, multimodal interaction designers or display and software developers struggling with delays in an implementation. In this paper, the target devices have been mobile touchscreen devices, but with minor modifications our tool could be used also in other domains.

2. EARLIER WORK

The measurement of latency actually means timekeeping between the action and the response. The following literature review shows some examples of how the timekeeping has been done in different modalities and contexts.

2.1 Human intermodal latency perception

The minimum perceived intramodal asynchrony 2 ms has been found in audio modality [6]. However, the perception of latency between different modalities is happening on tens of milliseconds level. That is supporting e.g. the research by Levitin *et al.* [6]. They found out that the perception of visual-auditory asynchrony is 40 ms and haptic-auditory asynchrony lies between 25 and 40 ms depending on the modality order. Adelstein *et al.* [7] found the just noticeable difference for haptic-auditory latency change being 25 ms. These figures give us a hint about the temporal resolution needed for a latency measurement tool.

2.2 Latency measurements in virtual environments

He *et al.* [2] introduce a video-based latency measurement system for Virtual Environment (VE) using a normal speed video camera. The VE system delay is determined by reading the video tape frame-by-frame. A similar video-based frame-by-frame reading approach was used by Liang *et al.* [8] when determining latencies in a virtual reality tracking device. Miller and Bishop [3] introduce a "Latency Meter" for measuring end-to-end latency in VEs. They used two high-speed 1-row CCD detectors and special algorithms to extract the latency between user movement and VE system display update. The aim of the work was to develop a standalone instrument that would estimate the visual latency without any additional electrical connection or change to the VE software.

2.3 Audio latency measurements in computer systems

Freed *et al.* [9] measured operating systems latencies with a audio recorder. They used a near-to-zero-latency device to transcode a computer network or MIDI event to a short audible "blip". That way they could use the simple audio recording software to record the input and the output at the same time in order to investigate the latencies. MacMillan *et al.* [10] used the same technique to perform an extensive set of audio latency experiments with different audio hardware in different operating systems. They investigated the suitability of general-purpose computer to real-time audio processing. Likewise, Nelson and Thom [11] measured MIDI latency under Linux, OS X and Windows with their own MIDI to audio paradigm. Wright *et al.* [4] measured system latencies in various computer operating systems from stimulus in to audio out.

2.4 Touch and tactile feedback latency measurement

Lehtosalo [5] used a force sensor to obtain latencies between finger touch and tactile feedback in touchscreen interaction. He used a proprietary setup where the mobile phone equipped with a touchscreen was located on a force sensor. The force sensor detected the finger press and the tactile feedback provided by vibration motor in the phone. Matlab was used for processing and analyzing the measurement data to computer the latency.

2.5 Commercial timing analysis products

OPTOFidelity (www.optofidelity.com) sells a product called WatchDog for automated timing analysis for mobile phone screen events. It shoots the screen with a video camera and automatically detects the changes on the screen. After the measurement it automatically creates a timing report in HTML. It also has an option to high-speed video and an add-on sensor for haptics. However, it does not measure audio feedback. BlackBox ToolKit (www.blackboxtoolkit.com) in turn is a timing analysis tool for pre-evaluation of a perception study setup or experiment application. It can detect for example colour changes on a screen, audio feedback as well as user actions, like button presses. In addition it can be used for simulating a participant performing a perception experiment by connecting signal cables to mouse switches on the computer running the experiment application. This enables the researcher to analyze the internal latencies of the experiment system before running the real experiment.

3. UNDERSTANDING A VIRTUAL MULTIMODAL BUTTON PRESS

Pressing a button is a very everyday thing and it seems to be a very simple and trivial task, as it actually is – when interacting with a real button. However, when we have to do everything artificially on a touchscreen device, an itemization of all the stages shows us that a virtual button press is far from trivial.

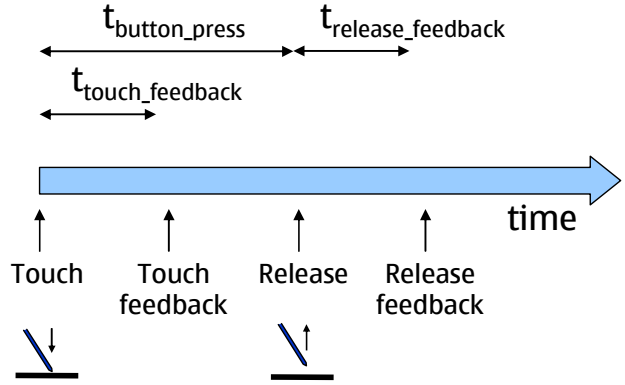

Figure 2. A rough timing diagram for a virtual button press.

