Sniper Pointing: Above the Surface Pointing with Multiple Resolutions

Master's Thesis at the
Media Computing Group
Prof. Dr. Jan Borchers
Computer Science Department
RWTH Aachen University

by
Kashyap Todi

Thesis advisor:
Prof. Dr. Jan Borchers

Second examiner:
Prof. Dr. Torsten Kuhlen

Registration date: 23.08.2012
Submission date: 25.02.2012

I hereby declare that I have created this work completely on my own and used no other sources or tools than the ones listed, and that I have marked any citations accordingly.

Hiermit versichere ich, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Aachen, February 2012
Kashyap Todi

Contents

List of Figures

List of Tables

Abstract

Manipulation of a pointing cursor is time-consuming because of two factors—homing actions to switch between the keyboard and the pointing device; and visual search time required for location of the cursor. In this thesis, the *'Sniper Pointing'* interaction technique is presented, which attempts to reduce the impact of these factors. The volume directly above the desk is utilized for freehand cursor manipulation with multiple levels of pointing granularity. On raising the hand slightly above the keyboard, cursor manipulation with absolute mapping is immediately offered, reducing homing time between devices and eliminating need for an initial visual search of the on-screen cursor. Raising the hand higher increases pointing accuracy and allows users to perform fine-grain positioning of the cursor. Using these techniques, speed and accuracy issues for pointing tasks are addressed. A set of user studies is performed to aid in design decisions and to quantitatively evaluate this input method for cursor manipulation.

Acknowledgements

Over the course of this Master's Thesis, I have received technical and non-technical assistance from various people and sources, and I would like to briefly acknowledge their support.

I would firstly like to thank Chatchavan Wacharamanotham, without whose guidance this thesis would not have been possible. He has been the most patient and helpful supervisor, and I am grateful to him for all his support and advice.

The Media Computing Group, at RWTH Aachen, has offered a wonderful work environment and has provided me with all the resources necessary. I am thankful to the group for giving me a chance to perform my thesis with them.

I would also like to thank Prof. Dr. Jan Borchers, my thesis advisor, and Prof. Dr. Torsten Kuhlen, the second examiner for this thesis, for their time and support.

Lastly, I would like to thank everyone else who has, at some point or the other, taken time out to help me with successfully accomplishing the aims of this thesis. I thank all the participants, who voluntarily took part in my user studies, and also my family and friends, who have provided me with mental and moral support over the course of this thesis.

Thank you!

Kashyap Todi.

Conventions

Throughout this thesis, the following conventions are used.

Text conventions

Definitions of technical terms or short excursus are set off in orange-coloured boxes.

EXCURSUS:
Excursus are detailed discussions of a particular point in a book, usually in an appendix, or digressions in a written text.

Definition:
Excursus

Definitions of hypotheses are set of in green-colored boxes.

HYPOTHESIS:
A supposition or proposed explanation made on the basis of limited evidence as a starting point for further investigation

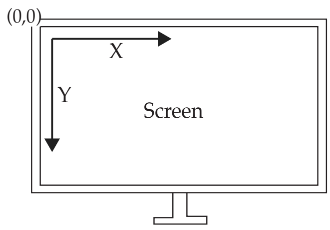
Hypothesis:
Description

The whole thesis is written in British English.

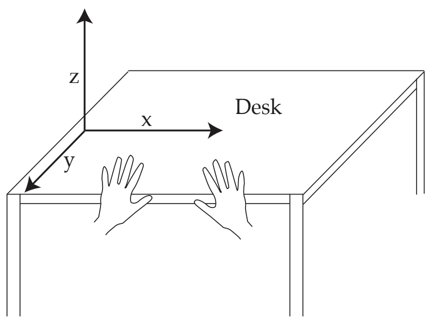
Graphical Conventions

Error bars are constructed using a 95% confidence interval of the mean.

Screen coordinates are represented in upper-case, according to the following axes convention:



Three-dimensional world coordinates are represented in lower-case, according to the following axes convention:



Chapter 1

Introduction

"We're going to use the best pointing device in the world. We're going to use a pointing device we're all born with—born with ten of them. We're going to use our fingers."

—Steve Jobs

In this chapter, a brief introduction to the subject of pointing provides some information with respect to how traditional techniques provide cursor manipulation, and outlines a basic sequence of actions which are performed in order to manipulate on-screen mouse cursors. Further, the proposed *Sniper Pointing* interaction technique is introduced and contributions of this thesis are highlighted.

Ever since the introduction of the mouse by English et al. [1967], it has been known to be an optimal pointing device in terms of speed and accuracy ([Card et al., 1978, p.16]). In a standard desktop environment, the main input interface for interaction with the graphical user interface (GUI) often consists of a keyboard and a pointing device (Figure 1.1).

Mouse known to be an optimal device.

Since both these input devices are physical in nature and each of them occupies its own space on a horizontal surface, switching between the two devices, known as *homing* (figure 1.2a), is a chore that needs to be performed during input and interaction tasks. Douglas and Mithal [1994]

Homing between mouse and keyboard is time-consuming. Accounts for 40% of total pointing time.

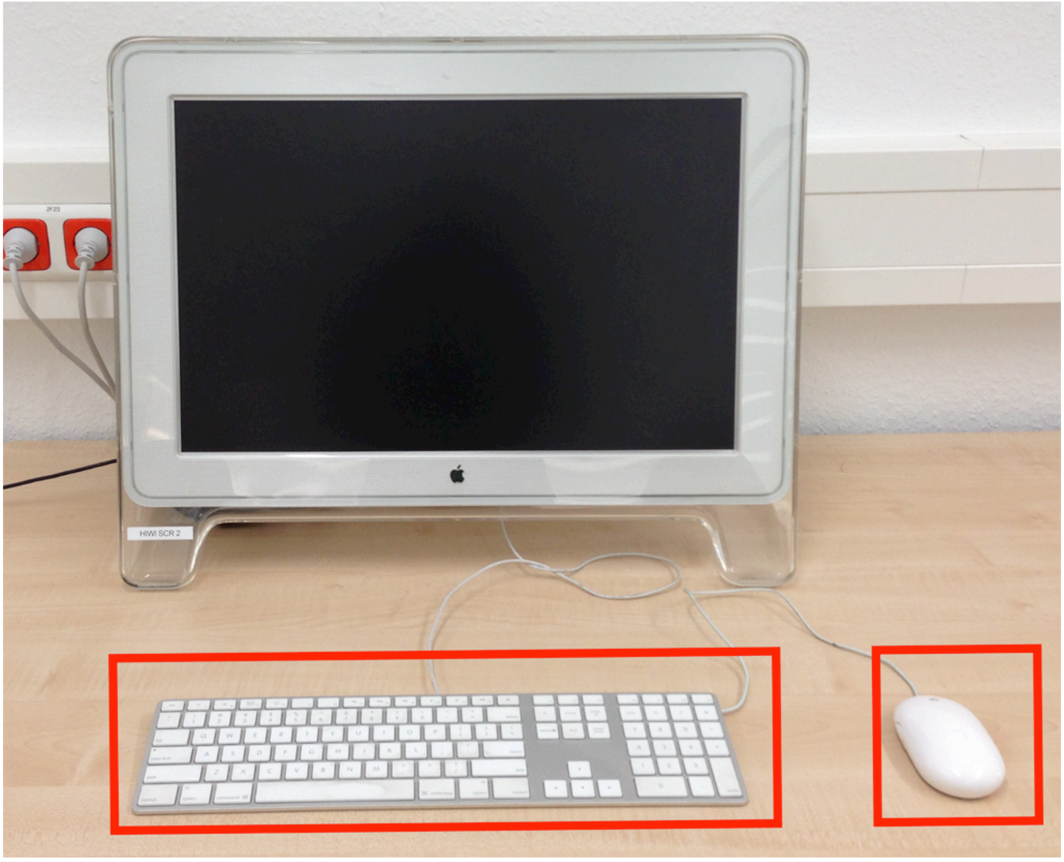


Figure 1.1: *A standard desktop computer set-up. A vertical display is used for output and input is provided using the keyboard and mouse, placed on a horizontal surface.*

showed that homing between the keyboard and mouse is a time consuming task and requires a total of 0.8 seconds, accounting for 40% of the total time[1] during the course of a single pointing task. This thesis also studies the effect of this homing time on the overall time required to complete pointing tasks (cf. Chapter 5—"Evaluation").

Initial visual search of on-screen mouse cursor is necessary before cursor manipulation.

After homing from the keyboard to the mouse, the task of cursor manipulation can be broken up into two sub-tasks. Since the mouse is a relative pointing device, the initial physical location of the device does not determine the on-screen position of the mouse cursor. Hence, the user is first required to visually locate the cursor (figure 1.2b).

[1]The total pointing time considered here is an aggregate of homing time from keyboard to pointing device, cursor manipulation time, and homing time from pointing device to keyboard.

We can use the model human processor presented by Card et al. [1986] to estimate that the time required for this initial visual search amounts to 470 milliseconds on average (Equation 1.1). This time is used solely for fixing the eye on the cursor, and does not involve its actual manipulation. Here, we assume that one single visual search is sufficient to find the on-screen pointer, which might not always be the case. Hence, this is just a lower bound estimate.

$$\begin{aligned} Visual\ Search\ Time &= Eye\ Movement\ Time + \tau_p + \tau_c + \tau_m \\ &= 230ms + 100ms + 70ms + 70ms \\ &= 470ms \end{aligned} \tag{1.1}$$

where,
τ_p = Perceptual Processor Cycle Time,
τ_c = Cognitive Processor Cycle Time,
τ_m = Motor Processor Cycle Time.

Large screen sizes result in need for mouse acceleration and clutching.

The second sub-task involves actual manipulation of the mouse cursor, to reach a given target location (figure 1.2c). This usually includes a quick ballistic movement towards the target, followed by a small corrective movement to reach the target. Although displays have grown larger in terms of size and screen space, the key physical characteristics of the mouse have remained the same. In order to minimize the device footprint, techniques like mouse acceleration are used, which make use of non-linear transfer functions for control-display gain[2]. Smaller input area also results in the need for *clutching*—the momentary act of lifting the device off the surface and repositioning it. Once

Hardware augmentations, absolute techniques, and direct manipulation attempt to overcome shortcomings of mice.

There have been several attempts to extend or replace the mouse. Hardware augmentations such as scroll wheels and multi-touch capabilities are used to add expressiveness to the device. Alternate input devices such as tablets and touch-screens eliminate the need for initial visual search of the mouse cursor. For example, Accot and Zhai [1999] has shown that the pen tablet reduces pointing time. However, being physical in nature, these too require a dedicated surface, and hence experience the effect of homing back

[2]Control-Display gain gives the relationship between motion of input device (e.g. mouse) to the respective movement of its displayed representation (e.g. cursor).

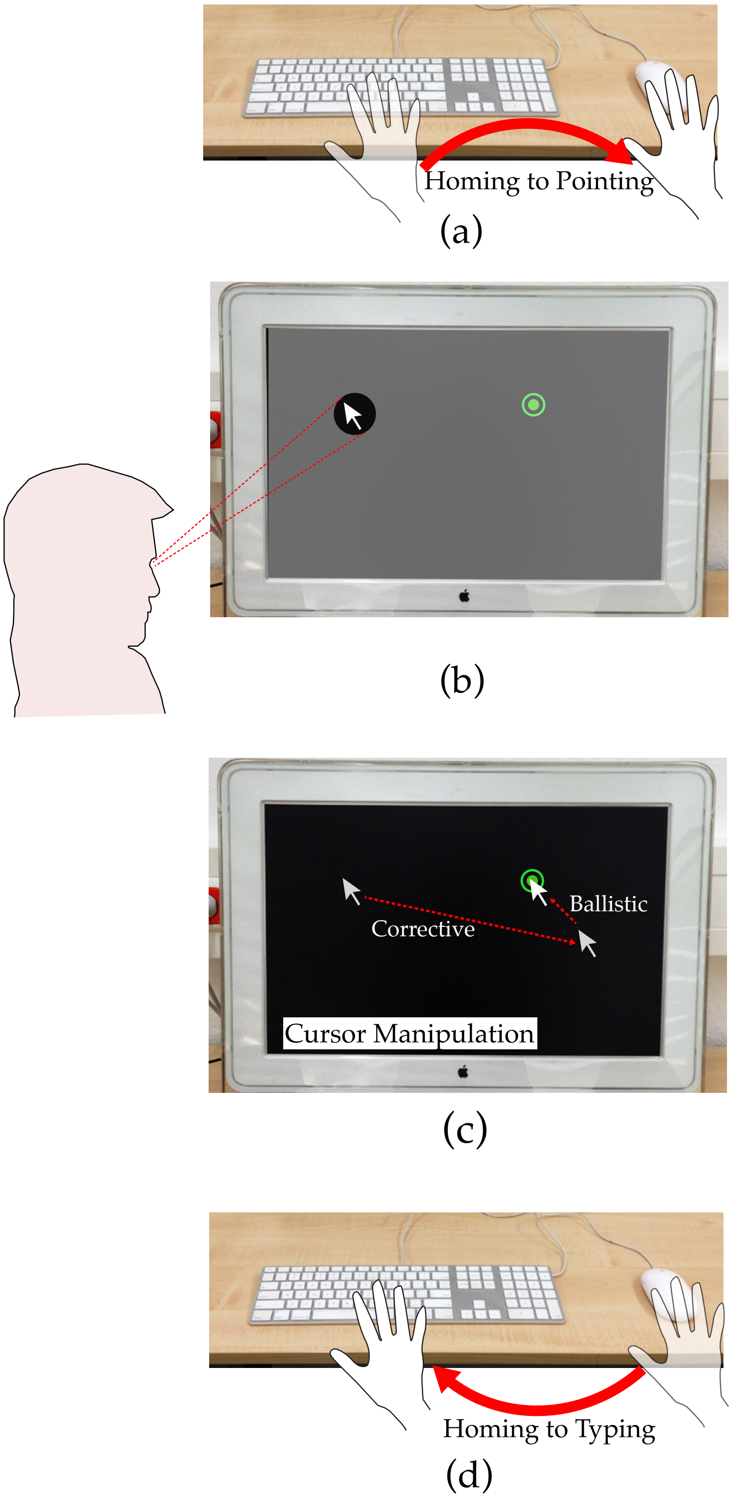
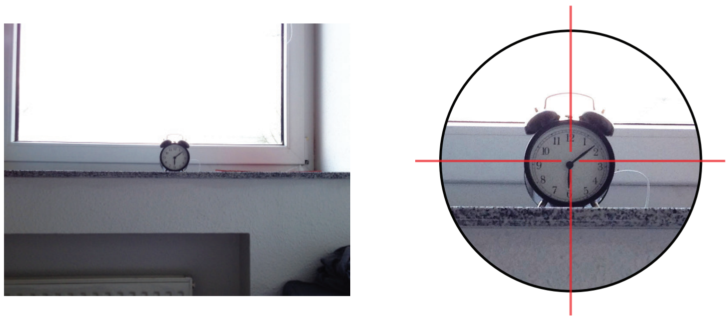


Figure 1.2: *Four phases of pointing with mouse. (a) Homing to mouse from keyboard. (b) Initial visual search of mouse cursor. (c) Cursor manipulation to reach a target—includes a ballistic and a corrective movement. (d) Homing back to resume typing on the keyboard.*

and forth between devices. Direct input devices such as touch-screens are fast and intuitive to use, but suffer from problems like occlusion and limited precision (Moscovich and Hughes [2008]). Reduced homing times have been achieved by using volume above the keyboard for mid-air pointing (Ortega and Nigay [2009]).

This thesis presents a new interaction technique, known as *'Sniper Pointing'*. The name has been derived from the metaphor of *'sniping'*—the act of aiming at a target from a long range. Sniper rifles are precision-rifles, equipped with multi-powered scopes, allowing for different levels of precision (Figure 1.3). Similar to this, the given interaction design also allows for multiple granularities of cursor manipulation. The technique combines absolute pointing with mid-air interaction, and provides multiple resolutions for pointing. The use of mid-air interactions aims at making the transition between typing and pointing seamless. Absolute techniques are used to eliminate the need for the initial search of on-screen pointers and cursor acceleration. The different resolutions of pointing are visualized with the aid of different cursors. A detailed explanation of this interaction technique is provided in chapter 3—"Interaction Design".

Sniper Pointing attempts to reduce pointing time through the application of mid-air interaction, absolute pointing technique and multiple layers of granularities.



(a) (b)

Figure 1.3: *Illustration of multiple zoom levels, showing resemblance of the interaction design to a sniper rifle. (a) A target is roughly aimed at initially. (b) Using the scope on the rifle, a precise aim can be acquired.*

1.1 Contributions

This thesis attempts to tackle problems faced by existing pointing devices by presenting a new interaction technique to reduce the overall time required for cursor manipulation. We survey and review existing literature to study the benefits and shortcomings of previous works done in the field. Moreover, a set of experimental analyses aid in design of the presented technique. The main contributions of this thesis are as follows:

- Design and implementation of an interaction technique for mid-air absolute pointing with multiple resolutions (Chapter 3).
- A user study to compare absolute and relative pointing and an investigation of the effects of input-space scale on the pointing time (Chapter 4).
- An evaluation of the proposed technique to study the effects on homing and pointing time (Chapter 5).

Chapter 2

Related work

The previous chapter has acknowledged the dominance of mouse as a pointing device, but it has been noted that it too is influenced by factors that negatively affect its performance, in terms of pointing speed. Other pointing devices apply different techniques and technologies to provide for on-screen object manipulation—for instance direct touch, absolute pointing, and mid-air interactions. This chapter discusses some techniques often used and provides information regarding some other works that are relevant to this thesis.

Previous works have compared various existing pointing techniques and devices, and some have introduced novel techniques to tackle specific problems faced by available technologies. In order to optimize the design and implementation of the proposed system, it is necessary to first study and analyse some relevant results previously obtained, and to summarize existing techniques, along with their advantages and disadvantages. This chapter highlights some of these works, divided into the following sections:

This chapter provide background information and a base, required for the design and implementation of *Sniper Pointing*.

1. Absolute and Relative Pointing—compares the two techniques and mentions an application of combination of both techniques.

2. Pointing with Multiple Granularities—illustrates the

use of different levels of pointing granularities by some systems.

3. Indirect versus Direct Manipulation—provides a description and comparison of the two, and the application of freehand interaction techniques.

4. Mid-Air Interaction—highlights the use of mid-air interactions in some existing systems.

   This chapter concludes with a *design space* summarizing the existing input techniques and positions *Sniper Pointing* relative to these techniques.

2.1 Absolute and Relative Pointing

Absolute pointing maps position in input space to corresponding position in output.

Cursor manipulation can be performed by devices using two different techniques of pointing—absolute and relative. *Absolute pointing* involves mapping the actual physical location of the input device to the on-screen location of the cursor. Pen tablets, touch screens and light pens are some examples of commercially available absolute devices.

Relative pointing uses techniques like cursor acceleration, and maps relative movement of input device.

Relative pointing, on the other hand, does not directly map the physical location of the pointing device to the cursor location. Instead, it maps the changes in the input position to changes in output position with a non-uniform transfer function (Buxton et al. [2002]). Consequently, the same movement in the input space may result in different movements of the cursor in output space, based on the transfer function. Mice, trackpads and joysticks are some examples of devices that apply relative pointing.

There has been an extensive amount of work done to compare these two techniques, and there have been input techniques designed, which attempt to combine the advantages of both these techniques.

2.1.1 Comparison of Absolute and Relative Pointing

Previous studies have compared absolute and relative techniques for various different scenarios. The two techniques have been compared in indirect as well as direct manipulation settings, and with different input devices such as tablet, mouse and pen-based input. Although the mouse is noted for being an optimal pointing device, studies have shown that absolute techniques can exhibit excellent performance, and have also shown absolute devices to perform better than their relative counterparts.

