Laboratoire de Recherche en Informatique

École Doctorale d'Informatique de l'Université Paris-Sud

Laboratoire de Recherche en Informatique

THÈSE DE DOCTORAT

présentée par : **Mathieu Nancel**

soutenue le : **5 Décembre 2012**

pour obtenir le grade de : **Docteur**

Discipline / Spécialité : **Informatique / Interactions Homme-Machine**

Designing and Combining Interaction Techniques in Large Display Environments

THÈSE dirigée par

Pr. BEAUDOUIN-LAFON Michel	*Université Paris-Sud XI, France*
Dr. PIETRIGA Emmanuel	*INRIA & INRIA Chile*

PRÉSIDENT DU JURY

Pr. Martin Jean-Claude	*Université Paris-Sud*

RAPPORTEURS

Pr. Balakrishnan Ravin	*University of Toronto*
Pr. Guiard Yves	*TELECOM-ParisTech*

EXAMINATEURS

Dr. Casiez Géry	*Université Lille 1*
Dr. Grossman Tovi	*Autodesk Research*

À Andrée Nancel, qu'a pas pu venir.

Remerciements

Ces quatre ans et demie de thèse ont été une expérience aussi agréable qu'enrichissante, et je souhaite en remercier tous les acteurs.

Je voudrais remercier en premier lieu mes directeurs de thèse, Michel Beaudouin-Lafon et Emmanuel Pietriga, qui ont accepté de m'accompagner dans cette expérience et qui m'ont guidé jusqu'à son aboutissement.

Via ses cours de Master 2, Michel m'a fait découvrir l'Interaction Homme-Machine et la recherche dans ce domaine. Il m'a ensuite accepté en stage et guidé dans le monde merveilleux des menus, des designs expérientaux et de la rédaction scientifique. C'est également lui qui m'a confié ce sujet de thèse en or : "*On va construire un mur d'écrans, tu pourrais faire une thèse là-dessus.*"[1]. Michel, j'ai énormément appris à tes côtés, sur tous les aspects de la recherche et de la vie de chercheur, et espère pouvoir continuer à travailler avec toi l'avenir.

Emmanuel a accepté d'être mon directeur de thèse sans qu'on n'ait jamais travaillé ensemble auparavant, et malgré mes amis banquiers. Il m'a fait confiance dès le début sur mes choix de projets, et a toujours pris le temps de répondre à mes questions, entendre mes explications parfois embrouillées, et soutenir mes idées malgré leur incrémentalité manifeste. Il a également supporté une quantité de mes blagues qui auraient poussé la plupart des gens à m'agresser physiquement. J'ai adoré travailler avec toi et espère aussi pouvoir le faire à nouveau. Live long and prosper.

Je voudrais également adresser mes plus chaleureux remerciements aux membres de mon jury: Yves Guiard et Ravin Balakrishnan, les deux rapporteurs de ma thèse, pour avoir pris le temps de lire mon manuscrit, d'en estimer la qualité et d'y suggérer des améliorations. Géry Casiez, Tovi Grossman et Jean-Claude Martin, pour avoir accepté d'être examinateurs de ma thèse ainsi que pour leurs questions et suggestions durant et après ma soutenance. Un merci supplémentaire à Jean-Claude Martin pour avoir accepté de présider à mon jury de thèse.

Ces cinq (!) années passées au sein d'InSitu, en stage puis en thèse, ont été de très loin les plus agréables et les plus intéressantes de ma scolarité et de ma vie professionnelle, ce que je dois à l'ensemble de l'équipe. Il y règne une ambiance que j'ai sincèrement peur de ne retrouver nulle part ailleurs : tout le monde parle avec tout le monde, blague avec tout le monde, travaille avec tout le monde, tous postes, statuts et affiliations confondus. Remercier tous ceux qui ont participé d'une manière ou d'une autre à mon travail ou à mon expérience de ces cinq années ne m'exposerait qu'au risque d'oublier quelqu'un, cependant il est certaines personnes à qui je suis spécialement redevables :

Wendy MacKay, chronologiquement la deuxième personne à m'avoir montré ce qu'était l'IHM et la recherche, pour m'avoir accepté dans l'équipe, pour ses conseils, et plus récemment pour m'avoir gardé une place de Student Volunteer au PC.

Olivier Chapuis ne s'est jamais lassé de répondre à mes questions sur le pointage, le design d'expériences et l'analyse de leurs résultats, qui sont respectivement le domaine pré-dominant de ma thèse et les outils principaux de la recherche qu'elle contient. Autant dire qu'il a passé beaucoup de temps à m'écouter et à me répondre, ce pour quoi je lui suis extrêmement reconnaissant.

Clément Pillias et Romain Primet sont virtuellement responsables de ce que j'ai fait durant ma thèse. Sans leur gigantesque travail pour rendre l'utilisation de la plate-forme WILD accessible aux

[1] Ça ne s'est pas vraiment passé comme ça.

gens du commun , e.g. moi, je n'aurais pas pu développer ni faire passer la moitié des expériences contrôlées qui composent ma thèse. Outre leur travail indispensable, ils ont également pris le temps de répondre à toutes mes (nombreuses et fréquentes) questions. Dans mon ingratitude, je suis même allé jusqu'à leur faire passer ces expériences qu'ils ont contribuer à créer. Clément, je n'oublie pas ces trois heures de vie que je te dois. Romain, être gaucher ne te sauvera pas toujours.

Enfin, un merci immense à tous les participants de mes expériences contrôlées et à leur infinie patience. Il en est peut-être qui ont participé à *toutes* mes expériences, du début de mon stage à la fin de ma thèse, soit plus d'une quinzaine. Vous êtes fous. J'aurais refusé au bout de la deuxième.

C'est à ce niveau de mes remerciements, entre les collègues et les proches, que je me dois de mentionner Julie Wagner, Fanny Chevalier et Stéphane Huot.

Stéphane partage le mérite (et la faute) de Romain et Clément quant à mon utilisation de WILD, en cela que sa contribution technique a été indispensable à mes travaux. J'ai beaucoup appris à travailler à ses côtés, dans tous les aspects de la recherche en IHM. Mais il ne s'est pas limité à ça, et m'a aussi énormément appris sur la cuisine, la musique, Benoit Poelvoorde, la bière, et j'en passe. Il m'a également mené à des situations aussi géniales qu'improbables, comme l'enregistrement d'une chanson qualité studio (à part le mastering), le tournage et le montage de *deux* clips, et le fait, probablement rare, d'enchainer des festivals avec un permanent de son équipe.

Je partage avec Julie mes plus prestigieuses publications, et travailler avec elle peut être une expérience des plus enrichissantes tant que l'utilisation de Skype peut être évitée. Elle a aussi partagé par sa présence l'essentiel de ma rédaction de thèse (praise the Starbucks) et la majorité de mes soirées bière ("*Une... blanche ? C'est ça ?*"), et a grandement amélioré mes connaissances sur la ville de Paris, le party-crashing et le métier de cascadeur.

Fanny, malgré la distance, a toujours été au bout de son Skype pour partager les bons comme les mauvais moments ; toujours de bon conseil, et incomparable pour raconter des conneries, les entretenir et les faire évoluer. Je pense pas n'avoir jamais développé un tel vocabulaire de private jokes avec qui que ce soit d'autre, ce qui n'est pas peu dire. Dromadaire.

Un immense merci à vos parents pour votre existence.
Changez rien.

Mes propres parents, Claude Abécassis et Robin Nancel, m'ont soutenu durant toutes mes études, m'ont fourni le confort matériel qui m'a permis de les effectuer avec l'insouciance qui fut la mienne, mais ont aussi su me ramener à la réalité quand c'était nécessaire. Merci pour tout.

Je remercie également tous les gens du JdR, qui se reconnaitront, pour les discussions qu'on a pu avoir sur ma recherche, qui m'ont souvent permis de mieux formuler mon travail et de le mettre en perspective, mais aussi simplement pour avoir été là et pour m'avoir fourni sans le savoir une alternative (au moins) hebdomadaire à mon travail.

Finalement, une mention spéciale pour Joyce, Quentin, Amandine et tous les autres employés du Starbucks de la rue Lafayette pour leur gentillesse, les cafés offerts et les coups de tampon en trop.

Cette thèse a été soutenue par l'attribution d'une allocation doctorale DIGITEO et la Région Île-de-France.

Abstract

Large display environments (LDEs) are interactive physical workspaces featuring one or more static large displays as well as rich interaction capabilities, and are meant to visualize and manipulate very large datasets. Research about mid-air interactions in such environments has emerged over the past decade, and a number of interaction techniques are now available for most elementary tasks such as pointing, navigating and command selection. However these techniques are often designed and evaluated separately on specific platforms and for specific use-cases or operationalizations, which makes it hard to choose, compare and combine them.

In this dissertation I propose a framework and a set of guidelines for analyzing and combining the input and output channels available in LDEs. I analyze the characteristics of LDEs in terms of (1) visual output and how it affects usability and collaboration and (2) input channels and how to combine them in rich sets of mid-air interaction techniques. These analyses lead to four design requirements intended to ensure that a set of interaction techniques can be used (i) at a distance, (ii) together with other interaction techniques and (iii) when collaborating with other users. In accordance with these requirements, I designed and evaluated a set of mid-air interaction techniques for panning and zooming, for invoking commands while pointing and for performing difficult pointing tasks with limited input requirements. For the latter I also developed two methods, one for calibrating high-precision techniques with two levels of precision and one for tuning velocity-based transfer functions. Finally, I introduce two higher-level design considerations for combining interaction techniques in input-constrained environments. Designers should take into account (1) the trade-off between minimizing limb usage and performing actions in parallel that affects overall performance, and (2) the decision and adaptation costs incurred by changing the resolution function of a pointing technique during a pointing task.

Keywords : large display environments, LDE, ultra-high-resolution displays, feedback location, task allocation, pan-and-zoom, on-body touch, dual-precision pointing, head orientation, pointer acceleration, decision, adaptation

Résumé

Les environnements à grands écrans (Large Display Environments, LDE) sont des espaces de travail interactifs contenant un ou plusieurs grands écrans fixes et divers dispositifs d'entrée ayant pour but de permettre la visualisation et la manipulation de très grands jeux de données. La recherche s'est de plus en plus intéressé à ces environnements durant ces dix dernières années, et il existe d'ores-et-déjà un certain nombre de techniques d'interaction correspondant à la plupart des tâches élémentaires comme le pointage, la navigation et la sélection de commandes. Cependant, ces techniques sont souvent conçues et évaluées séparément, dans des environnements et des cas d'utilisations spécifiques. Il est donc difficile de les comparer et de les combiner.

Dans ce manuscrit, je propose un ensemble de guides pour l'analyse et la combinaison des canaux d'entrée et de sortie disponibles dans les LDEs. Je présente d'abord une étude de leurs caractéristiques selon deux axes: (1) le retour visuel, et la manière dont il affecte l'utilisabilité des techniques d'interaction et la collaboration co-localisée, et (2) les canaux d'entrée, et comment les combiner en d'efficaces ensembles de techniques d'interaction. Grâce à ces analyses, j'ai développé quatre pré-requis de conception destinés à assurer que des techniques d'interaction peuvent être utilisées (i) à distance, (ii) en même temps que d'autres techniques et (iii) avec d'autres utilisateurs. Suivant ces pré-requis, j'ai conçu et évalué un ensemble de techniques de navigation, d'invocation de commandes tout en pointant, et de pointage haute-précision avec des moyens d'entrée limités. J'ai également développé deux méthodes de calibration de techniques de pointage, l'une spécifique aux techniques ayant deux niveaux de précision et l'autre adaptée aux fonctions d'accélération. En conclusion, j'introduis deux considérations de plus haut niveau sur la combinaison de techniques d'interaction dans des environnements aux canaux d'entrée limités : (1) il existe un compromis entre le fait de minimiser l'utilisation des membres de l'utilisateur et celui d'effectuer des actions en parallèle qui affecte les performances de l'ensemble ; (2) changer la fonction de transfert d'une technique de pointage durant son utilisation peut avoir un effet négatif sur les performances.

Mots clés : environnements à grands écrans, LDE, écrans ultra-haute-résolution, retour visuel, assignation de tâches, pointage haute-précision, orientation de la tête, fonctions d'accélération, décision, adaptation

Contents

Introduction **19**

I Combination of pointing and other interactions on wall-sized displays **23**

1 Designing Mid-air Interactions for Large Display Platforms **25**
1.1 Desktop Environments VS. Large Display Platforms 25
1.2 Location of the feedback 33
1.3 Task allocation strategies 38
1.4 Requirements 42

2 Pointing and navigation on wall-sized displays **45**
2.1 Introduction 45
2.2 Panning and Zooming in Mid-air 47
2.3 Design Space for Mid-Air Pan-and-Zoom Input 49
2.4 Design Choices 52
2.5 Experiment 54
2.6 Summary and Discussion 66

3 Device-less Discrete Selection in Large Display Platforms **71**
3.1 Introduction 71
3.2 Related Work 72
3.3 Design space for on-body touch interactions 77
3.4 Experiment 81
3.5 Conclusion 94

Conclusion of Part I: Parallelism and Limb Usage **97**

II Dual-Precision techniques **101**

4 The limits of single-precision techniques **105**
4.1 Introduction . . . 105
4.2 Matching the limits of human perception . . . 106
4.3 Experiment: Limits of single-mode Techniques . . . 115
4.4 Conclusion . . . 121

5 Dual-precision **123**
5.1 Introduction . . . 123
5.2 Experiment: Dual-Precision Techniques . . . 132
5.3 Implications for design . . . 142
5.4 Conclusion and Future Work . . . 145

6 Dual-Precision Techniques on Touch Devices **147**
6.1 Introduction . . . 147
6.2 Techniques . . . 148
6.3 Control-Display Transfer Functions . . . 155
6.4 Comparing the Dual-Precision Techniques . . . 159
6.5 Comparison with State-of-the-Art Techniques . . . 168
6.6 Conclusion . . . 173

Conclusion of Part II: Decision and Adaptation **175**

Conclusion and Perspectives **181**

Bibliography **187**

Appendices **213**

A General apparatus: the WILD platform **213**

B Preliminary Experiment with Single-Mode Pointing Techniques **215**
B.1 Participants . . . 215
B.2 Apparatus . . . 216
B.3 Task . . . 216
B.4 Predictions . . . 217
B.5 Results . . . 217

B.6 Summary 220

C Head and Chest Orientations **221**

C.1 Participants 222
C.2 Apparatus 223
C.3 Task 223
C.4 Design 224
C.5 Measures 225
C.6 Results 225
C.7 Summary 230

Glossary **230**

Glossary **231**

List of Tables

1.1 Design space for *Feedback Location* 37

2.1 Key Dimensions of the Design Space 50
2.2 Results of the full factorial ANOVA for MT. 61
2.3 Groups of techniques according to MT 65

3.1 Menu techniques available for mid-air interaction according to the *Feedback Location* design space 75

4.1 Device characteristics for relative pointing techniques 114

6.1 Transfer function parameter values. 159

C.1 $CursorPos = a.HeadPos + b$ correlation for all the data 227

List of Figures

1 Mid-air interaction in a large display platform 19

1.1 A user looking at a pixel higher than his point of view 28
1.2 Display physical size over visualization size 29
1.3 Main differences between interacting on a desktop and mid-air in a LDE . . 33
1.4 Examples of the *biTouch* framework 40
1.5 Wireless mouse held in mid-air 40
1.6 Examples of task allocation strategies 42

2.1 A typical sequence of virtual navigation steps 47
2.2 Dynamic zooming 50
2.3 The 12 techniques organized according to our key characteristics 51
2.4 Participant performing the task 56
2.5 Schematic representation of the pan-and-zoom tasks 57
2.6 Criterion for the practice blocks 59
2.7 *MT* per Hands, Guidance and Gesture 60
2.8 *MT* per Dist × Gesture, for each Guidance 63
2.9 Technique performance depending on groups 69

3.1 Body-part tally system of the Fasu, Southern Highlands Province 77
3.2 Morphological design space of on-body touch interactions. 79
3.3 Apparatus of the on-body touch experiment 82
3.4 18 body targets are grouped into five categories. 83
3.5 Timeline of the trials in the on-body touch experiment 84
3.6 Main conditions of the on-boty touch experiment 85
3.7 Mean Trial Time for all targets 87
3.8 Mean Trial Time for all tasks by pointing difficulty 89
3.9 Interaction Pointing×On-Body Touch on Pointing movement time. . . . 90
3.10 Effect of Pointing difficulty and Body target group on Cursor readjustment time. 91

3.11 Interaction Pointing×On-Body Touch on Body pointing time. 92
3.12 Preference and social acceptance of all body targets 94
3.13 Trade-off between limb usage, performance and parallelization of interactions 98

4.1 The interactive wall-sized used for our studies 106
4.2 Clutch and precision problems 110
4.3 The expressiveness paradigm of CD gains 113
4.4 Theoretical CD_{min} and CD_{max} for the four selected devices 115
4.5 *RayCasting* and *Gyro* 115
4.6 The platform used for the single-mode experiment 116
4.7 A trial in the single-mode experiment 117
4.8 (a) *Misses* and (b) *Pointing Time* per Technique × Width. 118

5.1 Typical use of a dual-precision technique 125
5.2 *Laser+Position* 129
5.3 *Laser+Gyro* 129
5.4 *Laser+Track* 130
5.5 Zones used for the *Laser+Track* technique 130
5.6 The dual-mode techniques cursor 131
5.7 Dual-precision cursor before switching back to precise mode 131
5.8 *Misses* and *Pointing Time* per Technique × Width 134
5.9 Subjective results for the one-handed, dual-precision experiment 138
5.10 *Pointing Time* and *Misses* for similar difficulties 141
5.11 Trade-off between reliability and number of commands 144

6.1 Interacting with a wall-sized display using a tablet 148
6.2 Ray-casting vs. indirect absolute mapping 151
6.3 Angular abscissa of the cursor depending on head orientation 152
6.4 Cursor of the *Discrete* techniques in coarse mode 153
6.5 The four dual-precision techniques 155
6.6 An example of logistic sigmoid curve 156
6.7 Example of curves with a varying λ 157
6.8 Location of the targets in all three experiments 161
6.9 Transfer functions used for the tablet experiment 163
6.10 *MT* for each Tech for the tablet experiment 163
6.11 Transfer functions used for the smartphone experiment 165
6.12 *MT* for each Tech for the smartphone experiment 166
6.13 Transfer functions used for the third experiment 169
6.14 The two input zones delimited by thick removable tape. 169

6.15 *MT* for each TECH by W and by A for the last experiment. 171
6.16 Preference, fatigue and perceived performance for each TECH 172
6.17 Decision and Adaptation 177

B.1 *RayCasting*, *Eye-Tip* and *GyroMouse* 215
B.2 Movement time (*MT*) per GAP × TECHNIQUE. 218
B.3 Error rate (*Outside Clicks*) per GAP × TECHNIQUE. 219

C.1 Apparatus of the head+chest orientations experiment 222
C.2 The task parameters of the head+chest orientations experiment 223
C.3 Perspective distortion of the targets 224
C.4 Movement time and error rate for each technique and instruction 226
C.5 An example of a correlation between the Head Screen Position and the cursor screen position 227
C.6 Head undershoot for each technique, with and without instruction. 228
C.7 Head undershoot by A × INSTRUC 228
C.8 Distribution of the prediction errors for *RayCasting* with instruction. 229
C.9 Example of device, head and body velocities during a trial. 229

Introduction

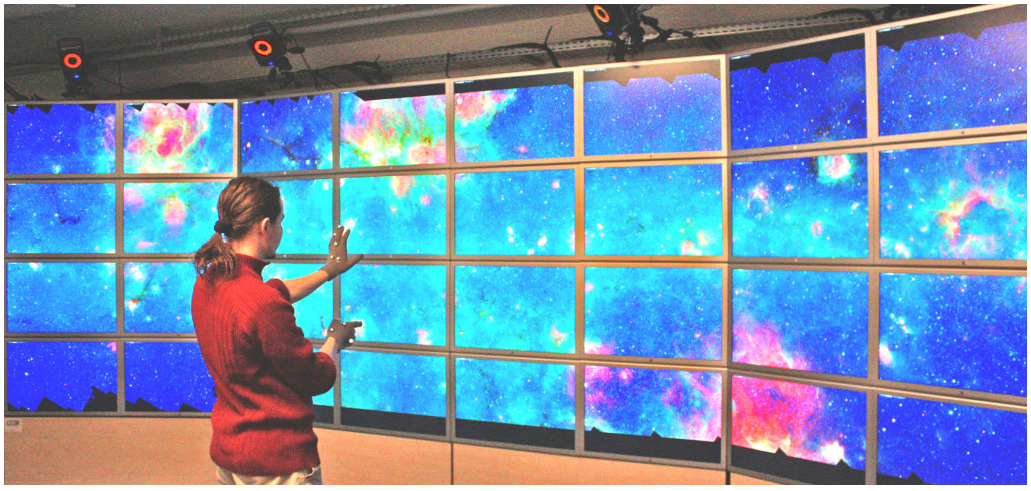


Figure 1: Mid-air interaction in a large display platform: pointing, panning and zooming over a very large visualization.

Large display platforms are physical workspaces designed to support the exploration and manipulation of very large datasets. They feature one or more large displays to render these datasets as well as interaction capabilities. Large display platforms are already in use, e.g. in control rooms [48], industrial design [38] and by NASA [85]. Some of these platforms can display several hundred megapixels and make it possible to visualize very large, heterogeneous datasets in many domains [2, 19, 176]. Astronomers can use them to display telescope images constructed from hundreds of thousands of frames stitched together, such as Spitzer's 4.7 billion $(396,032 \times 27,040)$ pixels images of the inner part of our galaxy (Figure 1). Biologists can explore the docking of complex molecules or 280 gigapixel electron micrographs[2]. Artists can create gigapixel images, such as the 272 gigapixel panorama of Shanghai[3] based on 12,000 pictures stitched together. Crisis management centers can interact with highly detailed maps of very large areas. For example,

[2]http://v.jcb-dataviewer.glencoesoftware.com/webclient/img_detail/201/
[3]http://www.shanghai-272-gigapixels.com/

OpenStreetMap[4] data range from a view of the world down to street level, resulting in an image that requires 18 peta ($18 \cdot 10^{15}$) pixels at its highest level of detail.

This thesis is focused on mid-air interactions in multi-display environments. Research on this topic has emerged in the past decade, and there is already a number of studies and interaction techniques available for elementary tasks such as pointing, navigating and command selection. These works have mainly focused on adapting existing interactions to physically large displays, but rarely take into account the very high pixel density that such displays can provide. Also, these techniques are often designed and evaluated separately on specific platforms and for specific use-cases or operationalizations, which makes them hard to choose, compare and combine. I believe there is a need for a unified framework for mid-air interaction in large display environments that would systematically consider factors such as the display and input capabilities of each platform, the required degree of collaboration, and the combinations of interactions that are needed during a given work session. In this dissertation I propose such a framework to inform the design of new interaction techniques and improve the usability and performance of existing ones, as well as to help combine them with other user actions.

The contribution of this dissertation is two-fold:

(1) Understanding how the system's input channels and the human sensory-motor system can be optimally combined in such platforms. Large display platforms can feature input devices and sensors adapted to their size and usage: motion capture or tracking [128, 117], touch-enabled displays [29, 22], interactive floors [10], interactive tables [173], hand-held [40, 41] or wearable devices [96, 80], etc. These input channels are often novel, at least compared to desktop input devices, and raise the questions of (1) how to effectively combine available input channels to enable simultaneous, higher-level mid-air interactions and (2) how can these techniques meet the performance and precision requirements of large display environments. Among other contributions I present controlled studies and usability analyses of input channels such as free-hand movements, on-body touch, head orientation, and number of fingers on a touch-enabled device. I also propose two meta-analyses, from the results of several controlled experiments, about how performance is affected by parallelization and limb usage, and how high-precision pointing strategies affect the users' sensory-motor system.

(2) Increasing and improving the vocabulary of mid-air input techniques for general-purpose tasks in wall display environments. While relatively new, interaction with large displays already exists as a research field, with previous work in perception

[4] http://www.openstreetmap.org/

[89, 176], collaboration [81] and interaction techniques [119, 128, 153]. However I will show that existing interaction techniques for navigation, command selection and pointing still need to be improved, especially considering the possibly high pixel densities of large display platforms and the need to be able to walk freely in such environments. I analyze existing interaction techniques in the light of the constraints of large display platforms, and develop new ones based on my analyses of the available input capabilities and contexts of use of wall-sized displays. I also propose a set of design requirements that ensures that interaction techniques for large display environments can be combined and used collaboratively.

In the first part of this dissertation I analyze the input and output channels typically available in a large display platform as opposed to on a desktop, and how these input channels can be used and combined in feature-rich applications. Using this analysis I describe the design and evaluation of efficient mid-air navigation and command triggering techniques that can be combined with other interactions. The second part of this dissertation is dedicated to high-performance, target-agnostic mid-air pointing techniques. I describe a study of human visual and input acuity and how they affect mid-air pointing. From this study I introduce a family of pointing techniques that support pointing tasks of virtually any difficulty. I then implement and evaluate a set of techniques from this family that can be used either one-handed or in combination with other commands on a hand-held device.

Part I

Combination of pointing and other interactions on wall-sized displays

Chapter 1

Designing Mid-air Interactions for Large Display Platforms

Large display platforms are collaborative work environments designed to enable the visualization and manipulation of very large datasets. A number of mid-air interaction techniques have already been proposed for these environments, mostly for specific needs or specific tasks studied independently. However, few existing systems and applications run on such platforms and they have been little studied. As a result, little to no research have yet focused on characterizing interaction techniques specifically for large display platforms: How does it compare to desktop environments? How does mid-air interaction affects performance, usability and fatigue? How can common tasks [35] be performed and combined in such platforms?

In this chapter I will analyze and discuss the core differences between desktop and large display environments in order to understand their specific constraints. From these I propose two distinct analyses of (i) the location of visual feedback relative to the manipulated data and how it affects collaboration and usability, and (ii) the different strategies to associate interactions with available input channels for multi-task applications. From the results of these analyses I generate a set of requirements for designing interaction techniques adapted to large display platforms.

1.1 Desktop Environments VS. Large Display Platforms

Historically, research about interaction techniques has been mainly focused on desktop environments, and only recently have "new" platforms and environments such as tabletops, tablets and large display platforms been investigated in depth (e.g. [55]). A majority of the existing models and interaction techniques were developed for the desktop. Some of

them have been shown to be applicable and usable as is in large display platforms, such as Fitts' law [114] that applies to mid-air pointing techniques [127]. Other techniques do not match the input and output constraints of large display platforms, such as mouse pointing when standing in front of a wall-sized display. In this case, the equivalent interactions have been implemented as new techniques for this platform , e.g. laser pointing [126]. Adjusting existing desktop interaction techniques to large display platforms is not trivial. Neither is designing new interaction techniques specifically for this context. I identify a number of fundamental differences between the display, input and user capabilities in desktop environments and in large display platforms. In this chapter I will show how these differences impact the adaptation of classical interaction techniques from the desktop to large display platforms and propose guidelines to develop interaction techniques for the latter.

1.1.1 Increased display capabilities

The term "large display" has been used in the literature to describe displays whose physical size is greater than the usual desktop screen or classic TV screen. The term encompasses a number of types of platforms, all of wich are "large" but differ on a number of characteristics:

Pixel density, pixel size and resolution – *pixel density* is the number of pixels contained in a given unit area. It is often measured in pixels per inch (ppi) or pixels per millimeter (ppmm). Its multiplicative inverse is the physical size of a pixel (here considered square for simplicity), thus often expressed in inches or millimeters. Pixel size can also be expressed in angular units, e.g. radians or degrees, when considering their visual size. This unit implicitly takes the location of the user relative to the display into account; in this case the pixels can no longer be considered square because of the angular distortion caused by perspective. Pixel *resolution* is the number of pixels of a display, regardless of its physical size. For example, a head-mounted screen, a TV screen and a 3 m diagonal projected screen can share the same resolution with different physical sizes, resulting in different pixels densities. In the remainder of this dissertation I will refer to *ultra-high-resolution* displays as displays featuring more than 10 megapixels.

Display technology – pixel density largely depends on the display technology. A classic HD projector can display up to (has a resolution of) 1920×1080 pixels; when projected on a rectangle 3 meter in diagonal, it provides a pixel density of 19 ppi. Multiple

projectors can be combined seamlessly with careful calibration, such as DGP[1]'s 5×1.8-meter display composed of 6×3 projectors and 6144×2304 pixels, resulting in a pixel density of 31 ppi. Unlike projected displays, the pixel density of tiled monitors does not depend on the distance. Platforms such as Digiteo's WILD [24, 25] (see Appendix A) or Calit2's OptIPresence[2] can display respectively 131 and 307 megapixels at approximately 100 ppi.

Projectors can theoretically reach the pixel density of LCD panels, provided they are close enough from the screen. However, their price is often much higher than LCD screens for the same number of pixels. Projected displays offer the possibility to be seamlessly tiled, i.e. with little-to-no visual interruption between one projection and the other. To date, LCD screens without bezels are almost inexistent and the available screens with very thin edges usually offer lower pixel density than typical desktop computers. LCD screens thus offer much higher pixel density but tiling them creates bezels that can hinder readability of the displayed data. Bi et al. [30] showed that interior bezels are not detrimental to either pointing or visual search performance (provided that visual objects are not split across bezels), but affect users' search strategies and steering behaviors. Ebert et al. [59] proposed to use projectors to display the missing information on bezels, which proved promising even though their pixel density is lower; however this additional information must be front-projected, meaning that it can be occluded by users walking in front of the display.

Physical size – I consider "large" displays as any single or tiled display whose physical size is bigger than a desktop screen, i.e. more than approximately 30-inch in diagonal. They thus include most large TV screens, single or tiled projected displays and tiled screens, as well as more specific setups such as CAVEs or specific display capabilities such as 3D stereoscopy. The largest existing large displays range from 5-meter to 10-meter wide and 1.8-meter to 3-meter high. These sizes are useful to display large quantities of information, especially when coupled with high pixel densities. However some areas of the display can become uncomfortable or physically impossible to reach while keeping a comfortable posture, especially the lower and higher parts. Users can walk to reach pixels located on the far left or the far right, but physical navigation and the human body provide few comfortable solutions to bring the eyes lower or higher than their normal location when standing (apart from crouching and jumping). While this might be of little importance with low pixel densities or if all interactions are performed at a distance, it can be problematic with high pixel density displays because of human visual acuity.

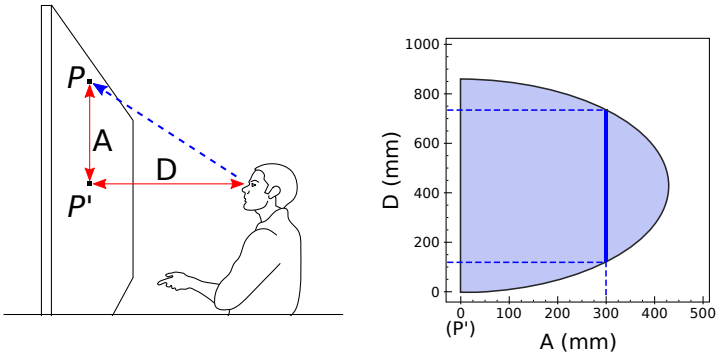
[1] http://www.dgp.toronto.edu/
[2] http://www.tacc.utexas.edu/resources/visualization/

To analyze this problem, I will focus on the effect of the vertical distance between a pixel and the user's point of view. There is a limit distance beyond which a user with normal vision can no longer perceive a pixel. The angular height (i.e. vertical width) of a pixel seen by a user at a distance D from the display is

$$\beta = \tan^{-1}\left(\frac{A+\frac{w}{2}}{D}\right) - \tan^{-1}\left(\frac{A-\frac{w}{2}}{D}\right) \tag{1.1}$$

where w is the pixel's (vertical) width and A the distance between the pixel and the orthogonal projection of the user's point of view on the display[3], i.e. the pixel directly facing him (P' in Figure 1.1-a). Following the theory of visual acuity [170], β must be greater or equal to $\frac{1}{60}^\circ$ (degrees) for a "normal" user (vision of 20/20) to be able to distinguish this pixel vertically. The solution to this inequality is a half disc of diameter $\emptyset = w/\tan\left(\frac{1}{60}\right)^\circ$ centered on $\left(A=0, D=\frac{\emptyset}{2}\right)$.

Figure 1.1-b shows the corresponding inequality plot for a user with 20/20 vision and $w = 0.25$ mm (100 ppi). This user must stand closer than 86 cm to the screen to be able to distinguish the pixel directly facing him (P' in Figure 1.1-a). An interesting reading of Figure 1.1-b is that regardless of the user's distance to the display, pixels higher or lower (or more generally, further away) than 43 cm from the "facing pixel" P' stop being accurately perceivable by users with normal vision on a 100 ppi display.



(a) (b)

Figure 1.1: (a) The user looks at a pixel P at a given altitude A (from his eyes) and from a distance D to the display. (b) The possible values of D for which the pixels of a 100-ppi screen can be accurately perceived for a given altitude A. As an example, the thick blue line represents the range of distances at which the user can accurately distinguish pixels that are 30 cm higher (or lower) of his projected point of view P'.

Using the same formula, we can compute the smallest pixel size w that can be accurately perceived at a given distance by a user facing the screen:

$$w = 2 \times D \times \tan\left(\frac{\beta}{2}\right) \tag{1.2}$$

[3]The point of view of the user is reduced to a point for simplicity.

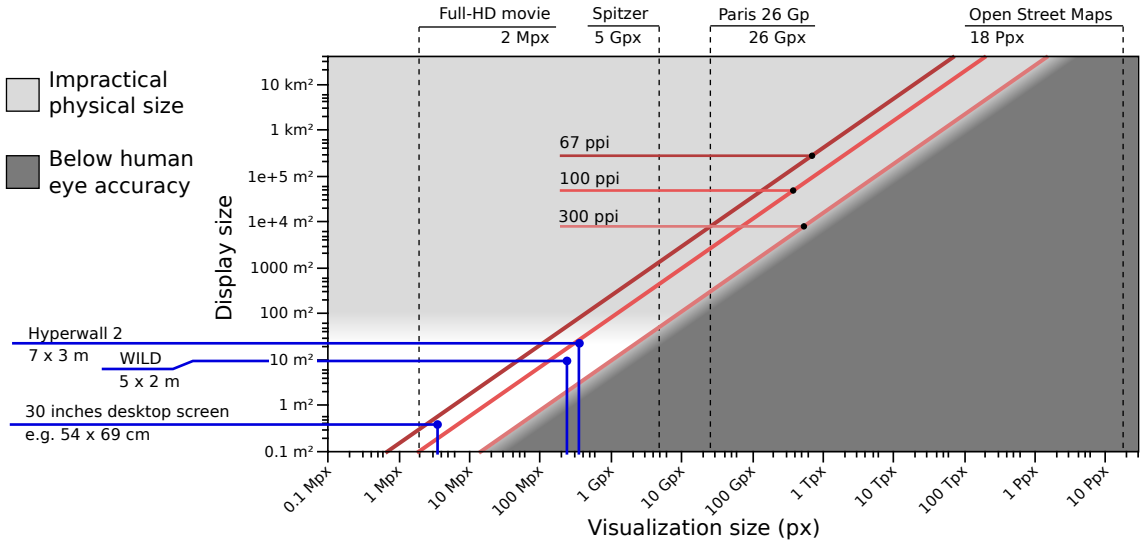


Figure 1.2: Display physical size over visualization size, with three typical pixel densities. Above a certain display size (light gray), parts of the display become physically unreachable. Above a certain pixel density (dark gray), the human eye can not differentiate pixels, thus can miss some of the displayed information. Display and visualization sizes are represented regardless of aspect ratio.

Using a conservative minimal distance D of 30 cm and the normal visual accuracy limit of $\frac{1}{60}^\circ$, this smallest size is $w = 0.087$ mm. It thus does not make sense to provide pixels much smaller than this. This pixel size corresponds to a 291 ppi display, i.e. a little below the latest commercial ultra-high resolution mobile devices[4]. Even with this tripled pixel density, displaying the Spitzer image ($396,032 \times 27,040$ pixels, see the Introduction) at full resolution would require a 35 m $\times$ 2.4 m display, which is unrealistic and impractical.

Figure 1.2 illustrates several examples of visualization vs. display size: the ordinate of the intersection between a red line and a dotted line gives the physical size needed to display the corresponding visualization (dotted line) with a given pixel density (red line). Blue dots represent the resolution and size of several displays. The quarter space to the left and to the bottom of a dot represents the visualizations that can be displayed at full resolution. Figure 1.2 shows that even the smallest of our "professional" examples can not fully fit existing displays. Ball et al.'s [19] results about physical versus virtual navigation do not invalidate the need for virtual navigation since some existing datasets are already several orders of magnitude too large to fit at once on even the largest sized displays.

Equations 1.1 and 1.2 apply when the display is vertical and flat; a solution is to tilt the top and bottom of the display towards the users to reduce the angular distortion on the upper and lower parts of the visualization. However this solution forbids cylindrical

[4]e.g. Apple iPhone 4 (326 ppi) or Samsung Galaxy III (306 ppi)

display setups, at least with tiled displays, which have been shown to improve performance and reduce the need for physical navigation [155]. Projected displays can accommodate both types of curvature at the same time, allowing for spherical surfaces [28].

However the limits of visual acuity do not invalidate the use of high-resolution large displays. Large displays improve collaboration and performance by allowing users to navigate physically [19]. Yost et al. [176] showed that the advantages of high pixel density are not limited by visual acuity. In particular, users were more efficient and sometimes more accurate in information visualization tasks on a display with a pixel density exceeding visual acuity limitations. Tan et al. [161] showed that *regardless of visual angles*, physically large displays improve performance and precision for spatial tasks such as mental rotation, 3D navigation and mental map formation and memory. They hypothesize that large displays improve the users' sense of immersion and presence and bias them into using optimal cognitive strategies.

1.1.2 Physical navigation

Navigation is the action of changing one's viewpoint of a given visualization when the corresponding dataset is too large to be fully displayed or comprehended at once. Navigation is used to either (i) focus on a specific area or data point, (ii) get an overview of an area of the dataset, possibly less detailed, or (iii) translate the viewport in order to explore a different area of the visualization. I identify three main types of navigation:

Physical navigation is the "natural" way of exploring an image or an environment by physically moving in front of, or around the object of interest. It can be either by walking or simply moving the head, similar to when observing a physically large object (i.e. a wide painting in a museum). It thus needs no learning.

Virtual navigation is a family of interaction techniques that lets users alter the visual representation of the dataset (e.g. by zooming its content) on the display, without requiring them to physically rotate their eyes or translate their head. Virtual navigation can be performed in datasets of virtually any number of dimensions, as with data-mining and visualization softwares. Virtual navigation can be performed in 1D, e.g. over lists [5], in 2D or 2+1D [68] over maps, graphs or any dataset rendered in 2D [128, 116], or within 3D-rendered worlds such as first person video games [176] or 3D-rendered datasets.

Mixed navigation techniques combine physical and virtual navigation by altering the visual representation of the dataset along with users movements, e.g. to provide content magnification over the user's Regions Of Interest [79] or to simulate 3D

navigation [57].

In typical single-screen desktop environments, head movements bring little to no additional information since the size and pixel density make eye movements efficient enough to cover the whole display. However head orientation can be useful in multiple-screen desktop environments [8]. In comparison, one of the main assets of interacting with wall displays is the possibility to actually *walk* in front of the displayed data to perform physical navigation, combining head rotation with body translation for more complex and "natural" navigation. Ball et al. [19] showed that physical navigation is more efficient than, and preferred to, virtual navigation for spatial visualization tasks on large displays. Finally, physical navigation does not change the displayed viewport, as opposed to virtual navigation. In collaborative environments, where several users work on different areas of the same visualization, one user changing the general viewport can conflict with every other user's focus and current actions.

1.1.3 Lack of a supporting surface

One of the main differences, in terms of features, between a desktop environment and a visualization platform is the table, or more generally any stable, fixed-location surface. Such static surfaces require users to sit or stand still in order to interact with the display, preventing physical navigation in the room and therefore not taking full advantage of very large high-resolution displays. This difference induces major changes in the way users can access and use input devices, which I categorize according to fatigue, *physical* support and accessibility of devices.

Fatigue – In a desktop or tabletop environment, users can use the table, the arms of their chair, or any stable element as support surfaces while interacting or to rest. For example, pointing with a mouse or trackpad [51] reduces fatigue by putting most of the arms' weight on the table or armrests. Keyboards are horizontally set so that users can put most of the weight of their arm on their forearm and palm while being able to access all the keys. More generally, input devices are laid on the table and thus require no additional holding effort when used.