The anatomy of virtual button press can be roughly illustrated as in Figure 2. The screen is touched with a stylus or a finger and feedback is given for the touch after time $t_{touch\_feedback}$ has passed.

This time is called touch feedback latency. A release of the stylus or the finger happens sooner or later depending on the user and the release feedback is given again after time $t_{release\_feedback}$ has passed. This time is called touch release latency.We can zoom in a little by separating the feedback in different modalities. To simplify, we focus only on the button press, or touch. After the stylus or finger touches the screen, the different feedback elements of the button press start to activate (after the individual latency period for each): Visual feedback, audio feedback, tactile feedback and action feedback (see Figure 3). Visual, audio and tactile feedback here mean everything that is related to the button itself or the content of the button, e.g. visual feedback can be the colour change of the button pressed, audio feedback can be an audible click designed for the button, and tactile feedback can mean a short vibration that can be felt by the stylus or finger pressing the button. The action feedback means the actions the button press initiates, for example, a number appearing on the screen, an application opening or a piece of music starting to play.

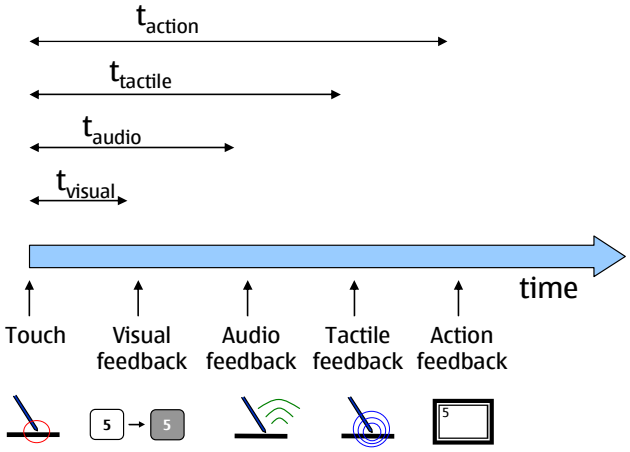

Figure 3. Diagram of a touch event and its different feedback types. This applies to both natural action and touchscreen events. All the feedback can also happen on touch release.

Visual, audio, tactile and action feedback latencies (t_{visual}, t_{audio}, $t_{tactile}$ and t_{action} in Figure 3) can differ remarkably from each other. In addition they can occur in an arbitrary order. However, for a good interaction it would be wise to have all the button-related feedback (visual, audio and tactile) before the action feedback. The aim of the button feedback is to indicate to the user that the button has been pressed correctly and to simulate some of the experience of pressing a real, physical button. In the real-world case, when we press a button we get all of the feedback from the physical action of pressing the button immediately, but the action caused by the press may occur some time later. With virtual button feedback in practice this is not true.

Figure 4. A visual feedback of a virtual button as a form of a popup (from the Apple iPhone) .

In order to be exact we can zoom further into the button-press: There usually are several instances of the different feedback types. For example, in addition to the basic colour of the button itself (changing from white to grey in Figure 5), visual feedback can include a pop-up of a number or key (Figure 4), or it can cause the whole keypad to change from uppercase to lowercase at the beginning of a sentence. In the audio domain, there can be a click imitating the button press and another sound playing a DTMF tone in the phone dialler, for example.