MacKenzie et al. [1991] applied Fitts' Law to compare mean movement times achieved by the mouse, tablet and trackball, for pointing and dragging tasks (Figure 2.1a). During the analysis of mean movement time for the devices, they observed that the stylus outperformed the mouse and trackball in both, pointing and dragging, tasks. For the pointing tasks, the movement times with the mouse (average time = 674 ms) and the tablet (665 ms) were comparable. However, for the dragging tasks, the movement time with the tablet (802 ms) was considerably lesser than that with the mouse (916 ms). In both cases, the trackball was significantly slower than the mouse and the tablet. The results from the studies performed by MacKenzie et al. [1991] illustrate that it is possible to use absolute pointing with performance comparable to that achieved by relative pointing using mouse.

Absolute pointing with stylus outperformed relative pointing with mouse and trackball.

Forlines et al. [2006b], at the Mitsubishi Electric Research Laboratories, studied direct pen input using absolute and relative techniques. Standard Fitt's 2D target acquisition tasks were performed by participants, on a screen capable of direct input. The study was performed using two different screen sizes—a tablet computer screen positioned horizontally; and a large wall-sized display positioned vertically. The results showed that absolute mapping outperformed relative mapping for small screen of the Tablet PC. However, relative pointing performed better than absolute on the large wall-sized display. Figure 2.1b shows the results obtained by for the study with the Tablet PC. The lower selection time in all conditions can be noted.

Absolute outperformed relative for small screens; relative pointing performed better for wall-sized display.

(a) MacKenzie et al. [1991] (b) Forlines et al. [2006b]

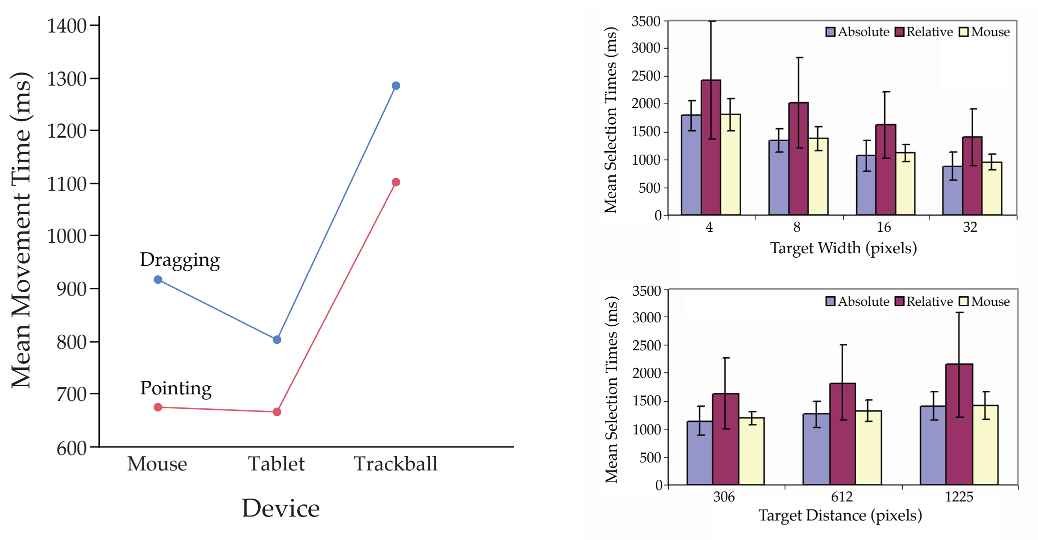


Figure 2.1: *A comparison of absolute and relative techniques. (a) MacKenzie et al. [1991] compared mean movement times for different devices and task. The tablet is seen to be faster than the mouse and trackball. (b) Forlines et al. [2006b] compared absolute and relative pen input for different target widths and target distances (Graphs are reproduced from the original papers).*

2.1.2 Combining Absolute and Relative Techniques

Absolute pointing for close-by targets and relative for targets further away.

Hybrid Pointing (Forlines et al. [2006a]) combines absolute and relative pointing techniques for input to wall-sized displays. The interaction design attempts to make use of the results obtained by the studies performed by Forlines et al. [2006b] to compare absolute and relative devices. Absolute pointing is used for targets that are closer to the user. To access out-of-reach targets, users can switch to relative pointing, in order to minimize physical movement. Figure 2.2 illustrates a series of manipulations and the process of switching between absolute and relative pointing.

ARC-Pad (McCallum and Irani [2009]) is another device that uses a combination of absolute and relative techniques, as a method to use a mobile touchscreen as an input device to large-sized displays. Fluid switching between the absolute and relative pointing modes is provided in the de-

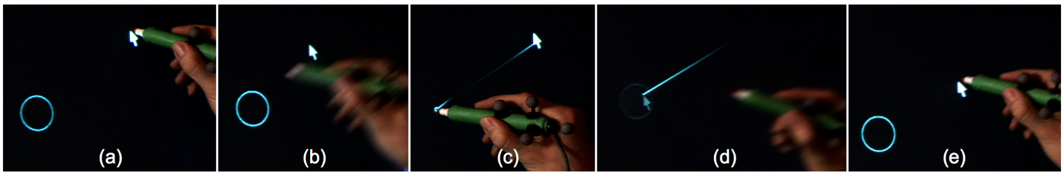


Figure 2.2: *Hybrid Pointing allows the user to switch between working in absolute or relative input modes (a) By default, the pen works in absolute input mode, (b) a quick click on the circular trailing widget switches to (c) relative mode. (d) Lifting the pen a certain distance away from the display while in relative mode switches back to (e) absolute mode. (Figure from Forlines et al. [2006a])*

vice. In the absolute mode, the landing position of user's finger on the screen determines the cursor position on the display. After the finger lands on the screen, sweeping the finger across causes relative movement of the cursor. Using a combination of these techniques, ARC-Pad also introduces an element of granularity in pointing, which is explained in the following section.

2.2 Pointing with Multiple Granularities

The granularity of pointing is the level of accuracy that can be provided by the pointing device. Traditional devices like the mouse offer fine-grain pointing, in that they allow users to select small target sizes accurately. On the other hand, the precision of devices like touchscreens is limited by the size of the user's fingers, and allows for only a coarser pointing granularity. Recently, there has been the application of multiple levels of granularities, to provide several levels of precision.

ARC-Pad uses mobile touchscreen for cursor manipulation on large-displays. Absolute pointing for coarse manipulation, and relative for finer granularity.

ARC-Pad, introduced in the previous section, compensates for the small size of mobile touchscreens, in comparison to the size of the display, by facilitating cursor manipulation using multiple granularities. This is achieved by using a combination of absolute and relative pointing techniques. A tap and release action enables absolute pointing, causing the cursor to jump to the corresponding position on the display. This is a technique to provide for coarse pointing to approximate positions. Dragging a finger across the screen

Figure 2.3: *ARC-Pad: Illustration of pointing modes. (Left) Cursor is initially at the top right corner. (Center) Tapping anywhere with ARC-Pad causes the cursor to instantly jump across the screen. (Right) For accurate positioning the user can clutch and slide the finger.*

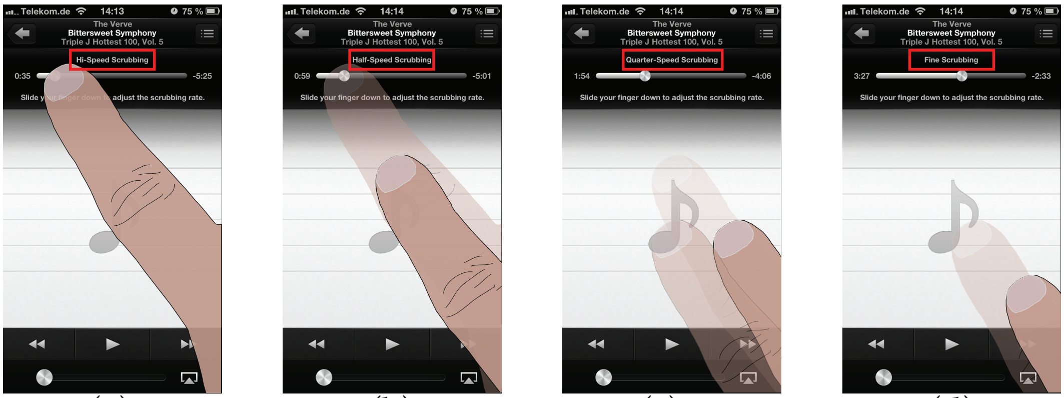
activates relative pointing, causing the cursor to follow the finger movement. Relative movement allows for fine pointing and provides precise control over the cursor.

Scrubbing on iPhone allows for different speeds of slider manipulation.

The Apple iPhone provides for *scrubbing* with multiple granularities in its *Music* and *Video* applications. The playback position can be altered by moving the slider in the horizontal direction. While doing so, four different granularities are provided in order to allow for precise positioning—hi-speed, half-speed, quarter-speed, and fine scrubbing (figure 2.4). When the user initially touches the slider, hi-speed scrubbing can be performed by directly manipulating the slider position along its horizontal axis. By dragging the finger vertically downwards, the granularity can be adjusted, as required. The level of granularity is determined by the vertical location of the finger on the screen—as the finger moves lower towards the home button, the granularity gradually becomes finer, in discrete increments.

It is evident that the use of multiple granularities can be successfully applied to tackle the problem of low resolution, inherent in input devices like the bare finger. By using different granularities, it can be possible to achieve higher accuracy, in cases where it would otherwise not be possible.

Note that while manipulation with multiple granularities in the iPhone is provided using direct manipulation of on-screen objects, the ARC-pad provides cursor manipulation



(a) (b) (c) (d)

Figure 2.4: *Scrubbing on the iPhone. Use of different scrubbing speeds for controlling playback position. (a) Initial touch on slider to activate fast scrubbing. (b) Finger moved down to switch to half-speed. (c) Finger moved further down to activate quarter-speed and moved to right to shift slider position. (d) Finger moved downwards to enable fine scrubbing.*

using indirect techniques. The choice of indirect or direct manipulation is context-sensitive and each of the two techniques have their own advantages and shortcomings. A comparison of indirect and direct manipulation follows.

2.3 Indirect vs. Direct Manipulation

There are two main techniques through which objects in the output space can be manipulated—direct and indirect. In direct mapping, input and output are provided in the same physical space. This is inherent in devices like touchscreens, light pens, laser pointers and other direct manipulation devices. On the other hand, devices that use indirect mapping do not provide input in the same physical space as the output (Hinckley [2003]). For instance, the mouse acts as an indirect pointing device in a standard desktop configuration, where a horizontal input space is mapped to a vertical output space. Since both techniques have their own advantages and disadvantages, their usage is highly dependent on the application area.

Direct manipulation is performed in same physical space; Indirect mapping has different input and output spaces.

Direct input faster; but screen occlusions are a problem.

Schmidt et al. [2009], in their comparison of direct and indirect multi-touch inputs, concluded that direct input was faster as compared to indirect techniques. However, direct input devices were noted to suffer from the effect of screen occlusion, which degrades user experience. The authors also note other advantages indirect input devices have, including the possibility of interacting with the output screen from a distance, and accommodation of a single input space for multiple output displays.

Directly pointing the finger to the screen is suitable for mixture of typing and pointing tasks, but causes fatigue during prolonged use.

In the analysis of the index finger as a pointing device, Gokturk and Sibert [1999] compare direct finger pointing, to the screen, with pointing using indirect mouse and trackpoint. The results from the study show that direct finger pointing is significantly faster than the trackpoint, but only marginally slower than the indirect mouse. The authors state a likelihood of natural finger pointing being a superior interaction technique for cursor manipulation, when keyboard and pointing tasks are mixed. However, qualitative results from the study also show that keeping the finger raised over a period of time resulted in fatigue, which is a disadvantage of using direct pointing techniques with vertical displays.

Occlusion caused due to 'fat finger'. Small targets are hard to acquire with direct finger manipulation.

Hinckley [2003] has summarized some aspects about the finger as a direct input device, and has pointed out the medium to high level of occlusion caused due to the 'fat finger' problem. Additionally, this low resolution of the finger also prohibits the effective acquisition of small targets during direct manipulation, limiting the minimum size of targets for devices that apply such a technique.

Combination of free-hand interactions and indirect manipulation can be advantageous. This is possible by providing over-the-surface interactions.

In summary, although using the bare hand as a pointing device can be fast and intuitive, direct manipulation with the finger can lead to occlusion of the screen, fatigue during prolonged use, and difficulty in acquisition of small targets. By combining free-hand interactions with indirect manipulations, it can be possible to combine the advantages of both these techniques. Free-hand interactions have traditionally been employed on the surface, by touch-sensitive devices like phones and tablets, limiting them direct manipulations. By moving over to interactions over the surface, in a three-dimensional volume, it is possible to achieve this combination of indirect and free-hand manipulation.

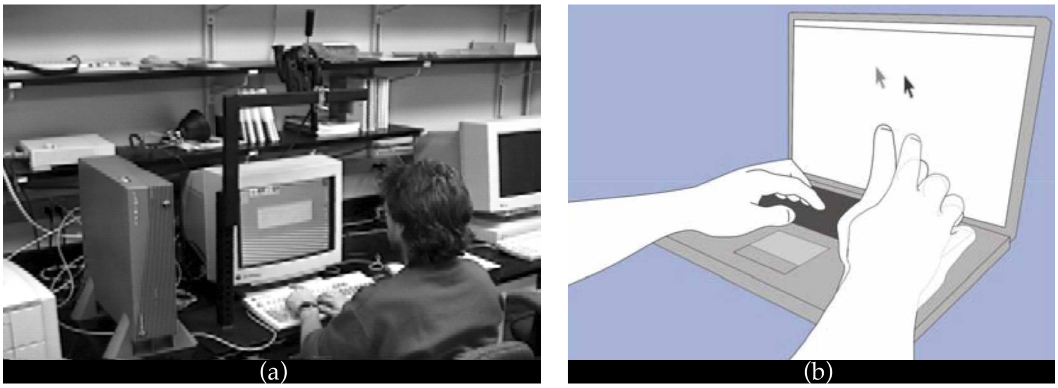


Figure 2.5: *Mid-air pointing on small screens. (a) FingerMouse (Mysliwiec [1994]) was one of the first systems which used this technique. (b) Illustration of the AirMouse system (Ortega and Nigay [2009]).*

2.4 Mid-Air Interaction

While most input techniques are based on interactions on a rigid surface, mid-air interaction adds a third dimension, and allows input within a given volume. This allows for both—on-surface interaction and above the surface object manipulation. Two different settings where mid-air techniques have been applied to provide for interaction and cursor manipulation are in the standard desktop setting, and in environments where large wall-sized displays are used.

2.4.1 Mid-Air Interactions with Desktop-Sized Screens

Over the surface pointing techniques have been developed for desktop settings. FingerMouse (Figure 2.5a), developed by Mysliwiec [1994], was one of the first systems to apply mid-air pointing techniques for free-hand interaction. It proposed colour segmentation techniques to detect hand positions and fingertip tracking through principal-axis based approaches. Using such methods, FingerMouse was able to provide for free-hand pointing techniques, without the need for additional accessories like specialized gloves.

FingerMouse introduced freehand pointing in mid-air for desktops; Color segmentation used for hand detection.

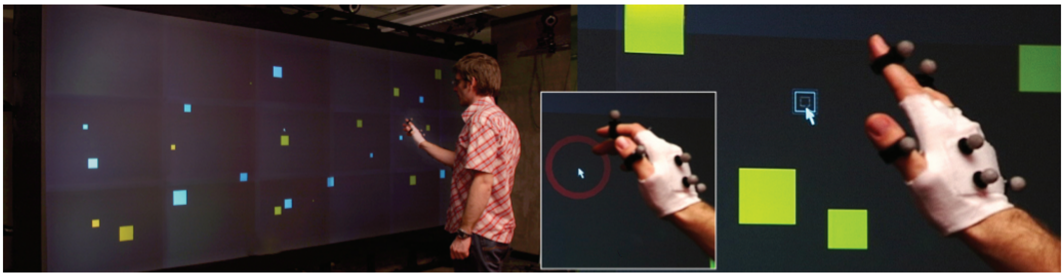


Figure 2.6: *Mid-air interactions on a wall-sized display. Vogel and Balakrishnan [2005] illustrate freehand pointing and clicking on such displays.*

AirMouse uses volume above the keyboard for freehand relative pointing.

AirMouse (Figure 2.5), presented by Ortega and Nigay [2009], developed a mid-air pointing system, which operated in a small volume above the keyboard surface. The mid-air system was used for two-dimensional and three-dimensional pointing and adopted relative techniques, very similar to that applied by a trackpad, for cursor manipulation. The problem of large homing times was partially addressed by the system, but overall pointing times were slower compared to the mouse.

FlowMouse attempts to improve input data accuracy

Wilson and Cutrell [2005] introduced the FlowMouse technique for mid-air pointing. While other techniques used absolute hand and finger tracking to perform transformations, FlowMouse used the notion of 'flowing' to obtain input data with higher accuracy. To do this, hand motion was modelled using optical flow techniques and a capacitive touch sensor was used to enable and disable interaction. Although this technique provides an insight into possibilities for improving input data, and can be useful for relative pointing, it can not be applied for absolute pointing, which depends purely on the physical location of input devices for input–output transformations.

It is evident from previous works that potential for performing mid-air cursor manipulation for desktop environments, using the volume above the keyboard, has been identified by researchers and attempts have been made to provide the same, however, with only limited success.

2.4.2 Mid-Air Interactions for Wall-Sized Displays

Mid-air interactions have also been applied in scenarios involving large, wall-sized displays. The colossal size of these output screens makes the use scenario different from that in desktop environments. Distance between the users and the screen, and high resolutions of these displays play an important role in determining the requirements for interaction in such environments.

Nancel et al. [2011] presented a system which allows panning and zooming using free-hand gestures, for very large displays. The work highlights the use of bimanual interactions and also points out the disadvantage of clutching—an additional hand movement, necessary while using devices like the mouse.

Bimanual gestures used for panning and zooming on wall-sized displays.

Vogel and Balakrishnan [2005] illustrated the application of distant freehand pointing and clicking on such displays. The research illustrates the use of gestural pointing techniques to provide for cursor manipulation. The lack of kinesthetic feedback is also addressed and the authors use visual and auditory feedback in order to compensate for this. Vogel and Balakrishnan [2005] point out the drawbacks of using hand-held, physical devices to perform pointing actions and highlight the advantages freehand pointing can have. One of the main contributions of this work is the presentation of different pointing and clicking techniques in settings involving large-sized displays.

Distant free-hand pointing and clicking techniques have been developed for large displays; highlights disadvantages of hand-held devices used for pointing.

It can be observed from these works that the importance of freehand pointing and cursor manipulation has been noted and it has shown to be advantageous in different settings, like desktop environments as well as with wall-sized displays.

Although previous works have developed various solutions and techniques for providing mid-air interactions, there is still room for further improvement and more efficient techniques, which can make such interactions more user-friendly while exhibiting high performance. By reducing homing time between typing and pointing, switching between devices can be made less taxing on users, and overall time required for performing pointing can be reduced. Additionally, mid-air interactions have the advantage that they do not require a specific physical space, and hence can reduce the device footprint. Absolute pointing techniques eliminate the need for a visual search, of the cursor, before each pointing operation, thereby reducing the time required for pointing. For prolonged usage, it has been observed that indirect mapping is preferred. By combining these input techniques with a novel interaction design, there is a possibility of improving pointing.