When standing in front of a large display, most input devices must be held at the same time as they are used, which requires extra effort and limb coordination that could otherwise be used for input. Users cannot rest their arms on a stable surface while interacting and sometimes create strategies to minimize fatigue, such as keeping the upper arm still when performing mid-air pointing, changing the way they hold a tablet over time [167] or putting some of the tablet's weight on the abdomen.

Physical support – In a desktop environment, input devices most often lay on the table. Most user inputs are either parallel to the horizontal plane, e.g. moving a stylus or reaching for a key, or normal to it, e.g. pressing a mouse button or a key. This 2+1D input provides a motor separation between continuous and discrete interactions. Interactions on a table top can thus take advantage of friction for stability and precision. For example, pressing a mouse button is orthogonal to the mouse pointing movements and thus does not impair pointing precision. More generally, keeping the cursor still requires little to no additional effort with typical desktop pointing devices (i.e. mice, trackpads, styluses), which gives users more opportunities to recover from fatigue [51].

Mid-air interactions cannot take advantage of a stable object other than the user's body, which makes surface-dependent devices such as mice and keyboards less practical or unusable. Expressive mid-air devices such as portable trackpads, keyboards and tablets often need two hands to be operated while standing. Multitouch smartphones can be used for multiple purposes. They can be held and operated with one hand, however only the thumb can touch the screen with limited operating range [147]. Gyroscopic mice can be used in mid-air and provide very precise input. However clicking a button can cause hand tremor that impacts pointing precision [53]; some devices feature an additional button to immobilize the cursor. Other wearable devices provide more pointing stability and accuracy [53]. However, as for any mid-air pointing device, holding it requires additional effort [51] that can increase fatigue. It also requires fingers for support and stability which could be used for other buttons (for instance, gaming mice that can make use of all fingers).

Accessible devices – A tabletop can also be used as a "storage" area for input devices that are not currently used, e.g. a mouse that can be left aside when the user types text on the keyboard. Other examples include game controllers like joysticks, or work-related devices such as Phantom devices[5], 3D mice or drawing tablets that can be put away near the user's hand for easy access.

The "storing" function of the tabletop has little equivalent in mid-air. Users can quickly grasp only a limited number of input devices, namely the ones they are able to carry, unless they interrupt their workflow to go get the device they need. Compared to a desktop situation, users must be able to perform a maximum number of actions with a limited number of portable input devices.

Figure 1.3 summarizes the main differences between desktop and large display environments that affect interaction. Having established the fundamental differences between desktop and large display environments, I will focus on the design constraints that apply

[5] http://www.sensable.com/haptic-phantom-omni.htm

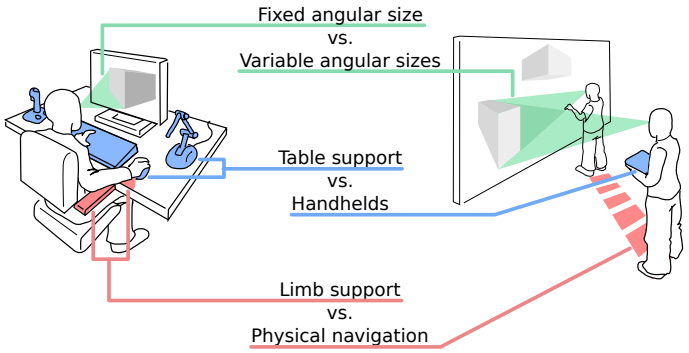


Figure 1.3: Main differences between interacting on a desktop and mid-air interaction in a large display environment, in terms of visual perception (green), accessible devices and types of interactions (blue) and user movements (red).

to mid-air interaction in the latter case. In Section 1.2, I will analyze how shared and personal displays can be combined and the associated constraints, namely split attention and occlusion. The corresponding design space will allow me to classify existing and new interaction techniques in terms of usability and collaboration in the remainder of this dissertation.

1.2 Location of the feedback

In his definition of the properties and behavior of an instrument, Beaudouin-Lafon [27] distinguishes the *response* of an action on a virtual object, i.e. its direct effect, from the *feedback* of the corresponding instrument. For example, most web-based zoomable interfaces such as Google Maps can be operated with the mouse wheel (or sliding on a trackpad) but also provide a visual instrument (widget), e.g. a slider, indicating the level of zoom. Regardless of how the zoom is controlled, the magnification of the map is the *response* of the interaction while the change of relative location of the slider knob is the *feedback* corresponding to this specific widget. In this section I will focus on the physical and virtual locations of instruments in large display platforms.

1.2.1 Distance- and visibility-dependence

Shoemaker et al. [153] proposed a design space for mid-air interaction techniques that features two orthogonal factors, *distance-dependence* and *visibility-dependence*:

Distance dependence – A technique is *distance-dependent* when its response changes with the distance between the user and the display, such as with ray-casting: the same gesture (elbow + wrist rotation) will produce cursor movements of greater

amplitude if the user is farther away from the display. Widgets on a wall-sized display also are distance-dependent since their readability varies with user distance. Distance-dependent techniques can only be used optimally within a certain range because of input precision or output readability [153], or require larger visual feedback as the user moves further away from the display.
The output of a *distance-independent* technique does not depend on this distance. For example, hand-held or virtual keyboards such as on smartphones or tablets can be used anywhere in the room. Distance-independent techniques favor location-independent mid-air interactions by not constraining the user to a physical area.

Visibility dependence – A *visibility-dependent* technique requires users to refer to visible feedback (other than the response of the action) during the interaction while a *visibility-independent* technique does not. For example, zooming with the mouse wheel is *visibility independent* (or eye-free) if the user only refers to the response of his action (e.g. the map magnification) and does not look at the corresponding slider. Conversely, the lack of tactile feedback makes typing on a virtual keyboard *visibility-dependent*: users need to look at the screen during the whole typing process.
Visibility-independent techniques do not require users to switch their visual attention, but are often more limited in expressiveness (see [53] and Section 3.2 page 72 for an analysis of existing menu techniques).

While eye-free techniques have been shown to improve performance and usability [110, 180, 131], visual feedback is sometimes unavoidable. Switching from *visibility-dependent* to *visibility-independent* interaction is one of the key characteristics of systems that encourage novice-to-expert transitions, such as WIMP linear menus and their associated keystrokes or marking menus [179, 150]. Novice interactions often are visibility-dependent as users need *feedthrough* [82] to control and learn them. Also, techniques such as pointing perform poorly without feedback [53].

1.2.2 Occlusion and attention switch

While Shoemaker et al. [153]'s taxonomy accurately describes most pointing-based mid-air interactions, it ignores an important trade-off between two potential issues for visibility-dependent techniques: *Distraction* and *Attention switch*.

Distraction occurs when a widget or any other interaction feedback is displayed over the visualized data. It ranges from minor distraction, e.g. a moving cursor, to complete hiding of a region of the data, e.g. invoking a linear menu over the visualization. A

user's actions can cause occlusion over his own Region Of Interest (ROI) but also over his collaborators' ROI when several users manipulate the same visualization. Occlusion problems have been addressed in the literature on multi-touch surfaces (e.g. [164, 165]) and desktop interactions (e.g. [77]). In their toolkit of group awareness, Hill and Gutwin [82] raised the problem of how shared widgets can occlude collaborators' local work areas. They propose remote versions of pull-down menus that limit this occlusion in collaborative contexts: one transparent and one with a smaller size ("summary"). Vogel and Casiez [165] proposed a set of geometric models capturing the shape of hand and forearm occlusion on a multi-touch table and provided a corpus of corresponding input data. Vogel and Balakrishnan [164] proposed an algorithm to define what area of the display is hidden by the user's arm and hand and what important data is contained in this area, as well as a visualization technique to displace this data in close, non-occluded areas of the display. This technique has not yet been evaluated with remote or collaborative interactions. Apart from these examples, collaboration-induced occlusion has not been, to our knowledge, studied in the literature.

Attention switch occurs when a command or interaction widget is displayed physically far away from the user's ROI: the user then has to switch his gaze and focus from the data to the commands and then back [144]. The amplitude of the attention switch depends on the distance between the widget and the ROI: moving palettes located near the ROI on the same display cause a small attention switch, while controlling widgets on a hand-held tablet when the manipulated data is on a wall-sized display requires users to move their eyes and head by about 45°. The observed effects of gaze shifts vary from negative to neutral: Rashid et al. [143] observed that the number of gaze shifts is correlated to completion time in mobile device + large display situations; on the contrary, Cauchard et al. [46]'s results suggest that the number of eye context switches does not affect performance.

1.2.3 Physical and virtual location of the feedback

In order to assess the causes of these issues I introduce a design space for *Feedback Location* that further develops *visibility-dependence*: if an interaction technique does require visual feedback to be operated (other than the direct response of the technique), the *physical location* of this feedback relative to the user(s) and the *virtual location* of the feedback relative to the data have an impact on usability, attention and collaboration.

The *physical location* of the visual feedback can be either on the *Main display* or on a *Secondary display*:

The **main display** is the core display of a large display platform. It is composed of any stationary display or set of displays (e.g. a wall display [25] or a CAVE) facing the users. It features the greatest number of pixels in the platform and is meant to be the primary output for the visualized dataset. In order to be readable at any distance, interaction feedback rendered on the main display must adapt its size to the user's distance. In this case feedback on the main display is *distance-dependent*[6] and may cause more occlusion when the user is far away from the display. Hawkey et al. [81] observed collaborative benefits when users are positioned close together in a collaborative task using a large display. However, physically close users are more likely to share portions of the same ROI and distract each other if interaction feedback and data are displayed on the main display.

Secondary displays are any displays used within the platform that do not fall into the first category, such as hand-held interactive screens and tabletops that are not part of the main display. Rekimoto [145] did an early exploration of *Secondary displays* with prototypes of "palettes" used in combination with shared whiteboards. They observed that displaying too many tools on a palette can degrade mutual awareness among collaborators, as users spend more time concentrating on their own devices. Lindeman et al. [112] report that "hand-held windows" can be positioned to allow the user to work effectively and do not clutter her view.

The *virtual location* of the feedback describes where it is rendered relative to the visualization. Since part of the data can be displayed on secondary displays, I consider that this factor is orthogonal to *physical location*. *Virtual location* can be either *On-data* or *Fixed*:

On-data feedback is displayed near the user's Region Of Interest (ROI) over the rendered data, either on the main display or on a secondary display featuring a subset of the dataset. Being displayed over the rendered data, they cause limited cursor movement and attention switch, similar to WIMP contextual menus and movable toolboxes. However they also occlude the data underneath, either briefly or constantly, depending on the type of visual feedback. For example, toolboxes are often always visible and movable over the visualization while pull-down menus disappear (mostly) after an item is selected. Occlusion can impair the user if the feedback is displayed on a secondary (personal) display, but it can impact his collaborators if the feedback is displayed on the main display.

Fixed feedback is always visible and has a fixed location and visible size on the host display (main or secondary). Examples include toolbars (as opposed to pull-down menus) and tablets when these do not feature parts of the visualization. Being

[6] Making Shoemaker et al. [153]'s factors non orthogonal.

visually separated from the data, they cannot cause occlusion, however they may decrease the screen real estate dedicated to data visualization of their host display in hybrid [143] configurations. Fixed feedback may also cause attention switch since the location of the feedback is no longer related to the user's ROI, unless a greater amount of virtual navigation is performed to bring the ROI near the widgets.

In the context of Multi-Display Environments, Wallace et al. [168] studied *content redirection*, i.e. displaying parts of shared data on personal displays, versus *input redirection*, i.e. moving a user's control focus from one display to another. They found that content redirection can improve the performance of users in suboptimal seating positions. However they did not study situations where users can stand and walk. In Rashid et al.'s taxonomy [143], *Hybrid* configurations feature subsets of the rendered data on the *Secondary display* (a tablet) for direct manipulation while in *Large Display* configurations the *Secondary display* only displays indirect commands. They report that the *Hybrid* configuration was worst or equal to worst for map, search and photo search tasks with hand-held smartphones, while the *Large Display* configuration was best or equal to best for all three tasks. As we consider the main display as the primary output for the manipulated dataset, both *Hybrid* and *Large Display* configurations cause attention switch when users need context or feedback about their local manipulations.

<table>
<tr><td colspan="2" rowspan="2"></td><td colspan="2">Physical location</td></tr>
<tr><td>Main display (MD)</td><td>Secondary display (SD)</td></tr>
<tr><td rowspan="3">Virtual location</td><td rowspan="2">On-data</td><td>+ Minimal attention switch (with MD data)

− Occlusion on the MD</td><td rowspan="2">+ Minimal attention switch (with SD data)
+ No occlusion on the MD

− Occlusion for the user
− Focus switch with MD</td></tr>
<tr><td>− Distance-dependent size</td></tr>
<tr><td>Fixed</td><td colspan="2">+ No occlusion
− Possible attention switch</td></tr>
</table>

Table 1.1: Design space for *Feedback Location*: pros and cons of the possible physical and virtual locations of widgets and visible interaction feedback.

Table 1.1 summarizes the pros and cons of the combinations of physical and virtual locations for interaction feedbacks. We can observe a trade-off between occlusion and attention switch: no combination of physical location and display location is free of one or both. The choice of one combination over another depends on the cost of occlusion and attention switch in a given context. Choosing a combination depends strongly on the required interactions and collaborative needs of a task. For example, collocated collaborative tasks

performed on a shared visualization on the main display may suffer from occlusion and distraction caused by other users' actions. Repetitive, short-term commands cause minimal attention switch and can be limited in time, thus causing only short-term occlusions; context-independent tasks such as typing text may take time and thus would cause lengthier occlusions, but can be performed out of the main display without the need to look at the main display data before the end of the typing. Non-collaborative environments can focus less on avoiding occlusions and use on-data feedback to minimize attention switch.

In this section I introduced a design space of *Feedback Location* that describes and qualifies the possible combinations of personal and shared displays in large display environments. I will use this design space in the remainder of this dissertation to classify existing techniques and assess the usability and collaborative potential of new techniques. From this analysis of the potential interactions of the visual *outputs* in large display platforms, I will now present a taxonomy of how *input* channels are allocated to interactions when several input channels must be available for a given task.

1.3 Task allocation strategies

There is a growing number of mid-air interaction techniques in the literature about large displays, but few articles (e.g. [85, 107]) mention whole applications with multiple tasks or real setups, except with public displays. Most research articles about interacting with wall-sized displays and large display platforms focus on specific tasks and interactions. They are evaluated in controlled experiments with operationalized tasks and controlled environment and factors. Controlled experiments are indispensable to understand the precise correlations and effects between these factors, e.g. interaction techniques, and dependent variables such as performance, user preference and reaction to a stimulus. "Real" applications such as the ones on desktop environments are usually more complex than what is tested in controlled experiments. They allow and require many different interactions an commands that must be mapped onto the available user capabilities, such as movements of their limbs. For instance, in desktop environments, interactions and commands are mostly controlled by cursor movements (e.g. selecting a menu item or drag-and-dropping an object) and keyboard strokes (e.g. shortcuts and modifiers), which mainly involve movements of the hand and forearm [51]. However, as pointed out in Section 1.1, users are more limited in terms of available input devices when standing in front of a wall-sized display. The lack of a stable surface means that users can have quick access only to the limited set of devices they are holding or carrying. In particular having no support surface makes the use of a keyboard cumbersome, which greatly limits the number of discrete events users can send easily. On the other hand, input is not constrained by a table or

any supporting surface, thus arms and hand movements can benefit from more degrees of freedom.

1.3.1 Resource allocation

A user standing in front of a display can move several limbs at once, e.g. fingers, hands, arms, head, eyes, etc. These movements can be interpreted as interaction inputs via physical interactors (key, button or mouse wheel events) or tracking systems (location and orientation of the hands and eyes, postures, etc.). Depending on the sensing capabilities of a given large display platform, including its input devices, a subset of these movements can be detected with more or less accuracy. Designing rich applications requires solving a resource allocation problem: the set of functionalities must be mapped to the morphologically and technologically limited input expressiveness. This expressiveness is formally defined as the measurable and comfortably usable degrees of freedom of a user's limbs and segments. The lack of supporting surface also means that part of the available limbs and muscular effort must be allocated to holding the input device prior to interacting with them.

Wagner et al. [167] followed a similar approach with *BiTouch*, an extension to the kinematic chain model [73] for touch-based mid-air interactions that takes the support function into account, whether the device is held by the dominant hand, by the non-dominant hand or by both hands. They observed that the limb segments used to hold a device are allocated three ordered functions (Figure 1.4) with decreasing priority, depending on their physical proximity to the device: (i) *Framing* is the general positioning of the device relative to the body and point of view of the user, (ii) *Support* encompasses holding and keeping the device stable, and (iii) *Interaction* covers actually using the device with the remaining free limbs or segments. For example, when interacting using a single hand with a smartphone in the dominant hand, the arm and forearm location and orientation set the general *frame* of the phone relative to the user, the palm and fingers *support* the phone, and the thumb is the only remaining segment able to *interact* with the screen. Skipping one of these functions provides more expressiveness for the following one(s) in the chain: if a user attaches a smartphone to her non-dominant forearm, thus delegating the *Support* function to mechanical means, then the palm and fingers of the non-dominant hand become available for other purposes.

BiTouch does not cover non-touch interactions such as mid-air pointing, however one could observe that the model of decreasing limb resource still applies, if the *Framing* function is no longer constrained to the user only. Consider laser-pointing with an IR-tracked wireless mouse (as in e.g. [127]). The arm, forearm and hand *frame* the pointing, i.e. orient the

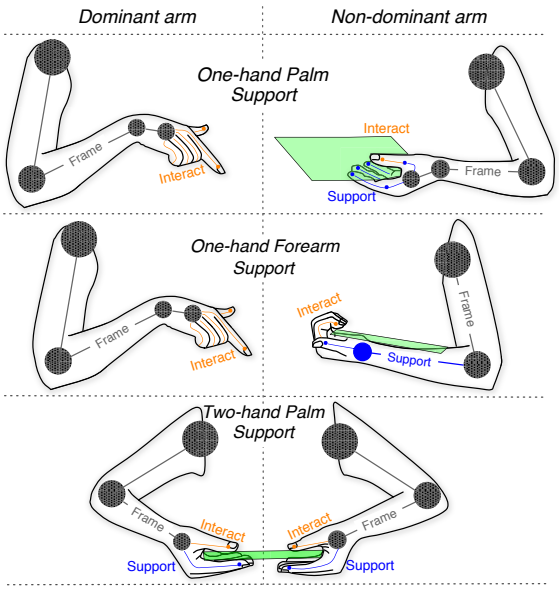


Figure 1.4: Examples of the *biTouch* framework (picture from [167]).

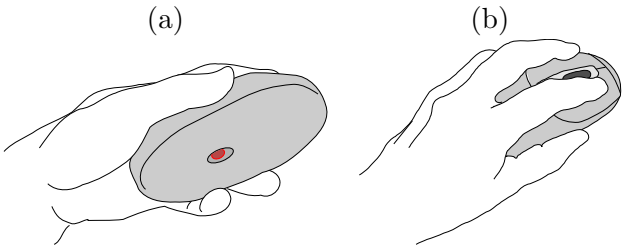


Figure 1.5: Wireless mouse held in mid-air, seen from below (a) and above (b).

mouse towards the desired location of the cursor. The thumb, ring and pinky fingers are used for *support*, so that the mouse does not fall (Figure 1.5-a). The index and middle fingers are available to interact with the mouse buttons and wheel (Figure 1.5-b). Limbs such as the head and eyes can also be used to *frame* an interaction (e.g. [178, 79]) but provide much less *interaction* capability compared to the hands and fingers, except maybe for blinking [9].

1.3.2 Existing strategies

The number of possible combinations of tasks and segments becomes quite high for applications with numerous interactions and platforms featuring high detection capabilities. Existing systems exhibit three main strategies to allocate a large number of tasks onto limited limb movement capabilities:

Maximizing limb usage: Research on multitouch devices and mid-air interaction has focused on exploiting unused limb expressiveness and "natural" movements to enrich the input vocabulary. Examples include tracking the proximity of the head to the screen [79] or gaze location [178] in desktop environments, exploiting the large vocabulary of available finger movements and hand postures on tabletops [12], using the skin as an interaction device [80, 111] or simply controlling parallel tasks [94] with both hands [128]. If well designed, these solutions can provide interactions with a high degree of parallelization and limit the need for visual feedback. However explicitly using more limbs increases the chances of causing fatigue, especially when standing [51]. By nature, this approach is also more limited in terms of allocatable tasks compared to, e.g., visual menus that can contain hundreds of commands at once. It also requires advanced tracking capabilities to correctly interpret user movements, which can be difficult to achieve uniformly across very large areas.

Factorizing task allocation: Another solution consists in allocating many functions to a single or small set of limb(s) or expressive input device(s). For example, multitouch tablets now feature as many pixels as typical desktop screens, e.g. the 2048×1536 pixels iPad, with high-resolution finger tracking capabilities. They can thus be used as hand-held toolboxes without the need to display part of the visualized dataset [145, 143], causing no occlusion, manageable fatigue [167] and offering a high number of commands and functionalities. However they require both arms at once since the non-dominant arm and hand cannot do much more than hold the tablet. All widgets being displayed on the tablet, it is also likely to cause attention switches since users have no feedback (other than visual) of their location. Finally, such one-handed interaction style means that most actions will be performed sequentially.

Everything through cursor movement is a particular, visibility-dependent case of the latter. Desktop-based applications usually allocate many interactions to cursor movements and pointing actions through menus, widgets and modifiers. This strategy is applicable to large display platforms and requires a limited number of limbs to control a whole application since only one arm and hand are needed to control the cursor's location. However it relies heavily on visible widgets and feedback, which increases the chances of occlusion as introduced in Section 1.2. Also the cursor is the sole interacting object, which prevents parallel tasks such as bimanually orienting, moving and scaling an object with two control points as on multitouch devices.

These strategies can be mixed, as illustrated in Figure 1.6: applications can favor limb usage for eye-free direct manipulation (e.g. virtual navigation and object manipulation), expressive personal devices for fine parameter settings or text entry and pointing-based

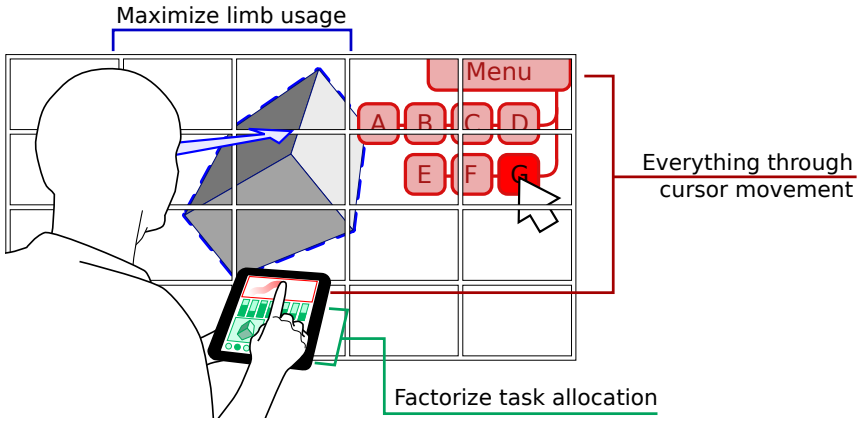


Figure 1.6: Examples of task allocation strategies: head orientation defines the object of interest (blue – *Maximizing limb usage*) while a tablet-controlled cursor selects an item on a menu (red – *Maximizing cursor usage*). The remaining tablet space is dedicated to various control widgets (green – *Factorizing task allocation*).

widgets for more general commands.

In this section I introduced a taxonomy of task-to-limb allocation intended to qualify and predict how multiple interactions can be combined within the limited input channels available when standing in front of a wall-sized display. Chapters 2 and 3 will explore this taxonomy with two goals: (1) validate my preliminary assessments about existing task allocation strategies and (2) enrich its predictive power by investigating complex combinations of interaction techniques. Section 1.4 will summarize the theoretical findings of this Chapter by drawing a list of requirements that interaction techniques for large display environments should meet for optimal usability, performance and collaboration.

1.4 Requirements

Based on the analysis and factors presented in this chapter, I introduce four main requirements for designing mid-air interaction techniques in large display platforms:

R1: **Human perception** – Display technology makes it possible to display ever-growing datasets at decreasing costs, and yet some recent interaction techniques in the literature are evaluated in much less constrained conditions. It is often unclear how a technique designed for "high-resolution large displays" will perform in more recent and expressive setups. While it is difficult to predict which new display technology will be available in the future, the capabilities and limits of human perception are known, and display capabilities are already reaching them.
On the basis that "he who can do more can do less", I defend that *the capabilities*

of mid-air interaction techniques in large display platforms should match the limits of human perception. Users should not need software magnification or physical navigation to be able to interact with already visible or readable elements of the visualization *unless necessary*; that is, physical navigation should be instigated by the user's will rather than need. A direct application of this requirement is that all targets visible from a given physical distance should be selectable regardless of task amplitude and display technology.

R2: **Location independence** – Walking is a "natural" way of navigating in front of a large image and has been shown to improve performance in visualization tasks [19]. Even when not necessary, e.g. when virtual navigation does not disturb a user's collaborators, she should have a choice between virtual and physical navigation. Conversely, equivalent input capabilities should be available anywhere in the platform. This means that mid-air interaction techniques in large display platforms should be location-independent: *users should be provided the same interaction capabilities wherever they stand, walk or sit in front of the display.* In particular, R1 must be true wherever the user stands. As an example, mid-air interactions constrained to a fixed location [117] or techniques with fixed-size visual feedbacks [153] are in conflict with R2.

R3: **Input channels** – The design of interaction techniques for large display platforms should take scarcity of input devices and fatigue into account. Users standing and walking in front of a wall-sized display have access to a smaller set of input devices compared to a desktop environment. Standing is also more tiring, as is the lack of a supporting surface for the arms.

Interaction designers should ensure that *a single interaction technique does not overload the input channels of users and their available input devices.* For example, pointing by touching, with the dominant hand, a device held in the non-dominant hand that cannot be used for any other interaction (e.g. invoking a command or a mode) is likely to be less usable in a real application than expected from the results of a controlled experiment.

R4: **Collaboration** – Mid-air interaction techniques in large display platforms should not impair collaborative work efficiency. As described in Section 1.2, it is possible that a user's actions or feedback affects his collaborators negatively, e.g. by changing a shared virtual viewport.

Several users should thus be able to use the platform simultaneously, and *a user's interactions must not hinder tasks carried out by other users.* For example, the visual feedback of a given user's interactions should cause as little occlusion and distraction as possible on other users' regions of interest.

Based on the theoretical findings of this chapter, I suggest that meeting these requirements when designing interaction techniques will improve the general usability, performance and collaborative power of large display platforms. In the remainder of this dissertation I will present the design and evaluation of a set of comprehensive interaction techniques in accordance with these requirements: Chapter 2 presents an analysis of the input factors available for pointing, panning and zooming in mid-air, as well as a thorough exploration of these factors resulting in a set of mid-air pan-and-zoom techniques. Chapter 3 introduces a design space for highly expressive discrete selection techniques and reports on a controlled experiment about combined menu and mid-air pointing techniques.

Chapter 2

Pointing and navigation on wall-sized displays[1]

2.1 Introduction

As discussed in Section 1.1, ultra-high-resolution wall-sized displays can accommodate very large visualizations that benefit from physical navigation, but do not discard the need for virtual navigation: some datasets are already several orders of magnitude larger than existing wall-sized displays. Many interaction techniques have been specifically designed to help users navigate large multiscale worlds on desktop computers, using zooming and associated interface schemes [54]. However, high-resolution wall-sized displays pose different sets of trade-offs. It is critical to their success that interaction techniques account for both the physical characteristics of the environment and the context of use, including cooperative work aspects. As defined in the previous chapter (Requirement R2), input should be location-independent and should require neither a hard surface such as a desk nor clumsy equipment: users should have the ability to move freely in front of the display and interact at a distance [19, 176]. This precludes the use of conventional input devices such as keyboards and mice, as well as newer interaction techniques: For example, the powerful multi-finger gestural input techniques designed by Malik *et al.* [117] were devised for interaction with lower-resolution large displays from afar; they require sitting at a desk, and are thus not optimal for very high-resolution displays that afford more physical forms of navigation. The recent Cyclostar approach [116] is very elegant, but requires the display surface to be touch-enabled, a feature that wall-sized displays often lack. Cyclostar is also not well-suited to wall-sized displays, as it requires users to be within arm's reach of the display surface. While this is perfectly acceptable for displays up to 1.5m in diagonal

[1] A subset of this chapter has been published at the CHI'11 conference [128] and received a Best Paper Award.

such as SMART Boards^{TM}, users of larger displays such as the one in Figure 6.1 (5.8m in diagonal) would only see a very limited portion of the display while navigating. This lack of an overview would be a hindrance as navigation is mostly driven by contextual information.

The work presented in this chapter serves three purposes.

First, our initial goal was to study which input channels available in mid-air can be used and combined to perform panning and zooming actions on wall-sized displays. More specifically, we sought to answer questions related to the performance and subjective preference of users relative to these combinations, including:

- Beyond their almost universal appeal, do gestures performed in free space work better than those input via devices operated in mid-air?
- Is bimanual interaction more efficient in this context?
- Is it more tiring?
- Do circular, continuous gestures perform better than those that require clutching (restoring the hand or finger to a more comfortable posture)?

We grounded our work in both theoretical and experimental work on bimanual input [37, 73, 109], the influence of limb segments on input performance [18, 177], on types of gestures [124, 172] and on the integral nature, in terms of perceptual structure, of the pan-zoom task [93]. In particular, we were interested in comparing the following dimensions:

- bimanual vs. unimanual input;
- device-based vs. free-hand techniques;
- degrees of freedom (DOF) and associated kinesthetic and haptic feedback;
- types of movements: linear gestures vs. circular, clutch-free gestures.

Second, at a higher level, I also investigate how three integral tasks, namely panning, zooming and pointing, behave with the limited set of input channels and devices available in a large display platform. I integrate this study into the *Task Allocation* design space introduced in the previous chapter (Section 1.3 page 40) by exploring two strategies for task allocation, namely *Maximizing limb usage* and *Factorizing task allocation*.

Finally, this work contributes to the vocabulary of navigation techniques available in mid-air for high-resolution large display platforms.

2.2 Panning and Zooming in Mid-air

Virtual navigation is meant to displace a user's viewport on the visualized dataset towards a remote Region Of Interest (ROI) with a possibly different zoom level, in order to get either an overview of a region of the scene for general observation or interaction, or a detailed view of a smaller portion of the scene for precise observation or interaction. Guiard and Beaudouin-Lafon [74] listed three main virtual navigation techniques: bi-focal, fish-eye and pan-and-zoom, which we study in this work.

Panning and zooming are usually operated in sequence, as illustrated in Figure 2.1[2]: unless familiar with the dataset or given additional guidance, a user cannot zoom towards a ROI that does not appear within his viewport and does not know in which direction to pan (Figure 2.1.1). In this type of "blind" situation, the usual strategy [74] is to zoom out until the requested ROI enters the viewport (Figure 2.1.2). The user can then reach his target ROI with a sequence of zooming and panning actions to adjust the focus (Figure 2.1.3-6).

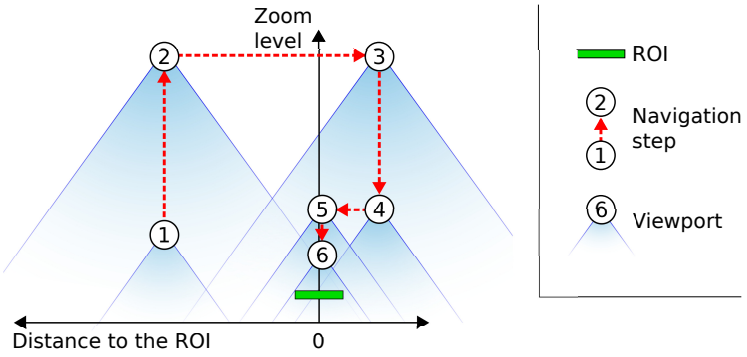


Figure 2.1: A typical sequence of virtual navigation steps. If at first the viewport does not contain the target ROI (1), the user can zoom out until he sees it (2). He can then pan to center the ROI near his own physical position (3) and zoom (4). If a lot of zooming is needed, step (3) can be too imprecise and lead to the ROI almost leaving the viewport while zooming (4). It is then necessary to repeat the last two steps once or more (5-6).

Pan and zoom is a 2+1 DOF task: the user controls the view's position (x, y) and its scale (s). Pan and zoom actions can be mapped to three input channels in numerous ways, and a large body of literature is devoted to the design and evaluation of input devices that feature a high number of DOF. The available degrees of freedom have a direct impact on the ability to parallelize the actions required to achieve the task. For example, 6DOF input devices can increase the degree of parallelization of docking tasks [177], although

[2]Figure 2.1 is the counterpart to space-scale diagrams [68]: the y-axis represents the altitude of the camera as opposed to the scale of the visualization, and the x-axis is centered on the target viewport. This visualization shows high-altitude panning movements more clearly: they would be too small to be visible in a classic space-scale diagram.

studies report limits in terms of human capacity to handle all DOFs simultaneously.

The film industry offers interesting and visually attractive scenarios with movies such as *Minority Report*, which show users interacting via freehand gestures to navigate a scene in a seemingly fluid and efficient way. The technology to achieve this type of interaction is now available in research laboratories and beyond [181]. However, it remains unclear how freehand gestures actually fare when compared to device-based input techniques that take advantage of the human ability to use physical tools [41] and suffer fewer problems commonly associated with spatial input [83], such as precision and fatigue. Years of research in virtual reality have demonstrated that devising efficient navigation techniques for immersive virtual environments is still a challenge.

Our goal is to study families of input techniques that let users pan and zoom from any location in front of wall-sized displays. Regarding the *Feedback Location* and *Task Allocation* dimensions presented in Chapter 1, we made two design choices for ease of use and performance: all our techniques are eyes-free and integrate pointing as part of their control mechanism.

Eyes-free – Pan-and-zoom is a rather simple 2+1D task that, unless aiming for exact coordinates or numerical zoom level, can be achieved with only the visual *response* [27] of the navigation. Its control can thus be *visibility-independent* [153], e.g. eyes-free, in the sense that a single user does not need to switch his attention from his ROI when performing a navigation task. Eyes-free interactions have been shown to improve performance and usability [110, 180, 131]. We thus applied immediate navigation responses to user's input, as opposed to via visual widgets such as in the Google Maps™ interface.

Pointing-based techniques – Navigation often happens before other interactions on the visualized data, which are likely to require pointing. We chose to make pointing available in all our pan-and-zoom techniques to simplify later use in real applications. We thus have three interactions to combine: pointing, panning and zooming. Rather than separating them completely we tried to improve navigation performance by integrating pointing into panning and zooming:

Cursor-based panning: We consider panning as a coarse "grab, drag and release" operation similar to applications such as Adobe Illustrator™ or Google Maps™ with their typical hand-shaped cursor, as opposed to rate-based panning that can be achieved with joysticks or by pressing a key to trigger camera movements with predefined velocities. Panning from a distance using laser-like pointing is a rather fast interaction that benefits from direct control: users always have a coarse knowledge of

where they are pointing at through proprioception [154], even with very fast movements that make the cursor harder to follow visually; in contrast, rate-based control relies heavily on visually following the cursor to know when to accelerate or stop the movement, thus could be less efficient to control panning.

Focused zooming: Pointing plays an important role when zooming, as it specifies the focus of expansion (zoom in)/contraction (zoom out). Letting users specify this focus point is very important on wall-sized displays, as they will typically not be always standing at the same position and be interested in the same areas. A focus of expansion implicitly located at a fixed location on the screen would make zooming operations tedious and hard to control as every zoom operation would require multiple panning actions to compensate drifts induced by the offset focus.
Focused zooming also allows varying the level of zoom while translating its focus of expansion, which we call *dynamic zooming*. When the targeted ROI is visible on screen (i.e. does not require any further zooming out), dynamic zooming can lower the duration of a navigation task by parallelizing the movement of the focus of expansion and the zoom-in as illustrated in Figure 2.2.

This integration of pointing into both aspects of pan-and-zoom also reduces the number of tasks to allocate to the users' input capabilities (Section 1.3).

We integrated pointing with the control mechanisms of both panning and zooming. Navigation manipulates viewports, i.e. virtual objects with dimensions often much larger than the actual data that is to be observed and/or interacted with. It is a rather coarse interaction at the scale of the display, a preliminary *view pointing* [74] task intended to set the frame for more precise actions. We thus chose to use a very simple and direct pointing technique, laser pointing (simulated with ray-casting), leaving aside pointing facilitation mechanisms (e.g. [71, 32]) and velocity-based transfer functions [44]. Using the motion tracking capabilities of the WILD room (Appendix A page 213), the cursor jitter of laser pointing [126], measured as the standard deviation of the cursor position when the device is held stationary, was 1.66×3.87 pixels at a distance of 3 meters from the display with a simple 10-sample window low-pass filter, which we expect to be precise enough for most navigation tasks.

2.3 Design Space for Mid-Air Pan-and-Zoom Input

An extensive design and testing phase allowed us to contemplate a number of candidate techniques and input possibilities. We eventually identified a design space composed of three key dimensions (Table 2.1), informed by related empirical studies reported in the

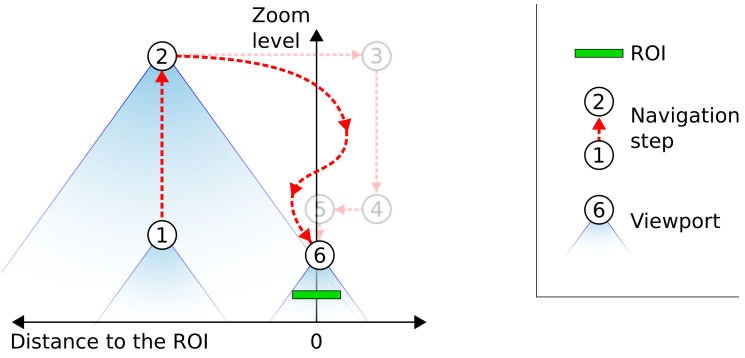


Figure 2.2: With dynamic zooming, expert users can move the focus of expansion as they zoom, allowing for more integral use of panning and zooming.

literature and refined through prototyping and pilot testing.

Factors		Advantages	Disadvantages
Hands	One	• One hand available for other actions	• Pan and zoom are performed sequentially
	Two	• Pan and zoom can be performed in parallel	• No hand available for other actions
Gesture	Linear	• Direct, natural mapping to zoom actions	• Potentially requires clutching
	Circular	• No clutching (continuous gesture)	• Less natural mapping to zoom actions
Degree of Guidance	1D path	• Input guided by strong haptic feedback • Mainly involves fingers	• Only 1 degree of freedom
	2D surface	• Multiple degrees of freedom • Mainly involves fingers	• Input guided by limited haptic feedback
	3D free hand	• Many degrees of freedom • No device	• No haptic feedback • Mainly involves whole hand and arms

Table 2.1: Key Dimensions of the Design Space

2.3.1 Unimanual vs. Bimanual Input

In their paper on the perceptual structure of multidimensional input, Jacob and Sibert claim that panning and zooming are integrally related: the user does not think of them as separate operations, but rather as a single, integral task such as "*focus on that area over there*" [93]. Buxton and Myers [37] and later Bourgeois and Guiard [34] observed high levels of parallelism for pan-zoom operations, further supporting this argument. The level of parallelism correlates with task performance and is typically enabled by the use of

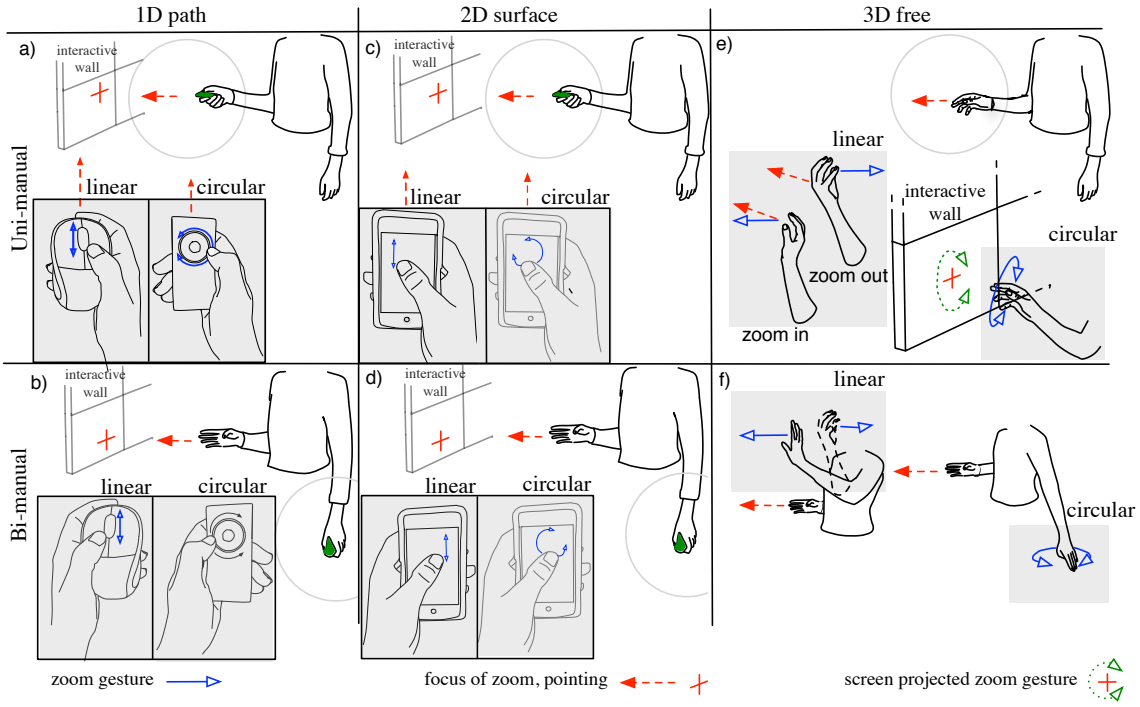


Figure 2.3: Matrix of the 12 techniques organized according to key characteristics: uni- vs. bimanual, degree of guidance, linear vs. circular gestures. 1D path involves guiding gestures along a particular path in space; in 2D surface gestures are made on a touch-sensitive surface; while in 3D free gestures are totally free.

bimanual input techniques [73, 109]. While we expect bimanual techniques to outperform unimanual ones, we are still interested in comparing their performance, as the latter might still be of interest in more complex, real-world tasks that require input channels for other actions.