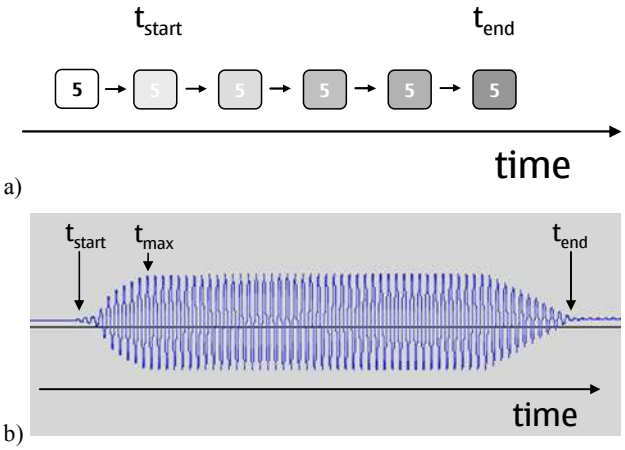

Figure 5. a) Propagation of virtual button's visual feedback colour change, b) timeline of a 70 ms audio feedback in a virtual button.

Visual feedback is usually shown on the button as long as the press action is happening, whereas audio and tactile feedback is usually designed to be as a fixed length. In both cases, we can still consider more deeply the fundamentals of the feedback. The visual feedback needs some time to show. Think about the colour change of a virtual button. The start of the feedback is that the colour starts to change; the end of the feedback is the moment when the colour has fully changed. This takes a certain amount of time depending on the implementation (see Figure 5a). In this paper, visual feedback latency is the time t_{end}, when the exact moment of touch is the start of the timekeeping. Then think about the simple beep and click of audio and tactile feedback. They also need some time to start and reach maximum intensity (see Figure 5b). After that they are played and stopped and take some time to return to their initial, pre-press state. In this paper, the latency is the time between the start (the touch) and t_{max}. Of course, the same is true also for action feedback, although the phases will usually be much more complicated.

We do not concentrate here to the sources of the latencies inside device hardware or software architecture. Instead we focus on the fact that there is end-to-end latency which should be measured and understand from the user's point of view.

4. MULTIMODAL LATENCY MEASUREMENT TOOL

In order to be able to investigate the latencies in virtual button presses in touchscreen interaction, we needed to build a measurement tool that would be able to detect latencies in the visual, auditory and tactile modalities. Our aim was that the device would be beneficial (or even a necessity) for both researchers conducting perceptual experiments and for product designers struggling with

delays in hardware, software and user interfaces. We had the following design drivers in mind:

1. As inexpensive as possible so as to be usable by every researcher and designer;
2. Easy to build. That means we want to use off-the-shelf equipment and components as much as possible;
3. Absolutely no hardware or software modifications to the devices to be measured;
4. Capable of measuring latencies between visual, audio and tactile modality combinations starting from the moment of touch;
5. Capable of measuring devices with resistive or capacitive touchscreens;
6. For laboratory use, but still as portable as possible.
7. Based on the human perception capabilities, measurement resolution and inaccuracy should be minimum 1 ms in audio and tactile, and 10 ms for visual modality.

4.1 General methodology

In any measurement there is a stage of data capture and a stage of data analysis. The measurement of latency actually means timekeeping between the action and the response. We can capture the data by recording it as such and then find the moment of the start and end of the clock in order to extract the latency. Sometimes, however, special capture methods are needed to make either the recording or the analysis easier. Based on earlier work in latency measurements, we made a simple classification of these advanced capture methods: Simplify, oversample or transcode.

Simplification means that the data acquisition compresses and filters the data so that changes in the time domain are more easily observable. The time domain can be also stretched with oversampling in order to zoom in time and simplify the analysis. Both of them will make the investigation of the relevant information easier. Transcoding means that the data is transformed from another modality to another in order to be more easily captured or analyzed. In this paper, we used oversampling and transcoding methods.

Data analysis can be done either visually by a human from the captured data, or it can be automated with signal processing methods. In this paper, and for this initial version of our latency measurement tool, analysis is done visually.

An example of simplification as well as oversampling is the Latency Meter [3]. Instead using normal speed video of a user and the VE, they use high-speed CCD sensors to simplify the "picture" by calculating two single numbers from the brightness distribution from the CCD sensors. These quickly updated numbers are used to calculate the latency between the user action and the VE response. Another example of simplification, but also transcoding is the work of Freed *et al.* [9] in which they measured operating systems latencies. They transcode network events into audible beeps, but also simplify by transcoding only the start and the end of the events.

Another practice used in latency measurements is the use of audio recorder. It can be a two or multiple channel soundcard found in a standard PC or a separate piece of hardware. This method need the events to be transcoded into audio first. Events are recorded in the different channels in order to easily extract the latencies between them using an audio editor. This methodology is used in [4, 9, 10].