2.5 Design Space of Input Devices

Summarizing the related works performed in this field allows for the definition of a design space of input devices. The space is a cross product of 'Mapping Functions' $\in$\{*Absolute*, *Relative*\}, 'Pointer Acceleration' $\in$\{*Accelerated*, *Unaccelerated*\} and 'Granularities' $\in$\{*Coarse*, *Fine*\}. Traditional devices such as mouse and tablet do not provide for different granularities of pointing. ARC-Pad (McCallum and Irani [2009]) uses absolute mapping for coarse pointing and relative mapping with cursor acceleration for fine-grain pointing. The proposed interaction technique (explained in Chapter 3—"Interaction Design") is designed to implement absolute pointing without acceleration for both, coarse and fine, pointing. However, another possible interaction technique, similar to ARC-Pad, is feasible. In this alternate approach, the coarse pointing layer could use absolute pointing without any acceleration, and the fine pointing layer could use relative pointing with cursor acceleration. The comparison between these alternatives is out of the scope of this thesis.

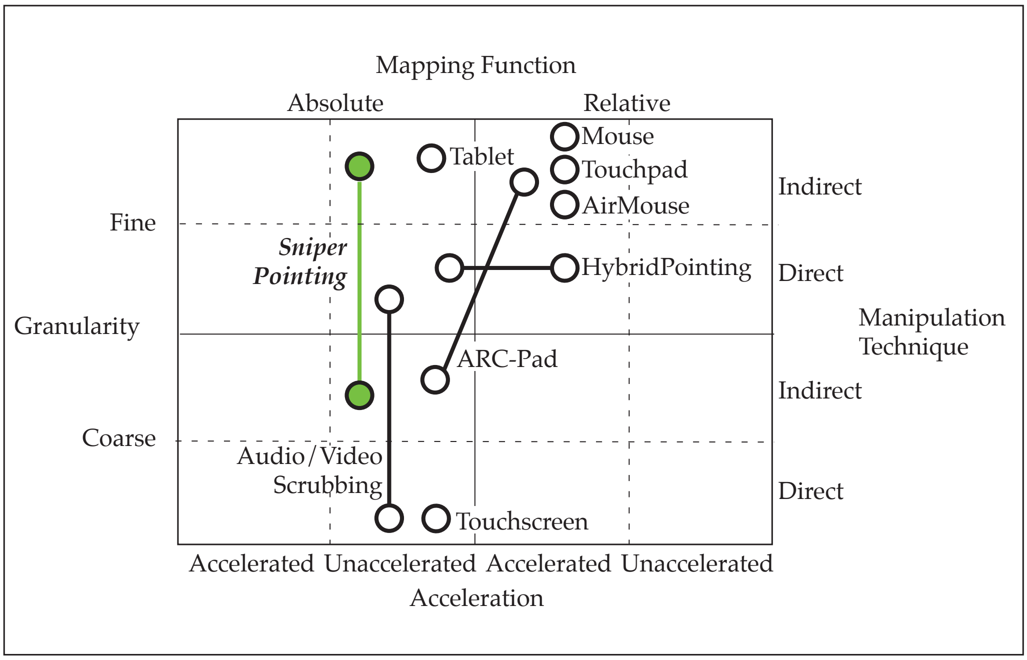
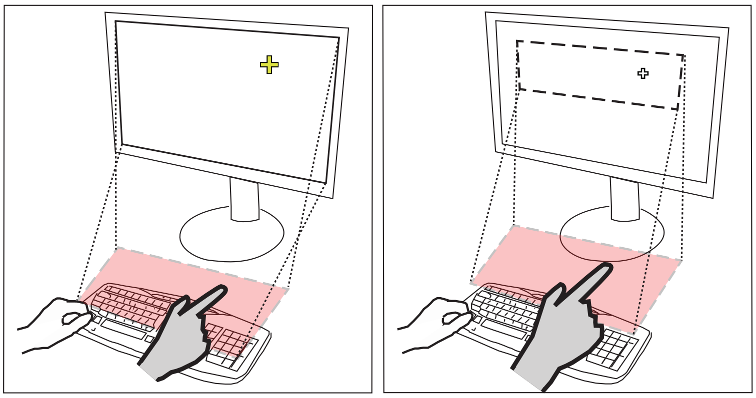


Figure 2.7: *Various input devices visualized in the design space. Devices are categorized by the use of absolute and relative techniques, cursor acceleration, and granularities of pointing.*

Chapter 3

Interaction Design



(a) Coarse Pointing (b) Fine Pointing

Figure 3.1: *Interaction using the two different pointing granularities and mapping of input space to screen space. (a) Coarse Pointing—The entire screen area is mapped to the input area. (b) Fine pointing—A small portion of the screen is mapped to the input area, to allow finer manipulation.*

There is a possibility for new pointing techniques

Work previously done in the field of pointing and cursor manipulation attempts to achieve accuracy and speed through the use of various different techniques, which can be visualized with the aid of a design space (Figure 2.7). A closer look at this visualization makes it possible to observe some other plausible combinations of techniques that could be further developed.

Novel interaction technique presented, combining absolute pointing with mid-air interactions. Allows for cursor manipulation with multiple levels of granularities.

The presented mid-air interaction design makes use of a combination of absolute pointing technique and multiple pointing granularities in an attempt to facilitate fast as well as accurate pointing. This chapter provides details about the interaction design and some implementation details vital to the interaction. Relevant design principles, that are important for the interaction, are highlighted. The interaction is explained in detail along with a storyboard, and a mathematical explanation is provided. Some important aspects relevant to the implementation of the interaction design are highlighted at the end of this chapter.

3.1 Design Principles

In order for a device to be successful, it is important to first outline a set of design principles which need to be fulfilled. These serve as goals for the proposed device and highlight the basic characteristics that should be exhibited. A pointing technique that could be on par with the mouse should be successful in the following criteria:

- Lower the homing time required to switch between typing and pointing.
- Allow fast pointing to targets on the screen.
- Pointing accuracy should be comparable to mouse input.
- Allow prolonged use without much fatigue.
- Minimum or no screen occlusion.

3.2 Storyboard

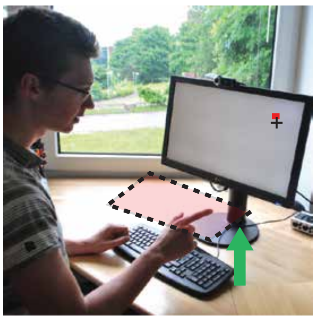
Storyboard illustrates layered interaction for multiple pointing granularities.

A sample usage scenario of Sniper Pointing can be illustrated through a storyboard (Figure 3.2). This highlights how a user can employ the different pointing layers in order to manipulate the cursor and position it on screen, as

The user is initially typing on the keyboard

A target on the screen has been sighted

Coarse manipulation of cursor is activated by raising finger slightly.



Raising finger to a higher level allows fine manipulation of cursor

The user returns to the keyboard to continue typing

Figure 3.2: *An example usage scenario of Sniper Pointing. This illustrates the usage of coarse pointing for quickly placing the cursor at an appropriate location, and fine pointing to accurately acquire the given target.*

desired. This scenario illustrates the application of both pointing layers—coarse as well as fine. It should be noted that in case only an approximate location needs to be acquired, that is, when the target is large in size, the coarse pointing action can be sufficient to do so.

3.3 Detailed Explanations

The presented technique proposes the use of absolute pointing for both, coarse and fine, granularities. Any given location on the input space corresponds to a precise loca-

Absolute pointing with indirect mapping has been used. Allows usage of either of the hands.

tion of the cursor. The mapping of the input–output space is indirect, in that the vertical display is mapped to a horizontal space, parallel to the surface of the keyboard. This helps in elimination of occlusion and reduces the fatigue caused when arms are held at a greater height, without support. When users' hands are placed on the keyboard, typing gains focus and pointing is disabled to avoid distraction. Pointing is activated as soon as the user raises his index finger[1] slightly above the keyboard. The core aspect of the interaction is the layered design of the input space. The absolute position of the finger on the horizontal area is not the sole factor that effects the cursor position. Instead, the vertical height of the finger, above the keyboard, also effects the mouse location and the granularity of mouse manipulation.

Layered Interaction

Coarse layers allows for fast cursor positioning and fine layer enables accurate target acquisition.

The interaction technique divides the volume above the desk into two different layers—one used for coarse, quick pointing; the other for fine grain, accurate pointing (Figure 3.3). The first layer, directly above the keyboard, is used for coarse pointing. This is enabled on raising the finger slightly, and allows for fast cursor manipulation. In the coarse layer, the entire screen area is mapped directly to the input surface area, allowing cursor movement across the entire screen. Coarse pointing can be used to quickly position the cursor at an approximate location, or to acquire large-sized targets. Once this action has been performed, raising the finger higher causes the pointing granularity to become much finer, allowing accurate cursor manipulations. In this case, only a small portion of the screen maps to the entire input area, leading to higher pointing resolution. Larger finger movements in the fine layer, result in comparatively smaller change in cursor position, hence allowing precise positioning of the cursor and acquisition of small targets.

[1] For this thesis, the tip of the index finger is used for input. Although it is possible to use other properties such as the tip of the thumb or the centre of mass of the hand, this investigation is beyond the scope of this thesis.

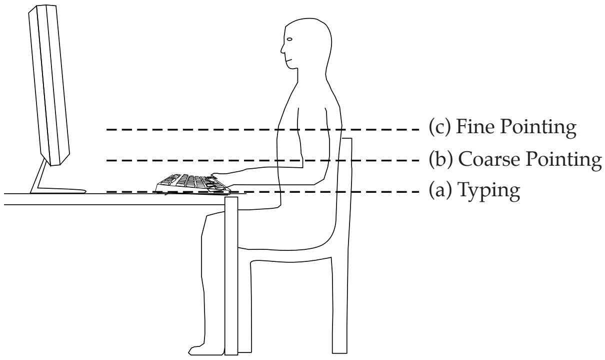


Figure 3.3: *Layers of pointing granularities. A side-view illustrating the three different layers presented in the interaction design. (a) At the keyboard level, typing is disabled. (b) Coarse pointing as activated as soon as the finger is raised above the keyboard. (c) Raising the finger higher enables fine pointing.*

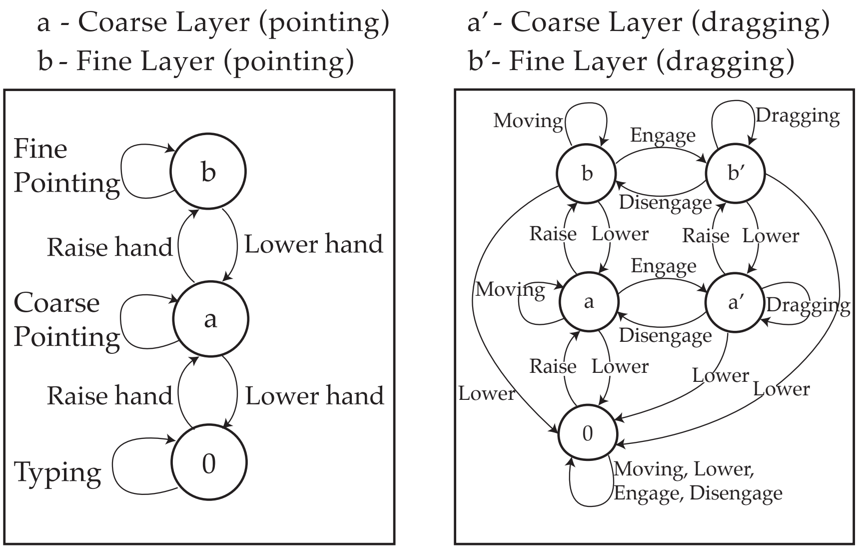


Figure 3.4: *A state diagram showing the transition between different stages. (Left) Cursor manipulation in different layers. (Right) Extended form to emulate mouse movement, including dragging.*

Targeting transitions are formulated with a state machine (Figure 3.4—left). Note that the overall interaction can be expanded to emulate mouse movement (Figure 3.4—right), but design of full mouse emulation is outside the scope of this thesis.

Cursor Manipulation

Figure 3.5 provides an illustration of cursor manipulation using *Sniper Pointing*. While performing coarse pointing, moving the pointing finger to a new position above the keyboard results in the cursor being repositioned in the corresponding location on screen. On switching to fine pointing, since the input volume is mapped to only a small portion of the screen (shown by the bounding box), moving the finger a large distance results in a smaller displacement of the cursor.

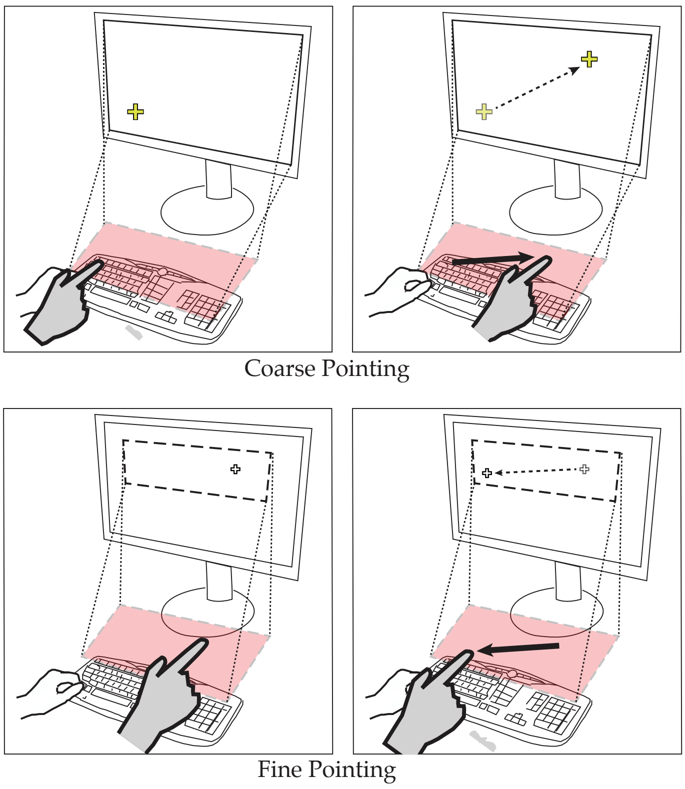


Figure 3.5: *Cursor manipulation using Sniper Pointing.*
(a) User pointing in the coarse granularity layer. (b) Moving the pointing finger in coarse layer causes cursor to jump to corresponding location on screen. (c) User raises his hand to the fine layer. (d) Finger movement in fine granularity layer results in smaller displacement of cursor.

3.4 Mathematical Explanations

This section describes mathematical transformations used to map the position of users' hands, in three-dimensional input space, to the cursor position, in screen space. The coordinate system used is shown in Figure 3.6. Here, lower case variables refer to desk (input) coordinates and upper case variables to screen-space coordinates.

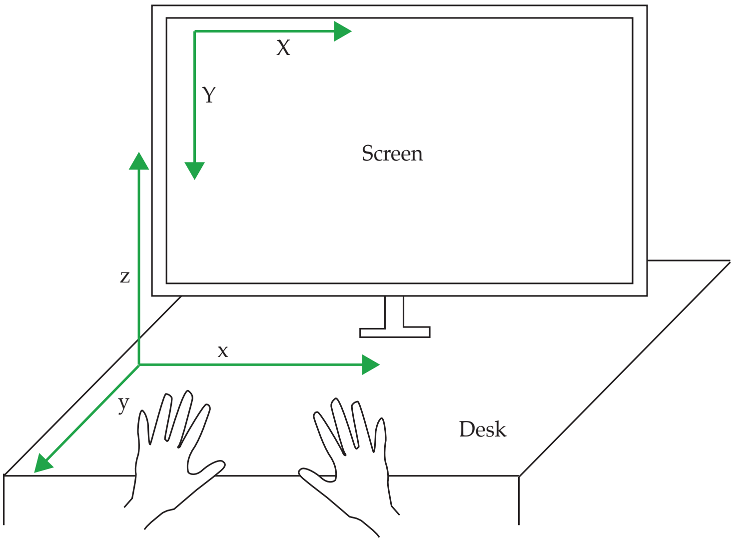


Figure 3.6: *The coordinate systems used for screen area and for input volume*

The input volume is mapped to the screen area (output) in the following way:

- The input volume is divided into two layers by an xy plane. The lower layer is used for coarse pointing and the upper for fine pointing.
- The coarse pointing layer provides quick, coarse absolute pointing. The user points his finger in order to position the cursor at an approximate desired location. Bigger targets can be effectively acquired by using this layer exclusively.
- The fine pointing layer provides higher accuracy,

once the approximate area is located by coarse pointing. The user raises his finger to a higher z-coordinate in order to activate this mode. In this layer, large displacements of the finger result in only minor displacement of the cursor on screen, therefore increasing the precision of the cursor movement. This can be used for accurately locating smaller targets.

3.4.1 Coarse Pointing Layer

Entire screen mapped to input area

During cursor manipulation in the lower (coarse) layer, screen coordinates (X, Y) are linearly mapped onto the desk coordinates (x, y). The entire screen area is mapped to the input area available above the keyboard (Figure 3.1a).

p_x, p_y specify the ratio of output–input space.

The ratio of screen to input space is specified by the constant factors p_x and p_y, for the x and y axes respectively, and are mathematically defined in equation 3.1. Equation 3.2 represents the transformation from input space (x, y) to screen coordinates (X, Y), in extended coordinates.

Definition: p_x and p_y

p_x AND p_y:

$$p_x = \frac{Width_{screen}}{Width_{input}} \quad (3.1a)$$

$$p_y = \frac{Height_{screen}}{Height_{input}} \quad (3.1b)$$

$$\begin{bmatrix} X_0 \\ Y_0 \\ 1 \end{bmatrix} = \begin{bmatrix} p_x & 0 & 0 \\ 0 & p_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix} \quad (3.2)$$

where,
(x_0, y_0) is any point in the input space, and
(X_0, Y_0) is the corresponding point in the screen space.

3.4.2 Fine Pointing Layer

In the upper layer, a 1–1 mapping of the screen to input space is used. Only a fraction of the screen area, equal in size to the input area, is mapped to the entire input space (Figure 3.1b), allowing for finer-grain cursor manipulation.

A fraction of the screen area is mapped to the entire input area

(x, y) represents the initial coordinates in the input space when the hand switches to the fine pointing layer; (x_n, y_n) represent the final input coordinates after manipulation in the fine layer is performed. Equation 3.3 represents the transformation from input space (x, y) to screen coordinates (X, Y), in extended coordinates.

$$\begin{bmatrix} X_n \\ Y_n \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & (x_n - x_0) \\ 0 & 1 & (y_n - y_0) \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x & 0 & 0 \\ 0 & p_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix} \tag{3.3}$$

Alternatively, a 1–<1 mapping could be possibly used, in order to achieve even higher accuracy. In this case, a smaller area of the screen space corresponds to the given input space.

The amount of gain in accuracy can be specified by the constant factors k_x and k_y, for the x and y axes respectively, and can be mathematically defined as per equation 3.4. The transformation from the input space to output coordinates is given by equation 3.5.

k_x AND k_y:

$$k_x = \frac{Width_{input}}{Width_{output}} \tag{3.4a}$$

$$k_y = \frac{Height_{input}}{Height_{output}} \tag{3.4b}$$

Definition: k_x and k_y

$$\begin{bmatrix} X_n \\ Y_n \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & k_x.(x_n - x_0) \\ 0 & 1 & k_y.(y_n - y_0) \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x & 0 & 0 \\ 0 & p_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix} \tag{3.5}$$

where,
(x_0, y_0) is any point in the input space, and
(X_0, Y_0) is the corresponding point in the screen space.