2.3.2 Linear vs. Circular Gestures

Navigating in the scale dimension (zooming in and out) is a task typically performed through vertical scroll gestures on, e.g., a mouse wheel or a touchpad. The mapping from input to command is natural, but often entails clutching as the course of mouse wheels and touchpads is very limited. An alternative consists in mapping continuous circular gestures to zooming: clockwise gestures zoom in; counter-clockwise gestures zoom out. Despite the less natural mapping from input to command, such continuous, clutch-free gestures have been successfully applied to vertical scrolling in documents [124, 172], and to panning and zooming on large, touch-sensitive surfaces in CycloStar [116]. Circular gestures potentially benefit from an automatic Vernier effect [60]: as zooming is mapped to angular movements, the larger the circular gesture's radius, the greater the distance

that has to be covered to make a full circle, and consequently the more precise the input.

2.3.3 Guidance through Passive Haptic Feedback

Two main categories of techniques have been studied for mid-air interaction on wall-sized displays: freehand techniques based on motion tracking [163, 181]; and techniques that require the user to hold an input device [23, 41, 112, 119]. Input devices provide some guidance to the user in terms of what gesture to execute, as all of them provide some sort of passive haptic feedback: a finger operating a knob or a mouse wheel follows a specific path; gestures on touch-enabled devices are made on planar surfaces. Freehand techniques, on the contrary, provide essentially no feedback to the user who can only rely on proprioception [122] to execute the gesture. We call this dimension the *degree of guidance*. Gestures can be guided to follow a particular path in space (*1D path*); they can be guided on a touch-sensitive surface (*2D surface*) ; or they can be totally free (*3D free*). These three values correspond to decreasing amounts of passive haptic feedback for the performance of input gestures.

These dimensions combine into the twelve techniques represented in Figure 2.3, that were implemented following a set of design choices oriented towards performance (task time and accuracy) as well as other usability issues, such as fatigue and ease of use.

2.4 Design Choices

2.4.1 Bi-manual interaction

Our bimanual techniques for panning and zooming (Figure 2.3, bottom row) are grounded in Guiard's [73] study of asymmetric division of labor in bimanual actions that led to the Kinematic chain model: motion of the dominant hand typically finds its spatial reference in the results of motion of the non-dominant-hand.

Zooming out is the first step of an open pan-and-zoom task (step 2 in Figures 2.1 and 2.2). Panning only occurs when the target ROI is visible, either to bring it closer to the user's physical location (steps 3 and 5 in Figure 2.1) or to compensate for focus drift caused by dynamic zooming (final step in Figure 2.2); zooming thus brings context to panning as well as to any action occurring after the navigation is performed. Controlling the zoom level is thus assigned to the non-dominant hand in bi-manual configurations, as is typically the case for bimanual pan-zoom techniques on the desktop [34, 37]. One could point out that pointing the focus of expansion of the zoom interaction sets the frame of reference to zooming, thus should be allocated to the non-dominant hand as well. However

providing the focus point is not the sole purpose of pointing in a zoomable interface: users often navigate in order to optimize their viewport for further interaction, which is often performed through pointing. We consider navigation as a preliminary task to facilitate further interaction, so we allocate pointing to the dominant hand in all configurations. As it relies on pointing, panning is also assigned to the dominant hand and enabled through a quasi-mode, i.e. pressing a button.

Pointing and panning with the same hand implements the *Factorizing task allocation* strategy of my *Task Allocation* taxonomy. Zooming with the non-dominant hand in bi-manual configurations is an example of the *Maximizing limb usage* allocation strategy. In uni-manual configurations, zooming is also assigned to the dominant hand, increasing the "factorization" of task allocation.

2.4.2 Input Gestures via a Device

The main limb segments involved in the input of gestures via a handheld device are the fingers, palm and, to a lesser extent, the forearm (for the dominant hand). This group of techniques is illustrated in Figure 2.3, columns *1D path* and *2D surface*.

Column *1D path* illustrates techniques that provide a high degree of guidance for executing the zooming gestures. The first row corresponds to *one handed* techniques: the device is operated by the dominant hand, which also controls pointing via ray-casting. The second row corresponds to *two handed* techniques: the dominant hand controls pointing via ray-casting, while the non-dominant hand controls zoom using the device. *linear* gestures can be input using, e.g., a wireless handheld mouse featuring a scroll wheel; *circular* gestures using, e.g., any type of handheld knob. Depressing a button on the device activates drag mode for panning.
Panning and zooming are essentially two continuous tasks: users can virtually access any zoom level and any location in the visualization depending on the zoom level.

Column *2D surface* illustrates techniques that use a touch-sensitive surface for input, providing a lesser degree of guidance than *1D path*. The surface is divided horizontally in two areas. Users zoom in the upper area either by moving the thumb up and down (*linear* case), or by drawing approximate circles (*circular* case). Touching the lower area activates drag mode for panning. Users only rely on proprioceptive information to switch between both areas and do not have to look at the device. These techniques can be implemented with a touch-sensitive handheld device such as a PDA or smartphone.

1D path techniques employing *circular* gestures provide more guidance, but do not benefit from the earlier-mentioned Vernier effect, as input is constrained to one specific trajectory.

However, the range of amplitudes that can be covered with the thumb is limited [147]. This should minimize the trade-off between *1D path* and *2D surface* in that respect. For *2D surface* techniques, rubbing gestures [134] were considered to avoid clutching when performing linear gestures, but were found to be impractical when performed with the thumb on a handheld touch-sensitive surface. As a technique designed specifically for thumb input, we were also interested in MicroRolls [147]. However, these were originally designed for discrete input. Cardinal MicroRolls would have had to be mapped to first order of control, which we discarded as discussed earlier, and circular MicroRolls are not precise enough for zoom control.

2.4.3 Input Gestures in Free Space

The main limb segments involved in performing gestures in free space are the wrist, forearm and upper arm. This group of techniques is illustrated in Figure 2.3, column *3D free*.

The first row illustrates *one handed* techniques using either *linear* or *circular* gestures. The technique using *circular* gestures is actually very close to the CycloStar zooming gesture, but performed in mid-air, without touching any surface. Users perform circular gestures with the dominant hand and forearm oriented toward the display. As in CycloStar, the focus of expansion is the centroid of the round shape corresponding to the cursor's circular path, here projected on the display surface (dotted arrow in Figure 2.3-e). The technique using *linear* gestures consists in pushing the dominant hand forward to zoom in, as if reaching for something, with the palm towards the target. Turning the hand and pulling backward (away from the display) zooms out. Users point orthogonally to the palm of the same hand (blue arrows in Figure 2.3-e, left side), with the arm slightly tilted for greater comfort. The second row illustrates *two handed* techniques (Figure 2.3-f). The *linear* zooming gestures are similar to the ones above, but are performed with the non-dominant hand, the dominant hand still being used for pointing and specifying the focus of expansion. In the *circular* case, users adopt a potentially less tiring posture, pointing at the floor with their non-dominant hand and making circular movements. All other postures and movements being ignored by the system for the non-dominant hand, the user can easily clutch. Several options can be considered for engaging drag mode: specific hand postures such as pinching, or using a small wireless actuator (e.g., a button).

2.5 Experiment

We conducted an experiment using a [2×2×3] within-subjects design with three primary factors: Handedness ∈ {*OneHanded*, *TwoHanded*}, Gesture ∈ {*Circular*, *Linear*}, and Guid-

ance $\in$ {*1DPath*, *2DSurface*, *3DFree*} to evaluate the 12 unique interaction techniques described above. We controlled for potential distance effects by introducing the Distance between two consecutive targets as a secondary within-subjects factor. We systematically varied these factors in the context of a multiscale navigation task within the WILD room (Appendix A page 213).

Measures include performance time and number of overshoots, treated as errors. Overshoots occur when participants zoom beyond the target zoom level, and indicate situations in which the participant has less precision of control over the level of zoom. A practical example could be, from an overview of Canada, zooming down to street level in Google Maps when the user actually wanted to get an overview of Vancouver.

2.5.1 Hypotheses

Based on the research literature and our own experience with the above techniques, we made the following 7 hypotheses.

Handedness: prior work [34, 37, 74, 109] suggests that two-handed gestures will be faster than one-handed gestures (*H1*) because panning and zooming are complementary actions, integrated into a single task [93]. Two-handed gestures should also be more accurate and easier to use (*H2*).

Gesture: Linear gestures should map better to the zooming component of the task, but should eventually be slower because of clutching, the limited action space compared to zoom range requiring participants to repeatedly reposition their hand/finger (*H3*). Prior work [124, 172] suggests that users will prefer clutch-free circular gestures (*H4*).

Device vs. Free Space: Zhai et al. [177] suggest that techniques using the smaller muscle groups of fingers should be more efficient than those using upper limb segments. Balakrishnan *et al.* [18] moderate this observation with findings suggesting that the fingers are not performing better than forearm or wrist for a reciprocal pointing task. Nevertheless, they acknowledge that differences exist in the motor system's ability to control the different limb segments. Based on the gestures to be performed and taking into account the physical size and mass of the segments involved, we predicted that techniques using fingers (*1DPath* and *2DSurface* conditions), should be faster than those requiring larger muscle groups (hands and arms, *3DFree* conditions) (*H5*).

We also predicted that *1DPath* gestures would be faster, with fewer overshoots than techniques with lesser haptic feedback, i.e., *2DSurface* and *3DFree* (*H6*). Finally, we predicted that *3DFree* gestures would be more tiring (*H7*).

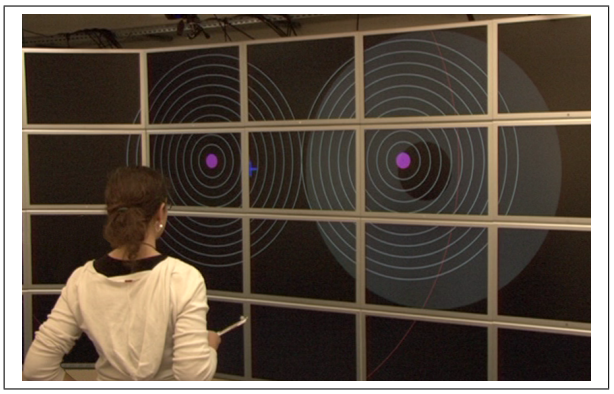


Figure 2.4: Participant performing the task

2.5.2 Participants

We recruited 12 participants (1 female), ranging in age from 20 to 30 years old (average 24.75, median = 25). All are right-handed daily computer users. None are color-blind.

2.5.3 Apparatus

Hardware.

This experiment was run in the WILD room (see Appendix A page 213): the graphics were displayed on the wall display and the participants' gestures were tracked with the VICON motion tracking system. The *Linear 1DPath* condition used the wheel of a wireless Logitech M305 mouse (Fig. 2.3-a,b). The *Circular 1DPath* condition used a wireless Samsung SM30P pointing device, normally used for presentations (Fig. 2.3-a,b). All *2DSurface* conditions used an iPod Touch. In order to avoid failures from gesture segmentation algorithms that would impact task performance in an uncontrolled manner, we used an explicit mode switch to unambiguously engage drag mode (panning). As mentioned earlier, we used the device's main button for *1DPath* conditions, and the lower area of the touch-sensitive surface for *2DSurface* conditions. While in real-world applications we would use specific hand postures such as pinching in *3DFree* conditions, for the sake of robustness we used a wireless mouse button whose activation is seamlessly integrated with the gesture.

Software. The experiment was written in Java 1.5 running on Mac OS X and was implemented with the jBricks [138] library. Touchstone [113] was used to manage the experiment.

The circular gestures of the *2D surface* and *3D free* conditions were recognized by fitting an elliptic function to a corpus of points, i.e. finding values for a, b, c, d, e, and f that

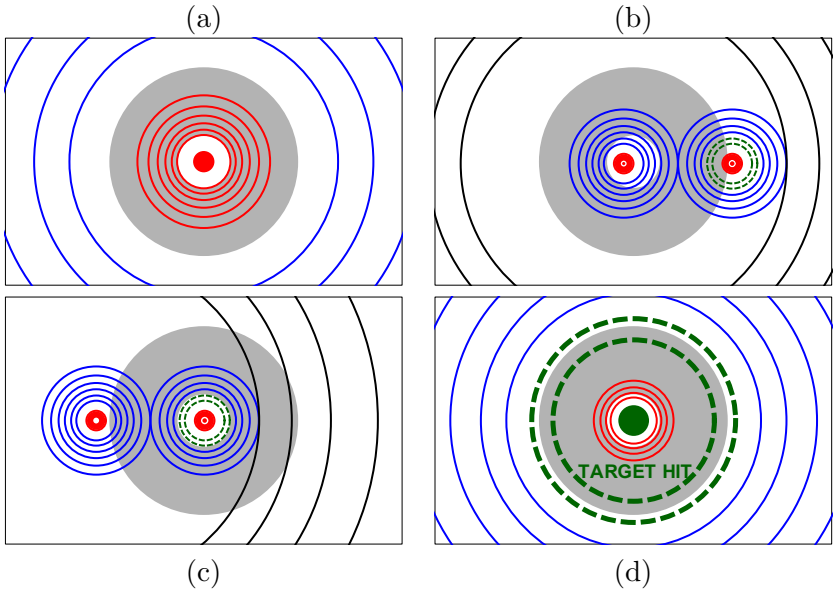


Figure 2.5: Task (schematic representation using altered colors): (1) Groups of concentric circles represent a given position and zoom level. (2) Zooming out until the neighboring set of circles appears. (3-4) Pan and zoom until the target (green inner disc and circles, dashed for illustration purposes only) is positioned correctly with respect to the stationary gray ring.

minimized $a.x^2 + b.xy + c.y^2 + d.x + e.y + f$ for all the coordinates in the corpus. Angular events were computed from the center of the fitted ellipsis. The velocity of the zoom was proportional to the average instant velocity of the last five angular events. The circular gestures were performed eye-free and without visual feedback other than the direct zoom response of the technique, so we used a few heuristics to improve the behavior of this algorithm:

1. The recognition only started when the corpus of points contained at least 6 coordinates ($\approx$ 100 ms at 60 Hz) in order to improve the reliability of the initial zoom movements. This caused a small lag at the beginning of zooming phases, but ensured that the initial zoom direction and amplitude was under control; we will discuss this effect in the Results subsection (p. 60).

2. Angular velocities beyond a threshold ($> .15$ rad.ms^{-1}) were discarded because they were likely to be caused by recognition errors.

3. Changes in rotating direction, e.g. clockwise to counter-clockwise, were only taken into account after 10 angular events ($\approx$ 167 ms) in the same direction in order to prevent zoom direction noise caused by imprecise circular movements.

We carefully tuned these heuristics in order to improve the behavior and performance of *circular* techniques.

2.5.4 Pan-Zoom Task

The task is a variation of Guiard *et al.*'s multiscale pointing task [74], adapted to take overshoots into account. Participants navigate through an abstract information space made of two groups of concentric circles: the *start group* and the *target group*. Each group consists of seven subgroups of 10 concentric circles symbolizing different zoom levels, each designated by a different color (Fig. 2.5.b). The target group features two additional green circles (dashed in Fig. 2.5.d) and a central, smaller disc, referred to as D, C_1 and C_2 from smallest to largest.

Participants start at the zoom level of the target but centered on the start group (Fig. 2.5.a). They zoom out, possibly through several subgroups depending on the DISTANCE secondary factor value, until the neighboring target group appears (Fig. 2.5.b). It may appear either on the left or right side of the start group. Then they pan and zoom into the target group until they reach the correct zoom level and the target is correctly centered. A stationary gray ring symbolizes the correct zoom level and position (Fig. 2.5-(a-d)). Its diameters are $d_1 = 1120$ pixels (259 mm) and $d_2 = 5280$ pixels (1219 mm). All three criteria must be met for the trial to end:

1. D is fully contained within the stationary ring's hole of diameter d_1,
2. (C_1) is visually smaller than the stationary ring of diameter d_2,
3. (C_2) is visually larger than the stationary ring.

Overshoots occur when the zoom level is higher than the maximum level required to meet criteria (2) and (3), in which case participants have to zoom out again (D becomes white instead of green in that situation). When all conditions are met, the message TARGET HIT appears and the thickness of C_1 and C_2 is increased (Fig. 2.5.d). The trial ends when the position and zoom level have stabilized for at least 1.2 seconds (all trials must be successfully completed).

2.5.5 Procedure

The experiment presents each subject with six replications of each of the 12 techniques at three DISTANCES: 49 920 px (11 529 mm), 798 720 px (184 464) and 12 779 520 px (2 951 390 mm)[3]. The experiment is organized into four sessions that each present three techniques: One combination of the GESTURE and HANDEDNESS factors and all three degrees of GUIDANCE. Each session lasts between 30 and 90 minutes, depending on techniques and

[3]The most difficult of these *view pointing* tasks [74] has a Fitts' ID of 13.4 bits. Acquiring a 50 mm target at such a distance means performing a pointing task of 15.85 bits

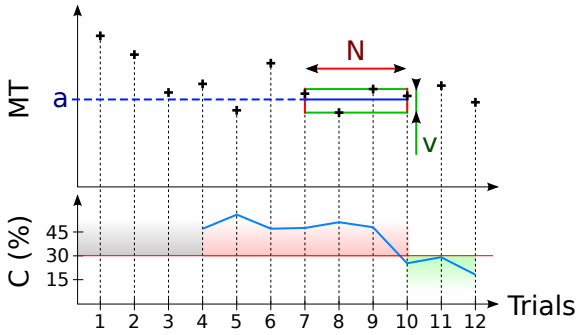


Figure 2.6: Criterion for the practice blocks: a user's performance is considered stable if the ratio $C = v/a$ is lower than a given criterion C (here C=30%) in the last N trials (here N=4). v is the variation of the movement times of the trials, simply computed as MT_{max} - MT_{min} within the last N trials. a is the average of MT within the last N trials. In this example, trials 7 to 10 met the criterion and trials 11 and 12 were performed as additional practice requested by the participant.

participant. Participants are required to wait at least one hour between two consecutive sessions, and to complete the whole experiment within four days or fewer, with a maximum of two sessions per day to avoid too much fatigue and boredom. Participants stand 1.7m from the wall and are asked to find a comfortable position so they can perform gestures quickly, but in a relaxed way.

Practice Condition: Participants are given a brief introduction at the beginning of the first session. Each technique begins with a practice condition, with trials at all three different Distances. Measures for the experimental condition start as soon as 1) participants feel comfortable and 2) task performance time variation for the last four trials is less than 30% of the task time average in that window, as explained in Figure 2.6. This criterion ensures that the performance of all participants stabilized at least once before starting the measured trials, regardless of the participant's previous experience with mid-air interaction, circular gestures and so on.

Experimental Condition: Each technique is presented in a block of 18 trials consisting of 6 replications at each Distance. Trials, blocks and sessions are fully counter-balanced within and across subjects, using a Latin square design.

Measures: We measure movement time MT and number of overshoots for each of 2592 trials: 2 Gesture × 2 Handedness × 3 Guidance × 3 Distance × 12 participants × 6 replications. Participants also answer questions, based on a 5-point Likert scale, about their perceived *performance*, *accuracy*, *ease of learning*, *ease of use*, and *fatigue*. They rank the techniques with respect to the Guidance factor after each session. When they have been exposed to both conditions of Handedness or Gesture, they rank those as well. After the last session, they rank the techniques individually and by factor. Participants

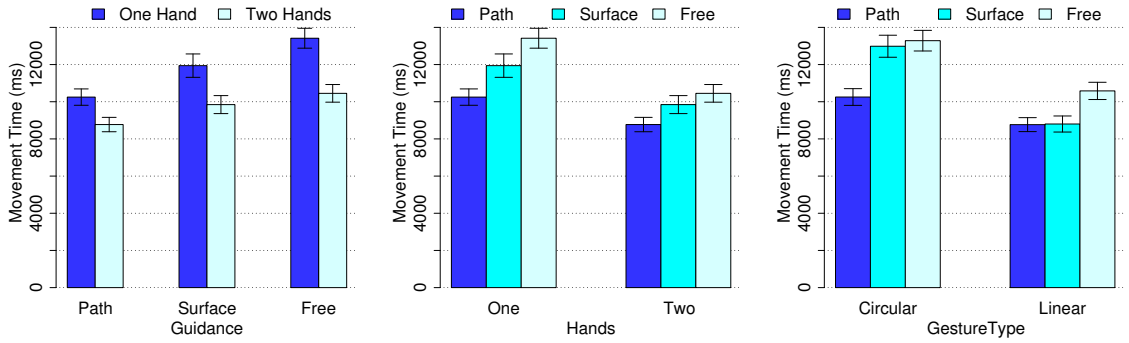


Figure 2.7: (a): *MT* per HANDS × GUIDANCE. (b) *MT* per GUIDANCE × HANDS. (c) *MT* per GUIDANCE × GESTURE.

are encouraged to make additional observations and comments about any of the above.

2.5.6 Results and Discussion: Movement Time

Prior to our analysis, we checked the performance for unwanted effects from secondary factors. We checked for individual performance differences across subjects and found that, for all 12 participants, movement time and number of overshoots were perfectly correlated with the overall performance measures.

As expected, movement time data are skewed positively; replications of unique experimental conditions are thus handled by taking the median (note that taking the mean yields similar results). In all remaining analyses, we handled participant as a random variable, using the standard repeated measures REML technique from the JMP 9 statistical package. We found no significant fatigue effect although we did find a significant learning effect across sessions. Participants performed about 1.4 s more slowly in the first session and then became slightly faster over the next three sessions. However, we found no significant interaction between session orders and main factors. As the factors were counter-balanced, this created no adverse effects in the analysis.

Table 2.2 details the results of the full factorial ANOVA for the model $MT \sim$ HANDS × GUIDANCE × GESTURE × DIST × Rand(Participant). We observe that HANDS has a significant effect on *MT* (Figure 2.7-a[4]). A post-hoc Tukey test shows that *TwoHanded* gestures are significantly faster than *OneHanded* gestures (avg. 9690 ms vs. 11869 ms). We found a significant interaction effect of HANDS × GUIDANCE (Figure 2.7-a). The interaction does not change the significance of the post-hoc test, but indicates that the magnitude of the

[4]Error bars in all the figures represent the 95% confidence limit of the mean of the medians per participants ($\pm StdErr \times 1.96$).

Factor	**DF, DFDen**	F **Ratio**	p
HANDS	1,11	24.65	0.0004 *
GESTURE	1,11	42.87	< 0.0001 *
GUIDANCE	2,22	58.80	< 0.0001 *
DIST	2,22	228.8	< 0.0001 *
HANDS×GESTURE	1,11	2.060	0.1790
HANDS×GUIDANCE	2,22	4.914	0.0172 *
GESTURE×GUIDANCE	2,22	10.38	0.0007 *
GESTURE×DIST	2,22	17.27	< 0.0001 *
HANDS×DIST	2,22	11.57	0.0004 *
GUIDANCE×DIST	4,44	3.828	0.0094 *
HANDS×GESTURE×GUIDANCE	2,22	1.127	0.3420
HANDS×GESTURE×DIST	2,22	0.790	0.4661
HANDS×GUIDANCE×DIST	4,44	0.650	0.6301
GESTURE×GUIDANCE×DIST	4,44	3.750	0.0104 *
HANDS×GESTURE×GUIDANCE×DIST	4,44	1.049	0.3929

Table 2.2: Results of the full factorial ANOVA for MT.

difference is greater for *3DFree* than for *2DSurface* and greater for *2DSurface* than for *1DPath* techniques.

Unsurprisingly, performance data strongly support (*H1*): all other conditions being equal, two-handed techniques are consistently faster than one-handed techniques. An interesting observation is that using two hands is more advantageous when the degree of guidance for achieving gestures is low.

GUIDANCE has a significant effect on MT (Figure 2.7-b). A post-hoc Tukey test shows that *1DPath* (avg. 9511 ms) is significantly faster than *2DSurface* (10894 ms), which in turn is significantly faster than *3DFree* (11934 ms). This time the HANDS × GUIDANCE interaction changes the significance of the test (Figure 2.7-b). The difference is that a post-hoc Tukey test shows no significant difference between *2DSurface* and *3DFree* for *TwoHanded*.

Both hypotheses (*H5*) and (*H6*) are supported: involving smaller muscle groups improves performance; providing higher guidance further contributes to this. However, this effect is less pronounced in *TwoHanded* conditions. This confirms the previous observation that a higher degree of guidance is especially useful when a single hand is involved.

GESTURE also has a significant effect on MT. A post-hoc Tukey test shows that *Linear* movements (avg. 9384 ms) performed significantly faster than *Circular* gestures (12175 ms). However, we have a strong significant interaction of GESTURE × GUIDANCE (Figure 2.7-c). A post-hoc Tukey test shows that:

(i) for *Circular* gestures: *1DPath* guidance is faster than both *2DSurface* and *3DFree* with no significant difference between *2DSurface* and *3DFree*;

(ii) for *Linear* gestures, there is no significant difference between *1DPath* and *2DSurface*, but a significant difference between *2DSurface* and *3DFree*;

(iii) for *1DPath* guidance there is no significant difference between *Circular* and *Linear* gestures, but there is a significant difference between *Circular* and *Linear* for *2DSurface* and *3DFree* guidance.

Surprisingly, *Linear* gestures are generally faster than *Circular* ones. (*H3*), which claimed that *Linear* gestures should be slower because of clutching, is not supported. Performance differences between gesture types are however affected by the degree of guidance: *Circular* gestures with *1DPath* guidance (e.g., a knob) are comparable to *Linear* gestures with low guidance. We tentatively explain the lower performance of *Circular* gestures with *2DSurface* guidance by the difficulty of performing circular gestures with the thumb [147]. The 100-ms lag at the beginning of a *circular* zoom gesture (see Apparatus page 56) could also partially explain this effect, although we observe that the average difference in movement time greatly exceeds 100 ms in all other conditions (see Table 2.3).

Another interesting observation is that our analogue to CycloStar in mid-air (*Circular* gestures with *3DFree* guidance) performs poorly. It seems that the lack of a surface to guide the gesture significantly degrades the usability of this technique. In the *OneHanded* condition, this can also relate to the control of the focus of expansion: specifying precisely the center of a circular gesture performed in mid-air, as with the *Circular* techniques, seems harder to control than pulling from or pushing towards a precise location, as with the *Linear* techniques. Another factor contributing to the poor performance of *Circular* in our study is likely related to overshoots, as discussed below.

As expected, distance to target (Dist) has a significant effect on MT. A post-hoc Tukey test shows that MT increases significantly with *distance*. There are several significant interactions between Dist and the main factors (Fig. 2.8), but none of these change the relative performance ordering for the main factors. These interactions are due to a change in the magnitude of the difference across conditions, confirming that the choice of an efficient technique is of increasing importance as the task becomes harder.

2.5.7 Results and Discussion: Overshoots

As detailed earlier in the description of task design, overshoots correspond to zooming beyond the target zoom level and are treated as errors. We consider the model *Overshoots* $\sim$ Hands $\times$ Guidance $\times$ Gesture $\times$ Dist $\times$ Rand(Participant).

We observe significant simple effects on *Overshoots* for Gesture ($F_{1,11} = 21.04$, $p = 0.0008$) and Guidance ($F_{2,22} = 53.80$, $p < 0.0001$), and one significant interaction effect for Gesture

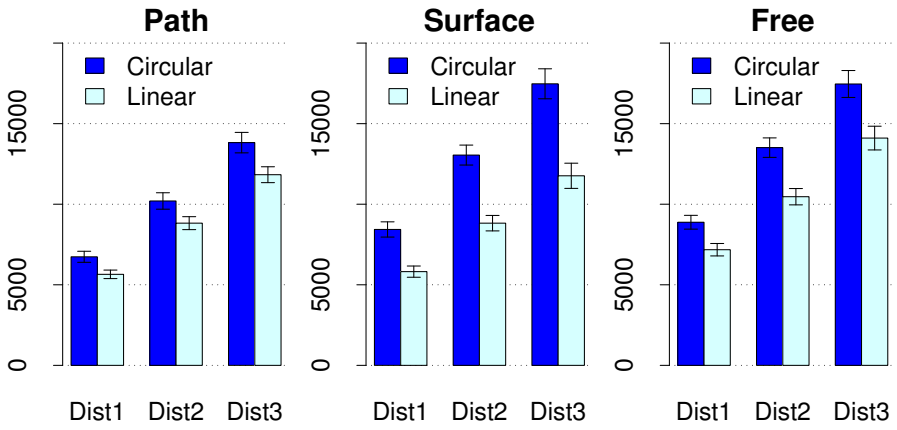


Figure 2.8: *MT* per DIST × GESTURE, for each GUIDANCE

× GUIDANCE ($F_{2,22} = 8.63$, $p = 0.0017$). *Circular* gestures exhibit more overshoots than *Linear* gestures (1.65 vs. 2.71). *2DSurface* gestures exhibit more overshoots than *1DPath* and *3DFree* gestures (3.75 for *2DSurface* vs. 1.52 for *1DPath*, and 1.26 for *3DFree*). There is a significant difference between *Linear* and *Circular* gestures for *2DSurface* and *3DFree*, but not *1DPath*. Moreover, overshoots exhibit the same interaction effect for *2DSurface* gestures: *Circular 2DSurface* result in significantly more overshoots than *Linear 2DSurface* (4.68 vs. 2.82).

The observed higher number of overshoots for *Circular* techniques helps explain the generally lower *MT* performance measured for this type of gestures. The best-fitting ellipse algorithm involved in the recognition of *Circular* gestures has an inherently higher cost of recovery, introducing a delay when initiating a zoom and reversing its course. The poor performance of our analogue to CycloStar is at least partially due to this. In addition, there seems to be a major difference between the zooming experiment reported in [116] and the present one: we included overshoots in our task design, whereas the CycloStar experiment apparently did not (there is no report of such a measure in the task design nor analysis of results).

2.5.8 Results and Discussion: Subjective Results

Quantitative subjective data confirms our results. Participants generally preferred *TwoHanded* to *OneHanded* techniques (8/12) and *Linear* to *Circular* gestures (10/12). Subjective preferences about degree of guidance were mixed, with 4 participants preferring the high degree of guidance provided by *1DPath* techniques, only 1 for both of *2DSurface* and *3DFree* techniques, and all others expressing no particular preferences. Looking at the details of answers to our 5-point Likert scale questions about perceived speed, accuracy, ease of use

and fatigue, significant results ($p < 0.002$) were obtained only for degree of GUIDANCE, with *1DPath* being consistently rated higher than *2DSurface* and *3DFree*; and for HANDS, *TwoHanded* techniques being considered less tiring than *OneHanded* techniques ($p < 0.03$).

Comments from participants suggest that in the *OneHanded* condition, zoom gestures interfere with pointing as they introduce additional hand jitter and consequently lower accuracy. Some participants also commented that pointing and zooming were confounded in the *OneHanded* conditions, making the techniques difficult to use (*H2*). However, two participants strongly preferred one-handed gestures, arguing that they were less complex and less tiring. They assumed their performance was better (even though it was not), probably because they experienced more overshoots in the *two handed* condition, which may have led to their conclusions. One of them mentioned that for the *one handed* condition there was *"no need for coordination"*, techniques were *"more relaxed"* and made it *"easier to pan and zoom"*.

Linear gestures were preferred to *Circular* ones. Participants commented that circular gestures were difficult to perform without guidance, that circular gestures for zooming interfered with linear gestures for panning, and that circular gestures were hard to map to zoom factor. All but one participants preferred linear gestures overall although one commented that he liked *"the continuity of circular gestures"*. Others commented that *"making good circles without a guide is hard"* and did not like having to turn their hands. These findings contradict our hypothesis that users would prefer clutch-free circular gestures (*H4*). This hypothesis was based on observations made for techniques operated on a desktop, not in mid-air, and involved different limb segments. In many of our conditions, the gestures had to be performed with the thumb, and were thus more complex to achieve than when using, e.g., the index finger in conjunction with hand or forearm movements. Several participants commented on this interaction effect: *"[It is] too hard to do circle gestures without a guide"*, *"Linear movements are easier on the iPod"* and *"[Is it] impossible to do circular movements on a surface, maybe with some oil?"*.

Finally, as hypothesized (*H7*), participants found *1DPath* guidance least tiring while *3DFree* caused the most fatigue.

2.5.9 Results and Discussion: Individual Techniques

The analysis of variance for the model $MT \sim$ HANDS $\times$ GUIDANCE $\times$ GESTURE $\times$ DIST $\times$ Rand(Participant) does not show a significant triple interaction between the three main factors (Table 2.2). Formally, we cannot say more than the above about the ranking of the twelve techniques. However, based on the results about MT above, we can observe

Group	Hands	Gesture	Guidance	Figure	MT (ms)
Gr1	*TwoHanded*	*Linear*	*2DSurface*	2.3-d	8 100
	TwoHanded	*Linear*	*1DPath*	2.3-b	8 377
Gr2	*OneHanded*	*Linear*	*1DPath*	2.3-a	9 160
	TwoHanded	*Circular*	*1DPath*	2.3-b	9 168
	TwoHanded	*Linear*	*3DFree*	2.3-f	9 185
	OneHanded	*Linear*	*2DSurface*	2.3-c	9 504
Gr3	*OneHanded*	*Circular*	*1DPath*	2.3-a	11 340
	TwoHanded	*Circular*	*2DSurface*	2.3-d	11 591
	TwoHanded	*Circular*	*3DFree*	2.3-f	11 718
	OneHanded	*Linear*	*3DFree*	2.3-e	11 981
Gr4	*OneHanded*	*Circular*	*2DSurface*	2.3-c	14 380
	OneHanded	*Circular*	*3DFree*	2.3-e	14 851

Table 2.3: Groups of techniques according to MT

four distinct groups of techniques, shown in Table 2.3. As a side note, if we consider the model $MT \sim$ Group $\times$ Rand(Participant), the ANOVA shows a significant effect of Group ($F_{3,33} = 65.35$, $p < 0.0001$) and a post-hoc Tukey test shows a significant difference between each groups.

Gr1 contains the two fastest techniques with similar MT : *TwoHanded*, *Linear* gestures with either *2DSurface* or *1DPath* degrees of guidance. Optimal performance in terms of movement time implies the use of two hands and a device to guide gestural input.

Gr2 contains the four techniques that come next and also have close MT : the *OneHanded* version of the two fastest techniques, the *TwoHanded Circular 1DPath* and the *TwoHanded Linear 3DFree* techniques. Techniques in this group are of interest as they exhibit a relatively good level of performance while broadening possible choices for interaction designers: all the values of our main factors are represented at least once. For instance, the unimanual techniques in this group leave one hand available to perform other actions. The *3DFree* technique is also of interest as it does not require the user to hold any equipment and is generally appealing to users.

Gr3 contains techniques that again have very close MT but about 2.3 s slower than the techniques in *Gr2*. This group consists of *OneHanded Circular 1DPath*, *TwoHanded Circular 2DSurface* and *3DFree*, and *OneHanded Linear 3DFree*. Techniques in this group are of lesser interest, except maybe for the *OneHanded Linear 3DFree* technique, which is the fastest unimanual technique using gestures performed in free space.

Gr4 contains the 2 techniques performing worst, *OneHanded Circular 2DSurface* and *3DFree*. These are significantly slower than all others, about 3 s slower than the techniques in *Gr3* and about 6 s slower than the techniques in *Gr1*. Our data suggest that these techniques

should be rejected.

2.6 Summary and Discussion

We have shown how the input channels available in mid-air can be used to manipulate a viewport for virtual multi-scale navigation. We studied location-independent, mid-air input techniques for pan-zoom navigation on wall-sized displays. These techniques must be usable with the input sources available in such platforms, thus we paid very careful attention to the combination of the resulting input channels. An extensive exploratory design phase allowed us to identify three key factors for the design of such techniques: handedness (uni- vs. bimanual input), gesture type (linear or circular), and level of guidance (movements restricted to a 1D path, a 2D surface or free movements in 3D space). We systematically evaluated each combination of these factors through a controlled experiment in which participants performed pan-and-zoom navigation in an abstract, very large multiscale environment, with distances up to 12 million pixels. The contribution of this work is two-fold:

First, we describe a comprehensive design space of combined input characteristics with specific advantages and drawbacks, including how these characteristics combine or interfere with each other and with direct pointing:

1. Physical guidance increases accuracy as opposed to gestures performed in free space; the latter is generally less efficient and makes users tired more quickly, but can be more appealing to users, benefiting from a "cool" effect. It is interesting to note that free-space gestures benefit from much more movement amplitude (approximately 30 cm, as opposed to less than 5 cm with *1DPath* and *2DSurface*) yet end up being less precise, due to the fact that finger movements are much more precise than hand movements.

2. Bimanual input performs very well. It provides more control over the two parameters of targeted zooming, namely the zoom level and the location of the focus of expansion. The critical importance of the latter makes unimanual input less precise since all zoom input causes hand tremor. However bimanual input also prevents users from holding other input devices for task-specific interactions. For situations where such tools are needed, some unimanual combinations of inputs still provide good performance.

3. Circular gestures perform less well than linear gestures, even though they do not need clutching for large-amplitude movements. Highly constrained circular gestures performed on a physical knob can still perform reasonably well if held in the hand

that does not control the location of the focus of expansion.

These results can probably be applied to tasks other than panning and zooming. Bimanual input has already been shown to improve performance and control in a number of situations. Our results about physical guidance are more novel and can probably be extended to other types of control. All combinations of *Linear* and *1DPath* and *2DSurface* factors belong to the two best performing groups of techniques, from which we infer that guided physical movements are the most efficient combinations when they cognitively match the expected virtual movement (here, bringing the camera nearer or further away from the user). *Circular* input is much harder to control and can only compete with *Linear* when using physically constrained movements performed with a dedicated hand, even though our criterion for the end of the training sessions required that participants' performance stabilized for all conditions. The fact that *Circular* movements, despite being clutch-free, performed poorly may also be due to the fact that zooming is inherently a linear movement of the camera relative to the scene. It might be interesting to evaluate *Linear* versus *Circular* movements for a circular task such as defining an orientation or choosing a value within a circular range such as a color hue.

Second, we propose a set of six mid-air pan-and-zoom techniques with good performance and with input characteristics that match the capabilities of a large display platform. An extensive design and tuning phase was necessary to map our design space, and the resulting techniques are usable beyond the scope of the controlled experiment presented here:

TwoHanded-Linear-1DPath is on average the fastest technique while at the same time the simplest to implement since it only requires calibrating the effect of a mouse wheel tick on the zoom level. The remaining mouse buttons can be used for a small number of application-specific actions, but more complex vocabularies need more input expressiveness.

TwoHanded-Linear-1DPath is less than 300 ms slower in average (i.e. less than 5% of the total movement time). It is a bit more expensive and possibly less simple to implement since it requires a hand-held tactile screen, but could accommodate more additional commands depending on the available screen real estate.