Figure 6. Casio Exilim EX-F1 (Casio Corp., exilim.casio.com)

4.2 Measurement of the latency between touch and visual feedback

We chose to use a high-speed camera to find out what exactly happened when a touchscreen button was pressed and the graphical UI changes. There are plenty of industrial high-speed cameras available, but they are big and expensive. Fortunately, there are consumer grade digital cameras currently available at reasonable prices that can shoot high-speed video, for example the Casio Exilim EX-F1 seen in Figure 6 (exilim.casio.com). It is capable of shooting 300, 600, and 1200 frames per second with 512x384, 432x192, 336x96 pixel resolutions, respectively. The 300 fps with the almost VGA resolution seemed to show good data for our purposes (see Figure 7). The high-speed video can be viewed with the freely available MiDas player (www.xcitex.com). However, the EX-F1 supports only the MOV video format, which is not supported by the MiDas player. Fortunately, we found MOV to AVI video converter to solve this problem (www.pazera-software.com). With these tools we could easily transfer the high-speed video from the EX-F1 to the MiDas player for analysis. One of the big advantages of the EX-F1 is that it can also be used as normal digital camera.

4.3 Detecting the moment of touch

There were multiple candidates for detecting the touch of the fingertip or stylus on the surface of the touchscreen. One was to use a programmable robot arm with a force gauge. That would have provided us a controllable stimulus, but is challenging for capacitive touch screens and a robot like this is expensive and far from portable. An alternative was to create a transcoder from touch to light, by building a stylus-like device with a sensitive switch on one end and an LED on the other. This way the moment of touch would have been seen in the high-speed video as an LED light. However, this would have again caused issues with capacitive touchscreens and required the building of new hardware, which not all users of our system would be capable.

After some investigation, we ended up with a simple solution: an inexpensive make-up mirror with an adjustable support. We placed it next to the device to be measured so that the stylus or finger can be seen from the side. This way also the user interface recording is inherently synchronized with the touch detection. Figure 10 shows our mirror arrangement more closely. This also enabled us to interact with the device as usual, with a stylus or finger. Figure 7 shows an example picture sequence of stylus ap-

proaching the touchscreen, where the moment of touch is easily seen in the mirror.

Figure 7. A picture sequence showing the stylus approaching and finally touching the touchscreen surface of a mobile phone. The mirror can be seen on the left and gives a clear view of when the stylus hits the screen. The picture is of the Samsung Omnia i900 dialler user interface.

4.4 Measurement of latency between touch and audio-tactile feedback

Using a microphone would work for some tactile feedback since the vibration generally causes audible noise that can be recorded by a microphone. But initial tests showed that the microphone did not pick up the sound of the tactile feedback accurately enough, especially if the tactile feedback intensity was low and its duration short. So we moved to an accelerometer. However, we wanted to use as little extra hardware as possible, so we transcoded the analogue output of the accelerometer into audio through the soundcard (Terratec Aureon 5.1 USB MKII (www.terratec.net)). It turned out to work well with the line-in input of the soundcard, so no extra data acquisition hardware was needed. The accelerometer circuit is the only piece of hardware that is not off-the-shelf, although is very simple. Figure 8 shows the principal circuit of the accelerometer used. In addition, the accelerometer picked up the touch event much better than the microphone (see Figure 9). It could be replaced by a off-the-shelf alternative by Phidgets (www.phidgets.com), although the accelerometer board is much bigger than ours.

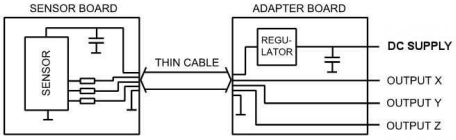

Figure 8. The accelerometer circuit. Only OUTPUT Z is used in the measurements.

In the final setup we used a Vivanco EM216 lavalier microphone (currently known as EM35, www.vivanco.de). The acceleration sensor within the accelerometer was Kionix KXPS5 series 6g model (www.kionix.com). For recording and analysis, we used Audacity v.1.3.6, a free open source software application (audacity.sourceforge.net), capable of showing timing information with millisecond resolution.