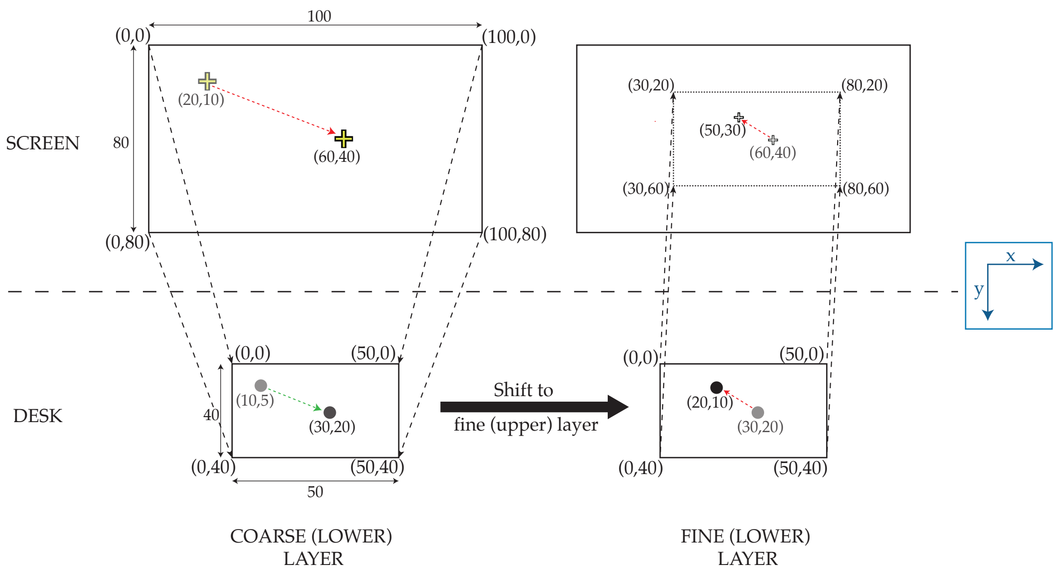


Figure 3.7: *An example to illustrate the mapping of input–output space. Displacement of the finger from input position (10,5) to (30,20) in the course (lower) layer causes the cursor to shift from (20,10) to (60,40). On lifting the finger to the fine (upper) layer, displacement of the finger from (30,20) to (20,10) causes the cursor to shift from (60,40) to (50,30).*

3.5 Example Transformation

The transformation from input to output coordinates, based on the given mathematical model, can be illustrated with the help of an example (Figure 3.7).
In the given example, the screen dimensions are 100 × 80 points and the corresponding input dimensions are 50 × 40 points. We can calculate the p_x and p_y values as:

$$p_x = \frac{Width_{screen}}{Width_{input}} = \frac{100}{50} = 2.0 \tag{3.6a}$$

$$p_y = \frac{Height_{screen}}{Height_{input}} = \frac{80}{40} = 2.0 \tag{3.6b}$$

The sequence of manipulations performed and the corresponding transformations are as follows:

1. The original position $(10, 5)$ of the finger, over the desk, can be mapped, using equation 3.2, to the initial cursor position using the coarse pointing transforma-

tion:

$$\begin{bmatrix} X_0 \\ Y_0 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 10 \\ 5 \\ 1 \end{bmatrix} = \begin{bmatrix} 20 \\ 10 \\ 1 \end{bmatrix} \tag{3.7}$$

2. The finger is moved in the course layer to a new position $(30, 20)$. This results in the movement of the cursor from the original position $(20, 10)$ to a new position on screen, given by:

$$\begin{aligned} \begin{bmatrix} X_0 \\ Y_0 \\ 1 \end{bmatrix} &= \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 30 \\ 20 \\ 1 \end{bmatrix} \\ \Rightarrow \begin{bmatrix} X_0 \\ Y_0 \\ 1 \end{bmatrix} &= \begin{bmatrix} 60 \\ 40 \\ 1 \end{bmatrix} \end{aligned} \tag{3.8}$$

3. The finger is now raised in order to shift to the fine (upper) layer. Here, a 1–1 mapping of input–output space is used ($k_x = 1; k_y = 1$). At the time of raising the finger, the initial input position (x_0, y_0) is equal to the last position of the finger in the coarse layer $(30, 20)$. Corresponding to this, the cursor is initially positioned at $(60, 40)$ (from equation 3.8).

4. Movement of the finger to the final position $(20, 10)$, causes the cursor to be relocated in accordance with equation 3.3, and can be given by:

$$\begin{aligned} \begin{bmatrix} X_n \\ Y_n \\ 1 \end{bmatrix} &= \begin{bmatrix} 1 & 0 & (20-30) \\ 0 & 1 & (10-20) \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 30 \\ 20 \\ 1 \end{bmatrix} \\ \Rightarrow \begin{bmatrix} X_n \\ Y_n \\ 1 \end{bmatrix} &= \begin{bmatrix} 1 & 0 & -10 \\ 0 & 1 & -10 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 60 \\ 40 \\ 1 \end{bmatrix} \\ \Rightarrow \begin{bmatrix} X_n \\ Y_n \\ 1 \end{bmatrix} &= \begin{bmatrix} 50 \\ 30 \\ 1 \end{bmatrix} \end{aligned} \tag{3.9}$$

3.6 Implementation

This section outlines some aspects of implementation important to the interaction. These help in making the proposed interaction more intuitive for the users and attempt to reduce the effect of factors such as fatigue and lack of feedback, which can play a crucial role in mid-air interaction techniques.

3.6.1 Separation of Layers

Effective separation of the two pointing layers allows the interaction to be intuitive, and helps in preventing confusion and errors. Some techniques used in order to make the separation efficient are:

Minimal separation between typing and pointing

- The coarse pointing layers starts directly above the keyboard. Making it closer to the desk allows for seamless switching between typing and pointing. Additionally, this allows users to perform coarse pointing without lifting their hands off the surface, hence minimizing fatigue.

Sufficient thickness of layers

- Each layer has sufficient thickness. This prevents for erroneous switching between coarse and fine pointing when the finger is unintentionally raised by a small height.

Minimal raising of arms required

- In order to further reduce fatigue caused by raising the arm, the vertical distance at which the fine pointing layer begins is made to be small enough in order to allow for continuous usage for a period of about 15 to 20 minutes, in accordance to the application of biomechanical principles to reduce stress on the shoulder (cf. Marras [2006]).

Fluid switching between the pointing layers

- There is no gap between the two layers. The fine pointing layer is stacked directly above the coarse, without any space separating them. The presence of a gap between the two layers was found to be confusing for users due to the lack of feedback when the finger is in this empty space.

3.6.2 Cursor Visualization

Two different cursors are used, in order to visualize the two layers (Figure 3.8). The cursor used for the coarse layer is larger in size, indicating lower resolution. Consecutively, when the user switches to fine pointing, the cursor becomes smaller in size, indication finer granularity. This acts as a tool to provide feedback about the current pointing layer being used.

Two distinct cursors are used to provide feedback.

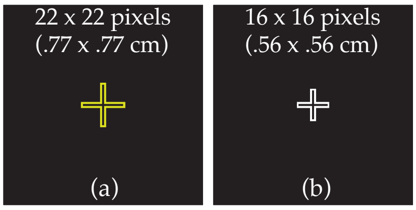


Figure 3.8: *Cursor visualizations used. (a) large yellow cursor for coarse layer (b) small white cursor for fine layer.*

3.6.3 Two-handed Interaction

The interaction is designed in such a way that the use of both hands is supported. At any given time, either the left or the right index finger can be used for pointing, and this can be dynamically switched according to preference or task. This is possible since free-hand pointing is used and there is no additional physical device responsible for pointing. The relative heights of the two hands, above the desk, is used to determine whether the left or the right index finger is used for pointing at the given moment. The hand that is raised higher is, by default, recognized as the pointing hand.

Usage of either hands is supported. Hand raised higher above the keyboard detected as the pointing hand.

3.6.4 Cursor Movement along Screen Edges

While pointing in the coarse layer, moving the finger to an edge of the keyboard causes the cursor to be positioned

Easy access to screen edges, which have infinite width, is provided.

along the corresponding screen edge. Additionally, when the finger is moved slightly beyond the edge, outside the input region, instead of moving the cursor out of the screen (rendering it invisible), it is made to retain its position along the edge. This allows for quick access to the screen edges, which have infinite width according to Fitt's laws, even when the finger is moved slightly outside the input region. This allows for easy access to, for instance, the 'Charms' bar in Windows 8, or hot corners in OS X.

3.6.5 Input Filtering

Filtering of input values to obtain stable results. Depth filtering used to limit the depth spectrum; 1€ Filter used for smoothing the input values.

Since devices like the Kinect do not always deliver accurate positions of the finger, and some jitter can be expected, input filtering is applied, on the raw data, to achieve a relatively stable input stream, which can be used to apply the input–output transformations. Firstly, the data is filtered depending on the depth value of every single packet received. If the depth value is outside the feasible range, it is assumed that the packet is not of interest, and is rejected. Secondly, the 1€ Filter (Casiez et al. [2012]) is used in order to smooth the input stream[2] Lastly, it is also possible to reduce the effect of hand tremor by specifying a minimum threshold for displacement in finger position, to qualify as a valid movement.

3.7 Design Principles Revisited

Section 3.1, highlighted some of the key aspects that are important for the design of a successful input device. The implementation of *Sniper Pointing* has attempted to satisfy each of these design goals, using the following mechanisms:

- Lower the homing time required to switch between typing and pointing—reducing the physical distance

[2]For the given Kinect implementation, the parameters used were: frequency=120, mincutoff=1.0, beta=1.0, dcutoff=1.0. These values are hardware-specific.

between the keyboard and the pointing device, and by using the volume directly above the keyboard for pointing.

- Allow fast pointing to targets on the screen—the coarse pointing layer, directly above the keyboard, maps the keyboard area to the entire screen area, and provides quick absolute pointing.
- Pointing accuracy should be comparable to mouse input—the fine pointing layer provides for a higher level of accuracy, in order to acquire small targets.
- Allow prolonged use without much fatigue—taking workstation design into ergonomics, the height of the pointing layers is optimized. Coarse pointing can be performed while resting hands on the desk or keyboard.
- Minimum or no screen occlusion—indirect cursor manipulation directly above the keyboard prevents the finger from occluding the screen area.

Chapter 4

Preliminary Studies

The interaction design of the Sniper Pointing system illustrates the application of absolute pointing and makes use of an input area which is smaller in size, as compared to the output screen. Additionally, the mid-air pointing technique is designed to allow for cursor manipulation with multiple granularities. This is achieved using a layered design, consisting of coarse and fine pointing layers.

Preliminary studies have served as a tool to aid in making design decisions relevant to some of these aspects and helped in optimizing the interaction. This chapter presents two preliminary studies which have been performed and their outcomes.

Two preliminary studies are presented

- The effect of absolute versus relative pointing and input-space scale on flat surface pointing—A comparison between indirect relative and absolute pointing on a horizontal surface is made and the effect of input space scale on pointing time is studied.

Absolute vs. Relative Pointing and Effect of Input-Space Scale

- Arrangement of the coarse and fine pointing layers, for mid-air pointing with multiple granularities—The ordering of the two granularity layers is determined through this study.

Arrangement of Pointing layers

4.1 User Study: Absolute vs. Relative Pointing & Effect of Input-Space Scale

This study dealt with the comparison of two pointing techniques—absolute and relative, when used on a horizontal surface, in a standard desktop environment. It also attempted to study the effect different input space scales had on pointing performance.

4.1.1 Aim and Rationale

To compare pointing speeds for the two methods, and to study the effect of input-space scale.

The aim of the user study was to compare pointing speeds for absolute and relative pointing. Additionally, an analysis of the effect of input-space scale on pointing speed was performed. This study was conducted in order to make design decisions for the proposed interaction. Using these results, we could determine the feasibility of using absolute pointing with input surface areas smaller than the screen size.

4.1.2 Experimental Design

Within-subject study; Participants performed multi-directional tapping tasks

A 'within-subject' user study was designed, in which participants were required to perform a multi-directional tapping task, as outlined by the ISO 9241-9 (ISO [2000]) standard and illustrated in figure 4.1. Users were required to point at targets, displayed on a vertical screen, using the given pointing devices, placed on a horizontal desk. All target acquisitions were explicitly confirmed by pressing the space-bar on a keyboard, using the non-dominant hand. This was done to avoid the occurrence of unintentional target acquisitions, caused by the cursor momentarily entering the target, while being manipulated, and to maintain uniformity between input devices.

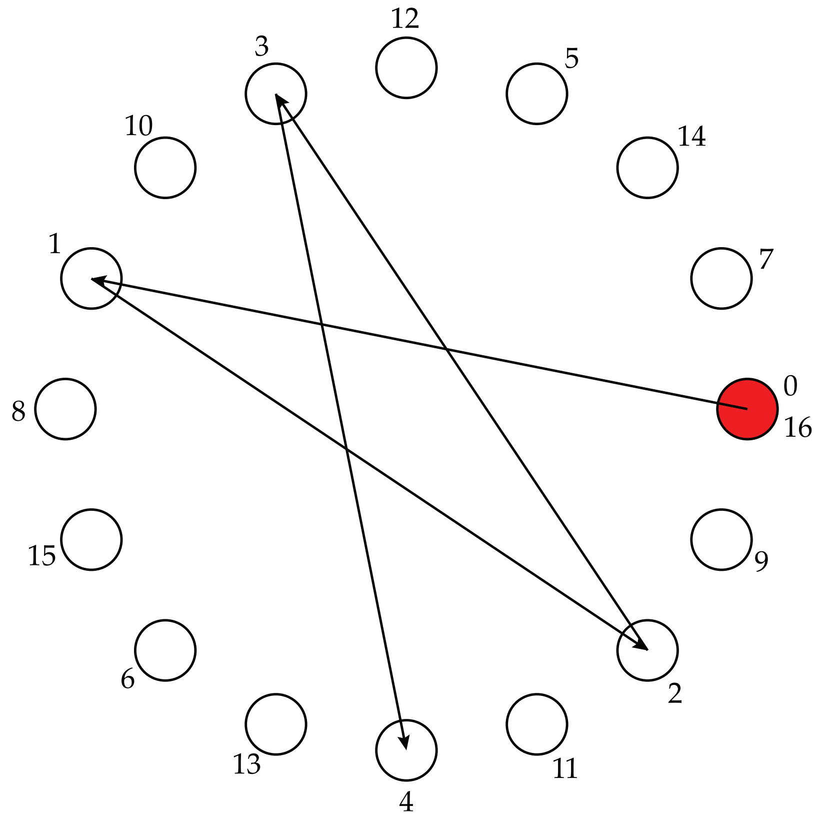


Figure 4.1: The multi-directional pointing task. Target highlighted in red is the next target to acquire. All other targets are outlined. The numbers and arrows indicate the order and pattern in which the targets are highlighted.

4.1.3 Independent Variables (IV)

The main independent variable of interest was *input method*. Three different forms of input were tested:

Input methods: Trackpad; 9.7” Tablet; 4.85” Tablet

1. *Trackpad*—A standard trackpad, which uses relative technique.
2. *Tablet with same size as screen*—A tablet input device, which uses absolute technique and has the same dimensions as that of the output screen. The diagonal length of the input surface used was 9.7 inches.
3. *Small-sized tablet*—A tablet device which had half the diagonal length (4.85 inches) as compared to the

screen, resulting in a 1:4 input—output screen ratio.

Three other independent variables were included in order to address the effect of the physical characteristics of targets on pointing performance:

2 target widths;
2 target distances;
16 target angles

1. *Target Width (W)*—24 pixels (0.25 inches); 32 pixels (0.33 inches).

2. *Target Distance (D)*—400 pixels (4.17 inches); 500 pixels (5.21 inches).

3. *Target Angle*—16 values, distributed uniformly along a circle, at intervals of 22.5°.

All independent variables were counter-balanced, in order to compensate for effects of task order on the results.

Additionally, a subset of the participants repeated the study for a screen which was twice the size (19.4 inches), as compared to the original (9.7 inches). The size of the input devices, however, remained the same. This resulted in the target distances doubling to 800 pixels (8.33 inches) and 1000 pixels (10.42 inches), while the target widths remained the same, allowing for a broader range of index of difficulties (ID) for targets.

Definition:
Index of Difficulty (ID)

INDEX OF DIFFICULTY (ID):
The index of difficulty of a target, based on Fitts' law, determines the theoretical difficulty of aiming at it. It is a function of *width (W)* of the target and the *distance (D)* to it, and is given in bits. In this thesis, the Shannon formulation is applied, given by equation 4.1.

$$ID = log_2(1 + D/W) \tag{4.1}$$

4.1.4 Dependent Variable (DV)

The dependent variable of interest for this study was the speed of target acquisition, measured in its inverse form as 'target acquisition time'. Participants were instructed to be as fast as possible, while performing the tasks.

TARGET ACQUISITION TIME:
The time taken for acquiring a given target. This is purely the time required to move the cursor from the initial position to the target.

Definition: *Target Acquisition Time*

4.1.5 Hypotheses

The following two null hypotheses were formulated before performing the study:

H1:
There is no difference in target acquisition time between relative trackpad and absolute tablet.

H1: No difference in speed between relative and absolute devices.

H2:
There is no difference in target acquisition time between a tablet having input area same as screen size and a tablet having a smaller input area.

H2: No difference in speed for different input sizes.

4.1.6 Hardware Configuration

A desktop computer, running Mac OS X, was set-up on a standard workplace desk. A desktop screen (23 inches; 1920 × 1200 pixels) placed vertically, was used for displaying the output. However, the entire area of the screen was not used. Instead, it was limited, according to size and aspect ratio of the input device used, by means of software. Two different effective display sizes were used (9.7 inches; 19.4 inches), represented by black rectangles on the screen. The input devices were placed horizontally on the desk. An

Desktop environment—input devices on horizontal surface; vertical output screen. Two different screen sizes used.

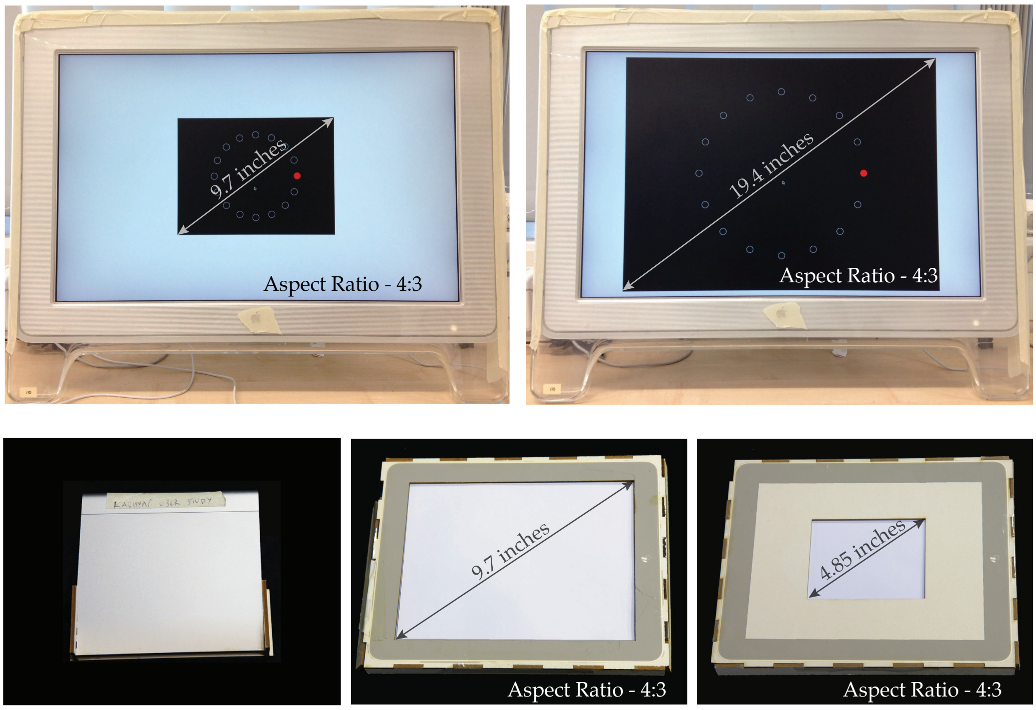


Figure 4.2: *Hardware set-up of the user study. Top: The two different output screen sizes used—Regular and Large. Below: Three input devices tested—Trackpad, Tablet (large) and Tablet (small). Custom cases were made to ensure that the surfaces had the same texture. Slope of surface was made to be 0°*

Apple Magic Trackpad was used as the relative pointing device and a first-generation Apple iPad (9.7 inch screen; 1024 × 768 pixels) as the absolute device. The pointing device was aligned with the centre of the screen, and a hardware keyboard, placed alongside the pointing device, was used to confirm target acquisition with the non-dominant hand. To eliminate influence of the difference in surface textures of the pointing devices, which may provide different amounts of friction, they were made uniform by overlaying them with plain paper. Figure 4.2 illustrates the two different screen sizes used, and the various pointing devices tested. In order to fix the size of the tablet surfaces, customized cases were built, and the input-output mapping was specified through software.