OneHanded-Linear-1DPath **and** *2DSurface* are the one-handed counterparts of the two best techniques. They are equally simple to implement. They are also the only single-handed techniques with good performance, therefore they should be preferred if the task requires users to hold specific tools in their non-dominant hand.

TwoHanded-Linear-3DFree is the only usable free-hand technique. It is more tiring than the

other *TwoHanded-Linear* techniques but also possibly more entertaining for demos and / or public applications. It also requires very limited input (i.e. a button) in addition to motion capture.

The performance groups presented in Section 2.5.9 coincide with the *Task Allocation* taxonomy from Chapter 1 (Section 1.3). Recall that two-handed techniques match the *Maximizing limb usage* strategy by allocating panning to the dominant arm and zooming to the non-dominant arm, i.e. two different limbs. Conversely, one-handed pan-and-zoom techniques fit the *Factorizing task allocation* by assigning the control of panning and zooming to the dominant arm. One of the results of the experiment is that "*Maximizing*" techniques have proven more efficient and were preferred over "*Factorizing*" techniques. Furthermore, being one-handed actually caused a systematic shift down in the performance results, as shown in Figure 2.9: for every combination of the GESTURE and GUIDANCE factors, the one-handed (*Factorized*) technique is systematically in the *group* below the two-handed (*Maximized*) technique. This observation emphasizes the performance gap between the two *Task Allocation* strategies. I hypothesize this gap is caused by the simultaneous control of zoom level and focus of expansion interfering with each other in the one-handed condition.

Nevertheless, the *Factorizing task allocation* strategy worked well for combining pointing and panning[5]. This is probably because participants never had to perform diverging pointing and panning tasks, i.e. pointing a target while panning towards a different region of the scene.

In the next chapter I will explore another combination of interactions, namely mid-air pointing and invoking commands. Similar to this chapter, pointing and command invocation will be combined following the *Task Allocation* taxonomy introduced in Chapter 1. However pointing will not be functionally linked to command invocation, as opposed to with panning and zooming in this chapter, which should provide additional data on the effects of task allocation strategies on usability and performance. The next chapter will also explore mid-air command invocation, which will contribute to the vocabulary of techniques usable in large display environments.

[5]Though we did not compare it to other solutions.

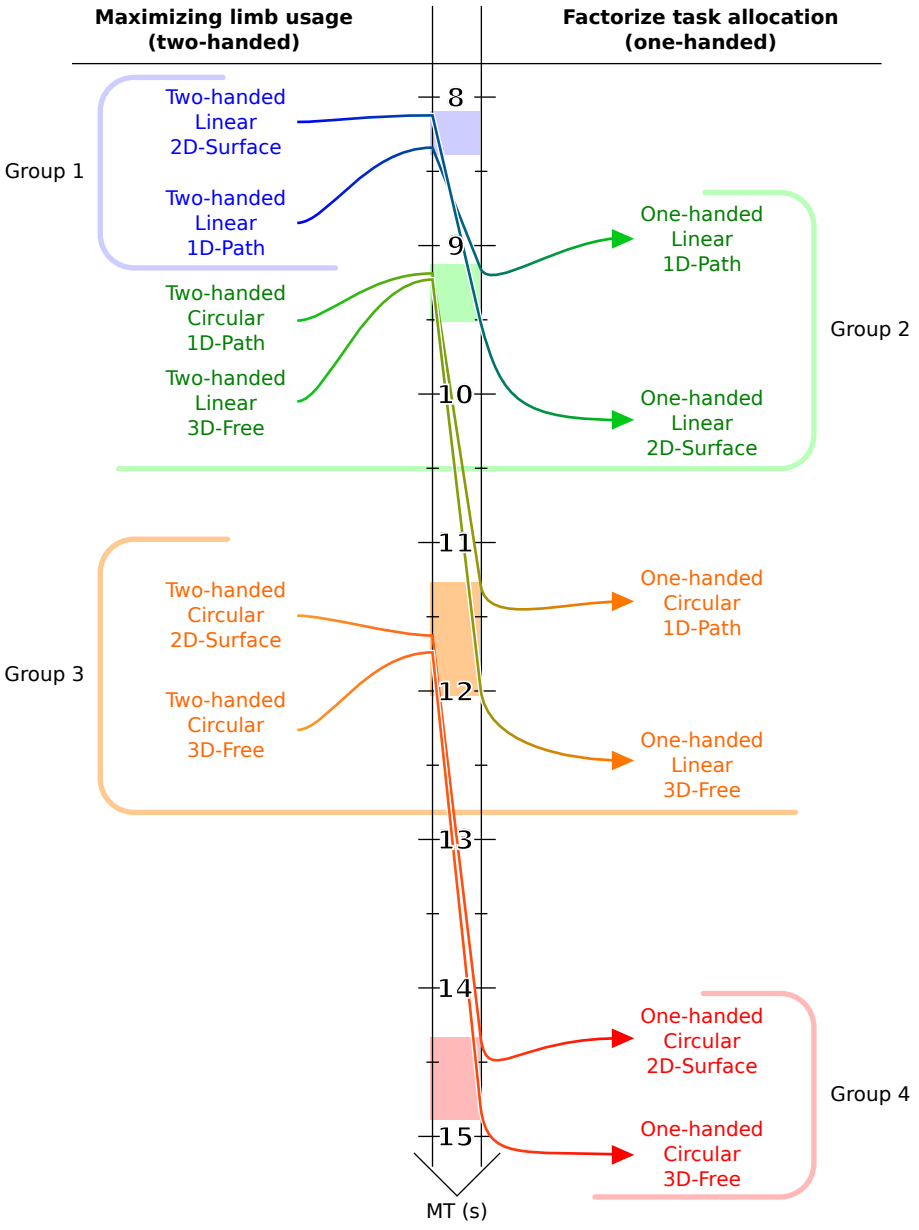


Figure 2.9: Technique performance depending on groups (Section 2.5.9) and on the *Task Allocation* taxonomy (Section 1.3). Arrows represent combinations of GESTURE and GUIDANCE going from two-handed (left) to one-handed (right). The left and right intersections with the MT (movement time) axis represent respectively the two- and one-handed average MT as reported in Table 2.3. The light areas on the MT axis represent the range of average MT for a given group.

Chapter 3

Device-less Discrete Selection in Large Display Platforms

3.1 Introduction

Existing real-world applications provide a number of discrete commands, actions and mode switches; extreme cases such as Autodesk Maya[1] can feature more than 1000 different items [107]. Some of these commands are general purpose and application-independent, e.g. selecting a target, copying a previously selected object or undoing an action; others are task- and sometimes software-specific. These items are often organized semantically into menus, sub-menus and groups. In single-user desktop environments, very large vocabularies of such commands can be used with pull-down menus and keyboard shortcuts. However in large display platforms, where several users may navigate physically in front of the display, new constraints apply that discard these solutions as discussed in Section 1.1.

The main problem I address in this chapter is menu breadth (the maximum number of items in a menu's hierarchical level) as opposed to menu depth (the maximum number of hierarchical levels). Snowberry et al. [156], Kiger [98], Landauer and Nachbar [108] and Jacko and Slavendy [91] have shown that breadth should be preferred over depth for learning and memorizing. Beaudouin-Lafon [26] observed the number of items in Bryce 2 and reports an average of 12.5 items in the first level and of 3.9 in sub-menus. Bailly [11] did the same observation on a more recent set of applications composed of Office Word 2003, Office Excel 2003, Adobe Reader 7, Mozilla Firefox 2.0, Mozilla Thunderbird 0.9 and Photoshop 7.0, and reports corresponding averages of 12.4 and 5.5 items. He also observed that 23.9% of menus contain more than 17 items. Linear or more recent menu techniques available on the desktop such as Flower Menus [13] can easily accommodate these numbers

[1] http://usa.autodesk.com/maya/

of items. The next section uses the *Feedback Location* design space introduced in Chapter 1 to analyze list of existing discrete command selection techniques (menus and lists) that were developed for large display platforms or that can be used in mid-air. I will pay particular attention to their theoretical or evaluated breadth and show that the *Main display* × *On-data* point of this design space is poorly addressed.

From this analysis I propose on-body touch as an eye-free, device-less input channel for high-breadth discrete selections or grouping. Formally, on-body touch is the action of touching one's body parts. It benefits from proprioception, i.e. the conscious knowledge of the location of one's body parts, which allows eye-free input: users can reach their body parts easily with high precision [135, 136]. On-body touch can also benefit from physical mnemonics, i.e. "ways to store/recall information relative to the body" [122]. As for tactile surfaces, on-body touch can be used for selecting specific locations, performing gestures or chords, or even indirect pointing provided proper sensing is available. In this chapter I will present a morphological design space for on-body touch interactions that I will use to illustrate the expressiveness of this input. I will then show how this space can be used to design discrete selection techniques. I will also report the results of an exploratory controlled experiment where body parts were evaluated in terms of performance, reachability, comfort and social acceptance.

3.2 Related Work

This section focuses on discrete selection techniques that can be used in mid-air and on on-body interaction.

Three families of discrete selection techniques are usable in mid-air: *Cursor-based* techniques, *Device-based* techniques and techniques that exploit *Unused input channels*. I will describe these families using the *Feedback Location* design space from Chapter 1 and illustrate them with corresponding related work (see Table 3.1). A particular case of this design space, *Main display* × *Fixed*, will not be considered in this chapter since it violates the "**Location independence**" requirement (R2) when applied to discrete selection techniques: a widget fixed on the main display can no longer be reached or even be seen if a user stands near the display but far from this widget. Techniques such as Frisbee [97], the Vacuum [29] or Drag-and-Pop [22] all make remote targets accessible so they can be interacted with. However these techniques were all designed for direct touch input on the main display, but on a very large display perspective distortion can make remote targets hard or impossible to read from a distance, making these techniques much less usable.

3.2.1 Cursor-based techniques

Cursor-based techniques use the pointing cursor to acquire menu items that are displayed on the *Main display* and over the visualized data (*On-data*). They provide contextual and visually close access to commands and thus cause minimal attention switch[2]. Since their feedback is unconstrained (*On-data*) on the *Main display*, they are not strongly limited by the size of the display; they can thus contain high numbers of items. However as discussed by [56] and following Requirement 2 (**Location independence**), items must be selectable regardless of the user's distance to the display. This requires either distance-dependent item sizes, which can cause more occlusion as users go far away from the display, or harder selection tasks.

In desktop environments, the main types of cursor-based menus are linear and radial menus [39], the latter having been improved with marking menus [106, 179]. These designs have been evaluated in mid-air on large displays as control conditions by Bailly et al. [15] (8 items in each hand) and Chertoff et al. [50] (9 items, one-handed), showing good performance against newer designs. Chertoff et al. [50], in particular, showed that radial menus were faster and most preferred. Cursor-based menus in 2D were also evaluated as an Immersive Virtual Reality (IVR) solution in 3D environments. The *No Haptics* × *World-Fixed* combination of the design space presented by Lindeman et al. [112] corresponds to a classic 2D linear menu adapted to 3D. It proved slower and generally less precise than the other menu designs evaluated in this article. Floating menus [36] are another 3D version of pull-down menus where the top-level menus are always visible in the user's field of view. Pull-down menus and item selection are triggered by "occlusion selection". Floating menus were proven efficient in IVR (max 8 items in a hand), yet less so than the new technique introduced in this article, TULIP, which I describe further in a subsequent section (Section 3.2.3 page 74). Finally, Wesche [171] introduced a full 3D, intersection-based selection + menu technique. A pencil-shaped 3D cursor with up to 8 distinct segments along the pen is used to select both the action (menu item) and its target object: the user manipulates the cursor to make the object and action intersect in 3D and then activates the item by clicking a side-button.

3.2.2 Device-based techniques

Device-based techniques integrate the set of items into input devices such as game controllers or *Secondary displays* such as tablets and smartphones. Some of these devices, e.g. tablets, can accommodate a large number of items and still keep input space available

[2] Apart from the attention needed to find an item in a very large list or tree, see [151].

for other interactions, as illustrated by Rekimoto [145]. However these techniques often require both hands to hold and interact with the device: even smartphones lose significant expressiveness when interacted with single-handedly [147]. Repetitive interaction with a *Secondary display* can also cause problems due to attention switch (see Section 1.2). The *Haptics* × *Hand-Held* condition evaluated by Lindeman et al. [112] uses a handheld physical board to embody an interactive command surface in IVR. It proved faster and more precise than other conditions.

Touch-based menu techniques designed for handheld devices can often be used in a large display platform. Because of the limited screen real estate of such devices, most device-based menus were designed as *On-data* menus. However they can also be used without data below. In this case they can be either *Fixed* or *On-data* if the device also contains part of the visualized dataset. Linear and 2D matrix menus are already widely featured in most mobile operating systems: the latest Android and iOS smartphones display 20+ items on their home-screens. Circular designs were presented by Huot and Lecolinet [87] and improved by [65] with up to 6 items per hierarchical level. Spiral designs [86] allow virtually any number of items but need constant visual attention, which precludes eyes-free expert mode. Barrel Menus [64] display menu items on three hierarchical levels of virtually any breadth, but can only display five items at a time at each level. Leaf menus [14] use stroke curvature to invoke menu items. The technique can differentiate up to 7 curvatures, and a selection gesture can start from any of the four corners of the device for increased expressiveness.

3.2.3 Unused input channels

Unused input channels such as pressure or unused fingers can be used to augment the vocabulary of a discrete selection technique and sometimes make it combinable with other interactions, as discussed in Section 1.3. Being mostly movement-based, they do not need an input device and their feedback is often on the *Main display*, similarly to *Cursor-based* techniques. An exception to that is the VisionWand [41], a 3D-tracked input device that combines ray-casting for pointing and other motion channels such as proximity to the display, orientation and back-and-forth gestures for simultaneous, possibly discrete interaction. These techniques are usually designed to provide eye-free interactions that can be performed in parallel with other ones in some conditions. However we will see that these input channels are often limited in their expressiveness, even when used in parallel, resulting in smaller breadths.

BiPad [167] takes advantage of the few remaining degrees of freedom of the non-dominant hand when holding a tablet, using the fingertips showing past the tablet. It can be used

in parallel with touch-based interaction using the dominant hand; examples of use include invoking discrete commands. However the movement range of the non-dominant fingers is small: only three items were selectable in the evaluation. [175] showed that users can input up to 10 levels of pressure on a handheld device when given feedback. TULIP [36] is an IVR menu system using finger pinch: the selected item depends on which finger is reached by the thumb; for menu structures with more than four items, the pinky finger's function is to display the next three or four items. In a similar vein, Finger-Count Menu [15] uses visual recognition to count the number of fingers showed by users to trigger commands; each hand can thus trigger up to five items. With Rotary menus [50] users rotate their wrist to trigger commands. The technique was evaluated successfully with 9 items.

		Physical location	
		Main Display	Secondary Display
Virtual location	On-data	C(N): Linear menus [15, 50, 112, 36] C(9): Radial menus [50] C(8): Marking menus [15] C(8): ToolFinger [171] U(5): Finger-Count Menu [15] U(9): Rotary menu [50]	D(6): ThumbMenu, ArchMenu [87] D(6): Wavelet menu [65] D(N): SpiraList [86] D(5): Barrel menu [64] D(7): Leaf menu [14] U(10): Pressure-based menus[175] (All these designs can be used as locally *Fixed* on the secondary display)
	Fixed	(Impractical in a large display platform)	D(N): "Palettes" [145] D(8): IVR tablet [112] U(3): BiPad [167] U(4): TULIP[3] [36]

Table 3.1: Menu techniques available for mid-air interaction according to the *Feedback Location* design space. The first letter stands for the type of technique (C-Cursor, D-Device, U-Unused); the number in parentheses is the highest breadth of the technique, either theoretical or evaluated. Techniques are listed in the *On-data* or *Fixed* row depending on how they were primarily designed.

As shown in Table 3.1, the only technique that supports more than 10 items available at once on the *Main display* is linear or matrix-shaped menus. However, as discussed in Sections 1.2 and 1.4, visual items that are selected by pointing need to be wide enough to be seen and acquired easily, which can lead to important occlusion of the visualized data.

[3]Being designed for IVRs, TULIP does not exactly fit into *Main* or *Secondary display*. However users have to look at their hands in novice mode, similar to using a tablet, so I chose to put it in the *Secondary display* column in Table 3.1.

3.2.4 On-body interaction

On-body interaction has three advantages for mid-air interaction: kinesthetic memory, haptic perception and proprioception.

Kinesthetic memory is the recollection of movement, weight, resistance, and position of parts of the body. It can be used to associate functions or virtual objects to specific body parts for quick recollection.

Haptic perception is the process of recognizing objects through touch.

Proprioception is the real-time knowledge of the location and posture of body parts using muscular and haptic feedbacks. It can be used to design eyes-free interactions.

Gustafson et al. [76] found that users retain spatial memory of the location of icons they habitually select on a mobile touch screen. They can transfer this knowledge to accurately select the associated locations of these icons on the palms of their hands. Angesleva [3] found that certain body locations have special significance and are easier to remember, specifically hip pockets, the stomach, the head and the heart.

I am interested in specific types of mid-air interaction that identify the user's gestures with respect to their *own* body. For example, Harrison et al. [78] use infra-red and depth cameras mounted on the user's shoulder to track arm and hand positions. *Skinput* [80] uses bio-acoustic fingerprinting to detect touches on the forearm. Lin et al. [111] use an ultrasonic sensor mounted on the user's wrist to track up to seven different input locations. They found that haptic feedback increases the accuracy of on-body touches and that users can discriminate among six distinct locations on the forearm. *PinStripe* [96] detects pinching and rolling gestures with clothing made of smart fabrics. All of these techniques require the user to wear specific clothing or hardware and are often limited to specific body parts.

Like the early VIDEOPLACE [105], Shoemaker et al. [154] extract the user's silhouette to provide visual feedback about the user's body position. Users can select menu items by pointing a Wii remote controller at specific parts of the body, such as the torso, without touching them. The authors argue that the associated proprioception enhances performance, with important implications for the design of *body-centric* interfaces to large displays. However, proprioceptive and haptic senses work together and Voisin et al. [166] show that removing one or the other may degrade performance.

Touching one's body parts can provide a very rich vocabulary. Ethnological studies [135, 136] showed that body parts-based tally systems could use up to 74 different body positions. These systems are used eye-free to communicate numbers between people and

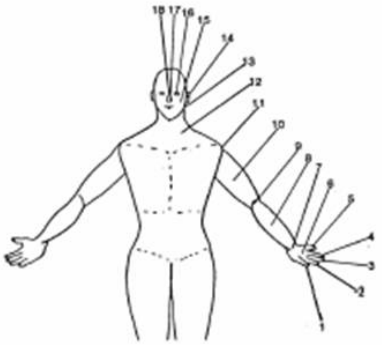


Figure 3.1: Body-part tally system of the Fasu, Southern Highlands Province. The numerals go from the pinky finger to the nose (central number) and continue increasing symmetrically to the opposite pinky finger, for a total of 36 numbers. From [136].

show how efficient proprioception can be when trained. Since they already know the locations of these targets, they can attend to the display without being forced to shift their attention to a hand-held device. Users can thus treat their own bodies as a sort of keyboard, pressing known spots to produce different results. When combined with pointing, users can achieve the equivalent of "control" keys that modify other commands.

3.3 Design space for on-body touch interactions

This section explores the action of touching body parts to trigger commands in large display environments. Such actions can be performed while sitting, standing or walking and do not require additional input devices. I study these commands as secondary actions, similar to right-clicking to invoke a menu in the context of the hovered item. In the context of a user standing in front of a wall-sized display and performing a specific task with (at least) her dominant hand, invoking a command has to be performed with the non-dominant hand. In the following the touching hand (i.e., the non-dominant hand) is called the *body pointer* and the touched body part is the *body target*.

One of the potential drawbacks of interacting by touching one's body is the "immersion syndrome" described by Baudel et al.[21]: some natural user actions such as scratching can be perceived as commands by the system. The vocabulary of systems using on-body touch should be designed so that the recognizer can accurately discriminate between command gestures and natural actions.

I propose the following morphological factors to characterize on-body touch:

Target locations can have an effect on performance and comfort. We expect the time to

reach a body target to depend on the physical distance between the initial position of the body pointer and the target. However, this effect could be reduced if users move both the body target and the body pointer to make the reaching faster. Body targets located on lower limbs can be harder to reach: users need to either bend their body or lift their leg, which requires suppleness and balance. Other body targets such as the crotch, chest, face and so on should probably be discarded for social awkwardness. Finally, users' movements may disable some body locations at a given time, e.g. walking probably discards all body targets located on the legs. Owens [135, 136] describes body part tally systems with up to 74 locations that use the whole body, including feet and toes. The most common number of numerals they observed is 27, which are always performed without using body parts below the navel (see Figure 3.1 for a 36-target example). One can also expect that touching one's limbs can achieve higher precision if the user does not rely on proprioception only but is also provided with real-time visual feedback. An on-body touch can also include more than one target location, resulting in a *gesture* rather than a touch. As shown in the previous chapter, one could expect that providing physical support (here the body) to perform gestures could improve their precision and stability.

Touch dynamics represents the temporal component of a given touch. A *simple touch* is a tap on a single position on the body. The only input channel is the location of the release event. A *dwell* happens on a single position but uses the duration of the touch as an additional input, e.g. for validating a critical command or specifying a continuous value. Dwell usually increases the input vocabulary by a factor 2: any tap event can have a corresponding dwell event. A *rhythm* is a temporal sequence of simple touches where additional input is given by the rhythm of the taps. Ghomi et al. [70] showed that users can accurately input 30 different rhythmic patterns and memorize up to 14.

Contact represents the parts of the body pointer that are used to perform the touch. In the default condition, no importance is given to the contact type: it can be one or several fingers, the palm, etc. The *number of fingers* [15] that perform the touch can be combined with the body target location to convey additional information and multiply a touch's expressiveness by 6 (including the palm). Similarly, the finger(s) that are used can be assigned different roles: different commands can be attached to, e.g., the forefinger and the middle finger, and even the palm. These roles can also be combined: the expressiveness of a touch event can be increased by a factor of 6 with only one command per finger and palm, by $\binom{6}{2} + 6 = 21$ if commands are assigned combinations of two fingers, and up to 63 if all combinations of fingers + palm are used. Users can perform *chords* [20] i.e. a spatial arrangement of finger

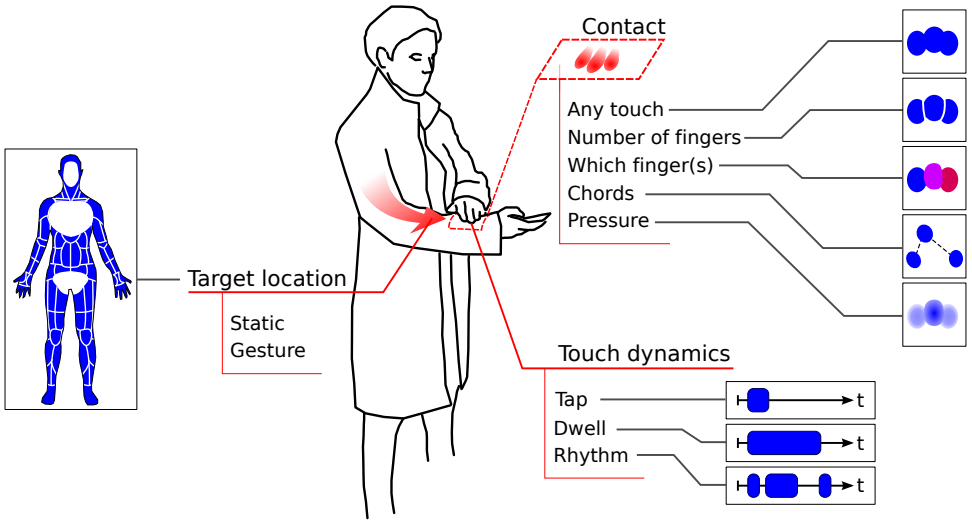


Figure 3.2: Morphological design space of on-body touch interactions.

contacts. Finally, pressure can be used as a discrete or continuous additional input channel with appropriate hardware, e.g. pressure-sensitive gloves [162], increasing the expressiveness of the touch by a factor of 10 [175] with feedback and by a factor of 4 [142] without.

These factors, illustrated in Figure 3.2, can of course be combined to provide high-breadth selection techniques. As an example, if we use the tally system described in Figure 3.1, considering that a user cannot touch targets located on his *body pointer* arm and discarding the targets on the face, we can expect a theoretical 14 on-body targets. Considering only non-temporal events (i.e. no dwell nor rhythm nor gesture) and without identifying individual fingers, we can obtain a theoretical 14 (body targets) $\times$ 6 (number of fingers + palm) = 84 items in a tap-based on-body touch discrete selection technique. Using rhythms, chords or differentiated fingers allow for an even higher number of items.

Other, non discrete combinations of touch dynamics and contact types are possible. For example, the abdomen can become a wide touch area for rhythmic patterns, gestures or chords. Other body parts could be used for the same input purpose with different semantics, e.g. the abdomen can be dedicated to manipulation commands while the back of the hand can be used for navigation commands. Body parts can also be used for item groups, i.e. semantic groupings of items within a menu's hierarchical level [13].

The expressiveness of these factors are of course limited by the available sensing technology available. Infra-red camera systems such as VICON's (see Appendix A) provide tracking with very high spatial and temporal resolution but, being camera-based, are dependent on occlusion, either from a user's collaborators or her own limbs. Kinect-based systems have

been shown to provide less resolution and are less accurate, but cost much less. Camera-based systems combined with interactive gloves [169, 162] can be used to detect which fingers are used for the touch, chords and pressure. Finally, one could imagine sensitive clothes that accurately detect and locate touch.

The combinations of factors evoked in this section have advantages beyond menu breadth. *Body locations* can be used to group similar targets; symmetric limbs or locations can hold targets with opposite effects; critical commands can be set on locations harder to reach or less likely to be touched by accident. Designers can also assign functions to semantically related limbs [72], e.g. "Save" and "Open" on the head to symbolize memory. Such associations could improve memorization with large command sets. *Contact type* can help differentiate commands from "natural" gestures such as scratching, e.g. by only considering touches performed with more (or less) than three fingers. It can also trigger related commands from the same body location, e.g. "Save" with one finger on the temple and "Save as" with two fingers, again possibly easing memorization. *Touch dynamics* can also differentiate intended commands, e.g. by tapping twice or dwelling, and make continuous gestures possible along limbs.

Menus or lists as large as those evoked above (84 items) are obviously not meant to be memorized entirely. The feedback of touch-based menus thus needs to minimize occlusion (Requirement 4: **Collaboration**) while remaining clear, such as a proxy of the user's body that would appear when she brings her non-dominant hand close to an "active" limb. Items could highlight as the hand hovers different body parts. An even smaller feedback could consist in using the non-dominant hand as a hovering scanner: a fixed-sized window appears on the *Main display* and displays the body part facing the user's palm with the corresponding commands. The user could then zoom in and out by moving her hand nearer or further away from her body to explore the set of available commands. In both cases the visual feedback would be triggered by a short dwell time while hovering close to the body, similar to Marking menus [179]. "Expert", feedback-less selection would then happen when the user does not wait for this timeout.

Most factors of this design space have already been evaluated in previous work, although often in a different context: pressure [175, 142] in desktop and mobile menu systems, rhythm [70] with tactile devices, number of fingers [15] and their nature [36], both in mid-air. However, what seems to be the most expressive factor, target location, has not. Together with Julie Wagner, Sean Gustavson, Stéphane Huot and Wendy Mackay we designed and ran a controlled experiment to evaluate which body parts are efficient and acceptable to use as input and how a simple static on-body tap interaction combine and interferes with mid-air pointing.

3.4 Experiment[4]

This section reports on an experiment designed to study how on-body touch can be performed on the whole body, alone (Body only) or in combination with mid-air pointing (Pointing+Body). Although pointing has been well-studied in the literature (e.g. [119, 163]), we know little of the performance and acceptability trade-offs involved in touching one's own body to control a multi-surface environment. Because it is indirect, we are particularly interested in on-body touch for secondary tasks such as confirming a selection, triggering an action on a specified object, or changing the scope or interpretation of a gesture.

Participants were asked to touch specific locations on their own body (On-Body Touch conditions), sometimes after having performed a pointing task on the wall display, i.e. moving a cursor inside a circular target (Pointing conditions). In this experiment we investigate two questions:

Q1. *Which on-body targets are most efficient and acceptable?* Users can take advantage of proprioception when touching their own bodies, which enables eyes-free interaction and suggests higher performance. However, body targets differ both in the level of motor control required to reach them, e.g., touching a foot requires more balance than touching a shoulder, and in their social acceptability, e.g., touching below the waist [96].

Q2. *What performance trade-offs arise with compound body-centric interaction techniques?* Users must position themselves relative to a target displayed on the wall and stabilize the body to point effectively. Simultaneously selecting on-body targets that force shifts in balance or awkward movements may degrade pointing performance. In addition, smaller targets will decrease pointing performance, but may also decrease on-body touch performance.

3.4.1 Method

3.4.1.1 Participants

We recruited sixteen unpaid right-handed volunteers (13 men, average age 28); five had previous experience using a wall-sized display. All had normal balance and wore comfortable, non-restrictive clothing.

[4] An article featuring a subset of this experiment section has been accepted at the CHI'13 conference (not yet in the proceedings).

(a) (b)

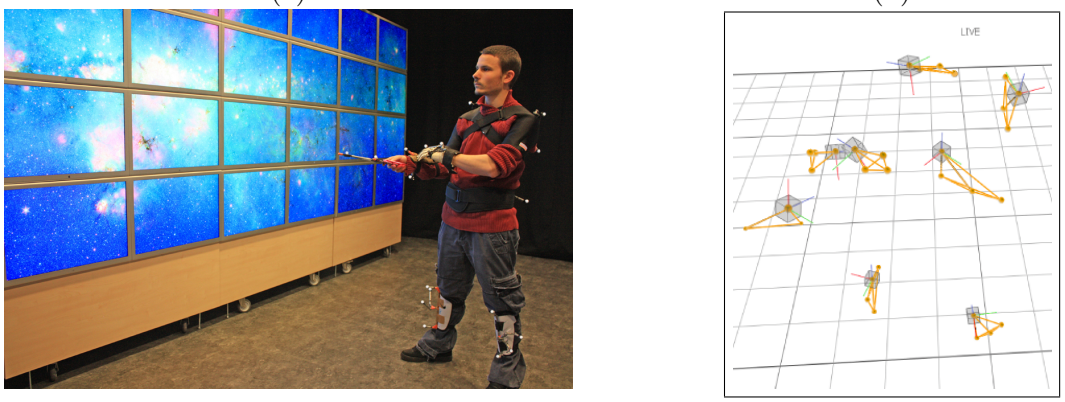


Figure 3.3: Apparatus of the experiment: passive infrared markers were mounted on several sports gear (a) to track the position of each body part of the participant with high precision (b).

3.4.1.2 Apparatus

Participants stood in front of the WILD wall-sized display (Appendix A). Participants' postures were tracked by the VICON system through a wireless mouse held in the user's dominant hand[5] for ray-casting, on the index finger of the non-dominant hand for on-body touches, and on protective sports gear – belt, forearms, shoulders and legs – to track on-body targets. The latter were adjustable to fit over the participants' clothing, as shown in Figure 3.3.

Based on pilot studies, we defined 18 body target locations distributed across the body (Fig. 3.4), ranging in size from 9 cm on the forearm to 16 cm on the lower limbs, depending upon location and density of nearby targets, grouped as follows:

Dominant Arm (D ARM) 4 targets: upper arm, elbow, forearm, wrist

Dominant Upper Body (D UPPER) 4 targets: thigh, hip, torso, shoulder

Non-dominant Upper Body (ND UPPER) 4 targets: thigh, hip, torso, shoulder

Dominant Lower Leg (D LOWER) 3 targets: knee, tibia, foot

Non-dominant Lower Leg (ND LOWER) 3 targets: knee, tibia, foot

In ON-BODY TOUCH conditions, participants wore an infrared-tracked glove on the non-dominant hand with a pressure sensor in the index finger. When a threshold pressure level was detected, the system computed an orthogonal projection from the index finger to the touched limb segment using a skeleton-based model (Figure 3.3-b) to calculate the body target closest to the index finger.

[5] All subjects were right-handed, so "dominant" refers to the right hand or side.

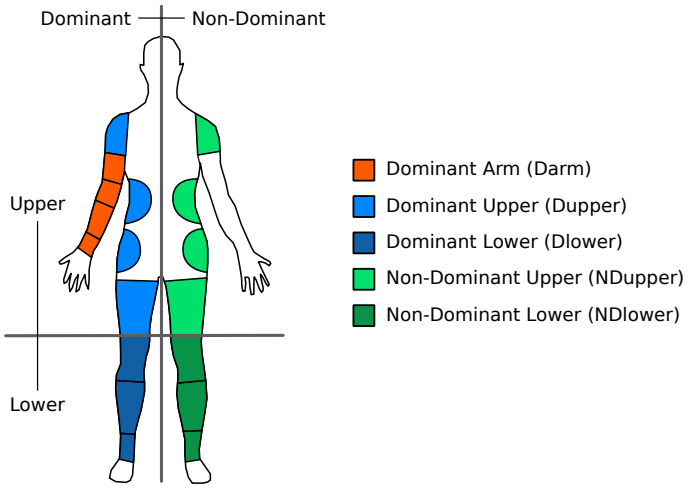


Figure 3.4: 18 body targets are grouped into five categories.

Wall pointing tasks varied in difficulty from *easy* (diameter of the circular target was 1200 px or 30 cm) to *medium* (850 px or 21.25 cm) to *hard* (500 px or 12.5 cm). Wall targets were randomly placed 4700 px (117.5 cm) from the starting target.

3.4.1.3 Measures

We collected timing (Figure 3.5) and error data for each trial, as follows:

Trial Time**:**
: From trial start to completion.

Pointing reaction time**:**
: From trial start to "initial" cursor displacement of more than 1000 px.

Pointing movement time**:**
: From initial cursor movement to entry into goal target.

Cursor readjustment time**:**
: From leaving goal target to final reentering goal target.

Body reaction time**:**
: From appearance of trial stimulus to leaving starting position.

Body pointing time**:**
: From leaving start position to touching on-body target.

Body errors**:**

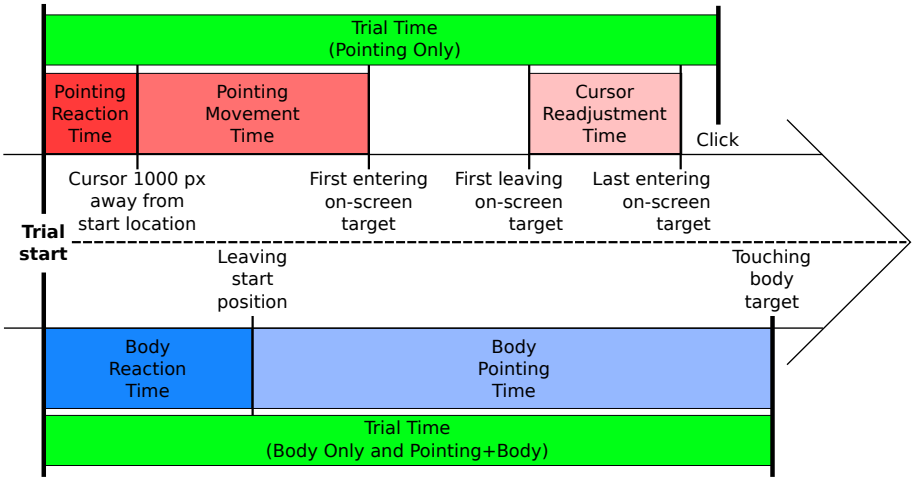


Figure 3.5: Timeline of the three main conditions: POINTING ONLY (top), BODY ONLY (bottom) and POINTING+BODY (both). The "Click" event at the end of the POINTING ONLY timeline only occurs in the POINTING ONLY condition. Every other POINTING ONLY events also occur in POINTING+BODY.

Number of incorrect touches detected on body target[6]; includes list of incorrect targets per error.

We debriefed participants at the end of the experiment and asked them to rank, on a Likert scale: (i) perceived comfort of each body target according to each POINTING condition ('1=very uncomfortable' to '5=very comfortable'); and (ii) social acceptability of each on-body target:"Would you agree to touch this body target in a work environment with colleagues in the same room?" ('1=never' to '5=certainly').

3.4.2 Procedure

Each session lasted about 60 minutes, starting with a training session, followed by blocks of trials of the following conditions, counter-balanced across subjects using a Latin square.

BODY ONLY**:** Non-dominant hand touches one of 18 on-body targets (atomic technique − 18×5 replications = 90 trials)

POINTING ONLY**:** Dominant hand points to one of three target sizes (atomic technique − 3×5 replications = 15 trials)

POINTING+BODY**:** Combines touching an on-body target with selecting a wall target (compound technique − (18×3)×5 replications = 270 trials)

[6]Includes both system detection and user errors.

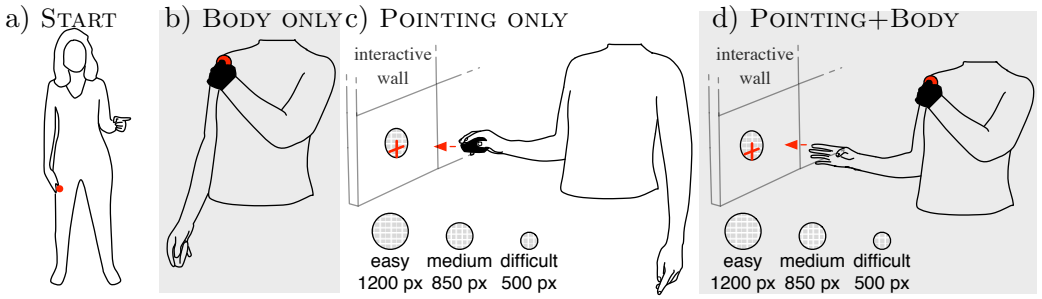


Figure 3.6: a) Starting position b) Body only c) Pointing only d) Pointing+Body Starting position: non-dominant hand at the hip and/or dominant hand points to a starting target on the wall display.
Body only and Pointing only are atomic conditions; Pointing+Body is compound: a body touch triggers the selected wall target.

Participants were thus exposed to 75 unique conditions, each replicated five times, for a total of 375 trials. Body only and Pointing+Body trials were organized into blocks of six, with the location of body targets randomized. No two successive trials involved the same body target group. Pointing only trials were organized into blocks of five trials and all wall pointing trials were counterbalanced across difficulty. The two atomic interaction techniques, Body only and Pointing only serve as baseline comparing performance with the compound interaction technique, Pointing+Body. Participants were instructed to perform trials as quickly and accurately as possible.

Body only – (Fig. 3.6b): The starting position involves standing comfortably facing the wall display, with the non-dominant hand at the thigh (Fig. 3.6a). The trial begins when a body-target illustration appears on the wall. The participant touches that target with the index finger of the non-dominant hand as quickly and accurately as possible. Participants were asked to avoid crouching or bending their bodies, which forced them to lift their legs to reach lower-leg targets. The trial ends only when the participant successfully selects the correct target; all intermediate incorrect selections are logged.

Pointing only – (Fig. 3.6c): The starting position involves standing comfortably facing the wall display and using the dominant hand to locate a cursor within a circular target displayed in the center of the wall. The trial begins when the starting target disappears and the goal target appears between $0.5s$ and $1s$ later, to reduce anticipatory movements and learning effects. The participant moves the dominant hand to move the cursor to the goal target and selects by pressing the left button of the mouse bearing the optical marker used for pointing. The trial ends only when the participant successfully clicks the mouse button while the cursor is inside the goal target.

POINTING+BODY – (Fig. 3.6d): The starting position combines the above, with the non-dominant hand at the thigh and the dominant hand pointing to the starting target on the wall. The trial begins with the appearance of a body-target illustration and the goal target on the wall display. The participant first points the cursor at the goal target, then completes the trial by touching the designated on-body target. The trial ends only when the on-body touch occurs while the cursor is inside the goal target on the wall.

3.4.2.1 Training

Participants began by calibrating the system to their bodies, visually locating, touching and verifying each of the 18 body targets. They were then exposed to three blocks of six BODY ONLY trials, with the requirement that they performed two on-body touches in less than five seconds. They continued with three additional blocks to ensure they could accurately touch each of the targets. Next, they were exposed to all three levels of difficulty for the POINTING ONLY condition: easy, medium and hard, in a single block. Finally, they performed three additional blocks of the compound POINTING+BODY technique.

3.4.3 Results

We conducted full factorial ANOVAs with PARTICIPANT as a random variable using the standard repeated measures REML technique from the JMP 9 statistical package. We found no fatigue or learning effects.