Figure 9 shows an example recording of touch, audio and tactile feedback in the Audacity audio editor. The audio feedback can be seen above on the left channel, and the touch (stylus hit) can be seen on the right channel in addition to the tactile feedback (and an attenuated trace of audio feedback). The latency between the touch can be measured with Audacity's selection tool.

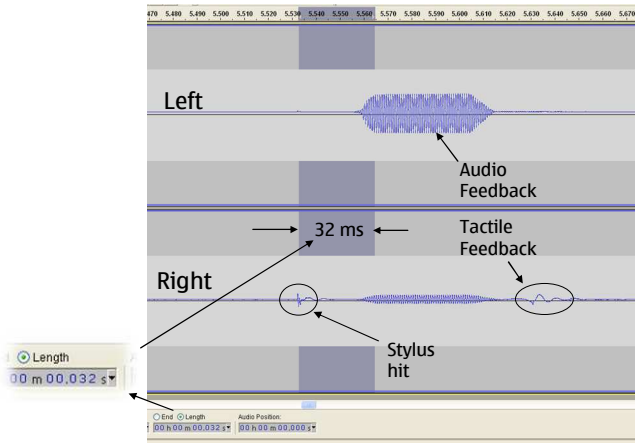

Figure 9. Example of a record of touch, audio and tactile feedback of a virtual button press.

The microphone was taped towards the loudspeaker of the phone being measured, and the accelerometer board was attached as near as possible to the tactile actuator of the measured device.

4.5 General measurement setup arrangement

Figure 1 shows the overall setup of the multimodal latency measurement tool. The centre of the tool is the camera attached to the table with a clamp and a 6 DOF adjustable arm. The mirror is placed next to the measured device and opposite to it a white background to make a clearer image (a folded sheet of paper). Two light sources can be seen on both sides of the camera (cheap LED lamps), as well as the microphone and accelerometer.

4.6 Extracting the latency between touch, visual, audio and tactile modalities

The high-speed video and audio streams are inherently synchronized by the stylus or finger hit seen in both streams. The visual feedback latency could be extracted by playing the high-speed video with MiDas player frame-by-frame and finding the frame where the stylus or finger touched the screen surface, in addition to the visual feedback t_{end} (Figure 5a). The latency was then measured as the number of frames from the touch event to the visual feedback multiplied by 1/300 s. The audio and tactile latency could be extracted as described above and in Figure 9. Latencies between different modalities can be extracted simply by subtracting the latencies of different modalities.

5. MEASUREMENTS

To evaluate the functionality of the latency measurement equipment, we ran a simple study to measure latencies in some commercial mobile phones. The phones all featured touchscreens and they all had audio and tactile feedback for the buttons. Our hypotheses were:

1. There will be **observable latencies** in the devices measured;
2. The latencies will **vary between devices**;
3. The latencies will **vary between applications;**
4. The latencies will **vary between modalities;**
5. The latencies will be **shorter with stylus rather than finger** interaction since the finger needs time to deform as it makes contact with the surface of the device.

Before starting to find answers to the hypotheses we had to make sure that the measurement setup was accurate and well synchronized.

Figure 10. Close-up of the mirror arrangement. The microphone (front) and the accelerometer (back) are not attached to the phone in this figure.

5.1 Calibration of the measurement tool

Although the internal clocks of the video camera and soundcard should have been accurate enough for our needs, we arranged a calibration test for the tool to validate the accuracy of the all the recording channels: An LED and a small loudspeaker were connected to the output of a calibrated Agilent 33120A Arbitrary Waveform Generator (AWG) (Figure 11). The high-speed camera was shooting the LED, the microphone was picking up the audio 2 mm on top of the loudspeaker and the accelerometer was attached to the bottom of the loudspeaker. The calibration method was to time two different time lengths: 100 ms and 1000 ms.