4.1.7 Software Configuration

An application was designed which displayed targets, arranged around a circle, to the users. At each instance, the target to be acquired was highlighted in red, while all other targets were outlined. A black rectangle indicated the screen area, and regions outside this rectangle were made inaccessible. The four combinations of target sizes and target distances were displayed sequentially. Once a given combination was completed, the next was displayed, until all four combinations were completed by the user. This procedure was repeated for each of the given input devices.

Data Logging

All relevant data was logged into comma-separated values (CSV) files along with timestamps. This included data such as the user identifier, device name, target description (size, distance, angle), action performed[1], the location of the mouse cursor (x,y), the touch location on the device (x,y), and the total target acquisition time, for each target.

Data entries with timestamps logged in CSV format.

Target Acquisition Time

The time taken for target acquisition was calculated by the application, and recorded into the log files. In order to eliminate factors such as time taken to perceive the target, and to measure solely the movement time taken, the start of each target acquisition action was noted when the cursor first moved, after the previous target was acquired. The end of target acquisition was noted when the user hit the space-bar to confirm acquisition.

Cursor Acceleration

Different systems and devices make use of different transfer functions to provide cursor acceleration. In order to standardize the cursor acceleration used for the study, the LibPointing toolkit (Casiez and Roussel [2011]) was used and the standard OS X transfer function was applied with default parameters.

LibPointing toolkit with default Mac OS X parameteres used.

Screen Size Limiting

In order to limit the screen size and aspect ratio to that

[1]The possible actions are 'Touch began', 'Touch moved', 'Touch ended' and 'False key-press'

of the 9.7 inch tablet device, with aspect ratio 4:3, a black bounding rectangle was drawn in the centre of the screen. This served as the effective screen area. The cursor was limited to movement within this bounding box, and other regions were made inaccessible.

For the case where the screen size was doubled (19.4 inches), the same techniques were applied, and the aspect ratio was maintained.

4.1.8 Experiment Procedure

For each participant, the experiment procedure was as follows:

1. The sequence of input devices presented to users was balanced using a 3×3 Latin Square. Additionally, the sequence in which the different target size and distance combinations appeared were also counterbalanced, in order to avoid task-order effects.

2. Participants were provided with an explanation of the tasks to be performed, and were requested to sign a consent form.

3. A training phase was provided, which allowed users to get acquainted with each of the devices presented. The training phase also involved target acquisition tasks, with a duration of approximately 5 minutes.

4. Following the training phase, the study was performed using the given devices. A pause was provided between trials for each device, during which participants were asked for qualitative opinions about the device they had experienced.

4.1.9 Participants

A total of 7 participants were recruited—6 male and 1 female. All participants were right-handed, between 20 and 30 years old, and were students at the local campus at RWTH Aachen. Participation was voluntary and no monetary compensation was provided. While all participants performed the user study with the 9.7 inch display, only 4 participants repeated the trials with the 19.4 inch display.

4.1.10 Statistical Methods

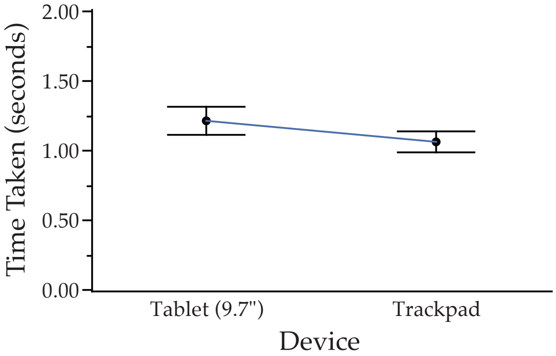
The within-subject design of the user study can be summarized as follows:

Input methods	=	3 (Trackpad, Tablet, Small Tablet)
Target widths	=	2 (24 pixels, 32 pixels)
Target distances	=	2 (400 pixels, 500 pixels)
Target angles	=	16 (22.5° intervals)
Total results	=	192 datapoints per participant. ($3 \times 2 \times 2 \times 16$)

Table 4.1: *Summary of the user study design. A combination of different target widths, distances and angles were used during the study.*

Target acquisition time aggregated and averaged; Repeated-Measure ANOVA used for analysis.

Before performing analysis, the target acquisition times for each set of 16 target angles were aggregated to obtain an average time for every target width and target distance combination. This resulted in a total of 4 average target acquisition times for each of the given input devices, for each participant. The data distribution was found to be not in the normal form, and hence a log transformation was applied on the 'target acquisition time' data. Considering the design of the user study, Repeated Measure ANOVA was used to statistically analyse the results.



(a)

Device	Mean	Lower CI	Upper CI
Tablet (9.7″)	1.212	1.111	1.313
Trackpad	1.061	0.986	1.137

(b)

Figure 4.3: *Results for comparison of pointing times for absolute tablet and relative trackpad. (a) Graph showing the pointing times for the two devices. The mouse was found to be only marginally faster than the tablet device. (b) Summary of average pointing times and 95% confidence interval values.*

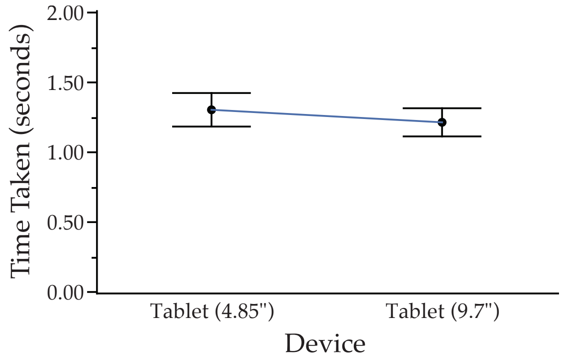
Source	DF	DFDen	F Ratio	Prob >F
Target Width (px)	1	42	11.04	0.0019*
Target Distance (px)	1	42	0.01	0.9051
Width * Distance	1	42	0.00	0.9753
Device	1	42	10.17	0.0027*
Width * Device	1	42	0.40	0.5305
Distance * Device	1	42	0.01	0.9390
Width * Distance * Device	2	42	0.01	0.9077

Table 4.2: *Results of fixed effect test comparing absolute tablet and relative trackpad. 'Target Width' and 'Device' showed significant effect on pointing time.*

4.1.11 Analysis: Absolute vs. Relative Pointing

Device type and target width had significant effect on results. Relative trackpad was slightly faster.

To compare absolute and relative pointing, results obtained using the 9.7 inch absolute tablet were compared to those obtained using the relative trackpad. Fixed effect tests showed significant effect of device type ($F_{1,42}$=10.17, p=.0027) and target width ($F_{1,42}$=11.0365, p=.0019) on the total pointing time. The main factor of interest here was the influence of device type on the pointing times. It was found that there was not a big difference between the relative trackpad (average time = 1.04 seconds) and the 9.7 inch absolute tablet (1.19 seconds). In fact, it can be observed in figure 4.4a that the 95% confidence intervals overlap for the



(a)

Device	Mean	Lower CI	Upper CI
Tablet (4.85″)	1.302	1.182	1.422
Tablet (9.7″)	1.212	1.111	1.313

(b)

Figure 4.4: *Results for study of effect of input space scale. (a) Graph showing the mean pointing times for the two tablet sizes—4.85″ and 9.7″. (b) Summary of average pointing times and 95% confidence interval values.*

two devices.

Source	DF	DFDen	F Ratio	Prob >F
Target Width (px)	1	42	18.24	<0.0001*
Target Distance (px)	1	42	0.84	0.3639
Width * Distance	1	42	0.00	0.9466
Device	1	42	3.91	0.0545
Width * Device	1	42	0.96	0.3328
Distance * Device	1	42	0.46	0.5021
Width * Distance * Device	1	42	0.00	0.9705

Table 4.3: *Results of fixed effect test for the study of effect of input space scale. 'Device' showed no significant effect on the pointing time.*

4.1.12 Analysis: Effect of Input Space Scale

No significant effect of input device size on results.

To study the effect of input space scale, the target acquisition times achieved using the 9.7 inch tablet were compared with that achieved using the 4.85 inch tablet. The fixed effect tests showed significant effect of target width ($F_{1,42}$=18.24, p<.0001) on the target acquisition time but no significant effect of device size ($F_{1,66}$=3.91, p=.0545) on the acquisition time. There was not much difference in pointing times for the 9.7 inch tablet (1.27 seconds) and the 4.85 inch tablet (1.19 seconds). Figure 4.4a shows that the confidence intervals for the two devices overlap.

4.1.13 Analysis: Fitts' Law Parameters

According to Fitts' Law, the mean (expected) movement time can be given by the formula:

Definition:
Movement Time (MT)

MOVEMENT TIME (MT):

$$MT = a + b \cdot ID \tag{4.2}$$

Target Width	Target Distance	Index of Difficulty
32	400	3.75
32	500	4.05
24	400	4.14
24	500	4.45
32	800	4.70
32	1000	5.01
24	800	5.10
24	1000	5.41

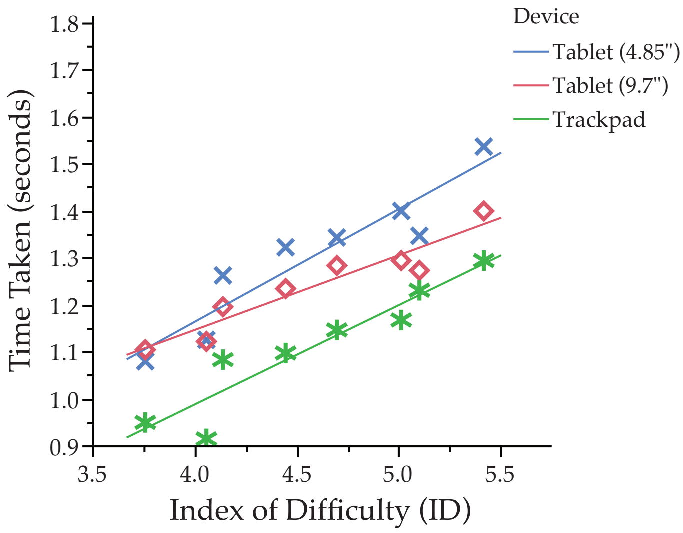
Table 4.4: *List of target width and distance combinations, and the corresponding index of difficulty (ID).*

Movement time influenced by factors 'a' and 'b'. Non-information aspect denoted by a; $1/b$ denotes throughput.

Here, a denotes the non-information aspect and b denotes the information aspect of the movement during pointing. Factors like regression error, modelling error, among others can lead to a non-zero value for the constant a. Additionally, a can also be non-zero due to factors like human visual, cognitive, or motor reaction/activation process, and is explained in detail by Zhai [2004]. The actual throughput of an input device can be given by $1/b$. This indicates how the input performance changes relative to changes in ID.

For each target width and distance combination, results among participants were averaged.

For the given user study, target acquisition times achieved using both screen sizes, 9.7 and 19.4 inches, were accumulated in order to achieve the movement times for eight different IDs (table 4.4). For each target width and distance combination, movement time results for all participants were aggregated and averaged. The linear fit for each of the three devices, given by equation 4.2, is illustrated in figure 4.5a. Figure 4.5b summarizes the various Fitts' Law parameters and also the throughputs ($1/b$) for the devices.



(i)

Device	Intercept (*a*) (seconds)	Slope (*b*) (seconds)	Throughput (1/*b*)	R^2
Trackpad	0.1419	0.2113	4.7326	0.8864
Tablet (9.7″)	0.5010	0.1589	6.2932	0.9082
Tablet (19.4″)	0.2043	0.2396	4.1736	0.8823

(ii)

Figure 4.5: *Results of Fitts' Law Analysis for the three different input devices. (i) Graph showing the relation between Index of Difficulty and target acquisition time. (ii) Summary of Fitts' Law parameters for the three devices. The 9.7″ Tablet was found to exhibit the highest throughput.*

It is observed that the 9.7 inch tablet exhibited the highest throughput for the given set of results. This indicates that the absolute tablet is potentially capable of exhibiting better overall performance, in terms of speed, when compared with the absolute trackpad, for higher values of ID. In Figure 4.5, linear fits for the 4.85 inch and 9.7 inch tablets intersect at a point. It can be observed that target acquisition time for targets with smaller index of difficulty (ID) is similar for both tablet sizes. However, as ID increases, the difference in acquisition times increases and the smaller tablet tends to become slower.

9.7" Tablet had highest throughput. 4.85" tablet slower for larger IDs.

4.1.14 Limitations

Output screen size and surface area of pointing devices limited range of results.

Since the user study was limited by the size of the output screen and available surface area of the pointing devices, an extended Fitts' analysis was not possible. A bigger screen would make it possible to repeat tests with a higher range of IDs. Also, different sizes of tablet devices could provide more concrete results, regarding the influence of input space scale on pointing times.

4.1.15 Implications

Hypothesis H1 can not be rejected.

A comparison of target acquisition times using the trackpad and the tablet showed that there was not much difference in pointing speeds between relative and absolute pointing. Hence, the hypothesis H1, stated earlier, can not be rejected. Analysis of Fitts' Law parameters showed that the tablet exhibits a higher overall throughput, and is likely to be faster for higher indices of difficulty (ID). However, this could not be verified due to the limitations of the experimental set-up.

From the given data, H2 also can not be rejected.

A comparison of two different sizes of absolute tablets showed that the size of the input area did not have significant effect on the target acquisition times. From the data, null hypothesis H2 can not be rejected. Analysis of the Fitts' law parameters indicates that scaling may have a greater effect for higher values of ID.

These results have been applied into the *Sniper Pointing* interaction, in a manner which seems feasible. The designed interaction uses absolute pointing over the surface, and maps the smaller keyboard area to the larger output screen area. The keyboard area has been chosen since it serves as a physical indicator of boundaries of the input area for pointing. Additionally, since the two hands can cover the entire keyboard area with ease, it also allows easy access to the entire screen area during pointing.

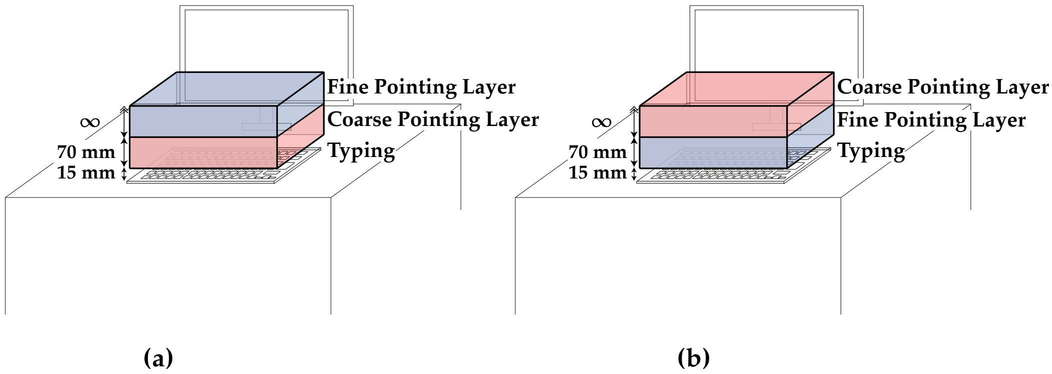


Figure 4.6: *Illustration of the two different layer arrangements tested during the study. (a) Coarse pointing directly above the keyboard, with fine pointing stacked above. (b) Coarse pointing higher above, with fine pointing closer to the keyboard.*

4.2 User Study: Arrangement of Coarse and Fine Pointing Layers

The *Sniper Pointing* interaction involves the use of multiple pointing layers—coarse and fine. These two layers, above the keyboard surface, are stacked one above the other. Shifting between layers is performed by raising and lowering the finger to the appropriate layer.

Study to compare two layer arrangements.

During the process of designing and implementing the *Sniper Pointing* system, another user study was performed to compare two possible arrangements of these pointing granularity layers, and aided in making decisions specific to this aspect.

4.2.1 Aim and Rationale

Determine suitable arrangement of pointing layers.

The aim of this preliminary study was to determine a suitable arrangement of the two pointing layers. Each of the two possibilities for layer arrangements, as illustrated in figure 4.6, can be backed with an appropriate rationale, as given below:

Coarse below and fine above
This arrangement offers coarse pointing near the keyboard, and fine pointing is accessible by lifting the finger higher. Having the coarse pointing layer close to the keyboard allows for immediate access to pointing by slightly lifting the finger. Hence, the switch between typing and pointing is seamless. Cursor manipulation using fine pointing is optional, and is not required for all targets. Additionally, large movements in fine pointing mode result in only small movements of the cursor. Therefore, fatigue and instability of the arm at greater heights should not play a major role in the user experience.

Coarse higher and fine close to keyboard
This arrangement offers coarse higher above, and lowering the finger towards the keyboard allows for fine pointing. By requiring users to raise the pointing finger to a greater height, involuntary activation of pointing, caused by slightly raising the finger, can be eliminated. Stability during fine pointing is increased, since the arm is closer to, or on, the desk. This arrangement also allows users to immediately begin typing after fine pointing is completed, resulting in faster homing to keyboard.

4.2.2 Experimental Design

Within-subject study; Participants performed multi-directional point-and-type tasks using two mid-air arrangements.

To compare the two layer arrangements, a 'within-subject' user study was designed. Four right-handed participants were required to perform multi-directional point-and-type tasks. Here, users pointed at targets on a vertical display using the *Sniper Pointing* technique for mid-air pointing. Once target acquisition was confirmed by pressing the space-bar on a keyboard, participants typed a standardized phrase into a text-box, displayed directly above the target. This sequence of tasks presented a typical usage scenario of typing and pointing to participants.

4.2.3 Independent Variables (IV)

The main independent variable of interest was *arrangement of pointing layers*. Two different arrangements were tested:

1. *Coarse lower, fine above:* Coarse pointing directly above the keyboard; fine pointing layer stacked above coarse layer.

2. *Coarse higher, fine below:* Coarse pointing layer higher above; fine pointing layer below coarse layer, and close to keyboard.

Two other independent variables were included in order to address the effect of the physical characteristics of targets on performance:

1. *Target Width (W)*—12 pixels (0.13 inches); 16 pixels (0.17 inches); 36 pixels (0.38 inches); 48 pixels (0.5 inches).