3.4.3.1 On-body touch only

We found a main effect of BODY TARGET GROUP on TRIAL TIME ($F_{4,60} = 21.20$, $p < 0.0001$). A post-hoc Tukey test revealed two significantly different groups: body targets located on the upper torso require less than 1400 ms to be touched whereas targets on the dominant arm and on the lower body parts require more than 1600 ms (Figure 3.7).

We found a significant effect of BODY TARGET on TRIAL TIME, since body targets located on the lower parts of the body require more time to be touched. Distributions in Figure 3.7[7] also shows that TRIAL TIME is consistent for targets belonging to the same group. In fact, grouping by BODY TARGET GROUP reveals a significant effect of BODY TARGET on TRIAL TIME only for the D UPPER ($F_{3,45} = 5.33$, $p = 0.0031$) and ND UPPER groups ($F_{3,45} = 3.4$,

[7]Error bars in all figures represent the 95 % confidence limit of the mean of the medians per participants ($\pm$StdErr $\times$ 1.96).

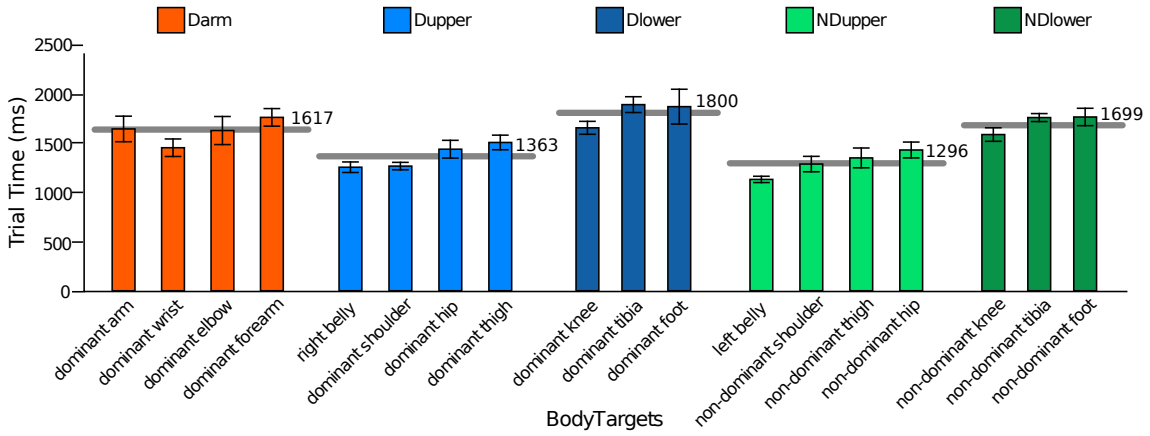


Figure 3.7: Mean TRIAL TIME is lower for body targets on the upper torso (D UPPER and ND UPPER) than for other targets and consistent across body targets. Horizontal lines indicate group means.

$p = 0.0257$), showing that the target on the dominant thigh takes more time to be touched than the ones on the dominant shoulder and torso and that the target on the non-dominant hip takes more time to be touched than the one on the non-dominant side of the torso.

We found 7.6 % of trials with at least one error in the BODY ONLY condition. We identified two main types of erroneous touches:

Vicinity (94.5 %) The participant's hand touched the frontier between the goal target and a nearby target.

Dominant arm position (5.5 %) The participant kept the dominant arm close to her torso, making it difficult for the system to distinguish between the torso and arm targets.

We found a main effect of BODY TARGET on error rate $F_{17,255} = 2.32$, $p = 0.0027$, but a post-hoc Tukey test shows no difference among BODY TARGETS. We also found a main effect of BODY TARGET GROUP on error rate ($F_{4,60} = 6.60$, $p = 0.0002$). A post-hoc Tukey test shows that targets on the dominant arm are more error prone than those in other groups (14.8 % against about 6 % for dominant and non-dominant upper, and 2.9 % for non-dominant lower). Similar to TRIAL TIME, we tested the consistency of error rate within BODY TARGET GROUP and found a significant difference only in the D UPPER group ($F_{3,45} = 4.42$, $p = 0.0083$): the dominant hip caused more errors than the dominant shoulder.

These results partially answer **Q1** about the performance of ON-BODY TOUCH: participants were able to quickly acquire targets on their body with an acceptable accuracy of 92.4 % at first try. As expected, performance degrades for the lower body parts. This is not surprising since these targets are further away from the default body position and require

more complex movements to be touched. However, the difference is small, about only 300 ms.

3.4.3.2 Combined task

Pointing only – We first analyze the POINTING ONLY results to have a baseline for how on-body touch affects pointing performance. We found a main effect of pointing difficulty on TRIAL TIME ($F_{2,30} = 40.23$, $p < 0.0001$). A post-hoc Tukey test shows that only difficult pointing tasks are significantly slower (1545 ms) than medium (1216 ms) and easy (1170 ms) pointing tasks (see Figure 3.8). Only 1.8 % of the trials required to relocate the cursor inside the target before validating the selection. This occurred mostly during difficult pointing tasks: 15 % readjustments against 1.2 % for easy and medium difficulties. Consequently, for CURSOR READJUSTMENT TIME, the ANOVA shows an effect of the difficulty ($F_{2,30} = 8.02$, $p = 0.0016$), CURSOR READJUSTMENT TIME being significantly higher for difficult tasks (77.3 ms) than for medium (3.8 ms) and easy (2.2 ms) ones.

Trial time – The ANOVA with the model POINTING[easy / medium / difficult] ×BODY TARGET GROUP shows a significant effect of POINTING difficulty on TRIAL TIME ($F_{2,30} = 48.51$, $p < 0.0001$), with the same significant difference than the POINTING ONLY baseline: TRIAL TIME is significantly slower for difficult POINTING (2545 ms) than both medium (1997 ms) and easy (1905 ms) (see Fig. 3.8). BODY TARGET GROUP also has an effect on TRIAL TIME ($F_{4,60} = 34.10$, $p < 0.0001$) showing the same significant groups as in BODY ONLY: TRIAL TIME is significantly faster when touching body targets on ND UPPER (1794 ms) and D UPPER (1914 ms) than on ND LOWER (2267 ms), D LOWER (2368 ms), and D ARM (2401 ms).

3.4.3.3 The impact of ON-BODY TOUCH on POINTING

Pointing time – We performed an ANOVA with the model POINTING[easy/medium/difficult] × ON-BODY TOUCH[none/BODY TARGET GROUP]. For POINTING MOVEMENT TIME, the ANOVA reveals significant effects of POINTING ($F_{2,30} = 100.53$, $p < 0.0001$) and ON-BODY TOUCH ($F_{5,75} = 17.22$, $p < 0.0001$), and a significant POINTING × ON-BODY TOUCH interaction ($F_{10,150} = 3.9$, $p < 0.0001$). A post-hoc Tukey test shows that POINTING MOVEMENT TIME is significantly higher for difficult pointing tasks (830 ms) than for medium (556 ms) and easy (496 ms) ones. For BODY TARGET GROUP, POINTING MOVEMENT TIME is significantly higher for D LOWER and ND LOWER (707 ms) than for all other groups of targets and than POINTING ONLY (551 ms). POINTING MOVEMENT TIME is also significantly higher for D ARM

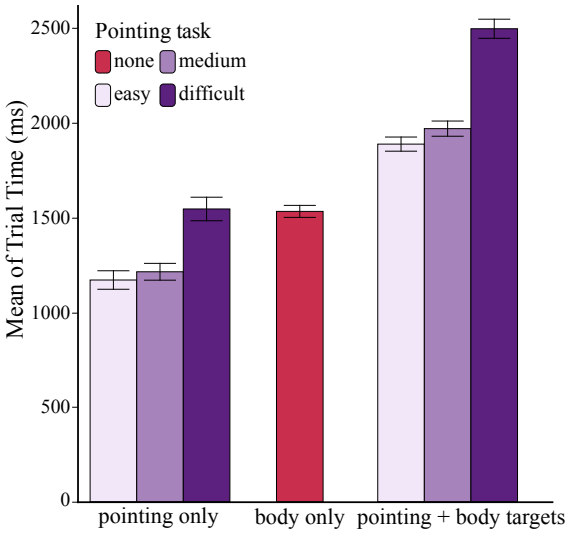


Figure 3.8: TRIAL TIME for all tasks (pointing only, body only, combined pointing + body) by pointing difficulty.

(634 ms) than for the POINTING ONLY baseline, and than D UPPER and ND UPPER (580 ms). Interestingly, these last groups are not significantly different than the body-only condition.

These results suggest that touching targets on the D UPPER and ND UPPER groups does not affect POINTING MOVEMENT TIME as much as targets on the pointing arm or on the legs. When associated with the effect of POINTING difficulty we can explain the interaction effect: while only D LOWER and ND LOWER have a strong negative effect on POINTING MOVEMENT TIME with the easiest difficulty, the other groups also have a negative impact when the pointing difficulty increases, especially for targets located on the D ARM (see Figure3.9).

Cursor readjustments – The impact of ON-BODY TOUCH on the POINTING task does not only impact the movement phase but also the cursor readjustments. For the combined POINTING+BODY task, 31 % of the trials required the participants to relocate the cursor inside of the target before validating the selection with a body touch, compared to only 6 % for POINTING ONLY. Thus, we found significant effects of POINTING ($F_{2,30} = 59.64$, $p < 0.0001$), BODY TARGET GROUP ($F_{5,75} = 23.03$, $p < 0.0001$) and POINTING×BODY TARGET GROUP ($F_{10,150} = 8.45, p < 0.0001$) on CURSOR READJUSTMENT TIME. As shown in Figure 3.10, CURSOR READJUSTMENT TIME increases significantly for each level of difficulty of POINTING but selecting body targets on some BODY TARGET GROUP, especially in D LOWER and D ARM, affects the body configuration and requires even more time to relocate the cursor inside of the on-screen target. We draw two interpretations from these results and our own observations during the experiment:

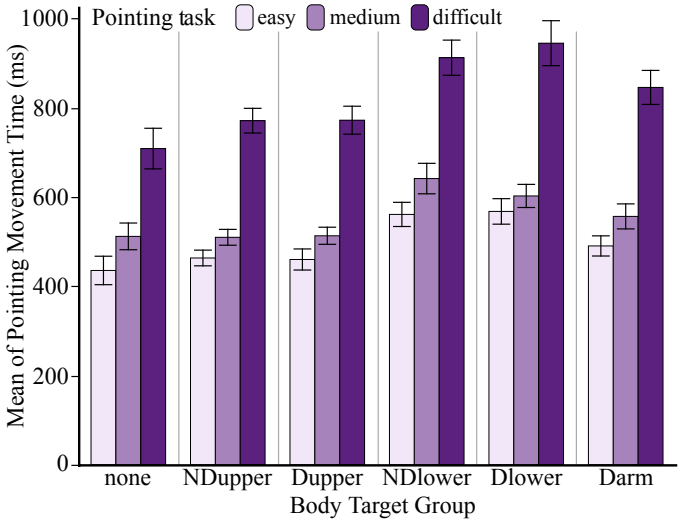


Figure 3.9: Interaction Pointing×On-Body Touch on Pointing movement time.

1. Touching the dominant arm while pointing affects the precision of pointing. Even though users know that a force will be applied to their pointing arm, the cursor still twitches. Users have to use "force-balance", i.e. resist to this force, but sometimes they could not prevent the cursor from leaving the target. This relates to the touch detection method we used, that required a certain pressure to be applied to the on-body targets.

2. Touching targets on the lower body parts affects the precision of pointing as well. Users were instructed to keep their back straight so they had to lift a leg, which affected their balance and thus their whole body posture, ultimately hindering their pointing precision. Touching lower limbs requires "movement-balance".

Overall, since the impact of both D lower and D arm is similar, we observe that maintaining force-balance is as difficult as maintaining movement-balance during the pointing task.

In summary, these results confirm and explain our hypothesis about the degradation of performance of Pointing while selecting on-body targets (**Q2**). Besides the expected overall time penalty of performing two tasks simultaneously (about 1 s in the worst case), the On-Body Touch task significantly impacts performance of the whole pointing task, and not only the Pointing movement times. Obviously, this drop in performance mostly affects the last step of the task, the selection phase, because the task involved body target selection *after* the pointing phase. However, our results also reveal that On-Body Touch on the lower parts of the body significantly impairs the movement phase of pointing, and that the overall negative impact increases with the difficulty of the pointing task, especially

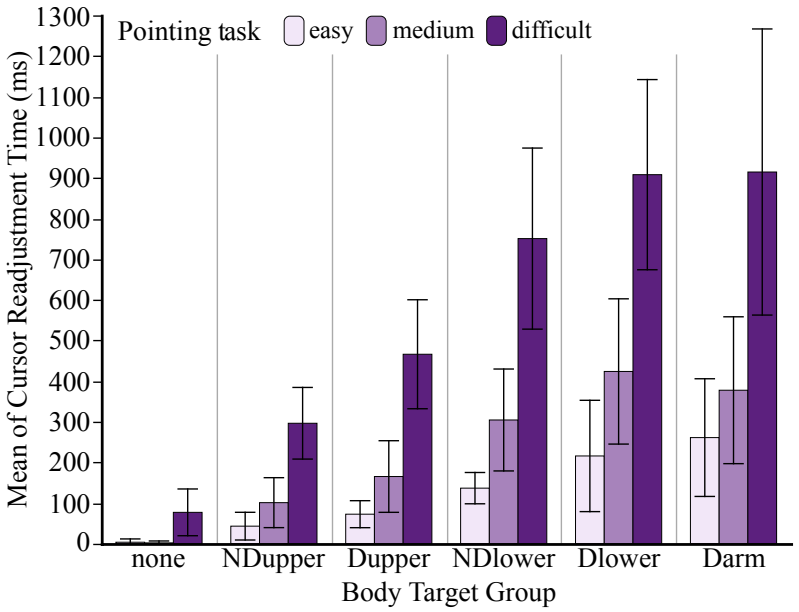


Figure 3.10: Effect of Pointing difficulty and Body target group on Cursor readjustment time.

when selecting a target on the pointing arm.

3.4.3.4 The impact of Pointing on On-Body Touch

Similarly, we studied the effect of Pointing on On-Body Touch by performing an ANOVA with the model Pointing[none/easy/medium/difficult]×Body target group.

Body pointing time – We found a significant effect of Body target group ($F_{4,60} = 38.69$, $p < 0.0001$), of Pointing ($F_{3,45} = 78.15$, $p < 0.0001$) and a significant Pointing×Body target group interaction ($F_{12,180} = 2.28$, $p = 0.01$) on Body pointing time. The main effect of Body target group is similar to the Body only condition: ND upper and D upper significantly faster than all other groups. The main effect of Pointing is also similar to those observed before, showing that difficult pointing tasks make simultaneous body touching slower than medium or easy pointing tasks. Obviously, all Body pointing time are significantly slower than in the Body only condition.

More interestingly, the Pointing × Body target group interaction effect reveals the actual impact of Pointing on On-Body Touch. As shown in Figure 3.11:

1. Body pointing time increases with the difficulty of the pointing task. In fact, despite the fact that our task required body target selection to be the last action, the reaction times indicate that both tasks start almost simultaneously (On-Body Touch even before Pointing);

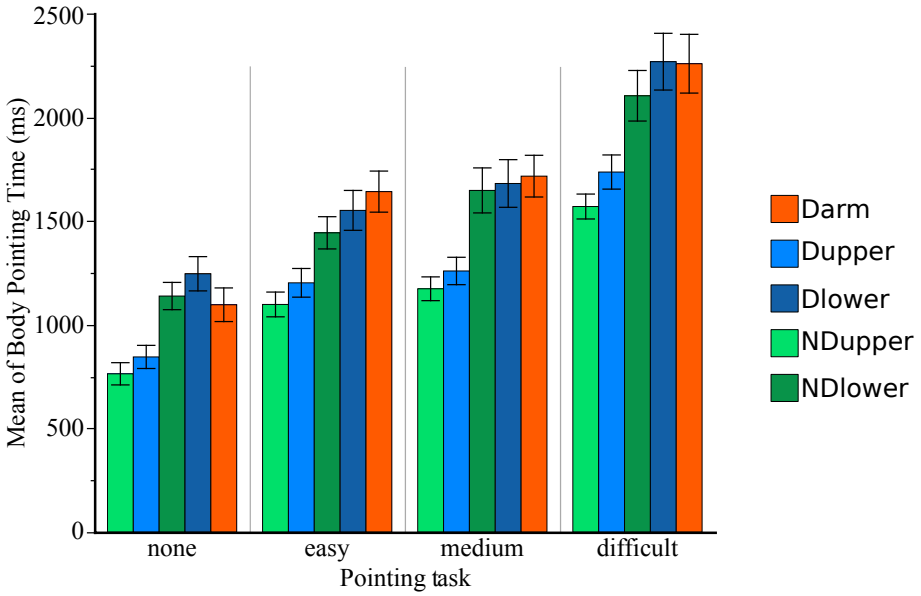


Figure 3.11: Interaction Pointing×On-Body Touch on Body pointing time.

2. This increase in difficulty does not affect the performance ordering of the groups of targets, but amplifies these differences. ND upper and D upper are still the group of targets that require less time to be touched.

Body target has more impact on Body pointing time in this body + pointing condition than it had on Trial Time in the body-only condition:

1. Targets on the *feet* and the *tibias* are now slower to touch than targets on the *knees*. In fact, the lower the target is, the more it requires to change the body balance to be reached, and participants were more careful in order not to impair their pointing precision;

2. For the D arm group, the target located on the *upper arm* is touched around 400 ms faster than the one located on the *forearm*, suggesting that participants were taking care to not displace the cursor by touching their forearm too precipitately.

These effects increase with the difficulty of the pointing task.

Error rate – Body target group and Pointing also have an effect on the On-Body Touch error rate ($F_{4,60} = 12.77$, $p < 0.0001$ and $F_{3,45} = 3.41$, $p = 0.0253$). Post-hoc Tukey tests show that difficult pointing tasks caused significantly less body pointing errors (5.2 %) than easy and medium (mean 8.1 %) ones, with the Body only condition in between, not significantly different from any other conditions (7.3 %). Although surprising, this lower On-Body Touch error rate with difficult pointing tasks can be explained by an increased

attention of the participants due to the difficulty of the primary task. For BODY TARGET GROUP, a post-hoc Tukey test shows that targets on the dominant arm are more error prone than those in others groups (15.6 % against about 6 % for dominant and non-dominant upper, and 2.7 % for non-dominant lower). This result is similar to the BODY ONLY baseline and suggests that targets on the dominant arm were the least reliable in both conditions, probably because of detection problems due to their proximity. We classified the errors into five groups:

Vicinity (57 %) and **Dominant arm position (33.3 %)**, as described previously.

Symmetry (4.2 %): the participant misinterpreted the stimulus and touched the mirrored target, e.g. the dominant shoulder instead of the non-dominant one.

Default target (1.3 %): the participant repeatedly pressed the default position target after the stimulus appeared, either consciously or as a result of a detection failure.

Completely wrong (4.2 %): none of the above, i.e. no easy explanation.

There were more dominant arm position errors than in the BODY ONLY condition, mostly on the D ARM and D UPPER target groups. We observed that participants were keeping their dominant arm close to their torso in order to help stabilize the cursor when touching body targets, likely causing more recognition errors when the body target was on the dominant arm or the torso. We also observed a few *Symmetry* and *Completely wrong* errors that did not occur in the BODY ONLY condition, probably because participants were sometimes confused by performing two tasks simultaneously. We expect these errors to decrease with experience.

In conclusion, our hypothesis about the impact of POINTING on ON-BODY TOUCH is partially verified (**Q2**): the difficulty of the primary task has a strong effect on the body pointing time and on the difference of performance between body targets. Indeed the least convenient targets on the lower body parts are even more difficult to acquire because users have to maintain body balance while pointing. Concerning errors, the primary task does not increase error rate but adds new error types and increases the number of *Dominant arm position* errors.

3.4.3.5 Body Targets Preference and Social Acceptance

We analyzed the subjective data about preference and social acceptance as ordinal data using nonparametric comparisons for each pair with the Wilcoxon method.

Participants generally felt more comfortable touching targets on the upper body than on the lower body or the dominant arm involved in POINTING (see Figure 3.12). For the most

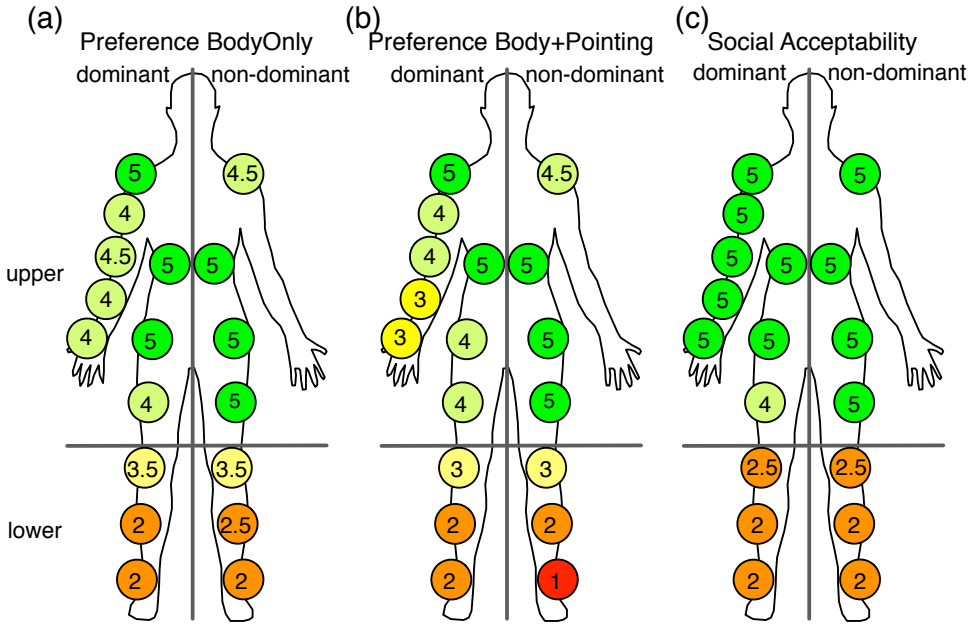


Figure 3.12: Medians of participants ranking preference and social acceptance of body targets in all conditions.

part, participants' assessments of preference for and within each BODY TARGET GROUP are consistent with their performance in the experiment. Wilcoxon tests show significant differences in the preference of each BODY TARGET GROUP: groups on the *upper body* are preferred over *lower body* parts, groups on the *non-dominant* side are preferred over groups on the *dominant* side, and in particular over the *dominant arm*.

Finally, as shown in Figure 3.12, participants clearly ranked targets on the upper body part as more socially acceptable than targets on the lower body parts, in accordance with previous studies [96]. Regarding **Q3**, these subjective results show that while being generally well-accepted by the participants, touching certain body parts while performing an interactive task could decrease perceived comfort of use and impair the performance of the main task.

3.5 Conclusion

In this chapter I introduce a morphological design space for on-body touch techniques that takes body location, touch dynamics and contact shape into account, resulting in theoretical vocabularies of hundreds of commands. The most expressive factor of this design space, body locations, has been seldom studied in the literature. I reported on an exploratory controlled experiment investigating the performance and acceptability of 18

on-body targets located on the trunk, legs and dominant arm for discrete selection. This experiment also explored how invoking commands using simple on-body touch affects simultaneous actions (here mid-air pointing) and conversely.

Participants were most effective with targets on the torso and least effective with targets on the lower body and on the dominant arm, especially in the POINTING+BODY condition: reaching targets on the lower legs requires additional balance and touching the dominant arm impairs the precision of mid-air pointing because of the force applied on the pointing arm. Users consistently preferred on-body targets located on the upper body.

These results suggest three guidelines for designing on-body interactions:

D1 *Task difficulty*: Designers should place on-body targets on the most stable locations, such as the upper torso, when a simultaneous task requires precise or highly coordinated movements.

D2 *Body balance*: Designers should detect anticipatory movements, such as shifts in balance to accommodate corresponding perturbations in a primary task, e.g. freezing an on-screen cursor. The precision of a pointing task can be adversely affected if the user must also touch an on-body target that requires a shift in balance or coordination, in particular, touching the dominant arm while it is performing a separate task.

D3 *Interaction effects*: Designers should consider which body parts negatively affect users' comfort while touching an on-body target. Designers should also consider side effects of each task, such as reduced attention or fatigue that may lead to unexpected body positions or increases in error rates.

Regarding the *Target location* factor of the design space presented in Section 3.3, this exploratory experiment tells us that tap-based on-body interactions are faster and preferred on the most static body parts, namely the shoulders and trunk. The non-dominant lower leg (ND LOWER group), while among the slowest in both BODY ONLY and POINTING+BODY conditions, caused fewer errors than other groups. It could thus be used for critical commands. The dominant arm, while heavily studied in previous work about on-body touch, showed lower performance and is highly sensitive to "external" movements. The number of targets in this experiment was limited by our sensing system: seven objects were tracked in real time with an optical system, which caused occlusions and mis-recognitions in some (limited) body configurations. A future step of this work could be to explore the limits of human accuracy when touching one's body parts, with or without feedback, with touch events detected by sensors located on the user's body, as opposed to computed from distant cameras. The combinations of the factors of the on-body touch design space should also be evaluated in detail. Finally, touching the dominant arm *before* mid-air pointing

could improve pointing stability, as when holding a gun[8].

Regarding the *Task Allocation* taxonomy presented in Section 1.3, the Pointing+Body condition was an example of the *Maximizing limb usage* strategy since mid-air pointing and on-body touch were assigned to respectively the dominant and non-dominant arm and hand. This combination proved less efficient than when mid-air pointing or on-body touch were performed alone, but still had acceptable performance. A notable exception is when the on-body targets were located on the dominant arm, which was used for two distinct interactions. This situation was an involuntary instance of *Factorizing task allocation* and showed the lowest performance, in accordance with the results from Chapter 2: simultaneous actions interfere with each other when they are controlled with the same limbs.

[8]http://www.hciforpeace.org/

Conclusion of Part I: Parallelism and Limb Usage

In the first chapter of this part, I presented an analysis of the differences between "classic" desktop environments and large display platforms in order to understand the constraints that apply to mid-air interaction in the latter. I then introduced a design space about *Feedback Location* that helps classify interaction techniques and understand how they impact collaboration and usability. I also introduced a taxonomy of *Task Allocation* strategies that describes how existing systems assign interactions to limbs in order to study how to combine interaction techniques in environments with limited input capabilities such as large display platforms. This taxonomy features three categories: *Maximizing limb usage*, *Factorizing task allocation*, and a specific case of the latter, *Maximizing cursor usage*. These categories can be combined to design rich, multi-task applications.

In Chapters 2 and 3, I studied mid-air interactions in combination: pointing and virtual navigation, and pointing and command invocation. These studies covered two items of the *Task Allocation* taxonomy:

Maximize limb usage: In Chapter 2, pointing/panning and zooming were assigned to, respectively, the dominant and non-dominant hands in the *two handed* conditions (Figure 3.13-a). Similarly, pointing and discrete selection were assigned to the dominant and non-dominant hands in Chapter 3 (Figure 3.13-b).

Factorize task allocation: In Chapter 2, pointing and panning were systematically assigned to the same limb, i.e. the dominant hand and arm (Figure 3.13-c). In the *one handed* conditions, zooming was assigned to the dominant hand as well (Figure 3.13-d). In the Pointing+Body conditions of Chapter 3, some on-body targets were located on the dominant, pointing arm (Figure 3.13-e). This arm was then used for mid-air pointing and as a target for on-body touch, making it a particular case of *Factorizing task allocation*.

The corresponding results can be analyzed by correlating performance with degree of

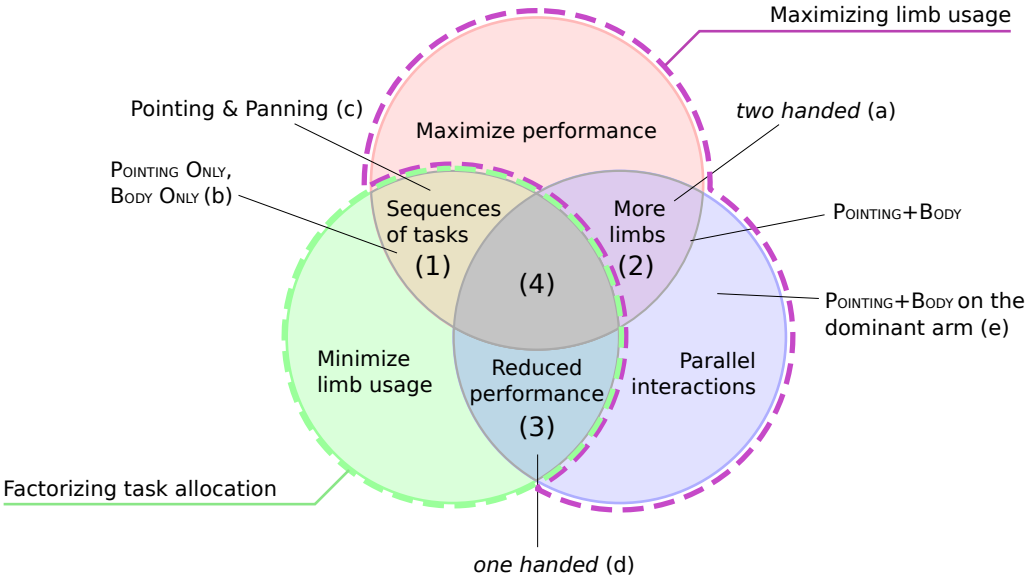


Figure 3.13: Trade-off between limb usage, performance and parallelization of interactions, within the *Task Allocation* taxonomy developed in Chapter 1. *one handed* and *two handed* refer to the corresponding conditions from Chapter 2. Pointing only, Body only and Pointing+Body refer to the corresponding conditions from Chapter 3. "Pointing & Panning" refers to the sequential arrangement of pointing and panning in Chapter 2.

parallelization. Two-handed pan-and-zoom (a) performed better than single-handed pan-and-zoom (d) because users had trouble varying the zoom level while keeping the focus of expansion steady in the single-handed condition. Conversely, panning was never used in parallel with pointing and zooming (c) and thus never interfered with them. Performance of on-body touch decreased when used simultaneously with mid-air pointing (b), especially when both interactions used the same limb (e).

We can see a pattern appear: In both studies, performance of simultaneous actions with the same limb was worse (*one handed* in Chapter 2 and D arm in Chapter 3) than with different limbs. Conversely, simultaneous actions with several limbs performed better. Actions performed in sequence with the same limb performed less well than simultaneously with other actions, as shown by pointing stability in Chapter 2 and lower general performance when combining on-body touch and mid-air pointing in Chapter 3.

This reveals a trade-off between maximizing performance, minimizing limb usage and designing parallel interactions (summarized in Figure 3.13):

(1) Maximizing performance while using a limited set of limbs will likely require interactions in sequence, as with our pointing and panning tasks (Chapter 2).

(2) Conversely, maximizing performance with parallel tasks requires a greater number of

limbs, e.g. both arms must be used, as with the bi-manual conditions in Chapter 2 and the simultaneous pointing and on-body touch condition in Chapter 3.

(3) Performing several tasks in parallel with a limited set of limbs can decrease performance, as with the uni-manual condition in Chapter 2.

(4) Maximizing performance of interactions controlled in parallel while using a limited number of limbs remains a challenge.

This trade-off underlines the impact of simultaneity between interactions. Performing actions in parallel (Figure 3.13-2) is clearly a key factor to better performance in general as it can interleave their completion time. However we observed that simultaneous interactions can also increase errors and have negative effects on users' comfort and opinion of the technique under certain conditions. The experimental results of Chapters 2 and 3 indicate that allocating simultaneous interactions to different limbs is an efficient strategy to design parallel interaction techniques. Within this trade-off, minimizing limb usage follows Requirement 3 (**Input channels**): interaction techniques should avoid using too much of a user's input capabilities if they are to be combined in an application.

Generally, the limited input expressiveness of fewer limbs often can not accommodate a seamless integration of parallel input movements. These results provide useful guidelines for the design of application using several interaction techniques in large display environments: designers should prioritize the need for high performance, parallel actions and input requirements, and choose *Task Allocation* strategies accordingly (see Figure 3.13).

Regarding the design of interaction techniques for large display platforms, I proposed four requirements that I believe increase the usability, performance and potential for collaboration based on an analysis of the specificities of such platforms. I also developed two morphological design spaces (Sections 2.2 and 3.3) that explore new input channels for mid-air interactions. Finally, I contributed new techniques to the corpus of available and evaluated interaction techniques for large display platforms.

Part II

Dual-Precision techniques

In Part I, I analyzed how several mid-air interaction techniques can be combined considering the limited input expressiveness available when standing or walking in front of a wall-sized display. I presented a taxonomy of strategies to allocate multiple interactions to the user's limbs. Preliminary hypotheses from this taxonomy were validated through controlled experiments with multiple interactions. The results of these experiments revealed a trade-off between minimizing limb usage, maximizing performance and parallelization of actions: favoring two of them is always at the cost of the third one when combining interactions.

Mid-air pointing was involved in both experiments of Part I because it is an elementary interaction for direct manipulation when users need to navigate physically in front of the display. For simplicity it was always implemented as simple ray-casting, which is enough for coarse pointing tasks such as view pointing (Chapter 2). However, large displays now feature very high pixel densities (up to 100 dpi) that create new challenges for pointing techniques as discussed in Section 1.1 (page 25).

I will now present a theoretical analysis of pointing mechanisms and define a family of target-agnostic pointing techniques called *Dual-precision* techniques. These techniques support pointing tasks with target sizes at the limit of human visual acuity, therefore meeting Requirement 1 (**Human perception**). I will first design and evaluate *Dual-precision* pointing techniques within the *Maximizing limb usage* strategy of task allocation defined in Section 1.3: all the techniques presented in Chapter 5 can be used single-handed with the dominant hand in order to enable other interactions or devices to be controlled with the non-dominant hand. I will then design and evaluate *Dual-precision* pointing techniques within the *Factorizing task allocation* strategy: all the techniques introduced in Chapter 6 are used on hand-held tactile surfaces that can accommodate other widgets.

Chapter 4

The limits of single-precision techniques[1]

4.1 Introduction

Pointing at large displays from a distance has been studied in various contexts, ranging from low resolution displays to high-resolution back-projected walls. However, it has been less studied in the context of ultra-high resolution walls that can display much smaller visual elements that users can still see. In a typical desktop situation with a resolution of 1920×1200 pixels, the maximum pointing amplitude is $\sqrt{1920^2 + 1200^2} = 2264$ pixels and the minimum target size is one pixel. Not taking edge pointing [6] into account, the corresponding, most difficult pointing task has a Fitts' ID of $log_2(2265) = 11.15$ bits. In comparison, the technically highest pointing amplitude on an ultra-high resolution display like the WILD room (Appendix A) is 21,457 pixels. It would corresponds to a Fitts' ID of 14.4 bits if users could see every pixels. I showed in Section 1.1 that users with normal visual acuity cannot distinguish pixels at 100 dpi when standing further away than 86 cm. However a pointing task could start out of the pixel-perceptive range of a user (provided that the cursor is perceivable) and end in front of her.

My goal is to make any *visible* target acquirable by pointing techniques, regardless of the amplitude of cursor movement, the distance of the user to the display and the pixel density of the display (Requirement 1). As said before, there is a maximal pixel density (≈ 300 ppi) above which the human eye cannot differentiate pixels at more than a reasonable reading distance distance (30 cm in my example page 28). Providing displays with greater pixel density thus makes little sense.

[1] A subset of this chapter has been published in an Inria technical report [127].

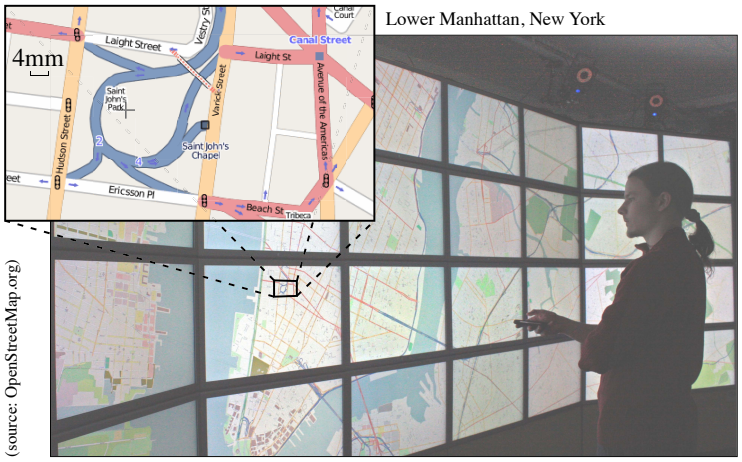


Figure 4.1: The interactive wall-sized (5.5m × 1.8m) ultra-high-resolution (20 480 × 6 400 = 131 million pixels) display used for our studies. Inset: magnification of a 9cm × 5cm area.

In this chapter I will address Requirement 3 (**Input channels**) by investigating distant pointing techniques that can be used with a single hand, in order to allow other interactions such as what was evaluated in Part I with the remaining hand. This discards, among other things, techniques that need the non-dominant hand to support an input device such as a tablet or phone like ARC-Pad does [119]. I also define a specific 5th requirement for mid-air pointing techniques: they should be **Eyes-free**, i.e. *users should not need to look at the input device that controls pointing*, either punctually or during the whole pointing, when pointing control is indirect. For example, moving a cursor on a wall-sized display using a hand-held smartphone as a trackpad does not require users to look at the tablet during pointing. Conversely, pointing a target *on the tablet* obviously requires looking at its screen, but is a direct control.

This chapter addresses the problem of existing target-agnostic, single-handed, mid-air and eye-free high-precision pointing on ultra-high resolution wall displays: given the very high pixel density, do existing pointing techniques enable users to efficiently select both large and small targets from a distance using a single hand and without switching their attention away from the wall display? I investigate this question by first identifying the limits of modeless devices in a theoretical study, then assessing these limits in a formative user study.

4.2 Matching the limits of human perception

Over the past ten years, a number of physical input devices have been explored for pointing on large displays. As with desktop pointing, some techniques map the absolute position

of the input device to the cursor's position, while others use its relative motion to control cursor displacements.

We are specifically interested in techniques that allow users to point from afar, while walking in front of the display, and that do not require both hands. Therefore direct techniques that use a pen [75] or direct touch [38, 160] are not considered since they require users to stand within physical reach of the display. Similarly, we do not consider systems that can work from afar but that require users to seat at a table. For example, Malik et al. [117] introduce a vision-based system for whole-hand gestural interactions performed on a constrained tabletop area. The system supports precise target acquisition on a back-projected wall-sized display from afar using asymmetric interactions, but is designed for people seated at a table. The rest of this section therefore focuses on single-handed techniques that can be used in mid-air.

4.2.1 Absolute mid-air pointing devices

Absolute pointing devices or techniques map a single input state, e.g. the location and orientation of a hand-held device, to a single cursor location on the display. Techniques based on the absolute position of the input device include the family of *ray-casting* techniques, also called laser pointing [126, 132, 133]. These techniques extend the user's finger, arm, or hand-held device with an imaginary ray whose intersection with the wall display is highlighted (Figure 4.5.a).

While intuitive, ray casting is essentially angle-based and thus degrades quickly with distance to the display because hand tremor and involuntary motion due to fatigue are amplified as the user is farther away from the display surface [126, 132]. It is therefore not adapted to small targets on ultra-high-resolution displays. Olsen and Nielsen [133] adapted existing interaction techniques to the limitations of this technology. Both Chen and Davis [49] and Oh and Stürzlinger [132] designed collaborative pointing devices based on laser pointers, enabling several users to interact with the display simultaneously. The latter also compared a laser pointer to a conventional mouse in a pointing task. The laser performed significantly worse than the mouse on a 1.83m $\times$ 1.22m low-resolution back-projected screen, but was preferred by users.

Myers et al. [126] studied the effect of human body limitations on laser pointing. They compared the pointing performance of a laser pointer, a regular mouse, a touch-sensitive SmartBoard^{TM} and *Semantic snarfing*. With the latter, users point with a stylus on a handheld that displays a copy of a region from the main screen. The technique requires users to look at the handheld device, creating a division of attention. Direct input standing

in front of the SmartBoard was the most efficient technique, followed by Semantic Snarfing. Laser pointer was the worst technique. Except for the SmartBoard which required direct contact, other conditions were performed seated about 1.52m away from the display.