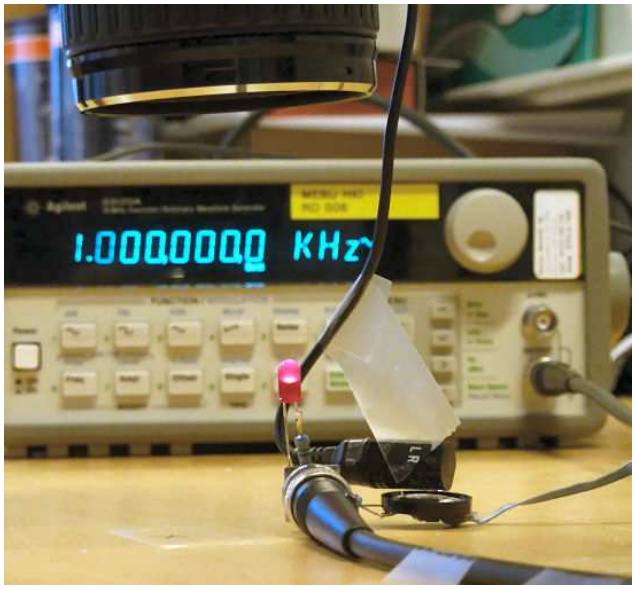

Figure 11. The calibration setup consists of an arbitrary waveform generator, LED and loudspeaker.

The AWG generated a continuous burst sequence of ten bursts per second for measuring 100 ms (the time between two bursts). For timing 1 s the AWG generated a continuous burst sequence of 1 burst per second. The length of the burst was 10 ms for 100 ms measurement and 100 ms for 1s measurements. The frequency of the bursts was 1 kHz in both cases. The measurement was repeated 10 times for each time length.

Table 1 shows the calibration results with standard deviations for the measurement resolution (σ_r) and the measurements (σ_m). The mean measurement error remained under 0.1% for all cases except for the visual measurement of 100 ms which gave a 0.37% mean error (0.37 ms). These errors as well as the standard deviations were small relative to human perception of latency differences, so we could be sure that our system has the resolution and accuracy needed to measure latency in the different modalities.

5.2 Measuring the touchscreen mobile phones

We chose four touchscreen mobile phones from different manufacturers. Two of them were equipped with resistive and two with capacitive touchscreens. Using both, we could show that our latency measurement tool was capable of measuring devices with both technologies. All of the phones selected featured audio and tactile feedback for virtual buttons. The phones measured were: the HTC Desire (www.htc.com), LG Chocolate BL-40 (www.lg.com), Nokia 5800 XpressMusic (www.nokia.com), and Samsung Omnia i900 (omnia.samsungmobile.com) (see Figure 12).

We needed to test similar applications in each phone so chose the dialler and text editor. The dialler is the application with a number keypad for making a phone call. The text editor is used for creating messages. We used the text editor in number mode to make it easier to compare to the dialler. We measured the keypress of number 5 key in both applications for ten repeats of each on each phone.

6. RESULTS

We can see from the Figure 13 that our hypothesis 1 is supported, since we could measure observable latencies from all the phones.The results show that the latency varies from 35 ms audio feedback latency in 5800's to 360 ms audio feedback latency in BL40 in dialler.

To go further, we compared the latency data from finger and stylus interaction to see if input device caused any differences in latency. The analysis was done only for the data from 5800 and i900, since other two phones featured capacitive touchscreens and did not support stylus input. A one-way ANOVA on input type showed no significant effect [$F(1,296)=0.0002$, $p=0.989$] on latency, which means that the use of finger did not cause significantly more latency than the use of stylus. So our hypothesis 5 is not supported. For further analysis we concentrated just on finger input so that we could compare all devices.

We analyzed the latency data (Figure 13) using a three-way ANOVA over phone, application and latency type. There were significant main effects found of all the factors: phone [$F(3,27)=400$, $p<0.01$], application [$F(1,9)=50.5$, $p<0.01$] and latency type [$F(2,18)=95.9$, $p<0.01$]. Also all the interactions between the factors were significant: phone between application [$F(3,27)=6.77$, $p<0.01$], phone between latency type [$F(6,54)=192$, $p<0.01$], application between latency type [$F(2,18)=99.0$, $p<0.01$] and finally between phone, application and latency type [$F(6, 54)=248$, $p<0.01$].

Table 1 Measurement resolutions and calibration results with standard deviations. All units are milliseconds.