2. *Target Angle*—8 values, distributed uniformly along a circle, at intervals of 45°.

The target distance was fixed to 800 pixels (8.33 inches) and the target widths were balanced among participants through the use of a 4x4 Latin Square. The order in which the two arrangements were tested was alternated among the participants.

4.2.4 Dependent Variable (DV)

The dependent variable of interest for this study was the 'total pointing time', which was measured as the interval in between two consecutive typing tasks. Participants were instructed to be as fast as possible, while performing the tasks.

Definition: *Total Pointing Time*

TOTAL POINTING TIME:
The time taken to perform a single pointing task. This is the aggregate of the homing time required to switch from typing to pointing, pointing time required to manipulate the cursor, and homing time required to return from pointing to writing.

4.2.5 Hypothesis

The hypothesis for the given experiment can be stated in the null form.

H1: No effect of layer ordering.

H1:
There is no effect of ordering, of the two pointing layers, on the pointing time achieved.

4.2.6 Hardware Configuration

Desktop environment; vertical screen for output; volume above the keyboard for input.

The hardware was set-up on a standard workplace desk (figure 5.1). A desktop screen (23 inches, 1920 $\times$ 1200 pixels) placed vertically, was used for displaying the output. A standard keyboard was placed on the horizontal surface of the desk. The mid-air pointing region was specified by the volume above the desk. A Vicon motion tracking system was used in order to accurately track the position of left and right index fingers of participants. More details about the Vicon set-up is provided in chapter 5—"Evaluation".

4.2.7 Software Configuration

Developed application to provide the point-and-type task.

An application was implemented in order to provide users with the point-and-type task. Targets, represented by circles, were displayed at intervals of 45°, and once a target was acquired, a standard phrase, to be typed, and a textbox appeared directly above it, with the keyboard gaining focus. The pointing cursor was hidden during typing, to avoid distraction. Once the given phrase was entered, the text-box and label disappeared, and the cursor regained the focus. All relevant data, including data from the Vicon, was logged into comma-separated value (CSV) files. In-depth explanations regarding the software configuration and further details are provided in chapter 5—"Evaluation".

4.2.8 Experiment Procedure

For each participant, the experiment procedure was as follows:

1. Participants were first provided with an explanation of the tasks to be performed, and were requested to sign a consent form.

2. A training phase allowed users to get acquainted with the input technique and the layer arrangement. This training phase lasted approximately 3 minutes, for each arrangement.

3. Once training was completed, the participants performed timed trials of the point-and-type task. Here, users were requested to be as fast as possible, while performing the tasks.

4. After trials with each technique, a break was provided, during which users were asked for qualitative opinions about the respective input device.

4.2.9 Participants

A total of 4 participants were recruited—3 male and 1 female. All participants were right-handed, between 20 and 30 years old, and students at the local campus at RWTH Aachen. Participation was voluntary and no monetary compensation was provided. The order of tasks was balanced among participants, to prevent any influence of learning or exhaustion.

4.2.10 Statistical Methods

The within-subject design of the user study can be summarized as follows:

Layer Arrangements	=	2 (coarse below & above)
Target widths	=	4 (12, 16, 32, 48 pixels)
Target distance	=	1 (800 pixels)
Target angles	=	8 (45° intervals)
Total results	=	64 datapoints per participant.
	=	($2 \times 4 \times 8$)

Table 4.5: *Summary of the user study design. A combination of different target widths and angles were used during the study.*

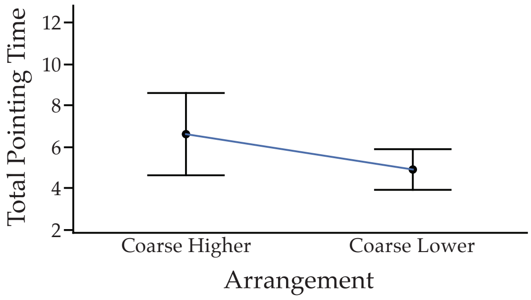
Total pointing time aggregated for targets. Paired-sampled *t*-test used for analysis.

For each set of 8 target angles, the total pointing times were aggregated, to obtain an average total pointing time for every target width, for each participant. To quantitatively analyse the results, a paired-sampled *t*-test was performed on the resultant data. For the given study, qualitative opinions obtained from the user were also vital.

4.2.11 Quantitative Analysis

Arrangement with coarse lower, and fine higher above found to be significantly faster.

To quantitatively analyse the two pointing layer arrangements, the total pointing times obtained from the point-and-type user study, for each of the two arrangements, were compared. The fixed effect tests showed statistically significant effect of arrangement ($F_{1,3}$=5.9526, p=0.0298) on the total pointing time. It was found that the arrangement with the coarse layer close to the keyboard, and the the fine layer stacked above the coarse, was faster (average time = 4.89 seconds) than the arrangement with the coarse layer higher above, and the fine layer below (6.59 seconds). This result, and a graph comparing the two arrangements, are shown in figure 4.7.



(a)

Device	Mean	Lower CI	Upper CI
Coarse Higher	6.589	4.607	8.572
Coarse Lower	4.886	3.906	5.865

(b)

Figure 4.7: *Results for comparison of total pointing time for two different layer arrangements. (a) Graph showing the resultant pointing times. Having the coarse pointing layer lower and the fine layer above, was significantly faster. (b) Summary of average total pointing times and 95% confidence intervals for the two arrangements.*

Source	DF	DFDen	F Ratio	Prob >F
Target Width	3	13	3.32	0.0523
Arrangement	1	13	5.95	0.0298*
Width*Arrangement	3	13	0.93	0.4561

Table 4.6: *Result of fixed effect test showing statistically significant main effect of arrangement on total pointing time.*

4.2.12 Qualitative Opinions

During the user study, users were asked subjective questions about the two given arrangements. The questions focussed on aspects like preferences, ease of use, tiredness, accuracy and speed of cursor manipulation.

Coarse lower, fine above also preferred by users.

In general, participants preferred the arrangement with the coarse layer close to the keyboard, and the fine layer stacked above. Some users also felt that this arrangement was less tiring. One of the users commented that it was easier to misjudge the height at which coarse pointing started, when the arrangement with the coarse layer above, and fine below, was used.Comments and opinions of users made it evident that the coarse layer close to the keyboard, and the fine layer stacked above, was more intuitive to use and was overall better.

4.2.13 Implications

Null hypothesis H1 rejected.

Quantitative as well as qualitative results of the user study rejected the null hypothesis H1, stated previously, and showed that the arrangement of the granularity layers, with the coarse layer close to the keyboard, and the fine layer higher above, outperformed the alternative arrangement.

Coarse lower, fine above arrangement has been used in *Sniper Pointing*.

This result has been used in the design and implementation of the Sniper Pointing system. In the final interaction design, typing is performed at the keyboard level, placed on the desk. On lifting the finger slightly above (15 millimetres) the keyboard height, coarse pointing was activated. The coarse layer had a thickness of 70 millimetres, and when the finger is lifted further above this, fine pointing is activated. The fine pointing layer does not have an upper boundary, and has infinite thickness.

Chapter 5

Evaluation

In order to evaluate *Sniper Pointing*, a user study was performed, with the aim to compare its performance with that of the mouse. While doing so, the main factor of interest was time required to perform various sub-tasks involved in a routine pointing task. This included tasks of switching between pointing and typing devices and cursor manipulation.

5.1 Rationale

Since the mouse has been one of the most widely used pointing devices in the past, it is feasible to compare the performance of *Sniper Pointing* with that of a standard mouse. Pointing is usually performed in conjunction with typing, hence it is apt to conduct a test that presents users with a scenario involving both, typing and pointing, activities. Another necessary activity, contributing to the overall task time, is the act of switching from one device to another to perform the relevant tasks, known as homing. It is important to compare this homing time for the two devices, and to study its effect on the overall performance, in terms of task completion time.

Mouse is the most widely used pointing device. Pointing is often used in conjunction with typing, making device switching necessary.

5.2 Experimental Design

Within-subject design; Multi-directional point-and-type tasks.

In order to evaluate *Sniper Pointing*, a 'within-subject' user study was designed. Participants were required to perform a multi-directional point-and-type task. Users pointed at targets on a vertical display, using each of the given pointing techniques. Once target acquisition was confirmed by pressing the space-bar on a keyboard, a standardized phrase was typed into a text-box, displayed directly above the target. This sequence of tasks presented a typical usage scenario of typing and pointing.

5.3 Independent Variables (IV)

The main independent variable of interest was the pointing device used. Two different devices were tested:

1. *Sniper Pointing*—The proposed mid-air pointing technique with multiple resolutions.
2. *Mouse*—A standard mouse with cursor acceleration.

Three other independent variables were included in order to address the effect of the physical characteristics of targets on performance:

1. *Target Width (W)*—12 pixels; 16 pixels; 36 pixels; 48 pixels.
2. *Target Distance (D)*—700 pixels; 1000 pixels.
3. *Target Angle*—16 values, distributed uniformly along a circle, at intervals of 22.5°.

The device order was alternated among participants to compensate for any influence of external factors like fatigue on the results. Additionally, target widths were balanced using a 4x4 Latin Square, and the target distances were accordingly counter-balanced.

5.4 Dependent Variable (DV)

There were four dependent variables of interest for the given point-and-type task, each measured in the form of 'time taken'. The time required to perform actions like homing to point, pointing, and homing to keyboard served as dependent variables. Additionally, the aggregate of these times, known as 'total pointing time', was also taken into consideration.
Participants were instructed to be as fast as possible, while performing the tasks.

HOMING TO POINT TIME:
Once typing has been completed, it is the time taken to move the pointing hand and begin cursor manipulation.

Definition:
Homing to Point Time

HOMING TO KEYBOARD TIME:
Once a target has been pointed at, it is the time taken by the pointing hand to move to the keyboard, in order to begin typing.

Definition:
Homing to Keyboard Time

POINTING TIME:
It is the pure pointing time taken to manipulate the cursor and move it from an initial position to the target position.

Definition:
Pointing Time

TOTAL POINTING TIME:
This defines the total time required to perform a single target acquisition task.

$$\begin{aligned} Total\ Pointing\ Time = & Homing\ to\ Point\ Time + \\ & Pointing\ Time + \\ & Homing\ to\ Keyboard\ Time \end{aligned} \tag{5.1}$$

Definition:
Total Pointing Time

5.5 Hypothesis

The hypotheses that are constructed before performing the study are stated in the null form:

H1: No difference in homing to point times

H1:
While homing from keyboard to pointing device, there is no difference in task completion times between *Sniper Pointing* and mouse.

H2: No difference in homing to keyboard times

H2:
While homing from pointing device to keyboard, there is no difference in task completion times between *Sniper Pointing* and mouse.

H3: No difference in overall pointing times

H3:
There is no difference in total pointing times between *Sniper Pointing* and mouse.

5.6 Hardware Configuration

The user study was executed on a standard desktop computer. Additionally, a Vicon set-up, consisting of Vicon motion capture cameras, and a computer, running the Vicon Tracker software, were used for tracking purposes. The details of the hardware set-up are provided as follows.

5.6.1 Desktop Arrangement

Desktop environment with Vicon arranged for tracking movements.

A desktop computer, running Mac OS X, was arranged on a desk surface, along with a 23 inch display (1920 × 1200 pixels). The entire study environment was enclosed within a rigid structure, made of aluminium profiles, which also hosted a set of cameras. A second computer, running Windows XP, processed input from Vicon cameras, through the

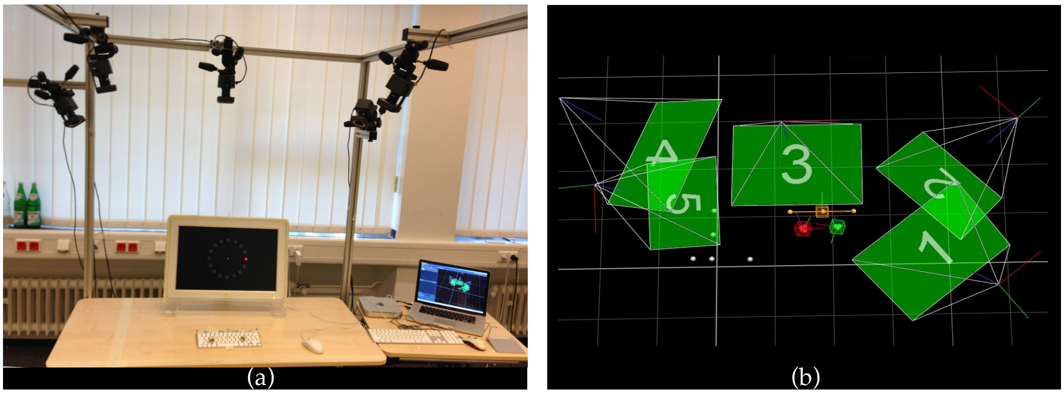


Figure 5.1: Illustration of the hardware arrangement and schematic view of cameras in the Vicon Tracker *window.*

Vicon Tracker software, and streamed this data to the application developed for the user study (Figure 5.1a).

5.6.2 Vicon Configuration

A cluster of five Vicon cameras was constructed (Figure 5.1b). The cameras were arranged in a semi-circular layout and were configured in such a way that the main area of focus was the entire volume directly above the desk. This allowed accurate and precise tracking of users' fingers, as well as keyboard position.

Vicon cameras captured movement of hands.

5.6.3 Finger and Keyboard Markers

Reflective markers served as data source for the Vicon camera set-up. Custom-made rings were made for each of the index fingers. To distinguish the left finger from the right, the markers were arranged in two different patterns (figure 5.2a). Additionally, markers were also attached to the keyboard (5.2b), to obtain its positional information. Corresponding objects, for each of the data source, were created in the Tracker software (5.2c).

Reflective markers to track both index fingers and keyboard.

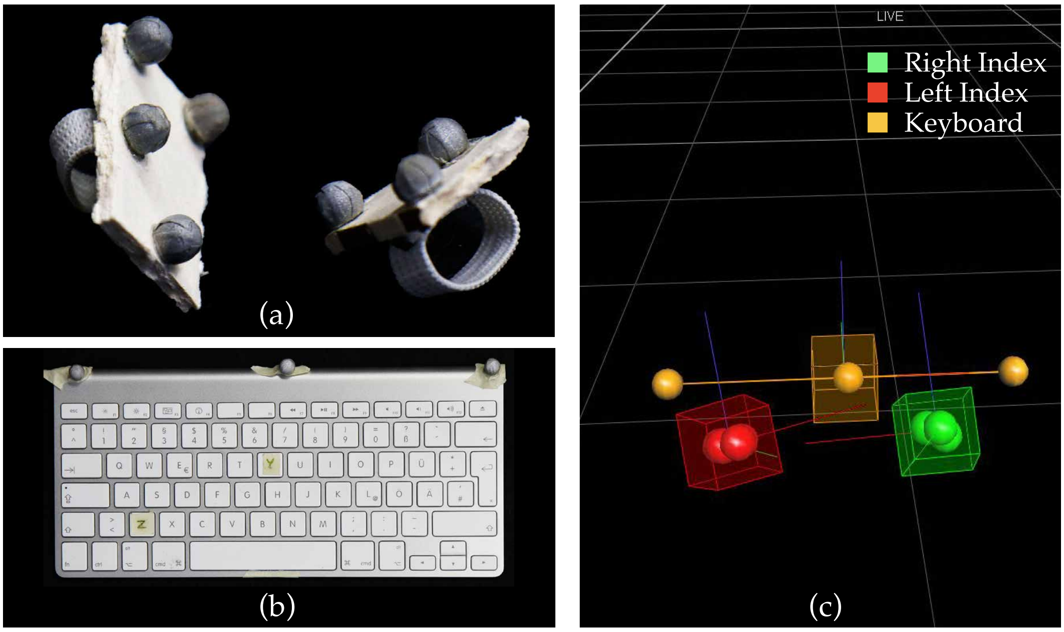


Figure 5.2: Illustration of the reflective markers used. (a) Custom-made rings for the left and right index fingers. (b) Keyboard with attached markers. (c) Objects represented on the Vicon Tracker screen.

5.7 Software Configuration

Software application presented the tasks to users, and logged all relevant data.

An application was implemented in order to provide users with the described point-and-type task. Targets, represented by circles, were displayed at intervals of 22.5°, and once a target was acquired, an empty text-box and a standard phrase, to be typed, appeared directly above the target, with the keyboard gaining focus. The pointing cursor was hidden during typing to avoid distraction. Once the given phrase was entered, the text-box and label disappeared, and the cursor regained the focus. All relevant data, including data from the Vicon, was logged into comma-separated value (CSV) files. Figure 5.3 shows the step-by-step activities performed by users, while executing the point-and-type tasks. The figure also shows the use of the two different cursor granularities, during target acquisition with *Sniper Pointing*.

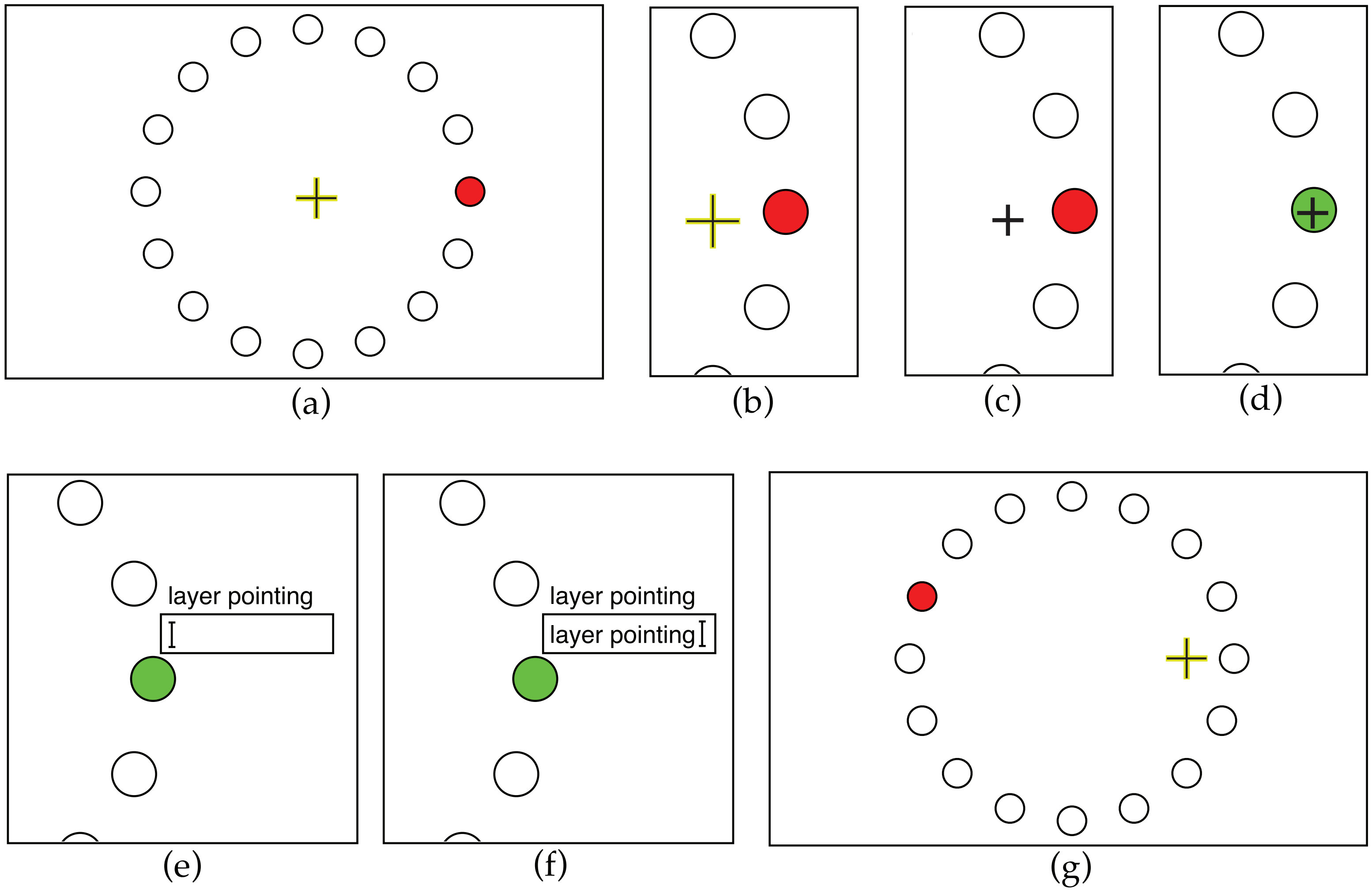


Figure 5.3: *Sequence of activities for a given point-and-type task, using* Sniper Pointing*. (a) Start of a trial, with target to be acquired highlighted in red. (b) User approaches the target; yellow cursor denotes coarse granularity. (c) Switching to fine-grain manipulation; change in cursor image and size indicate fine pointing. (d) Target colour changes to green on being acquired. (e) Standard phrase and text-box appear above the target; keyboard gains focus. (f) Phrase is typed into the box. (f) On hitting return key, next target to be acquired is highlighted in red.*

5.7.1 Initialization

Keyboard position initialized by each user.