Another ray-casting technique consists in holding a device at arm-length in front of the eyes so that the target is aligned with the tip of the device [137]. The technique is interesting as it resembles aiming, but our own tests (see Appendix B) revealed its limitations: it is more tiring and less precise than laser pointing, it causes visual occlusion, and it requires users to repeatedly switch between two very different focal lengths.

Some techniques use absolute input but focus more on the interaction vocabulary than on pointing performance. For example, VisionWand [41] tracks the position of a wand in 3D using two low-cost cameras. The two ends of the wand have different colors and can be distinguished by the vision system. While it does not improve distant pointing performance, it enables interactions such as tap, tilt, flip and rotate gestures. Other works use vision-based techniques to enable freehand pointing. Nickel and Stiefelhagen [130] recognize pointing gestures with two cameras. They introduce new pointing techniques using information such as head and forearm orientation, but focus on the recognition of relevant gestures among a sequence of arbitrary movements rather than precision of pointing gestures. With Shadow Reaching [152], users reach distant objects through the shadow of their body cast on the display surface by a light source. Because of projection perspective, the regions that can be reached depend on both the setup and the user's distance to the display.

Absolute mappings can also be used in combination with a small hand-held device. With the Touch projector [33], users manipulate objects located on a distant display using a smartphone (iPhone) through a live video feed showing that display.

Finally the Wiimote and other game controllers have also been studied as general-purpose pointing devices. Campbell et al. [40] evaluated a Wiimote operated as a zero-order or first-order pointing device, and found that participants were roughly 2.5 times faster in the zero-order condition. Natapov et al. [129] compared remote pointing with a Wiimote, a classic gamepad's joystick, and a mouse operated on a desk as baseline. They found that the mouse had the best throughput, followed by the Wiimote and the joystick, and reported that hand tremor and small movements greatly affected accuracy in the Wiimote condition for small targets.

4.2.2 Motion-based mid-air pointing devices

Techniques that map relative device motion to cursor displacements can be based on position control or rate control (zero or first order of control). Previous studies [40, 129] and our own tests show that techniques based on rate control are faster and more comfortable for coarse pointing across large distances, but perform poorly during the final precise pointing phase. MacKenzie and Jusoh [115] compared a regular mouse to a gyroscopic mouse[2] held on a table and then in mid-air, and to a handheld isometric joystick. The task was performed 1.52m away from a 15" screen. The joystick and the gyro-mouse held in mid-air performed poorly compared to the mice. Finally, Casiez and Vogel [45] evaluated rate-controlled pointing techniques with isometric and elastic devices and with several CD gains. The results are instructive, but they did not evaluate their designs against control-based techniques. Therefore we only consider zero-order techniques in the following.

Position-based pointing techniques require a *resolution function* [42] mapping input (device motion) to output (cursor motion). The most common resolution function uses a Control-to-Display (CD) gain, defined as the ratio between cursor movement and input variation. The CD gain can be constant or, more often, depends on the velocity of input motion[3]. Multiplying the magnitude of the input device motion by the CD gain gives the amplitude of cursor motion. Since a gain is theoretically without unit, both input and output amplitudes must be expressed in the same, or comparable, units. However, some techniques or devices map input and output of different natures. For example, a gyroscopic mouse transforms angular movements into cursor translations. In such cases the "gain" has a unit, such as $length.angle^{-1}$ in this example. For the sake of simplicity and since the same computational mechanism is applied whether the multiplier has a unit or not, we also refer to such values as CD gains.

The value of a CD gain is not necessarily constant over time. Most major operating systems use transfer functions between the input velocity and the CD gain, from the principle that slow inputs occur when precise cursor movements are intended and that fast inputs occur when coarse cursor movements are intended. This specific velocity-based relative technique is called Pointer Acceleration [44] (PA). However, the literature about such functions is rather scarce, and existing functions were not designed for wall-sized, ultra-high resolution displays.

The set of relative pointing techniques and devices that are likely to work for high-resolution wall displays can be refined by analyzing the devices' characteristics using the

[2] A mouse that uses gyroscopic sensors, also called gyro-mouse.

[3] The CD gain typically increases with input velocity according to the so-called Pointer Acceleration.

framework defined by Casiez et al. [43]. The framework provides formulae to compute upper and lower bounds for the CD gain, noted CD_{max} and CD_{min}. A gain below CD_{min} requires clutching which is likely to decrease performance. A gain above CD_{max} creates precision problems because of hand tremor and/or device quantization. If CD_{min} is greater than CD_{max} (Fig. 4.2), these problems are compounded: any CD gain value will trigger at least one of them, or both.

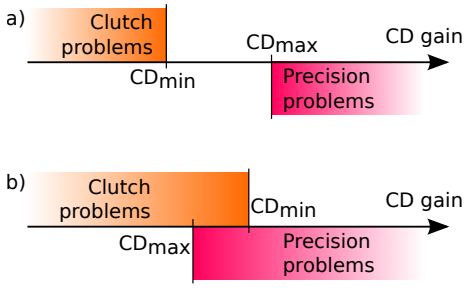


Figure 4.2: (a) Problems that arise when the CD gain value is lower than CD_{min} or larger than CD_{max}. (b) When CD_{min} is greater than CD_{max}, no CD gain value can avoid both problems.

These formulae are based on the minimum target width of the tasks (W_{min}), the maximum distance between targets (A_{max}), the pixel density of the display ($Screen_{res}$), the device's morphological characteristics —operating range (OR) and input resolution ($Device_{res}$)— and human motor precision ($Hand_{res}$). All these parameters must be expressed in the same distance unit, e.g., millimeters or inches, but not in pixels because display pixels and device *ticks* (or minimal input) often have different physical sizes. A CD gain used with a 600 dpi mouse on an 80 ppi screen should have the same effect than with a 1600 dpi mouse on a 100 ppi screen.

The formulae from Casiez et al. [43] use $Device_{res}$ and $Screen_{res}$ expressed as densities in dots per inches (ppi), which forces all other parameters to be expressed in inches or to convert units. We chose to use the multiplicative inverse of these measures instead, i.e., the physical size (ps) of pixels, so that they can be expressed in any length unit: $Device_{ps}$ and $Screen_{ps}$. Our formulation of CD_{qmax} (explained below) is thus written as the multiplicative inverse of the one described in [43] but has the same result[4]:

[4]We assume that $Screen_{ps}$ and $Device_{ps}$ are non null.

$$CD_{min} = \frac{A_{max}}{OR} \quad (4.1)$$

$$CD_{max} = \min(CD_{qmax}, CD_{lmax}) \quad (4.2)$$

$$CD_{lmax} = \frac{W_{min}}{Hand_{res}} \quad (4.3)$$

$$CD_{qmax} = \frac{Device_{res}}{Screen_{res}} = \frac{Screen_{ps}}{Device_{ps}} \quad (4.4)$$

Following this labeling, $Hand_{res}$ should also be renamed $Hand_{ps}$ for it represents a "point size", i.e. the smallest input that users can perform. However since we do not change its value, as opposed to $Screen_{ps}$ and $Device_{ps}$, we will keep this labeling for consistency with [43].

CD_{min} (Equation 4.1) is the ratio between the largest amplitude of a pointing task on a given display (A_{max}) and the largest device input movement (OR); it represents the minimum CD gain below which clutching is bound to occur, i.e., the maximal input that can be used to move the cursor by the maximal pointing amplitude in a single movement. If the CD gain is lower than this value, the input movement needed to make the cursor go through A_{max} is larger than the operating range of the technique / device. Clutching is thus necessary.

CD_{lmax} (Equation (4.3)) is the ratio between the smallest target size for a given task (W_{min}) and the smallest human input ($Hand_{res}$), i.e. the maximal human precision; it represents the maximum CD gain beyond which human precision problems start to occur, i.e., the minimal input that can be used to keep the cursor stable within the smallest targets. If the CD gain is higher than this value, acquiring targets of W_{min} requires more precision than normally available.

CD_{qmax} (Equation (4.4)) is the ratio between the smallest display output ($Screen_{ps}$) and the smallest device input ($Device_{ps}$); it represents the maximum CD gain beyond which quantization problems start to occur, i.e., the minimal input that can be used to move the cursor by one pixel. If the CD gain is higher than this value, the smallest device input results in a movement bigger than one pixel, thus some pixels become unreachable [7].

In the context of an ultra-high-resolution wall display and especially at a distance, it is not always necessary, or feasible, to reach every single pixel of the screen. In order to relax this constraint on CD_{qmax}, we replace $Screen_{ps}$ by W_{min}, the minimum target size. The resulting formula is the maximal CD gain beyond which targets of this size become unreachable:

$$CD_{qmax} = \frac{W_{min}}{Device_{ps}} \quad (4.5)$$

This is equivalent to Casiez *et al.*'s original formula when W_{min} is the size of one pixel. In addition to relaxing the constraint that every pixel must be reachable, our formulation can also be used to model sub-pixel targets, e.g., when using lenses [7].

Using equations (4.2), (4.3) and (4.5), we can rewrite CD_{max} as follows:

$$CD_{max} = \frac{W_{min}}{\max(Hand_{res}, Device_{ps})} \tag{4.6}$$

Fitts' law [157] is an empirical model that predicts movement time (MT) as a function of movement amplitude (A) and target width (W):

$$MT = a + b \times log_2(1 + A/W)$$

where a, b are determined empirically and depend on factors such as input device and user population, and $log_2(1 + A/W)$ is called the index of difficulty (ID) of the task and is measured in bits. Equations (4.1) and (4.6) allow us to compute a theoretical Fitt's index of difficulty, ID_{max}, beyond which CD_{min} cannot be lower than CD_{max} for a given technique:

$$\begin{aligned}
& CD_{min} < CD_{max} \\
\iff & \frac{A_{max}}{OR} < \frac{W_{min}}{\max(Hand_{res}, Device_{ps})} \\
\iff & \frac{A_{max}}{W_{min}} < \frac{OR}{\max(Hand_{res}, Device_{ps})} && (4.7) \\
\iff & ID_{max} = \log_2\left(1 + \frac{OR}{\max(Hand_{res}, Device_{ps})}\right) && (4.8)
\end{aligned}$$

This equation provides a theoretical limit for the Fitts' index of difficulty (ID_{max}) of a pointing task beyond which a given technique starts causing clutching and/or precision problems (Figure 4.2). Note that the *max* subscript in ID_{max} represents the maximum index of difficulty for the *technique* being considered, while in A_{max} it represents the estimated highest amplitude of the *pointing task*.

The intermediate equation 4.7 provides another reading of Casiez et al. [43]'s formulae. $Exp_{output} = A_{max}/W_{min}$ represents the number of possible target locations on a 1D interval: a punctual cursor on this interval can be over A_{max}/W_{min} different targets of size W_{min} (Figure 4.3-a). $Exp_{input} = OR/\max(Hand_{res}, Device_{ps})$ is the number of different input locations that a user with a given motor precision $Hand_{res}$ can input within a physical operating range OR with a device precision $Device_{ps}$ (Figure 4.3-b). Applying a CD

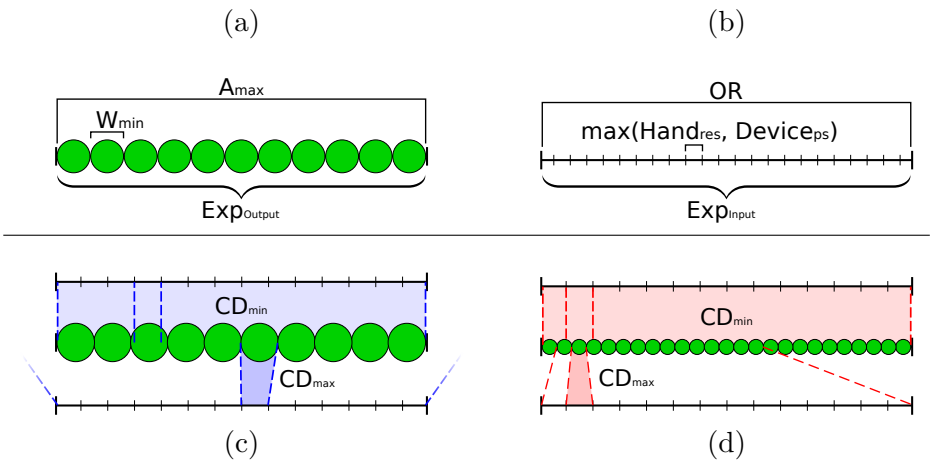


Figure 4.3: The expressiveness paradigm. (a) The output expressiveness $Exp_{output} = A_{max}/W_{min}$. (b) The input expressiveness $Exp_{input} = OR/\max(Hand_{res}, Device_{ps})$. (c) Range of mappings (CD gains) between Exp_{input} and Exp_{output} when $Exp_{input} \geq Exp_{output}$. (d) Possible mappings (CD gains) between Exp_{input} and Exp_{output} when $Exp_{input} < Exp_{output}$.

gain matches a given Exp_{input} to a task Exp_{output}. When $Exp_{input} \geq Exp_{output}$ (Figure 4.3-c), the user and input device have enough expressiveness to acquire targets of the required difficulty. The CD gain can then be set so that either OR matches A_{max} (Equation 4.1, Figure 4.3-c, top), $\max(Hand_{res}, Device_{ps})$ matches W_{min} (Equations 4.6, Figure 4.3-c, bottom), or any value in between. When $Exp_{input} < Exp_{output}$ (Figure 4.3-d), no CD gain value can match OR to A_{max} and $\max(Hand_{res}, Device_{ps})$ to W_{min} at the same time. If set so that the cursor can move as far as A_{max} in a single movement (Figure 4.3-d, top), targets of W_{min} will not be reachable. If set so that W_{min} is acquirable (Figure 4.3-d, bottom), users will have to clutch to perform cursor movements of A_{max}.

Based on these formulae, we analyze four candidate devices with relative input that are one-handed, position-controlled and usable in mid-air (Table 4.1):

1. *Soap* [23] wraps the tracking system of a mouse in a hull made of fabric. Users control the cursor by moving the tracking system inside the fabric, like a piece of soap in the hand. Relative motion of the hull enables both precise positioning of the cursor and moving across large distances. The resolution is that of a regular mouse (600 to 800 ppi), but the operating range without clutching is much smaller (about 3.5 cm).

2. Some *one-handed trackballs* can be operated in mid-air. Their operating range is rather small[5]. The best commercial, desktop trackballs have a resolution of 1000 ppi

[5]Some of these trackballs can be "thrown", i.e., the user can initiate a fast rotation in the direction of the target and let the ball roll until it approaches the target or stops by itself. In our understanding this use of a trackball is not covered by Casiez *et al.*'s formulae. However this is a rather coarse way of using a

Device	$Device_{ps}$	OR	$Hand_{res}$	ID_{max}	CD_{min}	CD_{max}
Trackball	32 μm (800 ppi)	40 mm	0.2 mm	7.65	79.7	35
Soap	32 μm (800 ppi)	35 mm	0.2 mm	7.46	91.1	35
Trackpad	51 μm (498 ppi)	51 mm	0.2 mm	8	62.5	35
GyroMouse	15.47×10^{-3} deg	90 deg	0.18 deg	8.98	35.41 mm/deg	41.61 mm/deg

Table 4.1: Device characteristics for relative pointing techniques. Red entries have $CD_{min} \gg CD_{max}$ and are therefore impractical.

and an operating range of approximately 4 cm.

3. A one-handed, thumb-operated *handheld trackpad* can be implemented using a touch-sensitive device such as a PDA or smartphone. We tested an iPod Touch running a full-screen trackpad application on its 51 × 76 mm surface. This corresponds to a theoretical resolution of 498 × 334 ppi.

4. A *gyroscopic mouse* (*GyroMouse*) converts angular movements of a mouse held in mid-air into conventional mouse events. Users can clutch using a button that freezes the cursor. We used a Logitech MX Air and, through informal testing, considered that the constraint on wrist motion limited its operating range to about 90 degrees.

In order to compute CD_{min} and CD_{max} for each of the above, we have to define the corresponding operating range (OR) and the hand, screen and device resolutions ($Hand_{res}$ and $Device_{ps}$). For the *Trackpad*, *Trackball* and *Soap*, we used Casiez *et al.*'s estimation for $Hand_{res}$ (0.2 mm) and the device resolution for $Device_{ps}$. For *GyroMouse* we adapted the formulae to obtain CD gains expressed in mm/deg. The corresponding $Hand_{res}$ value (0.18 deg) is the standard deviation of the device orientation when a user holds the device and tries to keep its projection (through ray-casting) still. We used conservative values for the smallest target size (W_{min} = 32 pixels or 7 mm) and largest amplitude (A_{max} = 13800 pixels or 3187 mm), resulting in a maximal Fitts' index of difficulty of 8.76 bits.

Table 4.1 summarizes the CD gain computations for the selected devices. The first three have a CD_{min} much larger than their CD_{max} (Figure 4.4). This is also indicated by their ID_{max}, which is smaller than the maximal Fitts' index of difficulty of the task. They are therefore very likely to create clutching and/or precision problems if used with constant CD gains. Had we taken more extreme values for the smallest target size, such as 10 pixels, and 20,000 pixels for the largest amplitude, differences would have been even more striking. We informally confirmed this assessment by trying various handheld trackballs and trackpads, concluding that only *GyroMouse* (Figure 4.5.b) could be a candidate for

trackball: if the target is far away from the cursor, the angular precision of the throw is reduced and the user may have to re-throw the trackball in the right direction. It also seems rather difficult to find and interpret hardware parameters that influence this type of input, e.g. trackballs may have different degrees of resistance to such movements. We leave the analysis of such throwing techniques for future work.

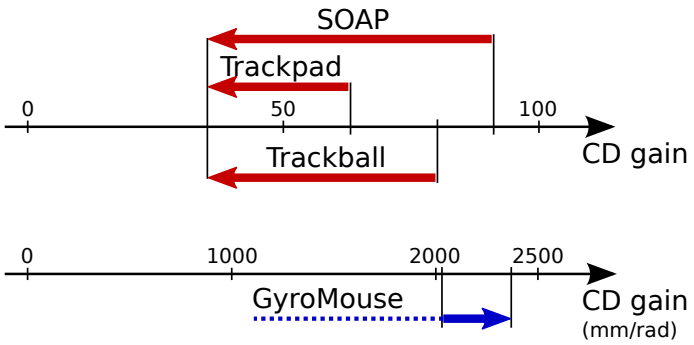


Figure 4.4: Theoretical CD_{min} and CD_{max} for the four selected devices. Arrows go from CD_{min} to CD_{max}. Red arrows indicate that CD_{min} is greater than CD_{max}.

our worst-case pointing difficulty.

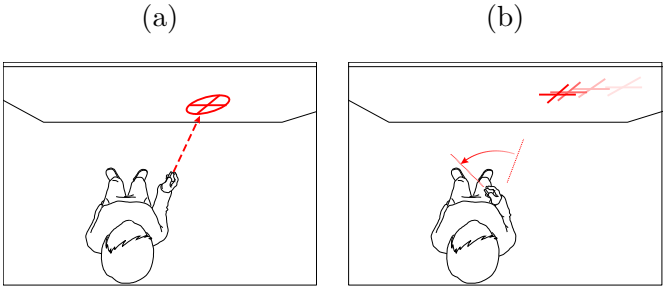


Figure 4.5: *RayCasting* (a) and *Gyro* (b).

4.3 Experiment: Limits of single-mode Techniques

The following formative user study was conducted to identify the limits of the two viable candidates identified in the previous section: an absolute technique, *RayCasting*, and a relative one, *Gyro*. We included two variants of the latter: the classic version with a constant CD gain, and *GyroAcc*, which dynamically adjusts the CD gain according to input device velocity. As explained before, we could not find any related work on how to tune an acceleration function depending on the pointing task parameters at the time where the study was conducted[6]. We designed one by choosing a simple shaped configurable function, namely a generalized logistic function of the form $f(x) = a + \frac{b}{1+e^{\lambda \times (x-M)}}$, and hand-tuned its parameters in an iterative design phase depending on the task's A_{max} and W_{min}.

[6]Since then, Roussel et al. [148] introduced a method to tune acceleration functions.

Figure 4.6: The platform used for the experiment: thirty-two 100 ppi screens (left picture), VICON cameras and VICON markers (white dots on the foreground object, right picture).

4.3.1 Participants

Twelve unpaid volunteers (2 female) served in the experiment, aged 24 to 43 (mean 29.5, std dev. 5.76), all right-handed, with normal or corrected-to-normal vision. All participants were familiar with remote interaction, having previously used at least a WiiMote.

4.3.2 Apparatus

The experiment was run in the WILD room (see Appendix A). All three techniques used a passive device with markers tracked by the motion-capture system.

The experiment was written in Java 1.5 running on Mac OS X and was implemented with the open source jBricks toolkit [138].

We chose to take into account the space behind bezels, making the cursor behave as if there were pixels under them, since a recent study found no effect of bezels on pointing performance [30].

We implemented the gyroscopic mouse by tracking the angular movements of the passive device. This gave us full control over Pointer Acceleration whereas both the operating system and the Logitech MX Air device driver feature native Pointer Acceleration functions that could neither be canceled nor finely tuned. A wireless mouse was attached to the passive device so that users could easily reach its left button to click. The maximum CD gain value of *GyroAcc* (35.41 mm/deg) was set so that users could move across the display without clutching. The minimum CD gain value (2.41 mm/deg) was set to allow enough precision to acquire the smallest targets (see target WIDTH below) while being close enough to the maximum CD gain value. Through pilot testing, we fine-tuned a logistic transfer

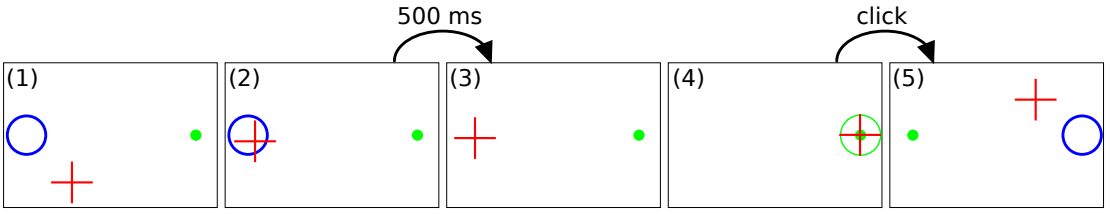


Figure 4.7: (1) The target (green dot) and dwell zone (blue circle) are both visible at the beginning of the trial. Participants have to keep the cursor (red cross) in the dwell zone for 500 ms (2) before it disappears (3). Then they click the target (4) and the next, symmetric trial starts (5). The background of the experiment was black but is shown here in white for clarity.

function based on the input velocity v in order to obtain a smooth transition between the minimum and maximum values of the gain:

$$CD(v) = 2.41 + \frac{33}{1 + e^{-16\times(v-.13)}} \tag{4.9}$$

4.3.3 Task and Procedure

The task was a Fitts reciprocal pointing task. Participants were asked to click targets located alternatively left and right from the center of the display. Targets were presented as bright green disks on a black background. When the cursor was inside the target, the target was highlighted white. An additional, wider green circle appeared (Fig. 4.7-(4)) so that participants could see the feedback unambiguously even for very small targets.

Before each trial, participants had to move the cursor inside a dwell zone using ray-casting (Fig. 4.7-(2)) and leave it there for half a second before the target appeared. The goal was to recalibrate the relative position and orientation of the hand-held device and the cursor at each trial so that the successive offsets caused by the relative techniques would not accumulate, causing undesired clutching. The dwell zone was a 500-pixel-wide circular area centered on the previous target. It disappeared at the end of the dwell time, signaling the start of the trial.

Participants stood at a distance of 2 meters from the display. This distance gave participants a good overview of the display while avoiding problems of visual acuity[7].

In this formative study, we were mainly interested in evaluating the limits of the three techniques in terms of precision. We thus always presented targets in decreasing order of width, stopping the experiment for each technique on a per-participant basis: if a given target was not selected after ten seconds, the trial timed out; When four successive

[7]The smallest target sizes were 12.7' of arc and 6.35' of arc in our two experiments, which is above point acuity (1' of arc, [170]) and minimum decipherable symbol height (5' of arc, [163])

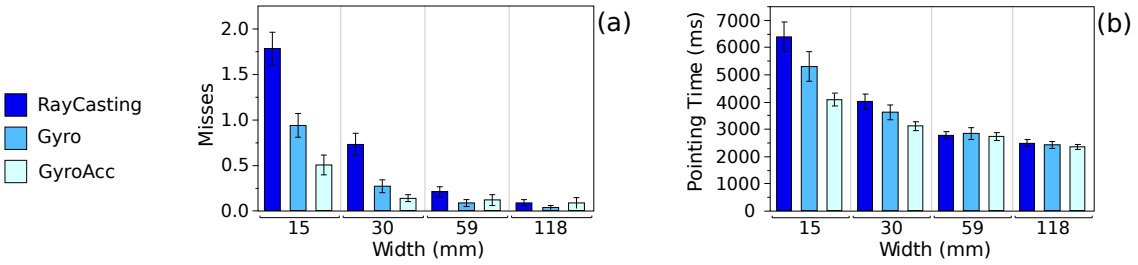


Figure 4.8: (a) *Misses* and (b) *Pointing Time* per TECHNIQUE × WIDTH.

TimeOuts occurred, we considered the task too difficult for the current TECHNIQUE and switched to the next technique, resetting WIDTH to the largest size. This was logged as a *Withdrawal*. In order to avoid edge-pointing effects [6], we set the targets at a distance of 956 mm ($1^1/_2$ screen width) from the left and right sides of the wall display. The distance between targets was thus fixed at 3187 mm (equivalent to the width of 5 screen tiles).

4.3.4 Design

The main factors were TECHNIQUE (*RayCasting*, *Gyro*, *GyroAcc*) and target WIDTH (118, 59, 30, 15, 7 mm). We checked that each participant could see all targets. We used a 3 × 5 repeated measures within-subject design with 10 replications, i.e., 1800 trials (3 × 5 × 10 × 12 participants). For each participant, we grouped trials into 15 blocks (TECHNIQUE × WIDTH). The presentation order for TECHNIQUE was counterbalanced across participants using a Latin square. For each technique, participants were asked to practice using 118 mm targets until they felt comfortable before starting actual measurements. A trial started as soon as the dwell phase ended. For each trial, we logged the time to click the target (*Pointing Time*), the number of clicks outside the target (*Misses*) and the number of times the cursor exited the target (*Crossings*).

At the end of the experiment, participants were asked to rank the three techniques, and to rate them for *Mental Effort*, *Accuracy*, *Speed*, *Fatigue*, *Comfort* and *Overall Easiness* on 5-point Likert scales.

4.3.5 Results

We analyzed the data using multiway ANOVAs, accounting for repeated measures using the REML procedure, and performed Tukey HSD post-hoc tests for pairwise comparisons. We used the mean for *Misses*. As expected, the distribution of *Pointing Time* was left-skewed for each condition, so we used the median instead as a simple method to discard outliers. We verified that there was no significant effect of TECHNIQUE presentation order

and observed that learning and fatigue effects were not significant. We now report the results relevant to assessing the limits of each technique, i.e., task time and mean number of errors (Figure 4.8)[8].

Timeouts and Withdrawal We observed no *Withdrawal* for WIDTH greater than 15 mm. Only *RayCasting* caused *Withdrawal*s for WIDTH = 15 mm (1 participant). For WIDTH = 7 mm, *GyroAcc*, *Gyro* and *RayCasting* caused *Withdrawal*s for 1, 3 and 9 participants respectively. WIDTH= 7 mm was too difficult for *RayCasting* and *Gyro*. We removed the corresponding blocks from all subsequent analyses.

In the remaining blocks, we observe significant effects on *TimeOuts* for TECHNIQUE ($F_{2,20.92} = 4.46$, $p < .025$), WIDTH ($F_{3,31.62} = 10.84$, $p < 0.0001$) and TECHNIQUE×WIDTH ($F_{6,61.1} = 7.03$, $p < 0.0001$). *GyroAcc* (mean 1%) causes significantly fewer *TimeOuts* than *RayCasting* (mean 6%), *Gyro* being in between. Significantly more *TimeOuts* are observed at WIDTH = 15 mm (mean 12%) than 30 mm (mean 1%), 59 mm (mean 1%) and 118 mm (no *TimeOuts*). For WIDTH = 15 mm, *GyroAcc* (mean 2%) causes significantly fewer *TimeOuts* than *Gyro* (mean 13%) and *RayCasting* (mean 22%).

We removed the trials with *TimeOuts* for all further analysis (3.14%).

Misses We observe a significant effect on *Misses* for TECHNIQUE ($F_{2,22} = 40.38$, $p < 0.0001$), WIDTH ($F_{3,33} = 90.64$, $p < 0.0001$) and TECHNIQUE×WIDTH ($F_{6,66} = 15.42$, $p < 0.0001$). Not surprisingly, *Misses* increase significantly as WIDTH decreases, except for WIDTH = 59 mm and WIDTH = 118 mm (Figure 4.8-a) which are not significantly different. The interaction does not change the significance of the post-hoc test, but indicates that the magnitude of the difference increases as target WIDTH decreases. For WIDTH = 15 mm, *RayCasting* (mean 1.78) causes significantly more *Misses* than *Gyro* (0.93) which causes significantly more *Misses* than *GyroAcc* (.5).

Crossings There is a significant effect on *Crossings* for TECHNIQUE ($F_{2,22} = 48.37$, $p < 0.0001$), WIDTH ($F_{3,33} = 88$, $p < 0.0001$) and TECHNIQUE×WIDTH ($F_{6,66} = 41.13$, $p < 0.0001$). *RayCasting* (mean 1.97) caused significantly more *Crossings* than *Gyro* (1.04), which caused significantly more *Crossings* than *GyroAcc* .63). *Crossings* significantly increased when WIDTH decreased, except for $Width = 59$ mm and $Width = 118$ mm which are not significantly different. For $Width = 15$ mm, *RayCasting* (mean 4.94) causes significantly more *Crossings* than *Gyro* (2.85) which causes significantly more *Crossings* than *GyroAcc*

[8]Error bars in all figures represent the 95% confidence limit of the mean of the medians per participants ($\pm StdErr \times 1.96$).

(1.18). *RayCasting* with $Width = 30$ mm causes significantly more *Crossings* than any other technique with $Width \geq 30$ mm.

Pointing Time We observe a significant effect on *Pointing Time* for TECHNIQUE ($F_{2,22} = 15.17$, $p < 0.0001$), WIDTH ($F_{3,33} = 59.59$, $p < 0.0001$) and TECHNIQUE×WIDTH ($F_{6,66} = 16.36$, $p < 0.0001$). *GyroAcc* (mean 3061 ms) is significantly faster than *Gyro* (3537 ms) and *RayCasting* (3902 ms). Unsurprisingly, *Pointing Time* increases significantly when WIDTH decreases (Figure 4.8-b), and the interaction does not change the significance of the post-hoc test, but indicates that the magnitude of the difference increases as target WIDTH decreases.

Subjective Results Pearson chi-square test shows that there is a significant effect on *Accuracy*, *Speed* and *Overall Ranking* for TECHNIQUE. 7 participants felt that *GyroAcc* was more accurate than the other techniques, 4 ranked it ex-aequo with one or both other techniques, and only 1 participant felt that *RayCasting* was more accurate than both other techniques. 9 participants felt that *GyroAcc* was faster than the other techniques, 2 ranked it ex-aequo with *RayCasting* and 1 felt that it was the slowest technique. 9 participants ranked *GyroAcc* best, one ranked *RayCasting* best and two participants could not rank any of the three techniques, feeling they were equivalent. 5 participants ranked *Gyro* second, 4 participants ranked *RayCasting* second and one ranked *GyroAcc* second. One participant felt that the maximum speed of *GyroAcc* was way too fast. When asked, this participant reported using a low Pointer Acceleration setting in his daily use of personal computers.

4.3.6 Discussion

The above results show that *Pointing Time*, *Misses*, *Crossings*, *TimeOuts* and participants preferences are strongly correlated. They also show the respective limits of each technique:

- *RayCasting* is not accurate enough to select targets such as those found on a map displayed on an ultra-high-resolution wall (Figure 4.1): Due to hand tremor and input resolution, the accuracy of this technique makes it difficult to acquire or keep the cursor stable within targets smaller than 30 mm (128 px) at a distance of 2 meters from the display. 59 mm is the smallest target width that caused less than 0.5 misses in average, corresponding to a Fitts' ID of 7.78 bits which we consider is the limit of this technique for usability.
- Relative techniques such as a *Gyro* mouse can be made precise enough by choosing a sufficiently low CD gain, making it possible to acquire targets 15 mm wide. However,

the high number of *TimeOuts* (12%) shows that it is not really usable for such widths with a 3-meter amplitude. 30 mm is the smallest target width that caused less than 0.5 misses in average, corresponding to a Fitts' ID of 6.74 bits which we consider is the limit of this technique for usability.

- *GyroAcc* alleviates this problem by dynamically adjusting the CD gain as a function of input device velocity, allowing users to control the tradeoff between speed and precision. It can be made efficient for pointing targets that are both distant and small. *GyroAcc* was precise enough to acquire the smallest targets (7 mm), while fast enough to move across more than 3 meters. While *GyroAcc* performed relatively well, *Withdrawal* also started to appear for 7 mm targets, suggesting that our prototype is reaching its limits at that difficulty (ID= 8.83 bits). Furthermore, for higher differences between A_{max} and W_{min}, the CD gain range of the transfer function will increase and we expect the resulting, much steeper, slope to become harder to control.

4.4 Conclusion

We investigated the problem of eye-free single-handed pointing on large ultra-high-resolution (100+ million pixels, 100 ppi) wall displays from a distance. We first explored the limits of existing single-mode remote pointing techniques, both absolute and relative. We showed that targets smaller than 30 mm (128 px on a 100 ppi display) could not be reached reliably with a 3m amplitude if a single static CD gain was used: precise pointing required a low gain, which caused too much clutching to cross large distances. With a dynamic gain (*GyroAcc*), the practical limit improved to about a quarter of this size.

We will now propose a framework for pointing techniques that can accomodate virtually any pointing task.

Chapter 5

Dual-precision[1]

5.1 Introduction

In the previous chapter we established the limits of existing single-mode, mid-air pointing techniques that can be used one-handed and eye-free. Being single-mode, they all use a single resolution function (whether it is a constant CD gain or a velocity-based transfer function). A pointing technique should adapt to the hardware and performance requirements of a task (Requirements 1 and 3). In particular, it should provide fast cursor movements for large amplitudes and precise control for small targets[2]. While these two aspects are well covered in the literature about pointing techniques in desktop environments, the results of Chapter 4 show that existing techniques hit a performance ceiling with very high Fitts' IDs such as the ones that can be reached on a ultra-high resolution wall display. Even *GyroAcc*, a technique that varied the CD gain with input velocity, caused *TimeOuts* and *Withdrawal*s in the previous experiment while the targets were still easily visible, failing to meet Requirement 1. While this could be due to our ad-hoc transfer function, we believe it reveals a limitation of the human sensory-motor system.

In this chapter we investigate an alternative solution. The first phase of a pointing task, especially with a high amplitude, has been shown to be a ballistic phase [121] in which users control the orientation and velocity of the cursor but only have a limited control of its location. The second phase of a pointing task consists in bringing the cursor within the target and may require high precision when the target is small, thus a finer control over the cursor location and a lower velocity. These two phases make users focus on two different aspects of motor and cursor movement, and the transition between them can be

[1] A subset of this chapter has been published in an Inria technical report [127].

[2] Other aspects can be considered, such as how a given technique allows accurate path following. Here we only consider pointing difficulty as modeled by Fitts' ID.

challenging with difficult pointing tasks.

Following these observation and using the parameters of Fitts' law, we propose a family of target-agnostic pointing techniques that explicitly divide pointing tasks in two phases of different velocities:

- a coarse phase in which users quickly approach the target by traversing most of the task amplitude with no or minimal clutching,
- a precise phase in which they can acquire nearby targets of virtually any visible Width.

Dual-precision pointing techniques thus feature two modes, *Coarse* and *Precise*, with explicit mode switches.

Such techniques have already been studied in the literature. With HybridPointing [63], users can reach distant objects by switching from absolute to relative pointing. However the technique still requires direct contact of the pen with the display surface, failing to meet Requirement 2.

ARC-Pad [119] uses a touch-sensitive mobile device for cursor positioning on large displays. When the user taps the screen, the position of the tap is mapped to the entire display, enabling coarse but fast repositioning of the cursor. When the user drags on the touch surface, the finger movements are interpreted as relative input, allowing precise adjustments to the cursor's position. Tapping vs. dragging is a practical way of differentiating the user's intention between coarse and precise cursor control, and was proven efficient on a 52" screen with a resolution of 1360×768 pixels. However, this approach does not scale: when mapping a 7 cm input device to a 5 meter display, a 1 millimeter error in the tap location corresponds to a 7.5 cm error on the display, resulting in multiple dragging gestures to reach the target. Since we are interested in eye-free techniques (Requirement 5), the error in the initial tap location is likely to be much larger, exacerbating the problem. Preliminary tests confirmed this hypothesis, so we did not include ARC-Pad in our study.

Vogel and Balakrishnan [163] use a high-precision 3D motion tracking system to develop and evaluate three techniques: pure ray casting, relative pointing with clutching, and ray-to-relative pointing, which combines absolute and relative pointing using two different hand postures. We adapted the latter to our environment and tested it in Exp. 2.

The family of adaptive techniques including PRISM [67], Adaptive Pointing [103] and Smoothed Pointing [69] are good candidates for this task. However only PRISM was published at the time of this study and was originally defined as a 3D pointing technique.

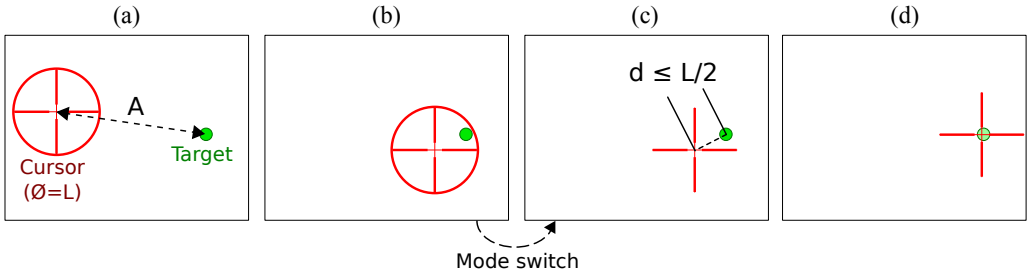


Figure 5.1: Typical use of a dual-precision technique: In *Coarse* mode the cursor features a circle indicating the range of the precise mode, L (a). The user brings the area cursor close over the target (b), i.e. at a distance $d \leq \frac{L}{2}$, then switches to *Precise* mode (c). She finishes the task in *Precise* mode (d).

5.1.1 Key Parameters

The design of dual-precision techniques relies on four key parameters: the resolution functions used in each phase, the point at which the user is expected to switch modes and the mode switch mechanism.

In order to be optimal, the mode switch should occur when the target is within the range of the precise mode, a distance we denote L. To assist users, we propose to visually represent this limited operating range by surrounding the cursor with a circle of diameter L when in coarse pointing mode (Figure 5.1-a). The coarse pointing phase can then be seen as an area cursor pointing task [95] that consists in bringing the cursor's circle over the target (Figure 5.1-b), while the precise pointing phase is a regular target acquisition task with a distance d lower or equal to $L/2$ (Figure 5.1-c,d).

As mentioned before, Fitts' law is expressed as a linear model of an Index of Difficulty (ID) computed from the task Width and Amplitude:

$$MT = a + b \times log_2(1 + A/W) \tag{5.1}$$

In our case, the *Coarse* phase corresponds to bringing the cursor close enough to the target to be within reaching range of the *Precise* mode. The corresponding index of difficulty is

$$ID_C = log_2(1 + A/L) \tag{5.2}$$

Similarly, the *Precise* phase consists in bringing the cursor from the location of the mode switch to the target, thus through a distance d supposedly lower or equal to $L/2$. The corresponding index of difficulty is

$$ID_P = log_2(1 + d/W) \tag{5.3}$$

We cannot predict the value of d, so we use a higher bound for the *Precise* phase's ID:

$$ID_P = log_2(1 + L/2W) \tag{5.4}$$

The movement time of the two modes can be expressed as follows (the subscripts P and C indicate values for the *Precise* and *Coarse* modes, respectively):

$$MT_C = a_C + b_C \times \log_2\left(1 + \frac{A}{L}\right) \tag{5.5}$$

$$MT_P = a_P + b_P \times \log_2\left(1 + \frac{L}{2\,W}\right) \tag{5.6}$$

Parameters a_C, a_P, b_C and b_P depend on a number of task characteristics such as the user, the pointing devices, the pointing techniques, etc. If we make no assumption on these values and just assume that parameters a and b are the same in both phases, pointing time will be optimal if both phases have the same difficulty, i.e.:

$$\frac{A}{L} = \frac{L}{2\,W} \iff L = \sqrt{2A \times W} \tag{5.7}$$

We are interested in target-agnostic techniques, so we cannot know precisely A and W for each target. However if we use the worst-case values from Casiez et al. [43]'s formulae ($W = W_{min}$ and $A = A_{max}$), Equation (5.7) becomes:

$$L^* = \sqrt{2A_{max} \times W_{min}} \tag{5.8}$$

Equation (5.8) provides a safe estimate of the value of L when we have little knowledge about pointing amplitudes and target widths.