Record channel	Refer-ence	Measurement resolution	Mean result	Difference	Relative Error	Stdev of measure-ment resolution (σ_r)	Stdev of measure-ments (σ_m)
Audio	100	0.0227	99.93	0.07	0.07%	0.0065	0.027
Audio	1000	0.0227	999.4	0.6	0.06%	0.0065	0.026
Tactile	100	0.0227	99.92	0.08	0.08%	0.0065	0.019
Tactile	1000	0.0227	999.4	0.6	0.06%	0.0065	0.023
Visual	100	3.4	99.67	0.37	0.37%	0.96	1.05
Visual	1000	3.4	999.3	0.7	0.07%	0.96	1.41

Figure 12. The mobile phones measured: HTC Desire, LG Chocolate BL-40, Nokia 5800 XpressMusic and Samsung Omnia i900.

Post hoc pair-wise analysis shows that the audio feedback latency in dialler differs significantly between all phones [p<0.01]. This supports our hypothesis 2. However, another example shows that it is not always supported: Visual feedback latency in dialler is not significantly different in i900 and BL40 [t(9)=0.04, p<0.97].

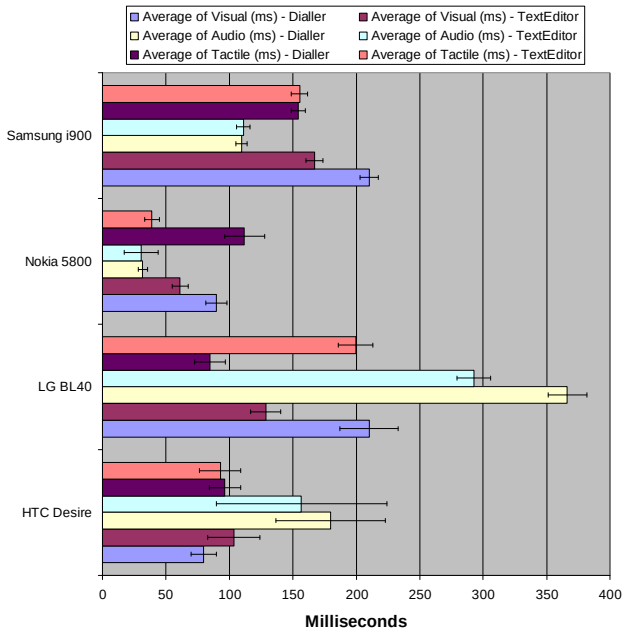

Figure 13. Mean latencies with standard deviations for visual, audio and tactile modalities and the two applications, dialler and text entry, and the four mobile phones.

An example shows that our hypothesis 3 is partly supported: The pair-wise post hoc analysis shows, that the visual feedback latency in i900 differs significantly between the two applications [p<0.01], whereas audio and tactile feedback latencies don't [t(9)=1.32, p<0.22 and t(9)=0.50, p<0.63].

The post hoc pair-wise analysis shows significant differences in latencies between different modalities when we take the text editor as an example: The visual and audio feedback latencies differ significantly in Desire [p<0.02], i900 [p<0.01], BL40 [p<0.01] and 5800 [p<0.01]. The visual and tactile feedback latencies also differ significantly in all phones [p<0.01]. The audio and tactile feedback latencies differ significantly in all phones [p<0.01] except 5800 [t(9)=2.25, p<0.051]. This partly supports our hypothesis 4.

We also can notice visually the remarkable variation in standard deviations, the smallest being in 5800 text editor and the largest in BL40 dialler (statistical significance was not checked for standard deviations).

7. DISCUSSION AND FUTURE STUDIES

The results of this initial study show that our tool can effectively measure multimodal latencies in different applications on a range of different phones. The results show also that the tool can detect latencies both with finger and stylus interaction. The reason for insignificant effect of input style on latencies might be the speed of the pressure. If the touch input would happed slower, the difference could be significant after some threshold. This could be validated with a controllable robot arm.

Although our multimodal latency measurement tool is working as planned, there is room for improvement. For example the manual analysis of the slow-motion video and the audio files containing auditory and tactile feedback is slow and time consuming. It also is a potential source of errors. Automating the analysis of the videos to find the moment of touch and the visual events would improve the speed and accuracy of the analysis, and also the reliability of the results. Automating the analysis of the audio files would again speed things up.