During each run of the application, the software was first initialized. This was done in order to allow users to adjust the position of the keyboard, according to preference. The location of the keyboard, on the desk, was obtained from the keyboard markers and Vicon system, and this was used to map the keyboard area to the display. Additionally, height information was obtained from the keyboard marker, and was used to determine the height at which typing was performed.

5.7.2 Layer Arrangement

Layers arranged derived from preliminary study—coarse pointing close to keyboard and fine above.

The appropriate layer arrangement for *Sniper Pointing* was directly obtained from results of a preliminary user study (cf. Chapter 4.2—"User Study: Arrangement of Coarse and Fine Pointing Layers"). The typing layer had a thickness of 15 millimetres above the keyboard, and the cursor was disabled when the finger was located within this layer. Raising the finger above this height resulted in activation of the coarse pointing. The cursor pointing layer had a thickness of 70 millimetres, providing users with a sufficient volume to perform coarse movements. On raising the finger further higher, fine pointing was enabled. There was no upper bound for the fine pointing height defined.

5.7.3 Homing Time Measurement

Time taken to begin pointing.

Homing to Point Time This was measured as the time taken from the moment the return key was pressed, to confirm end of the typing task, to the moment where pointing task was started, by initially moving the on-screen cursor. If users switched hands, to select a different hand for target acquisition, then this action contributed to the homing time. Hence, switching hands several times during a given trial effectively increased the homing time, and not the actual pointing time.

Homing to Keyboard Time This was measured as the time taken from the moment the space-bar was pressed, to confirm target acquisition, to the moment when the pointing hand was placed on the keyboard, to begin typing.

Time taken to return to keyboard.

5.8 Experiment Procedure

For each participant, the experiment procedure was as follows:

- The sequence of input devices used by participants was counter-balanced, to avoid task-order effects. The sequence in which the different target sizes appeared were balanced using a 4×4 Latin Square.
- Participants were provided with an explanation of the tasks to be performed, and were requested to sign a consent form.
- A training phase allowed users to get acquainted with the pointing technique and the point-and-type task. The training phase lasted for approximately 5 minutes, for each pointing method.
- Once training was completed, participants performed timed trials for the given point-and-type task. While doing so, participants were requested to be as fast as possible.
- After testing with each technique, a break was provided, during which participants were asked for qualitative opinions about the pointing method they had experienced.

5.8.1 Participants

A total of 16 participants were recruited—13 male and 3 female. All participants were right-handed, between 18 and 30 years old, and were students at the local campus at

RWTH Aachen. Participation was voluntary and no monetary compensation was provided. All participants were frequent users of desktop computers, and had considerable experience with mouse.

5.9 Statistical Methods

The within-subject design of the user study can be summarized according to table 5.1.

Pointing Methods	=	2 (*Sniper Pointing*, Mouse)
Target widths	=	4 (12, 16, 32, 48 pixles)
Target distance	=	2 (800 pixels; 1000 pixels)
Target angles	=	16 (22.5°intervals)
Total results	=	256 datapoints per participant ($2 \times 4 \times 2 \times 16$)

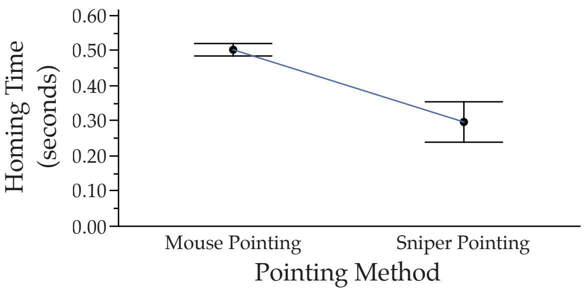
Table 5.1: *Summary of the user study design. A combination of different target widths, distances and angles were used during the study.*

Paired-sampled *t*-test used for analysis.

For each set of 16 targets (at 22.5°intervals), the resultant times were aggregated, to obtain an average time for each target width and distance combination, for each user. To quantitatively analyse the results, a paired-sampled *t*-test was performed on the resultant data. Repeated Measure ANOVA was used to analyse main effects and interactions between various independent variables.

5.10 Quantitative Analysis

This section elaborates on the analysis of the various factors that play a role in the total time required to perform cursor manipulations. This includes the homing time required to switch focus between keyboard and pointing device, and pointing time, required to perform cursor manipulation and acquire on-screen targets.



(a)

Method	Mean	Lower CI	Upper CI
Mouse Pointing	0.501	0.483	0.518
Sniper Pointing	0.294	0.237	0.352

(b)

Figure 5.4: *Results for comparison of homing to pointing times. (a) Graph showing mean homing times. Sniper Pointing was found to be significantly faster. (b) Summary of mean homing times and 95% confidence intervals for the two methods.*

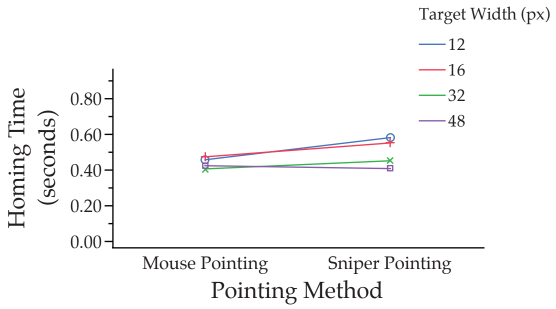
Source	DF	DFDen	F Ratio	Prob >F
Target Width (px)	3	194	1.64	0.1821
Target Distance (px)	1	194	0.01	0.9320
Width * Distance	3	194	0.31	0.8170
Pointing Method	1	193	127.31	<0.0001*
Width * Method	3	193	2.23	0.0856
Distance * Method	1	193	0.34	0.5597
Width * Distance * Method	3	193	0.20	0.8968

Table 5.2: *Results for fixed effect test of homing to point time. 'Pointing Method' showed significant effect on results.*

5.10.1 Analysis: Homing to Point Time

Homing to point time was measured as the time required to move the pointing hand from the keyboard to the pointing device, in order to begin cursor manipulation. The fixed effect tests (table 5.2) showed significant effect of 'Pointing Method' ($F_{1,193}$=127.32, p<.0001) on *homing to point time*. It was found that *Sniper Pointing*(average time = 0.200 seconds) was significantly faster than *Mouse Pointing* (0.492 seconds).

'Pointing Method' significantly effected homing time to pointing device. Sniper Pointing was faster than Mouse Pointing.



(a)

Method, Width	Mean	Lower CI	Upper CI
Mouse, 12	0.454	0.386	0.522
Sniper, 12	0.580	0.515	0.643
Mouse, 16	0.471	0.418	0.524
Sniper, 16	0.549	0.490	0.608
Mouse, 32	0.403	0.366	0.440
Sniper, 32	0.449	0.364	0.533
Mouse, 48	0.421	0.382	0.459
Sniper, 48	0.404	0.312	0.518

(b)

Figure 5.5: *Comparison of homing to keyboard time for the two given pointing methods, for the different target widths. (a) Graph showing mean homing times. Target width influenced the outcomes for homing time. (b) Summary of mean homing time and 95% confidence interval for the two methods, for each target width.*

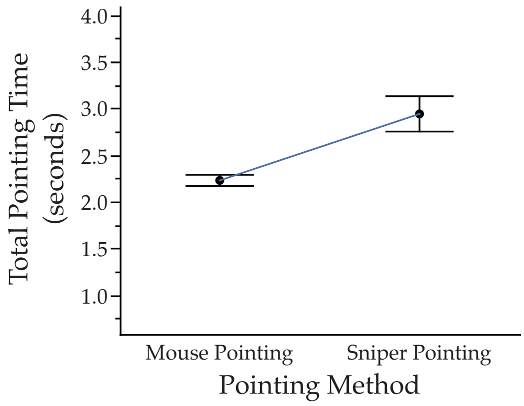
Source	DF	DFDen	F Ratio	Prob >F
Target Width (px)	3	194	8.73	<0.0001*
Target Distance (px)	1	194	0.74	0.3915
Width * Distance	3	194	0.56	0.6431
Pointing Method	1	193	2.67	0.1041
Width * Method	3	193	5.29	0.0016*
Distance * Method	1	193	0.09	0.7668
Width * Distance * Method	3	193	0.34	0.7934

Table 5.3: *Results for fixed effect test of homing to keyboard time. 'Target Width' had significant effect on results and there was interaction between 'Target Width' and 'Pointing Method'.*

5.10.2 Analysis: Homing to Keyboard Time

'Pointing Method' did not show significant effect on homing time to keyboard. 'Target Width' and 'Pointing Method' showed interaction.

Homing to keyboard time was measured as the time required to move the pointing hand back to the keyboard, to begin typing, after performing cursor manipulation. The fixed effect tests (table 5.3) showed significant effect of 'Target Width' ($F_{1,194}$=8.7255, $p<.0001$) on *homing to keyboard time*. Also there was interaction observed between 'Target Width' and 'Pointing Method'. For smaller target widths, homing time for the two methods were similar, but as target width became smaller, homing time for *Sniper Pointing* increased.



(a)

Method	Mean	Lower CI	Upper CI
Mouse Pointing	2.227	2.167	2.287
Sniper Pointing	2.939	2.750	3.129

(b)

Figure 5.6: *Comparison of total pointing time for the two different pointing methods. (a) Graph showing the resultant pointing times for the two methods.* Mouse Pointing *was overall slightly faster than* Sniper Pointing *(b) Summary of mean total pointing times and 95% confidence intervals for the two methods.*

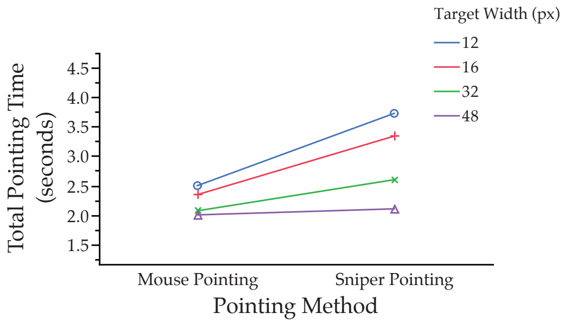
Source	DF	DFDen	F Ratio	Prob >F
Target Width (px)	3	194	70.56	<0.0001*
Target Distance (px)	1	194	8.12	0.0049*
Width * Distance	3	194	0.08	0.9704
Pointing Method	1	193	114.33	<0.0001*
Width * Method	3	193	14.09	<0.0001*
Distance * Method	1	193	1.48	0.2250

Table 5.4: *Results of fixed effect test for total pointing time. 'Target Width', 'Target Distance' and 'Pointing Method' had significant effect on the results and there was found to be strong interaction between 'Target Width' and 'Pointing Method'.*

5.10.3 Analysis: Total Pointing Time

Total pointing time was measured as the total time required to perform a single target acquisition task (given by equation 5.1). The fixed effect tests (table 5.4) showed significant effect of 'Target Width' ($F_{1,194}$=70.55, p<.0001), 'Target Distance' ($F_{1,194}$=8.11, p=0.0049), and 'Pointing Method' ($F_{1,193}$=114.33, p<.0001) on the results. 'Pointing Method' being the main independent variable, it was found that *Mouse Pointing*(average time = 2.205 seconds) was faster than *Sniper Pointing* (2.776 seconds).

'Pointing Method' significantly effected total pointing time. Mouse was overall faster than Sniper Pointing.



(a)

Method, Width	Mean	Lower CI	Upper CI
Mouse, 12	2.496	2.374	2.617
Sniper, 12	3.722	3.316	4.128
Mouse, 16	2.343	2.254	2.432
Sniper, 16	3.336	3.068	3.604
Mouse, 32	2.069	1.988	2.149
Sniper, 32	2.596	2.310	2.883
Mouse, 48	2.000	1.905	2.094
Sniper, 48	2.102	1.845	2.359

(b)

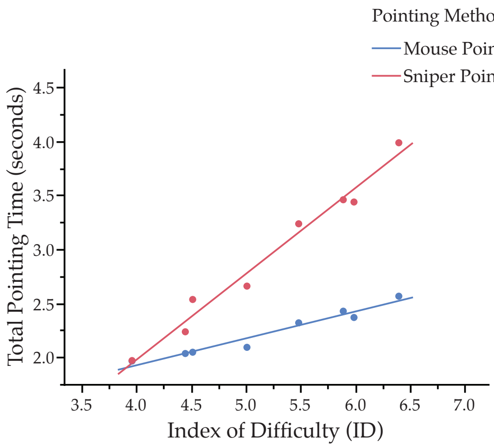
Figure 5.7: *Effect of target width on total pointing time for the given pointing methods. (a) Graph showing the resultant pointing times for the two methods. Significant difference between mouse and Sniper Pointing for smaller targets observed.(b) Summary of mean total pointing time for the two methods, for each target width.*

Index of Difficulty (ID in bits)	Distance (pixels)	Width (pixels)	t-ratio *t(13)*	Prob >|t| *p*
6.398	700	12	5.713	<0.0001*
5.989	1000	12	4.461	0.0006*
5.891	700	16	5.480	0.0001*
5.484	1000	16	5.696	<0.0001*
5.011	700	32	2.618	0.0213
4.516	1000	32	2.613	0.0215
4.448	700	48	0.023	0.9816
3.962	1000	48	1.110	0.2872

Table 5.5: *Results of paired samples* t*-test with Bonferroni correction. Prob >|t| is significant below 0.006 (0.05/8). Results show that the difference between mouse and* Sniper Pointing *are only significant for target widths below 32 pixels.*

Interaction between width and pointing method. Mouse significantly faster only for target widths less than 32 pixels.

Interaction between 'Target Width' and 'Pointing Method' ($F_{1,193}$=14.09, p<.0001) effected *total pointing time*. For larger targets (32 and 48 pixels), there was not much difference between the mouse and *Sniper Pointing*. As target width became smaller, the total pointing time for *Sniper Pointing* increased rapidly. In order to further analyse these results, a paired samples *t*-test with Bonferroni correction was performed for the different target width and distance combinations. It was found that *total pointing time* results were only significant for targets smaller than 32 pixels in width, for the two given pointing methods. Detailed results of the paired samples test are provided in table 5.5.



(i)

Pointing Method	Intercept (a) (seconds)	Slope (b) (seconds)	Throughput ($1/b$)	R^2
Mouse	0.9290	0.249	4.016	0.9533
Sniper	-1.2078	0.796	1.256	0.9767

(ii)

Figure 5.8: *Results of Fitts' Law Analysis for the two pointing methods. (i) Graph showing the relation between ID and total pointing time. (ii) Summary of Fitts' Law parameters for the two methods.*

5.10.4 Analysis: Fitts' Law Parameters

The total pointing time results were used to analyse the Fitts' law parameters for the two pointing methods. For each target width and distance combination, results were aggregated and averaged among all participants, for the given methods. This was used to estimate the Fitts' law parameters (a and b), which have been explained in detail in the previous chapter (cf. section 4.1.13).

Fitts' law parameters analysed using total pointing time results.

Figure 5.8 summarizes the results of analysis of Fitts' law parameters. It can be seen that the non-information aspect (a) for *Sniper Pointing* is much lower compared to that for mouse. On the other hand, *Sniper Pointing* also exhibits a higher value for slope (b), indicating that as ID increases, its performance degrades. It should be noted that the values for Fitts' parameters in figure 5.8 are theoretically derived from the results of the user study and although the intercept value (a) for *Sniper Pointing* is negative, this is not possible in practice.

Sniper Pointing exhibited a lower intercept (a) but greater slope (b) as compared to mouse.

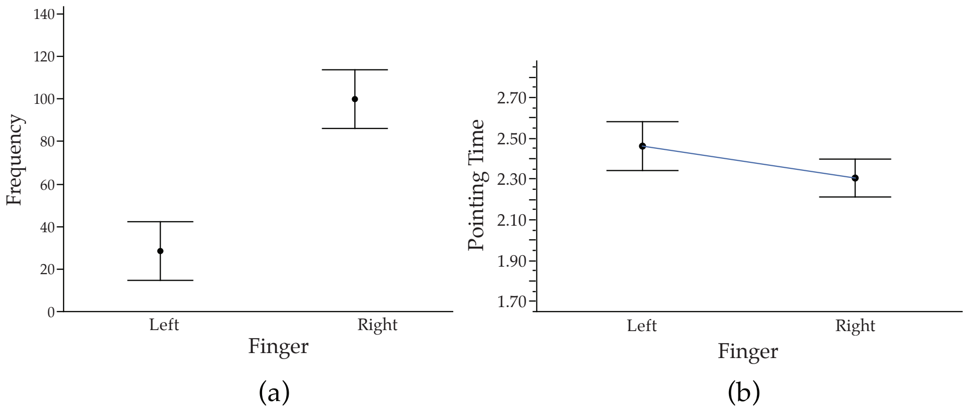


Figure 5.9: *Results for analysis of hand usage with Sniper Pointing. (a) Frequency with which right and left hand were used. (b) Average time required to point at targets.*

5.10.5 Analysis: Hand Usage with *Sniper Pointing*

Both hands used for pointing, with only small difference in pointing times.

While performing the point-and-type tasks with *Sniper Pointing*, users had the choice of using the right or left hand to point at targets. Figure 5.9 show that although the right hand was used with a higher frequency, users did also choose the left hand to point at targets. Figure 5.9 shows the average pointing time required for each of the hands. Pointing with the right hand was only slightly faster, and the confidence intervals (95%) for the two hands are seen to overlap.

5.11 Effect Size Measurement

The measurement of the effect size for various parameters involved during the quantitative analysis of *Sniper Pointing* was performed using *Cohen's d*. This is given by equation 5.2.

$$d = \frac{Sample\ Mean\ Difference}{Sample\ Standard\ Deviation} \tag{5.2}$$

The resultant effect sizes for the different phases involved in cursor manipulation have been outlined in table 5.6. It can be observed that while the results for *homing to point time* and *total pointing time* large effect sizes, the results for *homing to keyboard time* has only a small effect size.

Strong effect of *homing to point* and *total pointing time*.

Factor	Mean Difference	Standard Deviation	Cohen's d
Homing to Point Time	0.206	0.307	0.67
Homing to Keyboard Time	0.058	0.214	0.27
Total Pointing Time	0.712	0.874	0.81

Table 5.6: *Measurements of the effect size for the various factors evaluated during the evaluation of* Sniper Pointing.