We can now compute the bounds of L for single-CD gain techniques.

(i) The coarse phase of a pointing task consists in reaching a target of width L from a distance A. Casiez *et al.*'s equations for $CD_{min,C}$ and $CD_{max,C}$ are:

$$CD_{min,C} = \frac{A_{max}}{OR_C} \qquad CD_{max,C} = \min(CD_{qmax,C}, CD_{lmax,C})$$

$$CD_{qmax,C} = \frac{L}{Device_{ps,C}} \qquad CD_{lmax,C} = \frac{L}{Hand_{res,C}}$$

We want $CD_{min,C}$ to be lower than to $CD_{max,C}$:

$$\begin{aligned} & CD_{min,C} < CD_{max,C} \\ \iff \quad & \frac{A_{max}}{OR_C} < \frac{L}{\max(Device_{ps,C}, Hand_{res,C})} \\ \iff \quad & L > A_{max}\frac{res_C}{OR_C} \text{ , with } res_C = \max(Hand_{res,C}, Device_{ps,C}) \end{aligned} \tag{5.9}$$

This lower bound maps the operating range of the coarse mode (OR_C) to the estimated maximum amplitude of the pointing tasks (A_{max}). It ensures that L is larger than the smallest cursor movement in coarse mode, and that the user can cover a distance of at least A_{max} without clutching in coarse mode. If L is smaller than this bound, users may have to either clutch to reach a target further than A_{max} away in coarse mode, or move the cursor by more than $L/2$ in precise mode.

(ii) The precise phase of a pointing task consists in reaching a target of width W from a distance supposedly smaller than $L/2$. Casiez *et al.*'s equations for $CD_{min,P}$ and $CD_{max,P}$ are:

$$CD_{min,P} = \frac{L}{2OR_P} \qquad CD_{max,P} = \min(CD_{qmax,P}, CD_{lmax,P})$$
$$CD_{qmax,P} = \frac{W_{min}}{Device_{ps,P}} \qquad CD_{lmax,P} = \frac{W_{min}}{Hand_{res,P}}$$

We want $CD_{min,P}$ to be lower than $CD_{max,P}$, i.e.:

$$\begin{aligned} & CD_{min,P} < CD_{max,P} \\ \iff \quad & \frac{L}{2OR_P} < \frac{W_{min}}{\max(Hand_{res,P}, Device_{ps,P})} \\ \iff \quad & L < W_{min}\frac{2OR_P}{res_P} \text{ , with } res_P = \max(Hand_{res,P}, Device_{ps,P}) \end{aligned} \tag{5.10}$$

This higher bound maps the smallest input movement of the precise mode (res_P) to the smallest target size (W_{min}). It ensures that, in precise mode, a user can reach a target as small as W_{min} within an area of diameter L without clutching. If L is greater than this bound, users may have to clutch to reach the target or the target may be too small to be reachable.

By combining Equations (5.1), (5.9) and (5.10) we can compute ID_{max}, the Fitts' index of difficulty above which, in theory, a dual-precision technique cannot be used without clutching:

$$\begin{aligned} & A_{max}\frac{res_C}{OR_C} < L < W_{min}\frac{2OR_P}{res_P} \\ \iff \quad & \frac{A_{max}}{W_{min}} < \frac{2OR_C \times OR_P}{res_C \times res_P} \\ \iff \quad & ID_{max} = \log_2\left(1 + \frac{2OR_C \times OR_P}{res_C \times res_P}\right) \end{aligned} \tag{5.11}$$

Again, the *max* subscript in ID_{max} represents the maximum index of difficulty for the *technique* being considered, while in A_{max} it represents the estimated highest amplitude of the *pointing task*. Note also that Equations (5.9) and (5.10) are applicable only if the corresponding mode is relative, since they are inferred from CD gain formulae. However, if we consider that an absolute mode has a fixed CD gain of 1 and an operating range equal to the size of the display, we can still compute ID_{max} using Equation (5.11).

Finally, we can compute the limit CD gain for each phase (if it uses a relative mode). In the coarse pointing phase, the goal is to cover amplitude A as fast as possible. Users should be able to cover A_{max} in a single gesture, i.e., within the operating range of the coarse mode, OR_C. In the precise pointing phase, the target is at a distance smaller or equal to $L/2$. This distance should be reachable within the operating range of the precise mode, OR_P. The limit CD gains for coarse and precise mode are thus:

$$CD_{C^*} = \frac{A_{max}}{OR_C} \tag{5.12}$$

$$CD_{P^*} = \frac{L}{2OR_P} \tag{5.13}$$

For any given technique, Equations (5.8), (5.12) and (5.13) give the key parameters, while Equations (5.9), (5.10), and (5.11) allow us to test that the dual-mode technique meets the constraints for the task. Note that Equations (5.9), (5.10), (5.11), (5.12) and (5.13) were based on Casiez et al. [43]'s formulae for single-CD gain techniques. It thus applies primarily to pointing modes using constant resolution functions.

5.1.2 Techniques

As defined in the previous chapter, we are interested in location-independent (Requirement 2), high-precision (R1) techniques that are single-handed (R3) and eye-free (R5).

In the following experiment, we compare the most efficient technique from our formative experiment, *GyroAcc*, to three techniques that implement our dual-precision framework: *Laser+Position* (Figure 5.2), *Laser+Gyro* (Figure 5.3) and *Laser+Track* (Figure 5.4). All three techniques use *RayCasting* as their *Coarse* mode, because it is known to be intuitive and does not require clutching. The value of L given by Equation 5.8 only depends on task parameters (A_{max} and W_{min}). It is thus the same for all techniques: 160 mm.

Laser+Position (Figure 5.2) is an adaptation of Vogel's free-hand *RayToRelative* technique [163]. Instead of detecting hand gestures for mode switching and clicking, we use the buttons of a wireless mouse. *Laser+Position* combines *RayCasting* for coarse pointing

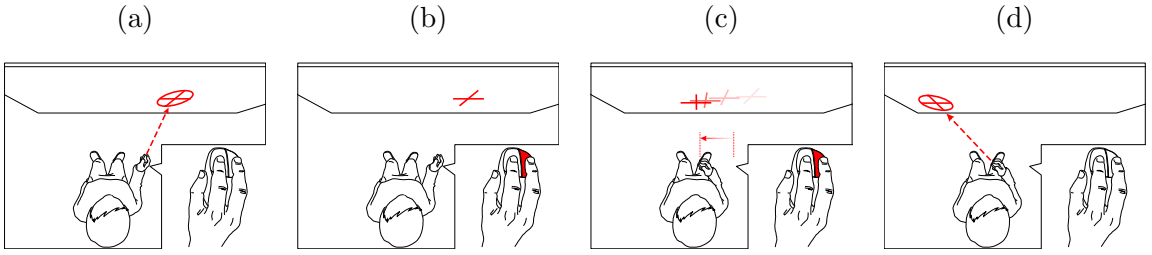


Figure 5.2: *Laser+Position*. *RayCasting* for coarse pointing (a). Switching to precise mode by pressing a button (b). Relative translational movements control cursor movements (c). Switching back to coarse mode by releasing the button (d). Click performed with left button.

and relative translational movements of the hand/device for precise pointing. In relative mode, the hand's translation is taken into account in a plane orthogonal to the orientation of the hand-held device at mode switch time (Figure 5.2-b). Precise pointing is activated by keeping a button depressed. A second button is used for clicking. Users can clutch in precise mode by releasing the first button and repositioning their hand quickly: If they press the first button again within less than 600 ms (tuned through pilot testing), the technique doesn't switch back to *Coarse* mode. Informal testing showed that an operating range of 300 mm for the *Precise* mode was large enough without causing too much fatigue. The theoretical limit of difficulty (Equation 5.11) for *Laser+Position* is approximately 18.2 bits, i.e., much higher than needed for a task that does not involve zooming or lenses ($A/W \approx 300,000$).

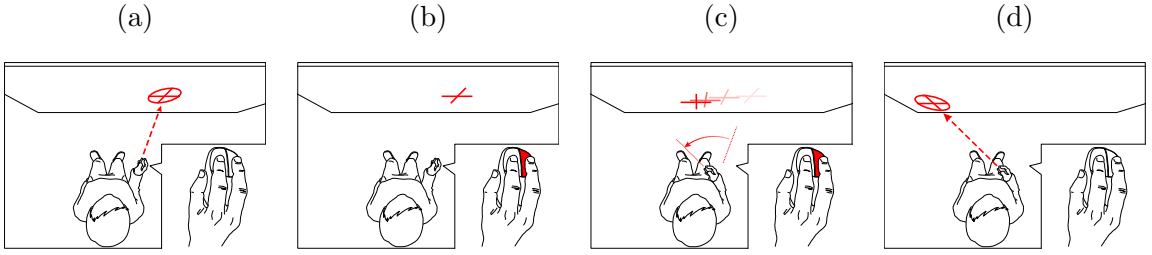


Figure 5.3: *Laser+Gyro*. *RayCasting* for coarse pointing (a). Switching to precise mode by pressing a button (b). Relative rotational movements control cursor movements (c). Switching back to coarse mode by releasing the button (d). Click performed with left button.

Laser+Gyro (Figure 5.3) combines *RayCasting* for coarse pointing and relative rotational movements for precise pointing. Compared to *Laser+Position*, which mainly involves upper limb segments (forearm up to shoulder) in relative mode, *Laser+Gyro* mainly involves the wrist and/or elbow and is potentially less tiring. Clutching, clicking and mode switching are identical to the *Laser+Position* technique. Our tests showed that an operating range of $\pi/2$ rad was large enough for the *Precise* mode while not causing too much fatigue. The theoretical limit of difficulty (Equation 5.11) for *Laser+Gyro* is approximately

19 bits.

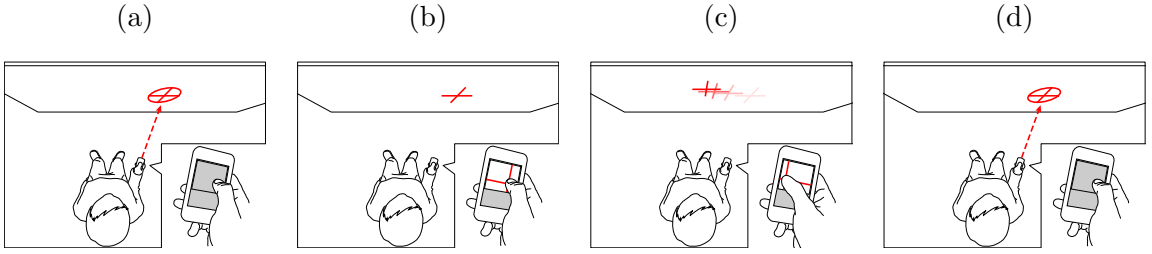


Figure 5.4: *Laser+Track*. *RayCasting* for coarse pointing (a). Switching to precise mode by touching the surface (b). Controlling cursor movements by moving the thumb (c). Switching back to coarse mode by releasing the thumb (d).

Laser+Track (Figure 5.4) combines *RayCasting* for coarse pointing and relative translational movements of the thumb on a touch-sensitive surface (PDA, smartphone, etc.) for precise pointing. The surface is divided into two areas (Figure 5.5): an upper zone (1) for tracking and a lower (smaller) zone (2) for clicking. Switching between the two zones can be achieved easily using proprioceptive information and does not require the user to look at the device after a short learning phase. Touching zone (1) switches to precise mode. Switching back to coarse mode only happens 300 ms (tuned through pilot testing) after the thumb has been released, thus enabling clutching. This clutching timer is reset each time a click occurs, so that users can stay in *Precise* mode if the click was a miss by going back to zone (1). To compensate for unintended finger movements at release time, we retrieve the coordinates of the click 200 ms before the finger-up event. The theoretical limit of difficulty (Equation 5.11) for *Laser+Track* is approximately 18 bits.

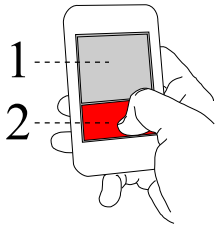


Figure 5.5: Zones used for the *Laser+Track* technique: (1) is for relative pointer movements and (2) is for clicking.

For all three techniques, the cursor is a crosshair surrounded by a circle (Figures 5.2-5.4). The circle's diameter is equal to the value of L for that technique. In *Precise* mode, the circle is decoupled from the crosshair and displayed as a ghost at the position where the cursor will be when the user switches back to coarse mode (Figure 5.6). This is because coarse mode is absolute and the cursor jumps when transitioning back from precise mode. The opacity of the ghost is inversely proportional to its distance to the crosshair so as to minimize visual interference, with a maximum value of 25%. This double cursor mechanism

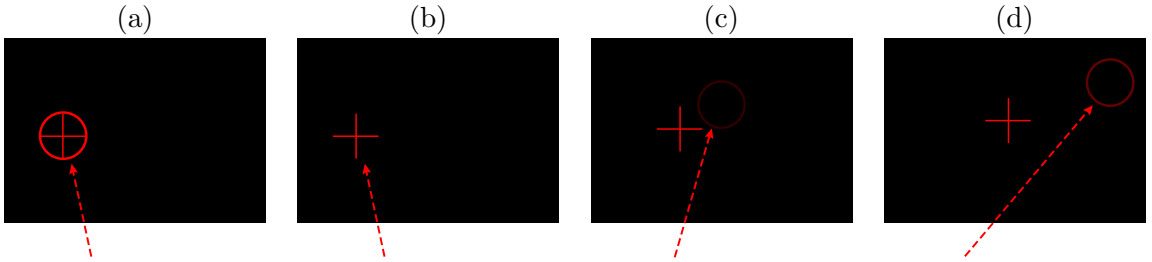


Figure 5.6: The dual-mode techniques cursor (a). The circle disappears when the user switches mode (b). In *Precise* mode (b to d), the circle is decoupled from the crosshair to indicate where the *Coarse* mode would be pointing. Opacity increases with distance to the crosshair (c and d).

has also been used in a more recent work by Debarba et al. [58], with a somewhat more elaborate opacity mechanism.

All dual-precision techniques support clutching in their relative mode (Figure 5.7). When the user releases the button or finger that controls relative mode, an animation shows how much time is left until the technique switches to absolute mode (Figure 5.7.b): four short strokes perpendicular to the cross' branches move from the center to the end of the cross. The user can use this time to clutch and continue moving in relative mode. When the four strokes reach the end of the cross, clutching times out and the technique switches to absolute mode.

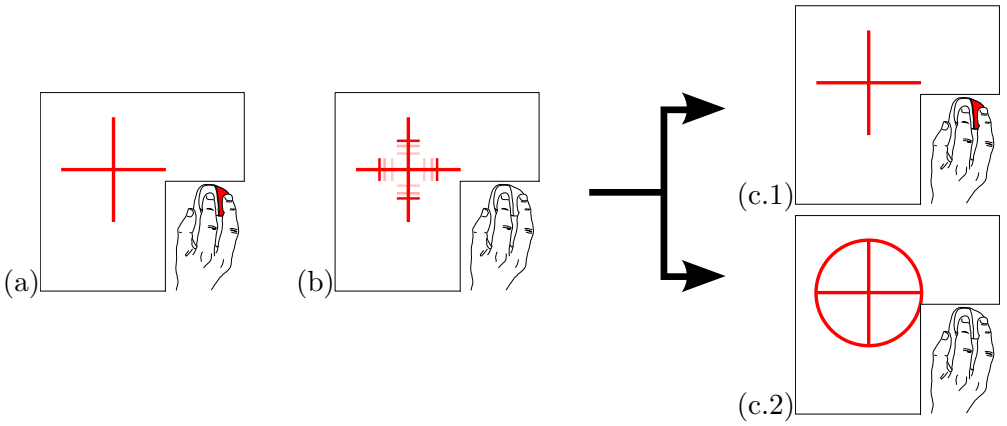


Figure 5.7: Visual feedback of the remaining time before a release event triggers a switch from precise mode (a) to coarse mode. When the user releases the mode activator (b), the technique does not switch back to coarse mode but moving lines appear on the branches of the cross to indicate the remaining time. If the user presses the mode activator before the moving lines reach the end of the cross (c.1), the technique stays in precise mode. If they do reach the end of the cross, the technique switches back to coarse mode (c.2).

5.2 Experiment: Dual-Precision Techniques

5.2.1 Participants, Apparatus and Task

The 12 participants of our formative user study (Chapter 4) served in this experiment, with at least a two-day interval between the two.

RayCasting was implemented as in Exp. 1 using the VICON motion tracker. Both *Laser+Gyro* and *Laser+Position* used a wireless mouse that was attached to a passive device equipped with retroreflective markers. The right button was used for mode switching and clutching, the left button for clicking. A mouse was also used for *GyroAcc*: the right button was used for clutching and the left button for clicking. *Laser+Track* used an Apple iPod Touch as a touch-sensitive surface.

The main factors were TECHNIQUE, AMPLITUDE and target WIDTH. The values of WIDTH were 30, 15, 7 and 4 mm. We checked that all participants could actually see all the targets. The values of AMPLITUDE were 637, 1912 and 3187 mm.

According to Casiez *et al.*'s literature review in [43], the highest Fitts' Index of Difficulty (ID) tested prior to their article was 7.6 bits in a desktop context (A= 30 cm, W= 1.5 mm). In this same article they report a univariate (1D) Fitts' experiment with IDs as high as 9 (A= 4.5 m, W= 9 mm) on a 25-ppi projected display. The highest ID tested in the following studies is 9.64 in bivariate tasks (2D) tasks, a difficulty never evaluated, to our knowledge.

For the three dual-mode techniques, L was computed using Equation (5.8): $L = 160$ mm. It was within the range defined by equations (5.10) and (5.9) (limit cases). The CD gain of each precise mode was computed using equation (5.13): $CD_{P*} = 3.07$ for *Laser+Track*, 0.51 for *Laser+Position* and 0.89 mm/deg for *Laser+Gyro*[3]. The maximum and minimum CD gains for the transfer function of *GyroAcc* were computed using equations (5.12) and (5.13): $CD_{C*} = 35.41$ mm/deg and $CD_{P*} = 0.89$ mm/deg [3]. We used the following logistic function:

$$CD(v) = 0.89 + \frac{34.52}{1 + e^{-16\times(v-.13)}} \tag{5.14}$$

[3]The CD gains of *Laser+Gyro* and *GyroAcc* are ratios between angular inputs (expressed in degrees) and linear outputs (in millimeters), hence the unit.

5.2.2 Design

We used a $4 \times 4 \times 3$ repeated measures within-subject design with three independent variables. For each participant, we grouped trials into 48 blocks, one per TECHNIQUE, AMPLITUDE and WIDTH. The presentation order for TECHNIQUE, AMPLITUDE and WIDTH was counterbalanced across participants using a Latin square. Each time a new TECHNIQUE began, participants had the opportunity to train with AMPLITUDE = 1912 mm and WIDTH = 7 mm. The actual trials started when the participant felt ready and their pointing time stabilized, i.e., when the task time difference between the slowest and fastest trials among the last four was within 30% of the mean of these trials.

To summarize, we collected 4 TECHNIQUE $\times$ 4 WIDTH $\times$ 3 AMPLITUDE $\times$ 5 replications $\times$ 12 participants = 2880 trials. The task was the same as in Exp. 1. For each trial, in addition to *Misses* and *Pointing Time*, we logged the time to acquire the dwell zone (*Recalibration*), the time to enter the target (*Reaching*), the time to perform the first click (*Clicking*), the time taken to perform the first switch to precise mode (*Switch Time*), the distance from the first mode switch to the target (*MS Distance*) and the number of times the cursor left the target (*Crossings*).

At the end of the experiment, participants were asked to rank the techniques and rate them for *Mental Effort*, *Accuracy*, *Speed*, *Fatigue*, *Comfort* and *Overall Easiness* on 5-point Likert scales.

5.2.3 Predictions

The *Precise* modes of *Laser+Gyro* and *Laser+Position* are similar and are controlled by the same limbs (forearm, wrist and hand). We expect them to have similar performance (prediction *P1*). However, *Laser+Position* should be more tiring since rotations are controlled more naturally than translations (prediction *P2*). *Laser+Track* is the only dual-mode technique whose *Precise* mode does not require moving the hand-held device, so we expect it to be faster for *Recalibration* (prediction *P3*). The minimum CD gain for *GyroAcc*'s transfer function had to be lowered compared with the first experiment because the targets are smaller. This means that there is a higher amplitude of CD gain to manage through the transfer function, which should negatively affect the performance of the technique (prediction *P4*). Finally, the value of L does not depend on the target WIDTH, so the time spent in *Coarse* mode should not be affected by the target WIDTH (prediction *P5*).

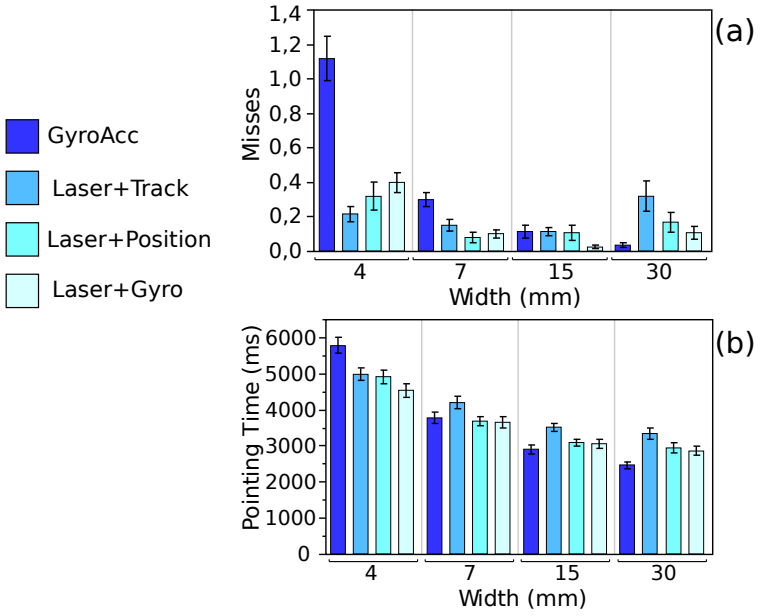


Figure 5.8: (a) *Misses* and (b) *Pointing Time* per TECHNIQUE × WIDTH.

5.2.4 Results

We analyzed the data using multiway ANOVAs, accounting for repeated measures using the REML procedure, and performed Tukey HSD post-hoc tests for pairwise comparisons. We used the mean for *Misses* and *Crossings*. As expected, the distributions of time measurements per condition were left-skewed, so we used the median. We verified that there was no significant effect of TECHNIQUE presentation order and observed that learning and fatigue effects were not significant. All reported results are significant at least at the $p < 0.001$ level unless noted otherwise.

Timeouts 3% of the trials were *TimeOuts*. There is a significant effect on the number of *TimeOuts* for TECHNIQUE ($F_{3,33} = 13.63$, $p < 0.0001$), AMPLITUDE ($F_{2,22} = 4.67$, $p = 0.0204$), WIDTH ($F_{3,33} = 25.44$, $p < 0.0001$) and TECHNIQUE×WIDTH ($F_{9,99} = 8.44$, $p < 0.0001$). As expected, larger amplitudes and smaller widths cause more *TimeOuts*. *Laser+Track* (mean 0.06) and *GyroAcc* (0.04) cause significantly more *TimeOuts* than *Laser+Position* (0.01) and *Laser+Gyro* (0). The effect increases with smaller widths: for $Width = 4$ mm, *Laser+Track* and *GyroAcc* caused significantly more *TimeOuts* than the two other techniques.

Misses There is a significant effect on *Misses* for TECHNIQUE ($F_{3,33} = 13.19$, $p < 0.0001$), WIDTH ($F_{3,33} = 43.21$, $p < 0.0001$) and TECHNIQUE×WIDTH ($F_{9,99} = 15.47$, $p < 0.0001$). As

expected, *Misses* increased with smaller widths. *GyroAcc* (mean 0.44) causes significantly more *Misses* than the other techniques (means from 0.16 to 0.26). The effect was even stronger with WIDTH = 4 mm (mean 1.36).

Clutching We found a significant effect on clutch for TECHNIQUE ($F_{3,33} = 281.01$, $p < 0.0001$), WIDTH ($F_{3,33} = 3.66$, $p = 0.0222$) and WIDTH×AMPLITUDE ($F_{6,66} = 2.24$, $p = 0.0497$). Smaller widths slightly increase the amount of clutching (the only significant difference is between WIDTH = 4 and WIDTH = 30 mm). *Laser+Track* (mean 2.27) causes significantly more clutch than *Laser+Gyro* and *Laser+Position* (1.01 and 0.98) which cause significantly more clutch than *GyroAcc* (0.02).

Crossings There is a significant effect on *Crossings* for TECHNIQUE ($F_{3,33} = 19.57$, $p < 0.0001$), AMPLITUDE ($F_{2,22} = 3.48$, $p = 0.0488$), WIDTH ($F_{3,33} = 79.66$, $p < 0.0001$) and TECHNIQUE×WIDTH ($F_{9,99} = 19.9$, $p < 0.0001$). As expected, smaller widths cause more crossings. The effect of AMPLITUDE is a bit surprising, with more *Crossings* for the medium amplitude. *GyroAcc* and *Laser+Track* (resp. 1.13 and 1.03) cause significantly more *Crossings* than *Laser+Position* and *Laser+Gyro* (resp. 0.7 and 0.61). *GyroAcc* causes almost twice as many *Crossings* than the second worst condition for WIDTH = 4 mm.

Dwell Time There is a significant effect on *Recalibration* for TECHNIQUE ($F_{3,33} = 7.35$, $p = 0.0007$), AMPLITUDE ($F_{2,22} = 65.42$, $p < 0.0001$), WIDTH ($F_{3,33} = 10.42$, $p < 0.0001$), TECHNIQUE×AMPLITUDE ($F_{6,66} = 3.83$, $p = 0.0024$) and AMPLITUDE×WIDTH ($F_{6,66} = 3.87$, $p = 0.0023$). As expected, *Recalibration* increases with AMPLITUDE. Recalibration takes significantly more time with *GyroAcc* (mean 1722 ms) than with all other techniques (1504 ms to 1417 ms), especially for the larger AMPLITUDES.

Reaching Time We found a significant effect on *Reaching* for TECHNIQUE ($F_{3,33} = 3.65$, $p = 0.022$), WIDTH ($F_{3,33} = 308.06$, $p < 0.0001$) and AMPLITUDE ($F_{2,22} = 134.3$, $p < 0.0001$). As expected reaching time increases with task difficulty. *Laser+Track* (mean 2710 ms) is significantly slower to reach the target than *GyroAcc* (2428 ms), with the other two techniques in between.

5.2.5 Pointing Time

There is a significant effect on *Pointing Time* for TECHNIQUE ($F_{3,33} = 13.09, p < 0.0001$), AMPLITUDE ($F_{2,22} = 71.14, p < 0.0001$), WIDTH ($F_{3,33} = 140.52, p < 0.0001$) and TECHNIQUE×WIDTH

($F_{9,99} = 8.56$, $p < 0.0001$). As expected, pointing time increases with task difficulty. *Laser+Track* (mean 4268 ms) is significantly slower than the other techniques (means from 3519 to 3847 ms). For WIDTH = 4 mm, *GyroAcc* is the slowest.

Modeling Pointing Time As described earlier, the task of pointing with a dual-mode technique consists of two phases, coarse and precise, with a mode switch in between. We expect the movement time of each pointing phase (MT_C and MT_P) to follow Fitts' law and assume that the mode switch takes a constant time MT_S dependent on the technique. We obtain the following model for the global pointing task time (MT_T):

$$MT_T = MT_C + MT_S + MT_P \tag{5.15}$$

$$= (a_c + MT_S + a_p) + b_c \, log\left(1 + \frac{A}{L}\right) + b_p \, log\left(1 + \frac{d}{W}\right) \tag{5.16}$$

$$= a + b_c \, ID_C + b_p \, ID_P \tag{5.17}$$

Remember that the value $d = L/2$ used in Equation 5.4 is a higher bound. In our analyses we observed that lower values for d fit our data better. We used $d = L/8$ and computed the following regressions and goodness of fit:

Model	Technique	Parameters	r^2	AICc
Our model	*Laser+Gyro*	$560 + 424 \times ID_C + 1070 \times ID_P$	.94	172
	Laser+Position	$349 + 446 \times ID_C + 1183 \times ID_P$	.96	173
	Laser+Track	$1393 + 385 \times ID_C + 986 \times ID_P$	.94	174
Fitts' law	*Laser+Gyro*	$-60 + 512 \times ID$	.87	179
	Laser+Position	$-328 + 564 \times ID$	.87	181
	Laser+Track	$779 + 475 \times ID$	.87	177

The goodness of fit ($r^2 = .94$ for *Laser+Gyro*, .96 for *Laser+Position* and .94 for *Laser+Track*) is consistently better than when modeling the global task time with Fitts' law ($r^2 = .87$, .87 and .87 respectively). Since this could be due to our model having two parameters instead of one, we computed the Akaike Information Criterion with correction (AICc)[4][1]. The results confirm that our model provides a better fit. We also note that the intercepts for two of the Fitts models are negative, and quite large in the case of *Laser+Position*, which typically indicates a problem.

Regarding our model, the constants in the regressions confirm that *Laser+Gyro* and *Laser+Position* are similar (prediction *P1*), with a slight advantage for *Laser+Gyro* in the precise phase. They also show a high intercept (a) for *Laser+Track*, indicating that

[4]This criterion measures the relative goodness of fit of a statistical model and assesses overfitting, i.e., increasing the number of free parameters to improve the goodness of the fit.

participants had problems with this technique. We also observe that b_p is consistently higher than b_c, which contradicts our assumption that Fitts' law parameters a and b are the same in both phases (Section 5.1.1, page 125). Users seemed to have more difficulty with the *Precise* mode, maybe due to mental readjustment caused by the change of input modality from *Coarse* to *Precise*. This discrepancy could affect the computation of L, which could itself affect overall performance. It is also consistent with the empirical value $d = L/8$ that provided the best fit with our model (Equation 5.17): a smaller L would probably have led to more time spent in coarse mode and less time spent in precise mode. However such adjustments probably depend on the techniques used in each mode and on individual user performance.

5.2.6 Mode Switches

These analyses were only performed on *Laser+Gyro*, *Laser+Position* and *Laser+Track*.

In some cases and despite our instructions, participants chose not to switch modes and perform the task in *Coarse* mode only (5 %). We call this behavior "coarse-only". We observe a significant effect of Width ($F_{3,33} = 4.13$, $p = 0.0136$), Amplitude ($F_{2,22} = 5.8$, $p = 0.0095$) and Width×Amplitude ($F_{6,66} = 5.12$, $p = 0.0002$) on the number of such trials. Unsurprisingly, participants performed more pointing tasks "coarse-only" with the largest targets Widths (30 mm) than with the two smallest ones (4 and 7 mm), Width=15 remaining not significantly different in between. More interestingly, participants finished more trials "coarse-only" with the smallest Amplitude (637 mm) than with the two other Amplitudes. Finally, the combination ($Amplitude = 637, Width = 30$) caused significantly more of this behavior than any other condition. In the following analyses of mode switch occurrences we removed the trials with no mode switch. *Switch Time* and *MS Distance* are measures related to the first mode switch. In order to discard data from erroneous first mode switches we also excluded the trials with *MS Distance*s greater than $3 \times L$ (13 %).

We observed no significant effect of Technique on *Switch Time*. Unsurprisingly, *Switch Time* increases significantly with Amplitude ($F_{2,20.86} = 147.96$, $p < 0.0001$). However we also found a significant effect of Width ($F_{3,32.33} = 8.84$, $p = 0.0002$), in contradiction with prediction *P5*. This is surprising since the size of the *Coarse* mode "target", L, does not depend on the actual target Width. The post-hoc test reveals that users took more time to switch modes with target Widths of 4 and 7 mm than of 15 and 30 mm.

Similarly, we did not observe any effect of Technique on *MS Distance*. However, we did observe one for Width ($F_{3,32.69} = 4.72$, $p = 0.0076$) and Amplitude ($F_{2,21.99} = 9$, $p = 0.0014$).

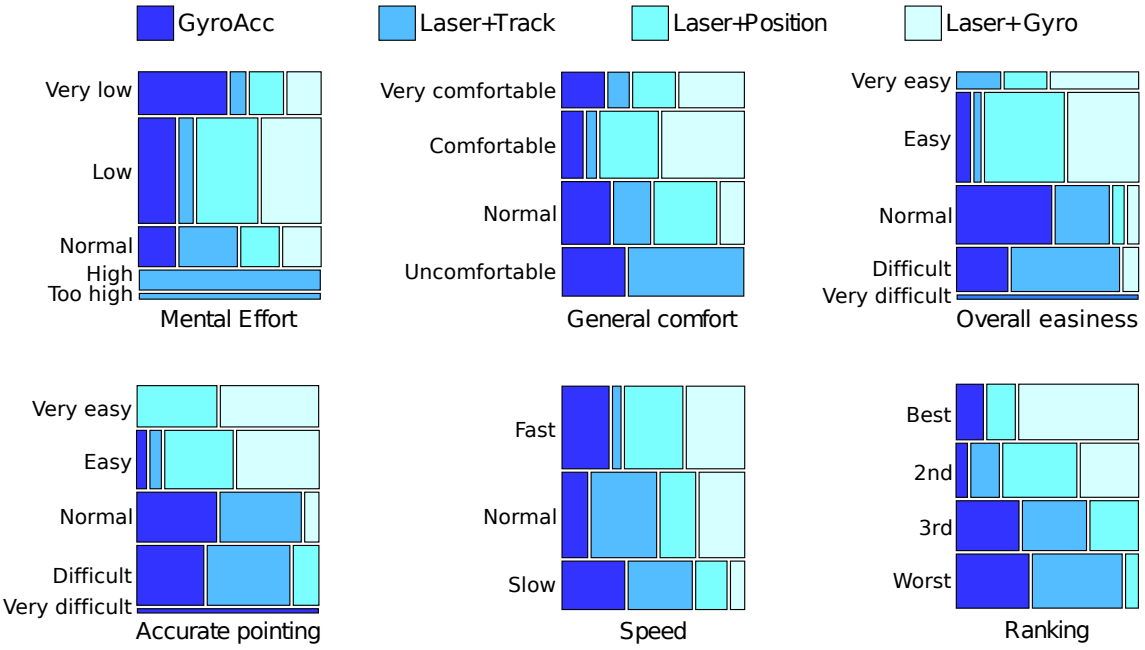


Figure 5.9: Mosaic plot of the subjective results for Exp. 2. The height of a box represents the number of participants that used this grade or rank overall (regardless of the technique). The width of a box represents the proportion of a particular grade or rank for a given technique.

Participants switched modes further away from the target with AMPLITUDEs of 3187 and 1912 mm than with the smaller AMPLITUDE 637 mm. Participants also switched modes further away with the largest target (30 mm) than with the two smallest (4 and 7 mm), WIDTH=15 remaining not significantly different in between.

We computed the time spent in precise mode as *Precise Time* = *Pointing Time* - *Switch Time* and found a significant effect of TECHNIQUE ($F_{2,19.19} = 11.57$, $p = 0.0005$) and WIDTH ($F_{3,28.33} = 81.48$, $p < 0.0001$). Unsurprisingly, *Precise Time* with targets of 4 mm were slower than targets of 7 mm, which were slower than targets of 15 and 30 mm. More interestingly, *Laser+Gyro* caused shorter *Precise Time* than *Laser+Position* and *Laser+Track*.

Finally, we observed that *Switch Time* follows Fitts' law with the model $Switch\ Time = a + b \times \log_2\left(1 + \frac{Amplitude}{L}\right)$ with goodnesses of fit equal[5] to 1 for *Laser+Gyro* and *Laser+Position* and 0.99 for *Laser+Track*.

5.2.7 Subjective results

A Pearson χ^2 test shows that there is a significant effect on *Mental Effort*, *Accuracy*, *Comfort*, *Easiness* and *Ranking* for TECHNIQUE. 6 participants graded mental effort for *Laser+Track* as *High* or *Too high*, and all 12 graded all other techniques *Normal* (3) or below. 6 participants graded precise pointing with *Laser+Track* and *GyroAcc* as *Difficult* or *Very difficult* and 9 graded it *Easy* or *Very easy* with *Laser+Gyro* and *Laser+Position*.

10 participants graded *Laser+Track Uncomfortable* or *Very uncomfortable*. 10 and 7 participants respectively graded *Laser+Gyro* and *Laser+Position* as *Comfortable* or *Very comfortable*, partially supporting prediction *P2*. 10 participants graded *Laser+Position* and *Laser+Gyro* as *Easy* or *Very easy* and 10 graded *Laser+Track* as *Normal* (3) or below. Finally, 8 participants preferred *Laser+Gyro* overall, 2 preferred *Laser+Position*, 2 preferred *GyroAcc* and none preferred *Laser+Track*. Overall, these results are consistent with the quantitative analysis.

5.2.8 Discussion

5.2.8.1 Coarse and Precise phases

Based on our results, we hypothesize that users adapt their *Coarse* pointing phase depending on the expected difficulty of the *Precise* phase in order to optimize general pointing time. Indeed, despite being functionally unrelated to target WIDTH, the *Coarse* phase lasted longer with, and ended closer to, the smaller targets. We suppose this behavior was intended to make smaller targets easier to acquire by lowering the amplitude of the *Precise* phase. Participants also switched modes closer to the target when they had smaller AMPLITUDE to go through, possibly capitalizing on a normally shorter *Coarse* phase to switch modes closer to the target.

Despite comparable *Switch Time* and *MS Distance*, participants spend less time in *Precise* mode with *Laser+Gyro* than with the other two dual-precision techniques. We hypothesize this is due to the angular control of the precise mode of *Laser+Gyro*, which is closer to the *Coarse* mode (ray-casting) than with *Laser+Position* and *Laser+Track*. Similarly, participants sometimes avoided switching mode with small AMPLITUDE and large target WIDTHS. We hypothesize they tried to minimize pointing time by neither switching modes nor adapting their input to the *Precise* phase control.

[5] Rounded at two decimals.

5.2.8.2 Techniques

GyroAcc was generally perceived as imprecise, which is consistent with the quantitative results. More precisely, despite the fact that it was the fastest technique to reach the target, contradicting prediction *P4*, it caused more crossings and clicks outside the target than the other techniques, especially with very small targets. This indicates that users could easily perform fast, coarse pointing but that stabilizing the cursor over a small target was too difficult. *GyroAcc* was also the worst for *Recalibration*, i.e., the offset between the hand-held device and the cursor often became quite large. This means that in real situations users will probably have to recalibrate the position of the cursor to the input device, i.e., clutch and loose time when going back from a precise pointing task to a coarse one. However, *GyroAcc* performed well in both speed and accuracy for WIDTHS of 7 mm and higher, indicating that pointing tasks of this difficulty can be performed without explicitly trading speed for precision. From both the quantitative and qualitative results, we suggest that *GyroAcc* be used when targets are consistently larger than 7 mm (about 18 pixels on a 100ppi display).

Laser+Track was the slowest and least preferred technique overall. Despite having the same coarse mode as the other dual-mode techniques, it was the slowest for *Reaching* and caused the most clutch; it also caused many *Crossings*, meaning that the *Precise* mode is neither precise nor fast enough to compete with other dual-mode techniques. In addition, it was not better for recalibration time, contradicting prediction *P3* (we expected that its precise mode would cause smaller displacements of the absolute cursor). These results are consistent with participants' opinion: hard to use, imprecise and uncomfortable. Part of this may be due to the following problem: 5 participants reported that lifting the finger from the tracking zone of the device (zone 1) caused a loss in precision despite our finger-release adjustment, and that they would have preferred a physical button. An improvement could be to implement the precise mode of *Laser+Track* with an input device that couples finger tracking with physical actuators, such as Apple's Magic Mouse or Microsoft's Touch Mouse.

Laser+Gyro and *Laser+Position* had very similar results, supporting prediction *P1*. They caused almost no *TimeOuts*, meaning that they were able to withstand very difficult tasks such as those in the experiment. They were also the most stable (best for *Crossings*) and fastest techniques overall. They were both perceived as precise, comfortable and easy to use. *Laser+Gyro* was the preferred technique and had a slight advantage when pointing at very small targets, which is consistent with prediction *P2*.