It would be interesting to expand the measurements to other touchscreen widgets and interaction patterns, such as sliders, scrollbars, and drag-and-drop to see how latency changes in more continuous interactions. With slight modifications to the setup, our multimodal latency measurement tool could also be also used for latency measurements of whole device gestures and their responses.

The creation of this tool is just the first step in our research. Our final aim is to study the effects different latencies and combina-

tions of latencies have on users and their interactions with touchscreens. We will conduct experiments about the usability and the user experience with controlled ranges of different latencies in different modalities, in button press, number and text entry, and usage of different touchscreen widgets. This tool will be used to validate the latencies in the experiments.

8. CONCLUSIONS

This paper introduced a multimodal latency measurement tool for touchscreen interaction. Off-the-shelf components and freeware software were used to make the system cheap and easy for all to use whilst still being capable of measuring latencies accurately between different events in the visual, auditory and tactile modalities. The tool features a high-speed camera, a mirror, a microphone and an accelerometer to measure the touch, visual, audio and tactile feedback events that occur in touchscreen interaction. The microphone and accelerometer were both interfaced with a standard soundcard that made the measurement and analysis simple. The latencies were extracted visually using a standard slow-motion video player and an audio editor. In this paper, the focus has been in mobile touchscreen devices, but with minor modifications our tool could be used also in other domains. To validate the tool, we measured four commercial mobile phones. Our results show that there are significant differences in latencies, not only between the devices, but also between different applications and modalities within one device.

9. ACKNOWLEDGEMENTS

We would like to thank Tom Ahola for building the accelerometer board and giving advice on metrology issues. We also would like to thank to Risto Kaivola for giving valuable advice on latency issues.

10. REFERENCES

[1] Mine, M. R., Characterization of End-to-End Delays in Head-Mounted Display Systems. University of North Carolina, Chapel Hill, 1993.

[2] He, D., Liu, F., Pape, D., Dawe, G. and Sandin, D., Video-Based Measurement of System Latency. In Proceedings of the IPT2000, International Immersive Projection Technology Workshop (Ames IA, USA, 2000) (2000).

[3] Miller, D. and Bishop, G., Latency meter: a device for easily monitoring VE delay. In Proceedings of the Stereoscopic Displays and Virtual Reality Systems IX (San Jose, CA, USA, 2002) (2002).

[4] Wright, M., Cassidy, R. J. and Zbyszynski, M. F., Audio and Gesture Latency Measurements on Linux and OSX. In Proceedings of the International Computer Music Conference (Miami, FL, USA, 2004) (2004), 423-429.

[5] Lehtosalo, A., Haptics Technologies and Electromechanical Latency Measurements in Tactile Feedback System. Master's Thesis, Tampere University of Technology, Tampere, Finland, 2009.

[6] Levitin, D. J., MacLean, K., Mathews, M. and Chu, L., The Perception of Cross-Modal Simultaneity. Invited paper. . In Proceedings of the 3rd International Conference on Computing Anticipatory Systems (Liège, Belgium., 1999) (1999).

[7] Adelstein, B. D., Begault, D. R., Anderson, M. R. and Wenzel, E. M., Sensitivity to haptic-audio asynchrony. In Proceedings of the 5th International Conference on Multimodal Interfaces (Vancouver, Canada, 2003). ACM (2003), 73-76.

[8] Liang, J., Shaw, C. and Green, M., On Temporal-Spatial Realism in the Virtual Reality Environment. In Proceedings of the UIST'91, Symposium of User Interface Software and Technology (Hilton Head, South Carolina, US, 1991) (1991).

[9] Freed, A., Chaudhary, A. and Davila, B., Operating Systems Latency Measurement and Analysis for Sound Synthesis and Processing Applications. In Proceedings of the International Computer Music Conference (Thessaloniki, Creece, 1997). ICMA (International Computer Music Association) (1997).

[10] MacMillan, K., Droettboom, M. and Fujinaga, I., Audio Latency Measurements of Desktop Operating Systems. In Proceedings of the International Computer Music Conference (Habana, Cuba, 2001). ICMA (International Computer Music Association) (2001).

[11] Nelson, M. and Thom, B., Interactive MIDI: Real-time Performance Evaluation. In Proceedings of the New Interfaces for Musical Expression (Hamamatsu, Japan, 2004) (2004).