5.12 Results and Implications

On evaluating *Sniper Pointing* and analysing the results from the performed user studies, a set of results can be deduced. The hypotheses, stated earlier in this chapter, can be verified or rejected using the given results.

Firstly, analysis of homing to pointing time for the two pointing methods have shown that *Sniper Pointing* is significantly faster than *Mouse Pointing*, rejecting null hypothesis *H1*, which states that homing to pointing time is not lesser while using *Sniper Pointing*, as compared to mouse. Measurement of *Cohen's d* indicates that the given result has a large effect size (0.67).

Null Hypothesis H1 rejected. Homing to pointing device significantly faster for *Sniper Pointing*.

Secondly, the quantitative analysis indicates no significant effect of 'Pointing Method' on the homing to keyboard time. Moreover, the effect size for these results is small (0.27). Hence, null hypothesis *H2*, which states homing to keyboard time is not lesser while using *Sniper Pointing*, can not be confirmed.

H2 can not be rejected.

Lastly, analysis of the overall pointing time shows that *Mouse Pointing* is faster than *Sniper Pointing*, rejecting null hypothesis *H3*, which states that there is no difference in total pointing time for the two methods. Measurement of *Cohen's d* for the given results indicate a large effect size (0.81). Along with 'Pointing Method', another factor

H3 rejected—Mouse overall faster than *Sniper Pointing*. Interaction between 'Target Width' and 'Pointing Method'.

that has significant effect on overall pointing time is 'Target Width'. It can be noted that the mouse is significantly faster than *Sniper Pointing* only for smaller target widths (32 pixels and less). As width increases, the difference between total pointing times for the two methods reduces, and *Sniper Pointing* exhibits pointing speeds similar to that of the mouse.

The results of this evaluation imply that although the mouse has been found to be faster than *Sniper Pointing*, the designed interaction technique does reduce the overall homing between pointing and typing time. Also, the interaction technique has been observed to be effective when targets are of adequate size and width. In the following section, further discussions and limitations of the user study are provided.

5.13 Discussion and Limitations

This section provides a brief discussion regarding the evaluation of *Sniper Pointing*, and factors that could have influenced the results are mentioned. This section also highlights the potential for using two-handed interactions, as observed during the trials performed by users, and mentions possibilities of improvements. Finally, some qualitative feedback from users about *Sniper Pointing* are also mentioned.

Experience with commonly used commercial devices, like mouse, plays a vital role and can influence the results.

In a comparison of a novel interaction technique, like *Sniper Pointing* to one that is used on widespread scale, like *Mouse Pointing*, it is important to take into consideration that most or all users have been exposed to the latter for prolonged periods of time, and that experience is an important influencing factor, which can affect the results obtained. Although the designed user study provided the users with a training stage, where they were given a chance to gain some experience with the given pointing methods, this can not entirely compensate for users' familiarity with a device like the Mouse. Hence, it could be possible that further training with the novel interaction technique could influence the results, and provide a fairer ground for comparison. However, due to the limited time frame of this thesis, this has not been possible. The alternative to this, finding users who are unfamiliar with the Mouse, is also not an easy task and was not a feasible option. It can be hence concluded, that although the outcome of the study favoured the Mouse, the results for *Sniper Pointing* are still promising.

Another interesting observation during the course of the user trials was the effectiveness of two-handed interactions (mentioned in chapter 3.6.3). Although all participants were dominantly right-handed, it was observed that participants were able to effectively perform target acquisition tasks with their non-dominant hand. An analysis of hand usage (chapter 5.10.5) also supported this observation. After a few attempts, the participants managed to successfully manipulate the cursor and acquire targets with their non-dominant index finger. This was particularly beneficial to divide the screen into two logical halves—left and right, and to utilize the respective hand for target acquisition, depending on the location of targets.

Two-handed interactions have been found to be effective. Users can also perform cursor manipulation with non-dominant hand, with the given interaction technique.

Observing users' performing the studies also revealed some shortcomings and possible room for improvement in the mechanism for switching the pointing finger, for cursor manipulation. The given application determined the pointing finger—left or right, based solely on the height of the two fingers above the keyboard. It was assumed that the finger higher above was currently responsible for manipulations. Hence, accidentally lifting the other index finger above the intended pointing finger caused the application to switch the pointing finger, and in turn re-position the cursor. This led to unintended actions, and was a possible source of errors. In order to eliminate this issue, a more sophisticated technique for determining the intended pointing finger should be developed. It is possible to either improve the recognition system, and allow for such accidental movements by the users, or to design the interaction in such a way that enforces users to explicitly indicate a finger switching action.

Mechanism used to detect hand switching is not perfect. Can be source of some errors, and more sophisticated solutions are needed.

Finally, during the user study, each of the participants were asked about their opinions and impressions of *Sniper Pointing* and the feedback were encouraging as well as informative. Users founds the interaction to be intuitive and fluid.

Positive qualitative opinions regarding usability and intuitiveness of *Sniper Pointing*.

Some users even stated that it made the task more *"pleasant"*, as compared to the mouse. Two participants indicated the willingness to use *Sniper Pointing* on a daily basis, as a replacement for the mouse, which was a particularly motivating feedback. One of the participants commented—*"[Since] it is novel, you are not used to it. But it is easy to get used to."* Participants also appreciated the availability of two different granularities, offered by *Sniper Pointing*, and admitted that having a quick, coarse movement followed by a slow, fine movement was advantageous for the pointing tasks.

Future works can build on the results of this evaluation to provide improved interactions.

To conclude, although the overall results from analysis of *Sniper Pointing* have indicated that the mouse exhibits better performance in terms of overall pointing speed, the designed interaction does successfully manage to achieve some of the design goals, like reducing time required to switch between devices, while being user-friendly and intuitive, as observed from qualitative opinions of test participants. Future works can use the results from this evaluation to optimize the design and to successfully apply this interaction technique in work environments. Chapter 6 provides a summary of this thesis along with some possible future works in this field.

Chapter 6

Summary and future work

This thesis has taken a look into various aspects related to cursor manipulation in desktop environments. Currently prevalent techniques and technologies have been reviewed and various aspects that contribute to the speed of performing this manipulation have been analysed. A novel interaction technique for cursor manipulation has been designed and evaluated. This chapter summarizes the work performed over the course of this thesis and also highlights some possible future work which could enhance performance and enrich the user experience.

6.1 Summary and contributions

Mouse pointing is not optimal.

Pointing task divided into initial visual search, homing between devices, and cursor manipulation.

Chapter 1 highlighted that although the mouse is one of the most widely used pointing devices, some of its physical and operational characteristics negatively influence its performance. Division of a single pointing task into a series of pointing phases showed that the factors that contribute to overall time required to perform such a task include visually searching a cursor, homing between the keyboard and pointing device, and cursor manipulation. The interaction technique presented in this thesis, titled *Sniper Pointing*, attempts to reduce this overall time by either eliminating or optimizing these pointing phases.

Sniper Pointing eliminates the need for time-consuming initial visual searches by using absolute pointing.

Before the mouse cursor can be manipulated, relative pointing requires users to visually locate its on-screen position. This initial visual search is time-consuming and redundant. *Sniper Pointing* addresses this issue by using absolute pointing techniques, which maps the position in input space to corresponding location of the cursor in the output space. Mapping the keyboard frame to the screen eliminates the need for an initial visual search, and allows for immediate cursor manipulation.

Fluid switching between typing and pointing reduces homing time in *Sniper Pointing*.

In typical usage scenarios, pointing is performed in conjunction with typing, making homing between the keyboard and pointing device necessary. The time required for homing is a function of the amount of physical separation between the two devices. Previous researches and the evaluation of *Sniper Pointing* (chapter 5) have shown homing between the keyboard and mouse to be time-consuming. On the other hand, *Sniper Pointing* utilizes the volume directly above the keyboard for cursor manipulation. By reducing the distance between the keyboard and pointing mechanism, and by making the physical separation less prominent, homing time is dramatically reduced, and fluid switching between typing and pointing is provided.

Layered interaction, involving quick coarse pointing and accurate fine pointing, addresses speed and accuracy issues.

After homing to the pointing device, cursor manipulation allows users to point at on-screen targets. This can include a quick ballistic movement towards the target, followed by a corrective movement to reach the specific location. *Sniper Pointing* also presents a new interaction technique for effective cursor manipulation. A *layered* interaction design provides users with two different pointing granularities—a coarse pointing layer offers fast pointing; and a fine pointing layer allows for fine-grain manipulation of the cursor. By doing so, Sniper Pointing attempts to address both speed and accuracy issues, while performing cursor manipulation.

User studies performed and ergonomic principles applied to optimize the design.

To optimize the design and implementation of the interaction technique, preliminary studies have been conducted over the course of this thesis (chapter 4). Additionally, factors like fatigue during prolonged use have been considered by paying attention to ergonomics and applying some principles from basic biomechanics and workstation design (Marras [2006]). This has been especially used to determine the height of the different pointing layers, so as to minimize strain on the arm caused by vertical movements.

Preliminary studies performed to compare absolute and relative pointing techniques, and to determine the optimal arrangement of pointing layers for *Sniper Pointing*.

The first preliminary study has been used to analyse the feasibility of using absolute techniques, in desktop environments, to provide cursor manipulation. Additionally, the effect of input-space scale on the pointing time has been analysed, in order to determine whether it is feasible to use input spaces which are smaller in comparison to the output space. Although comparison between relative and absolute pointing has yielded inconclusive results, is has been found that it is possible to reduce the size of the input space without significant degradation in performance. The second preliminary study has been conducted to determine an optimal arrangement of coarse and fine pointing layers, for the *Sniper Pointing* interaction technique. The results have clearly shown that the appropriate design for this technique is to provide coarse pointing directly above the keyboard, and to allow users to perform fine pointing by raising the pointing hand higher above.

User study performed to evaluate *Sniper Pointing* and to compare it with the mouse.

To evaluate *Sniper Pointing*, the resulting interaction technique has been tested with users and a quantitative analysis has been performed (chapter 5). While doing so, the *Sniper Pointing* interaction has been compared to traditional mouse pointing, in a standard desktop environment. Although results from evaluation show mouse pointing to be faster overall, *Sniper Pointing* has managed to address issues like homing between devices, and homing time has been observed to be significantly lesser using this technique, as compared to the mouse. Also, *Sniper Pointing* has shown to be effective for larger targets. It should be noted that these evaluation results could be influenced by factors such as experience. While participants of the study have had significant amount of experience using the mouse, or similar devices like the trackpad, *Sniper Pointing* is a completely new and unfamiliar technique. It can be advantageous to further investigate this technique and there is room further improvements, which can make it a stronger competitor to traditional pointing devices.

6.2 Future work

This thesis has introduced a new interaction technique which uses the volume above the keyboard surface, to provide for on-screen cursor manipulation. Being a novel technique, developed from ground-up, there is the possibility for performing further evaluation of the system and to improve the overall interaction and performance. This section highlight some possible scope for future works, and offers a brief insight into future prospects of *Sniper Pointing* and mid-air cursor manipulation.

Study the effect of initial visual search on results.

The introduction pointed out that a visual search for an on-screen cursor is time-consuming and redundant. However, the evaluation of *Sniper Pointing* has not taken this sub-task into consideration, while comparining it with the mouse. Although the human processor model provides an insight into the theoretical time required to perform such a search, a quantitative analysis of the effect of the initial search could impact the evaluation results of *Sniper Pointing*. Such an an analysis could be performed by conducting an eye-tracking study (Chen et al. [2001]).

Possible application in settings with larger displays.

Further, the integration of *Sniper Pointing* into settings with large output screens is a potential domain where such an interaction technique could be beneficial. The fact that *Sniper Pointing* uses a small input area, irrelevant to the size of the display, can be advantageous, since it allows for efficient scaling of display sizes, without drastic reduction in pointing performance (cf. 2.1.2—Hybrid Pointing). However, to verify this prediction, an analysis of the technique in such an environment must be conducted.

Possibility of adding more layers of granularities to the interaction technique.

Another interesting aspect for future research is the increase in the number of granularity layers. While *Sniper Pointing* uses two discrete pointing layers—coarse and fine, it can be possible to increase this number, to provide various levels of granularity. Another technique that can be investigated is the use of a continuous spectrum of granularities, where the vertical height in the input space is continuously mapped to the output granularity of the cursor.

The current implementation of *Sniper Pointing* uses two different screen cursors—coarse and fine (figure 3.8). Both these cursors use single points of activation, marked by the centres of their cross-hairs. Kabbash and Buxton [1995] presented the "Prince" technique for target selection using area cursors. Here, instead of having a single point of activation, a larger area of the screen is activated when the cursor is over it. A similar technique could be integrated into *Sniper Pointing*, to improve the pointing performance. In such an implementation, it could be advantageous to use an area cursor for coarse pointing, and a cross-hair for fine pointing (figure 6.1).

An area cursor can be used for coarse pointing.

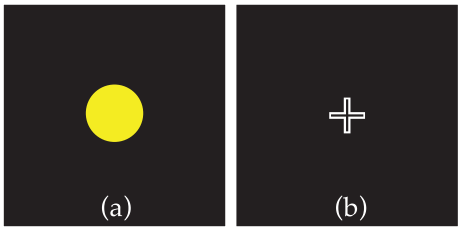


Figure 6.1: *An illustration of the use of an area cursor in* Sniper Pointing*. (a) An area cursor can be used for coarse pointing. (b) The cross-hair is used for fine pointing.*

While *Sniper Pointing* applies absolute pointing for both, coarse and fine pointing, the effect of using a hybrid technique can also be investigated. It could be feasible to apply absolute pointing in the coarse layer, followed by relative pointing while performing fine-grain manipulation.

Combination of absolute and relative techniques can be tested.

The combination of new hardware technologies like Leap Motion[1], which provide highly accurate input information in three dimensions, along with further development in interaction techniques like *Sniper Pointing*, which facilitate fast and accurate cursor manipulation, can make it possible to replace or augment traditional pointing devices, like the mouse, with newer techniques for cursor manipulation. These can be adapted appropriately for various settings and can be optimized for the given environment, making pointing more user-friendly and efficient.

[1] www.leapmotion.com

Bibliography

J. Accot and S. Zhai. Performance Evaluation of Input Devices in Trajectory-Based Tasks: An Application of the Steering Law. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: The CHI is the Limit*, pages 466–472. ACM, 1999.

W. Buxton, M. Billinghurst, Y. Guiard, A. Sellen, and S. Zhai. Human Input to Computer Systems: Theories, Techniques and Technology, 2002.

S.K. Card, W.K. English, and B.J. Burr. Evaluation of Mouse, Rate-Controlled Isometric Joystick, Step Keys, and Text Keys for Text Selection on a CRT. *Ergonomics*, 21(8):601–613, 1978. doi: 10.1080/00140137808931762.

S.K. Card, T.P. Moran, and A. Newell. *The Psychology of Human-Computer Interaction*, pages 23–44. CRC, 1986.

G. Casiez and N. Roussel. No More Bricolage!: Methods and Tools to Characterize, Replicate and Compare Pointing Transfer Functions. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, pages 603–614. ACM, 2011.

G. Casiez, N. Roussel, and D. Vogel. 1€ Filter: A Simple Speed-Based Low-Pass Filter for Noisy Input in Interactive Systems. In *Proceedings of the 2012 ACM Annual Conference on Human Factors in Computing Systems*, pages 2527–2530. ACM, 2012.

M.C. Chen, J.R. Anderson, and M.H. Sohn. What can a mouse cursor tell us more?: Correlation of Eye/Mouse Movements on Web Browsing. In *CHI'01 Extended Abstracts on Human Factors in Computing Systems*, pages 281–282. ACM, 2001.

S.A. Douglas and A.K. Mithal. The Effect of Reducing Homing Time on the Speed of a Finger-Controlled Isometric Pointing Device. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems: Celebrating Interdependence*, pages 411–416. ACM, 1994.

W.K. English, D.C. Engelbart, and M.L. Berman. Display-Selection Techniques for Text Manipulation. *Human Factors in Electronics, IEEE Transactions on*, HFE-8(1):5–15, March 1967.

C. Forlines, D. Vogel, and R. Balakrishnan. HybridPointing: Fluid Switching Between Absolute and Relative Pointing with a Direct Input Device. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*, pages 211–220. ACM, 2006a.

C. Forlines, D. Vogel, N. Kong, and R. Balakrishnan. Absolute vs. Relative Direct Pen Input. *Mitsubishi Electric Research Labs Tech Report, TR2006-066*, 2006b.

M. Gokturk and J.L. Sibert. An Analysis of the index Finger as a Pointing Device. In *CHI'99 Extended Abstracts on Human Factors in Computing Systems*, pages 286–287. ACM, 1999.

K. Hinckley. The human–computer interaction handbook. chapter Input Technologies and Techniques, pages 151–168. L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 2003. ISBN 0-8058-3838-4.

ISO. 9241-9 Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs)—Part 9: Requirements for Non-Keyboard Input Devices (Final Draft International Standard). *International Organization for Standardization*, 2000.

P. Kabbash and W.A.S. Buxton. The "Prince" Technique: Fitts' Law and Selection Using Area Cursors. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95, pages 273–279, 1995. ISBN 0-201-84705-1. doi: 10.1145/223904.223939.

I.S MacKenzie, A. Sellen, and W.A.S. Buxton. A Comparison of Input Devices in Element Pointing and Dragging Tasks. In *Proceedings of the SIGCHI Conference on Human*

Factors in Computing Systems, CHI '91, pages 161–166, New York, NY, USA, 1991. ACM. ISBN 0-89791-383-3.

W.S. Marras. Basic Biomechanics and Workstation Design. *Handbook of Human Factors and Ergonomics*, pages 340–370, 2006.

D.C. McCallum and P. Irani. ARC-pad: Absolute + Relative Cursor Positioning for Large Displays with a Mobile Touchscreen. In *Proceedings of the 22nd annual ACM Symposium on User Interface Software and Technology*, pages 153–156. ACM, 2009.

T. Moscovich and J.F. Hughes. Indirect Mappings of Multi-Touch Input Using One and Two Hands. In *Proceedings of the 22th annual SIGCHI Conference on Human Factors in Computing Systems*, pages 1275–1284. ACM, 2008.

T.A. Mysliwiec. FingerMouse: A Freehand Computer Pointing Interface. *University of Illinois at Chicago, Tech. Rep. VISLab-94-01*, 1994.

M. Nancel, J. Wagner, E. Pietriga, O. Chapuis, and W. Mackay. Mid-Air Pan-and-Zoom on Wall-Sized Displays. In *Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems*, pages 177–186. ACM, 2011.

M. Ortega and L. Nigay. AirMouse: Finger Gesture for 2d and 3d Interaction. *Human-Computer Interaction–INTERACT 2009*, pages 214–227, 2009.

D. Schmidt, F. Block, and H. Gellersen. A Comparison of Direct and Indirect Multi-Touch Input for Large Surfaces. *Human-Computer Interaction–INTERACT 2009*, pages 582–594, 2009.

D. Vogel and R. Balakrishnan. Distant Freehand Pointing and Clicking on Very Large, High Resolution Displays. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*, pages 33–42. ACM, 2005.

A. Wilson and E. Cutrell. FlowMouse: A Computer Vision-Based Pointing and Gesture Input Device. *Human-Computer Interaction–INTERACT 2005*, pages 565–578, 2005.

S. Zhai. Characterizing Computer Input with Fitts' Law Parameters: The Information and Non-Information Aspects of Pointing. *International Journal on Human–Computer Studies*, 61(6):791–809, December 2004. ISSN 1071-5819.

Index

Typeset February 23, 2013