It is interesting to note the sudden decrease in performance of *GyroAcc* for the smallest targets. This supports our hypothesis (Section 4.3.6, page 120) that there is a limit to

the output range of continuous transfer functions above which the steepness of the curve makes it difficult to control the transition between fast and precise movements. While this particular theory should be evaluated further, we can at least observe that dual-precision techniques do not have such a limitation since the transition between modes is discrete. Equation (5.11) shows that, in most cases, the theoretical limit of difficulty of dual-mode techniques (18.97 for *Laser+Gyro*, i.e., targets of 0.01 mm on a 5.5-meter wall display) is higher than human visual acuity.

5.2.9 Comparing single- and dual-mode techniques

In order to demonstrate the effectiveness of dual-precision techniques with respect to single-mode ones, we compared the techniques in Exp. 1 and Exp. 2 for *Pointing Time* and *Misses* for the conditions common to the two experiments: AMPLITUDE = 3187 mm and WIDTH $\in \{7, 15, 30\}$ mm. For both measures we found a significant effect ($p < 0.0001$) of TECHNIQUE, WIDTH and TECHNIQUE×WIDTH. Measures for *GyroAcc* are reported separately as *GyroAcc1* for Exp. 1 and *GyroAcc2* for Exp. 2 because the more challenging conditions in Exp. 2 called for different transfer functions in order to make smaller targets reachable (see Sections 4.3.2 and 5.2.1).

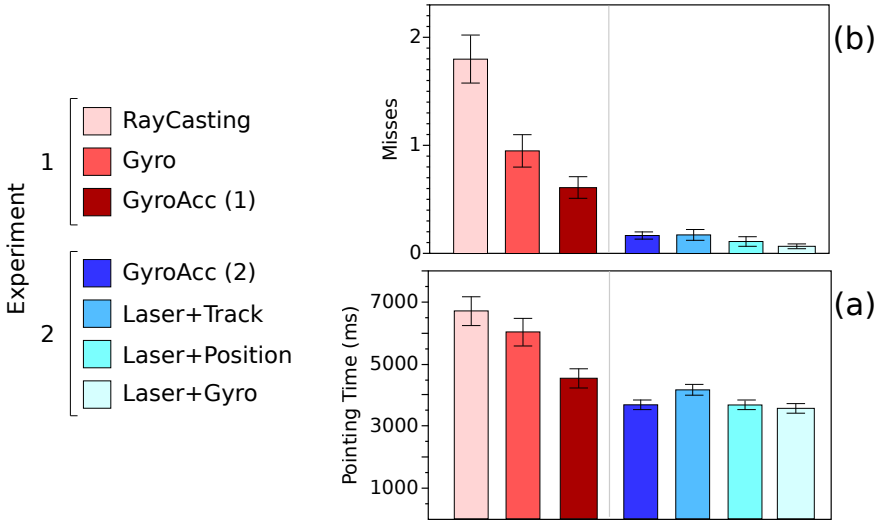


Figure 5.10: *Pointing Time* (a) and *Misses* (b) for all TECHNIQUEs and for similar difficulties.

As expected, *Misses* and *Pointing Time* increase significantly when WIDTH decreases. As shown in Figure 5.10.a, *RayCasting* and *Gyro* are significantly slower than all other techniques (means 6846 and 6018 ms). The next slowest technique is *GyroAcc1* (4527 *ms*), although *Laser+Track* (4152 ms) is not statistically different from it and from the remaining techniques. The pattern for misses is similar (Figure 5.10.b). *RayCasting* (mean 1.84)

causes the most misses, and *Gyro* and *GyroAcc1* (means .95 and .61) cause significantly more misses than the techniques in Exp. 2. For WIDTH = 7 mm, the techniques in Exp. 2 significantly outperform those in Exp. 1 for both time and error. For WIDTH = 15 mm, *RayCasting* and *Gyro* significantly underperform compared to all other techniques.

These results show that techniques that either are absolute or use a constant transfer function (CD gain) are less efficient than dual-mode techniques or those using dynamic gains, even for easily reachable targets. It may be surprising that dual-mode techniques with an explicit mode-switch perform better than *RayCasting* or *GyroAcc* for WIDTH = 30 mm since Exp. 1 showed that *RayCasting* had enough precision to reach those targets. Based on the error rates of both experiments and on the *Crossings* measured in Exp. 2, we suggest that switching to the *Precise* mode makes the click more stable: it reduces the impact of the tremor caused by clicking as well as the crossings and clicks outside the target.

5.3 Implications for design

Pointing techniques focus on increasing the speed and precision of a single task, target acquisition. However, as for desktop environments or mobile devices, interacting with a large wall display requires users to perform additional actions. For example, it must be possible to select a target once it has been acquired, to specify an arbitrary area of the display, to invoke and interact with visual widgets, or to navigate a large scene by panning and zooming. Integrating such actions with a mid-air pointing technique requires careful design in order to retain the benefits of the pointing technique while keeping the other interactions simple and effective. The rest of this section analyzes several approaches for this integration, as well as the compatibility of our dual-mode techniques with existing pointing facilitation techniques.

5.3.1 Integration with other basic interactions

Some pointing techniques embed their own specific action triggers, e.g. the VisionWand gestures [41]. For others, such as ray-casting, the selection mechanism depends on the devices at hand such as wireless mice, Wii remotes, etc. With the latter, interaction designers may still have to define how to perform these vocabularies of actions with sometimes limited input channels, while not hindering performance and user preference.

Invoking commands before, during or after pointing in mid-air can be achieved by a variety of techniques:

- *Hand positions* [21, 163] can be mapped to a set of commands. Current vision-based approaches are easy to set up and allow large vocabularies, however the recognition algorithms are not perfect, requiring techniques to disambiguate or correct input. Moreover, changing the position of the hand can hinder pointing precision unless a separate input channel, such as a hardware button, is used to segment pointing from position detection. These approaches also require the interaction space to be fully covered by the cameras, with high-enough resolution to detect hand positions accurately. In the context of large high-resolution displays where multiple users may walk around the interactive room, a full yet precise coverage of the whole area can be hard to achieve, especially because of visual occlusion.

- *Hand or cursor gestures*, such as pigtails [84], also support large vocabularies that can help remembering the corresponding command, e.g., a pinching gesture for rescaling. However users must learn the gestures and the recognition algorithms must be sufficiently accurate. In addition, performing such gestures after the pointing movement requires additional time and can be tiring when performed many times with a stretched arm. As with the previous approach (hand positions), this approach also typically requires a separate channel, such as a hardware mode switch, to segment gestures from pointing movements.

- *Dwelling* [125] can be used to trigger several commands, e.g. depending on the duration of the dwelling action. It uses no additional input channel and causes minor hand tremor, however the vocabulary is more limited than for the other approaches. It also requires extra time and can sometime be confused with a user hesitation or slow movement, the latter being common when pointing at very small targets. Choumane et al. [52] addressed this problem and propose *Buttonless Click*, a method based on trajectory and kinematic gesture analysis to discriminate selection, pick and release tasks.

- *Crossing* [4] uses the trajectory of the movement to trigger targets, and therefore does not require another input channel. However, crossing techniques are by definition target-aware since the system needs to know the position of the targets to determine which ones, if any, are crossed. Crossing is therefore outside the scope of this paper.

- *Mobile touch screens* such as smartphones can accommodate a number of virtual buttons, sliders and other widgets controlled with the thumb. However the number of such widgets that can be reached easily without looking at the device is limited. In addition our results show that interacting on a hand-held touch screen device whose location is tracked for pointing causes pointing tremor.

- *Physical devices*, e.g., mice and remote controllers such as the WiiMote, feature

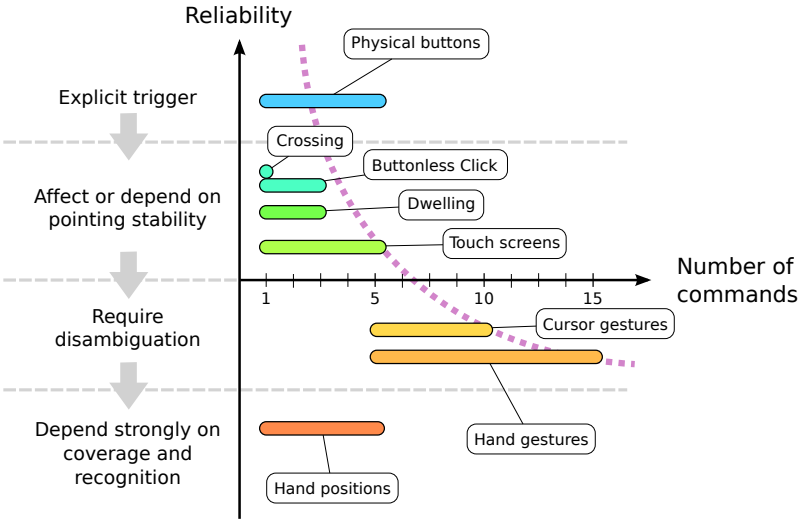


Figure 5.11: Trade-off between reliability and number of commands for one-handed selection and other pointing-related commands. Number of commands are rough estimates. The purple dotted line represents (informally) the corresponding front.

multiple physical buttons and wheels that take very little time to operate and work reliably anywhere in the room. However some of these devices can be cumbersome to hold, especially for long periods of time, and they prevent the use of other devices in the pointing hand.

We observe a trade-off between expressiveness and reliability for these different approaches, as illustrated in Figure 5.11: dwelling and physical buttons cause little to no pointing tremor and are very reliable, but they accommodate a limited vocabulary of commands. Hand and thumb gestures can offer many more commands but depend on the accuracy of the recognition algorithms. They also cause hand tremor and some of them require an additional input channel to distinguish between pointing and triggering commands.

In most of our techniques we chose to use mice buttons for target acquisition and mode switching because our studies were performance-driven: buttons generate very little hand tremor and can be triggered quickly. Because they are so widely used, mouse buttons require no learning and can be mapped to commands as in desktop environments. Since one button must be dedicated to the mode switch, applications that require dual-mode pointing and more than two other discrete commands cannot be controlled with a regular 3-button mouse without using one of the above techniques[6]. In order to give access to larger command sets, interaction designers can use dual-precision pointing techniques with any of the above trigger mechanisms, which they should select according to the trade-off between speed, precision and size of the command set.

[6] Note however that many mice on the market, designed for games, feature more than three buttons.

5.3.2 Compatibility with pointing facilitation mechanisms

As for most target-agnostic pointing techniques, dual-mode techniques can be combined with pointing facilitation mechanisms, i.e. techniques that attempt to decrease movement time either by reducing the task amplitude, increasing the target width, or a combination of both. Zoom-and-pick [62] uses hand-held projectors to make accurate selections by locally distorting the region around the cursor. Foehrenbach et al. [61] tried to enhance freehand pointing with tactile feedback but had limited success. The WorldCursor [174] uses a special wand and a laser projector that provides feedback about where the system thinks the user is pointing. Both Frees et al. [67] and König et al. [103] transpose the idea of dynamic control-display gain adaptation to ray casting, the former – called PRISM – in the context of 3D object selection in immersive virtual environments, the latter for large displays.

Except for Drag-and-pop [22] and the Vacuum [29], most target-aware pointing facilitation techniques have been designed for the desktop [17]. Since they often make few assumptions about the physical input device used for pointing, they can be adapted to facilitate distant pointing. For example, the Bubble cursor [71] uses proximity to differentiate between potential targets. Combined with a dual-precision technique, it could prevent the use of the *Precise* mode in areas with few potential targets, thus avoid mode-switch and mental adjustment times. Other techniques that could be considered include Expanding targets [120], the Adaptive Hybrid Cursor [146] and DynaSpot [47]. The present work focuses on generic, target-agnostic pointing techniques, thus all of our resulting techniques could be coupled with efficient pointing facilitation mechanisms.

5.4 Conclusion and Future Work

In this chapter we investigated a family of dual-precision pointing techniques. We introduced a model for predicting pointing time, a method to calibrate the techniques and formulae to compute the theoretical limits for their usage. We implemented three one-handed techniques and thoroughly detailed their design and settings. We compared their performance, error rate and user preference, and showed that targets as small as 4 mm (16 px) can be acquired reliably when standing 2 meters away from the display. In comparison, the smallest targets studied in previous work were at least four times as large. Our results show that dual-mode techniques perform better than classical techniques for targets smaller than 30 mm (15 mm for *GyroAcc*), and that a good precise mode is crucial for both performance and user acceptance. The best techniques combined ray-casting with either device rotational (*Laser+Gyro*) or translational (*Laser+Position*) movements

in precise mode, with a slight participant preference for *Laser+Gyro*.

Finally we presented recommendations for the integration of our pointing techniques with other basic interactions such as clicking, dragging and invoking widgets, as well as with existing pointing facilitation techniques.

All the dual-precision techniques developed and tested in this chapter had similar modes: *Coarse* pointing was performed using ray-casting and *Precise* pointing was using a constant CD gain. However ray-casting is perspective-dependent and thus looses accuracy with high eccentricities. Similarly, as shown in the experiment in Chapter 4, Pointer Acceleration performs better than constant CD gains for identical input channels. One could thus wonder if precise mode could not be made more efficient by using Pointer Acceleration. I will investigate these questions in the next chapter.

This chapter studied one-handed pointing techniques that favor the *Maximizing limb usage* strategy of the *Task Allocation* taxonomy from Chapter 1, on the basis that in both combinations of techniques studied in Part I, separating pointing from other simultaneous interactions performed best. The next chapter will be dedicated to another strategy of this taxonomy, *Factorizing task allocation*. It will explore touch-based pointing techniques performed from hand-held devices, such as tablets and smartphones, that allow other widgets to be displayed on the same input surface.

Chapter 6

Dual-Precision Techniques on Touch Devices[1]

6.1 Introduction

Touch input on a tablet or a smartphone screen has been evaluated and used for pointing in a number of previous studies [119, 140, 33]. Some of these works have successfully investigated mid-air pointing with a touch-screen for large displays but were often limited to low-resolution ones. For example, ARC-Pad [119] enables both absolute and relative pointing on a large display via the touch-screen of a mobile device. Alternatives include using sensors on the mobile device, such as tilt [140] or the phone's camera [33]. With Touch Projector [33], users aim the camera at the display of interest and manipulate its content by touching and dragging content on the screen. The techniques we have designed use some of these approaches for remote control of the cursor.

In Chapter 4 we showed theoretically that a smartphone screen does not provide enough input resolution to acquire small (visible) targets on an ultra-high-resolution wall-sized display, at least with a static CD gain; this is also true for tablets: even the latest tablet displays feature almost 45 times fewer pixels than a display such as WILD (Appendix A). We intend to use the smallest possible input area from a tablet or smartphone to allow for other interactions, so we expect that users will need several levels of CD gain. To address this problem we investigate three ways to augment the input expressiveness of a touch area: using additional input channels, using two modalities for touching the handheld, and using tuned velocity-based transfer functions.

[1] A subset of this chapter has been published at the CHI'13 conference (not yet in the proceedings)).

Figure 6.1: Interacting with a wall-sized display using a tablet that features both control widgets and an area dedicated to high-precision remote pointing (top-left).

6.2 Techniques

We consider contexts of use where the primary purpose of the handheld device is to accommodate widgets for the advanced control of objects selected via pointing. The physical area dedicated to pointing on the device, or *pointing zone*, should therefore not be too large. However, if too small, pointing will be inefficient. Our goal is to identify the best trade-off between screen real-estate allocation and good pointing performance.

In the previous two chapters, we investigated two approaches to provide fast and high-precision pointing. Pointer Acceleration functions dynamically adjust the control-display (CD) gain based on the velocity of the user's movements. Dual-precision pointing lets users explicitly switch between different CD gains, typically between a *Coarse* mode that allows fast repositioning of the cursor across large distances, and a slower but more *Precise* mode that users can engage when they want to adjust the cursor position to acquire a very small target.

We showed in these chapters that the main challenge when designing dual-precision techniques is to seamlessly integrate the two modes so that the mode switch minimizes cognitive and/or motor costs. All the techniques presented in this chapter assign the *Precise* mode to single-finger drag gestures performed with the dominant hand in the pointing zone. Given our Requirement 2 (**Location independence**), and since the non-dominant hand will typically hold the device [167] in a way considered comfortable by the user, this leaves two main options for the *Coarse* mode:

- Use the dominant hand in both modes but in two different configurations that are easily differentiable by the system;
- Use a different body part for the *Coarse* mode, that does not impair the control of the *Precise* mode.

We experimented with several options. Eventually, we identified a subset of two viable assignments through iterative design, prototyping, piloting and tuning: double-finger drag gestures with the dominant hand in the handheld's pointing zone, and head orientation. Both approaches are detailed below.

6.2.1 Two-finger pad-based coarse pointing

This approach is inspired by ARC-Pad [119] (Absolute+Relative Cursor Pad), a pointing technique that provides users with an absolute and a relative pointing mode. A typical ARC-Pad pointing task is composed of a tap (press-release) on a touch-sensitive handheld device, followed by a dragging gesture. The tap gesture coarsely positions the cursor on the large display according to an absolute mapping of the handheld device's surface to the large display. The following drag gesture is interpreted as relative movements of the cursor to adjust its position. The original technique was designed for, and evaluated on, much lower-resolution large displays than those considered here. It proved very difficult to use on ultra-high-resolution walls, mainly because in that context the absolute mapping is far too imprecise and often requires either several attempts (taps) in coarse mode to move the cursor close enough to the target, or several relative-mode drag gestures to adjust the cursor's position, causing much clutching in the second phase. We thus adapted the original technique so that users can adjust the location of their cursor in absolute mode. We named this variation of the original technique ARC-Pad$_2$.

ARC-Pad$_2$ distinguishes between absolute and relative pointing by the number of fingers involved in performing the pointing gestures rather than by the type of gesture performed. A single-finger drag gesture controls the cursor in relative mode (as before); a drag gesture performed with two conjoined fingers [99] is interpreted as absolute positioning of the cursor. As opposed to the original method that relied on tap gestures for absolute mapping, users can now adjust the cursor position in absolute mode by dragging with the two fingers, and then switch to relative mode for more precise, relative adjustments of the cursor position. The switch from absolute to relative mode is triggered whenever at least one of the fingers is lifted from the pad's surface. This means that users can either lift a single finger at the current location and continue dragging, or they can lift both fingers, adjust their hand position relative to the pad, and touch anywhere on the surface with a single

finger. This second option can be very useful when pointing at a target near the edges of the display, as it allows users to initiate relative drag gestures from the center of the pointing zone, in any direction.

We considered the symmetric approach, i.e. one finger for coarse pointing and two for precise pointing, but preliminary tests showed that lowering a second finger in order to switch to precise mode may cause the finger to fall out of the input area when the target is near the display borders. This has either no effect if the finger did not reach an interaction widget outside the pointing area, or it can trigger unwanted commands. In both cases users may wonder why lowering their finger did not have the expected effect and look at the input device, thus loosing their focus (going against Requirement 5). Leaving the input area when using two fingers is less likely to happen in *Coarse* mode since it uses an absolute mapping: users always have a feedback of their fingers' location on the input area through the position of the cursor on the display.

6.2.2 Head-based coarse pointing

The second viable approach that we identified for *Coarse* control makes use of the natural head movements that occur when remotely pointing at targets on a wall.

Object selection is often preceded by a visual search when the target is not located in the user's immediate field of view. Head orientation provides a good approximation of where a user is looking [158, 66, 130]. Head movements can also be exploited in conjunction with any positioning device used in the environment [100, 130] and has been shown to support a variety of interaction techniques [100, 123, 102]. Our approach integrates head motion with cursor selection. It was inspired by Ashdown et al. [8], who use head orientation for positioning a cursor on the monitor of interest, thereby reducing mouse trips in a multi-monitor setup. We provide a framework for designing target selection techniques that use head orientation, based on the results of a study of head and body orientations when acquiring targets in mid-air on a wall-sized display (Appendix C).

Stellmach and Dachselt [159] took a similar approach using gaze instead of head orientation. They introduce four techniques that follow a common design guideline: *gaze suggests, touch confirms*. While their results are promising, they did not compare their design to existing techniques, and only evaluated it in desktop environment. Wall-sized displays feature much greater sizes than desktop screens, regardless of pixel density. While desktop screens can often be skimmed through with minimal movement other than the eyes', wall-sized displays may require much larger head rotations. Confirming prior work [92, 178], we observed in the study described in Appendix C that users consistently stabilize their

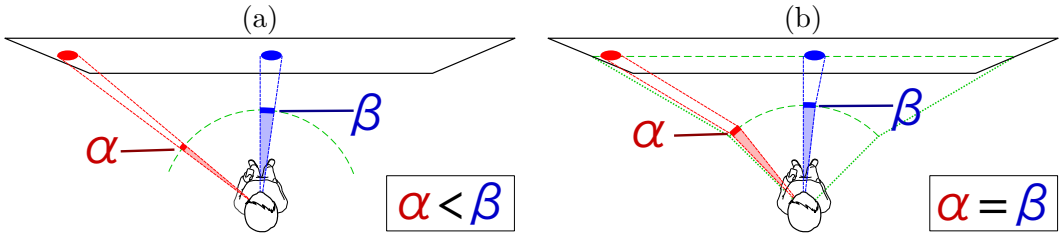


Figure 6.2: Varying angular motor size with ray-casting (a) can be fixed using an indirect absolute mapping (b).

gaze and lock it on the target while acquiring it. While tracking the users' gaze is not a practical option in our context, we found head orientation to be a good coarse indirect indicator of where users are looking[2], and thus a good predictor of the rough location of the next target on the wall display.

Our technique controls the cursor's location in *Coarse* mode through an absolute but indirect mapping of head orientation to cursor position on the wall. We favored an indirect mapping over direct ray-casting (or laser pointing) for two reasons. First, being perspective-based, this method would have caused targets of the same size on the display to have noticeably different *motor* sizes depending on their location with respect to the user's physical position, as illustrated in Figure 6.2-a (this effect gets amplified as users get closer to the display). We also wanted to optimize the users' input operating range, within the limits of comfortable neck positions.

As illustrated in Figure 6.2-b, we addressed these issues by mapping a location-independent, fixed-size angular operating range centered on the orthogonal projection of the user's location on the flat display surface. This ensures that when users move in front of the display, looking straight ahead always sets the cursor exactly in front of them. As they look further away, the cursor is progressively offset from the direction of the head (up to 12.5°, see Figure 6.3), accounting for the extra rotation of the eyes. This offset makes it possible to point at targets on the sides of the wall comfortably while maximizing accuracy in the central area.

This is an absolute yet indirect technique with a constant resolution function [42]: all targets with the same display size have the same motor size. Its precision directly depends on the operating range of the input channel, which cannot be endlessly extended. The limit is the angular range users can reach with their neck with minimal discomfort and minimal difference between eye and head orientation. Increasing the angular operating range makes small targets easier to acquire, but amplitudes become larger. This solution

[2]When a target is off-centered relative to the default head position, users rotates their head in order to minimize the effort put on ocular muscles [66].

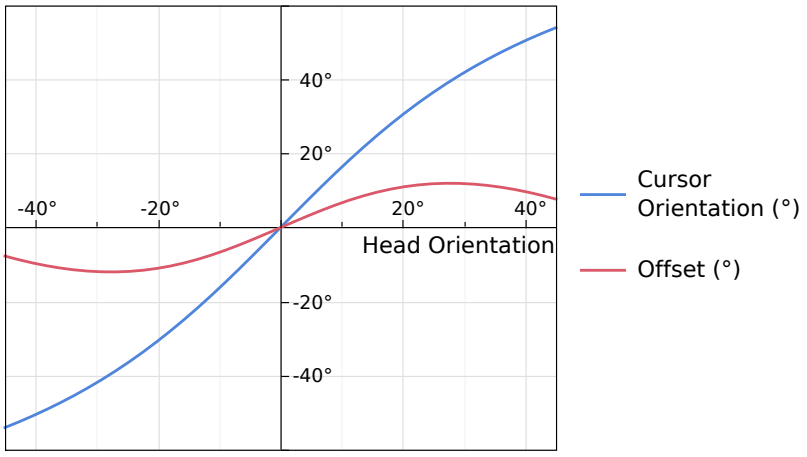


Figure 6.3: Angular abscissa of the cursor depending on the horizontal orientation of the head. Offset represents the difference between the head orientation and the cursor angle.

favors mid-air interactions since target motor sizes no longer depend on distance.

6.2.3 Cost of Switching

In the previous chapter we observed that the *Precise* phase of a pointing task affects how users' behavior perform in *Coarse* mode. Participants typically spent more time adjusting the location of the mode switch when the *Precise* phase looked challenging (i.e. smaller targets) or when the *Coarse* phase looked easy (i.e. small pointing amplitude). In the easiest cases they even skipped the *Precise* phase and achieved the selection in *Coarse* mode only. However this optimization behavior occurred in only 5 % of the trials and we did not find significant evidence of improved performance. The *Coarse* phase was originally designed to embody the initial, ballistic phase of a pointing task. While fast, unconstrained *Coarse* movements showed signs of accuracy optimization and thus of additional cognitive cost.

This raised a question that we had not anticipated: beyond minimizing the motor cost of switching between modes, is it possible to minimize the *cognitive* cost associated with making the decision to switch between the two modes?

We explored an approach where the cognitive load is transferred to the perceptual system, hypothesizing that this would significantly reduce the switching cost. We designed variations of the above techniques that artificially limit the precision of the *Coarse* pointing mode: users approach the target fast and know when to switch to precise mode simply because there is no other option. We *discretized* the wall display according to its constituent 30" LCD panels arranged in an 8×4 matrix. In *Coarse* mode, users could only jump

from screen to screen. We call this process *discretization* to emphasize that the pointing resolution in *Coarse* mode is artificially degraded while keeping the same physical input resolution.

The resulting technique, when coupled with *Head* control, is somewhat reminiscent of the Rake cursor [31, 141] and similar multi-cursor desktop pointing techniques that use eye gaze to select the active cursor in a matrix of cursors all controlled with the same input device, such as a mouse. However we chose to always relocate the cursor at the center of the currently head-pointed screen when in *Coarse* mode, as opposed to keeping its last *Precise* location relative to the screen. We expected users to always remember where they had left their cursor when they last switched to *Coarse* mode and to be able to take advantage of it. Preliminary tests showed otherwise, and we found that having the *Precise* cursor start at a constant location (relative to the screen where the mode switch occurred) yields better performance and is less cognitively demanding. Having the *Precise* mode start at the center of the screen also provides a constant average distance between the cursor and any target in that screen. Jumping to another screen thus puts the cursor at the center of that screen, and users have to switch to *Precise* mode to reposition the cursor within it.

6.2.4 Cursor Feedback

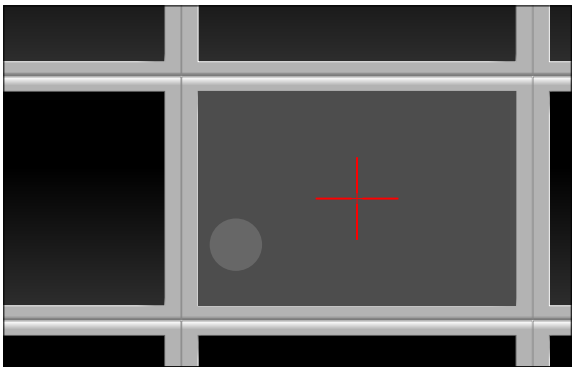


Figure 6.4: Cursor of the *Discrete* techniques in coarse mode. The precise cursor (red cross) is located at the center of the screen currently pointed at, which is highlighted (white, 5% translucency). A translucent proxy (white circle, 2% additional translucency) indicates the location of the continuous pointing input in order to enable users to anticipate when the pointed screen will change.

We use a red cross for the cursor and additional feedback in coarse mode. For the *Continuous* techniques we simply surround the cursor by a red circle. As in Chapter 5, the size of the circle gives users an indication of how far from the target they can bring the cursor in precise mode.

For the *Discrete* techniques, in coarse mode the visible cursor is a red cross always located

at the center of the pointed screen and the currently pointed screen is slightly highlighted (Fig. 6.4). Preliminary tests showed that users also need to know how close they are to switching screens. If not, screen flickering may happen that can cause imprecise mode switch location. A translucent disc (Fig. 6.4) is added to provide feedback about the exact, continuous location pointed by the coarse technique so that users know how close they are to leave the current screen. We tried other feedbacks such as highlighting the border of the screen when the user is about to change screen, but these indirect feedbacks require more attention to be understood than a simple proxy of the pointed location. The screen highlight and the proxy of the continuous pointing location are meant to be background feedback, thus their opacity is low (5% for the screen combined with 2% for the proxy).

6.2.5 Four dual-precision pointing techniques

We eventually narrowed our design down to four pointing techniques by combining the two coarse input control techniques (pad-based vs. head-based) with two coarse mode input precisions (*continuous* vs. *discretized*). As mentioned earlier, all four techniques use the handheld device's pointing zone to control the cursor in *Precise* pointing mode with optimized CD-gain transfer functions.

- ARC-Pad$_2$ + *Continuous* = *ContPad*: two conjoined fingers in the handheld's pointing zone activate *Coarse* mode (absolute mapping of the zone to the wall); a single finger activates *Precise* mode (relative cursor displacements).
- ARC-Pad$_2$ + *Discrete* = *DiscPad*: same as above but pointing precision in *Coarse* mode is artificially restricted: the cursor can only get positioned at the center of any given LCD panel, requiring a switch to *Precise* mode for any further cursor position adjustments.
- *Head*+ *Continuous* = *ContHead*: head orientation provides *Coarse* control of the cursor, without any artificial restriction on pointing precision. Touching the pointing zone on the handheld automatically switches to *Precise* mode (relative cursor displacements).
- *Head*+ *Discrete* = *DiscHead*: same as above, but pointing precision in *Coarse* mode is artificially restricted. Head orientation can only make the cursor jump to the center of any given LCD panel, as in the case of *DiscPad*.

With *ARC-Pad*$_2$ techniques, the cursor only moves if one or two fingers touch the screen, allowing users to easily perform clutch and tap gestures. However *Coarse* mode is the default mode for *Head* techniques: as long as the user does not touch the handheld's screen, the cursor follows the user's head movements. *Head*-based techniques thus feature

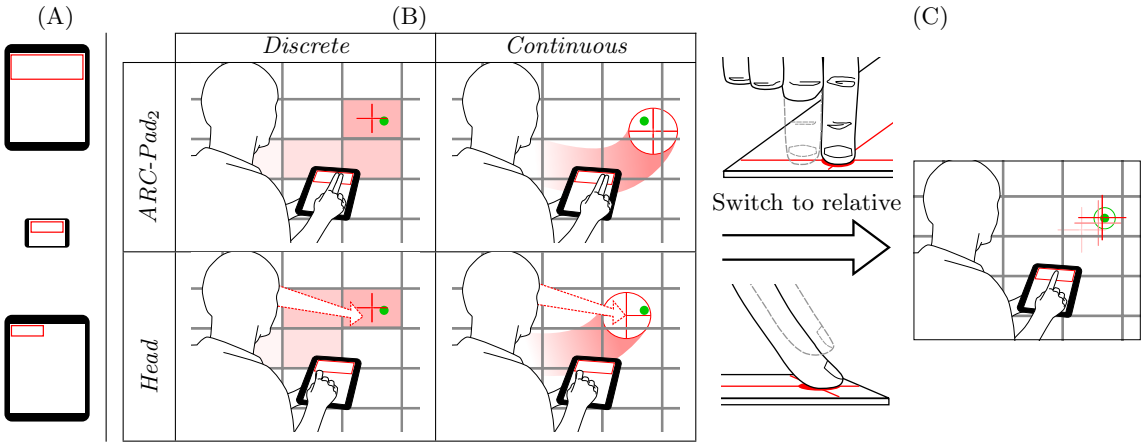


Figure 6.5: The four dual-precision techniques. (A) The three combinations of device and pointing zone used in our experiments: tablet with large zone (Experiments 1 and 3), smartphone with small zone (Experiment 2) and tablet with small zone (Exp. 3). (B) The four coarse (absolute) modes, combinations of *Head* vs. *ARC-Pad*$_2$ and *Discrete* vs. *Continuous*. (C) In all cases, using a single finger in the pointing area switches to precise (relative) mode.

a 500 ms delay after finger release from the handheld's surface before switching back to coarse mode, letting users perform clutch and tap gestures.

6.3 Control-Display Transfer Functions

All the techniques introduced above use relative pointing in either or both modes. As mentioned earlier in this chapter, a simple absolute mapping from the handheld device to a ultra-high-resolution wall display does not work, because one pixel on the handheld maps to several dozen pixels on the wall, even if we were to make the pointing zone full-screen. Pointer Acceleration consists in applying a transfer function to the Control-Display (CD) gain based on the dynamics of the users' movements. We showed in Chapter 5 that mid-air angular relative pointing, if tuned appropriately, could be a viable candidate for high precision pointing on wall displays. We expect touch-sensitive tablets to also enable users to point precisely with carefully tuned Pointer Acceleration functions. As mentioned earlier, optimizing the transfer function that controls Pointer Acceleration for ultra-high-resolution wall displays is challenging and the previous literature about tuning a transfer function in a specific context (here ultra-high resolution wall-sized displays) is scarce[3]. The acceleration functions implemented in major operating systems have been parameterized for desktop environments [44]. They work for single- and multi-monitor display configurations, but are not adapted to ultra-high-resolution walls, which typically

[3]A recent study from Roussel et al. [148] partially addresses this problem, but was published after the work described here was conducted.

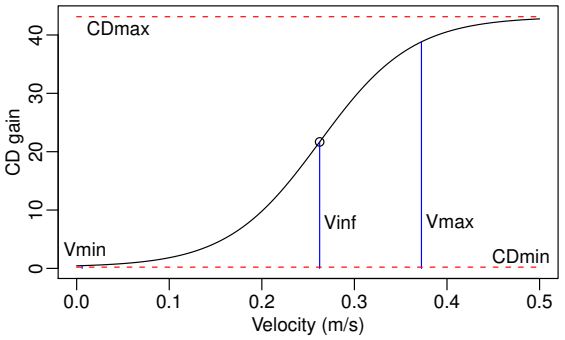


Figure 6.6: An example of logistic sigmoid curve (large zone *RelaLarge*). In this example V_{min}= 0.006, V_{max}= 0.37, $CD_{min} = 0.22$, $CD_{max} = 43.1$, $\lambda = 20$ and $ratio_{inf}$= 0.5.

feature a 200 to 400 inches diagonal and a very high resolution, e.g., 20 480 × 6 400 for the WILD room (Figure 6.1). Beyond recent work by Casiez et al. [44, 43] that provides a general framework but does not address contexts such as ours, the only documented calibration methods are those related to PRISM [67] and its subsequent refinements [104, 69]. These calibration methods, however, were designed to support absolute-to-relative transfer functions that enable pointing techniques that feature an implicit switch between absolute and relative pointing. While we later compare our techniques with the latest developments in this area, their calibration methods are of little use for tuning our transfer functions.

In this section, we propose a method to calibrate those functions. In our context, Pointer Acceleration must be adapted to enable relocation of the cursor across the display (corresponding to amplitudes of up to 22 000 pixels) at high input speeds, minimizing clutching by setting the CD gain to a high value. Conversely, CD gain must be set to a low value at low input speeds so as to enable high-precision cursor control.

Some operating systems use sigmoid transfer functions [44, 43] that are characterized by a slope that smoothly gets steeper before decreasing again, as illustrated in Figure 6.6. The lower slopes at both ends of the curve enable higher precision at low input velocities and bound cursor speed.

To model such curves, we use a simple form of the generalized logistic function:

$$CD(x) = \frac{CD_{max} - CD_{min}}{1 + e^{-\lambda(x - V_{inf})}} + CD_{min} \tag{6.1}$$

where

$$V_{inf} = ratio_{inf}.(V_{max} - V_{min}) + V_{min}.$$

This function can be tuned with six parameters:

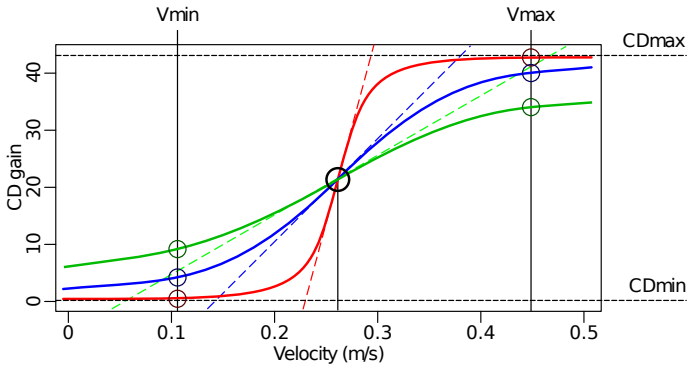


Figure 6.7: Examples of curves with the same values for V_{min}, V_{max}, $ratio_{inf}$, CD_{min} and CD_{max}, and a varying λ. The red, blue and green curves feature decreasing values for λ. The values of the function at the extremities of the input range (small circles) illustrate the effect of λ on the reachability of CD_{min} and CD_{max}.

- V_{max} and V_{min} bound the input range of the function by specifying lower and upper velocities beyond which accurate control becomes difficult. Defining the curve outside of this range makes little sense since it represents velocities harder or even impossible to reach precisely.
- $ratio_{inf}$ positions the function's inflection point within this range: 0 sets the inflexion point above V_{min}, 1 sets it above V_{max} and 0.5 sets it in between. We preferred using a ratio rather than absolute values because the input velocity range is already defined.
- CD_{max} and CD_{min} are asymptotic values that define the output range of the function. They depend on the task (A_{max} and W_{min}) and on users' perceptions.
- λ defines the curve's slope at its inflection point:

$$CD'(V_{inf}) = \frac{\lambda \times (CD_{max} - CD_{min})}{4} \tag{6.2}$$

  $\lambda = 0$ yields a constant function, $\lambda = \infty$ a step function.

λ is a trade-off between the smoothness of the curve and how close to CD_{min} and CD_{max} the resulting transfer function can go within the input range $[V_{min};V_{max}]$. Low values (Figure 6.7, green) make for smoother curves but CD_{min} and CD_{max} are harder to approach. High values (Figure 6.7, red) increase the effective output range of the function but make it steeper, and thus possibly harder to control around the inflexion point.

We tune these parameter values as follows:

1. V_{min} and V_{max} are respectively the 90^{th} quantile and the median of two corpora of velocities corresponding to voluntarily slow and fast finger movements on a tablet;
2. CD_{min} maps a lower bound for input movement amplitude that is considered usable

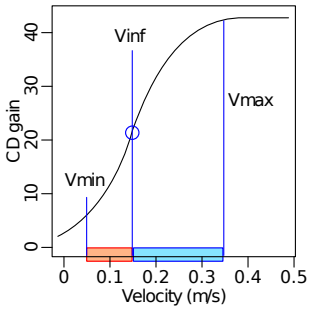
for selecting a target (the equivalent of 100 *input* pixels) to the minimum target size W_{min}:

$$\frac{W_{min}}{100 \times \text{res}_{device}}$$

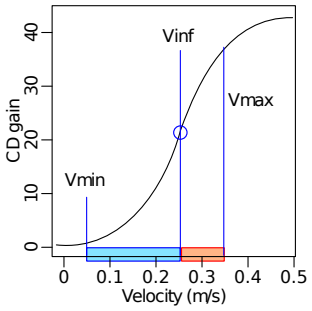
The initial value for CD_{max} is the maximum pointing amplitude, i.e. A_{max} for modeless techniques or the expected *Precise* mode amplitude L (see Chapter 5) for precise modes, mapped onto the input zone width:

$$\frac{A_{max} \text{ or } L}{\text{width}_{input}}$$

3. $ratio_{inf}$ defines how "early" (or "late") the inflexion happens in the user's velocity range:



earlier (low value) means that more of the input velocity range (higher part, blue) is dedicated to fast cursor movements, making them more controllable than precise ones; users must be able to finely control their slowest input motions (red) so that small targets are still reachable.



later (high value) means that more of the input velocity range (lower part, blue) is dedicated to precise cursor movements, making them more controllable than fast ones; users must be able to perform movements fast enough (red) to travel through the highest amplitudes.

4. λ is directly proportional to the curve's maximum steepness (Equation 6.2, Figure 6.7). It should be as low (smooth) as possible while allowing cursor movements both fast and slow enough to perform the worst-case pointing tasks without clutching. If this output range cannot be achieved without resulting in a step-like function around the inflexion point, go to step 5.

5. In the most constrained cases, lowering CD_{max} decreases the range of the function's output. This solution makes it easier to smoothen the curve but lowers the cursor's maximum velocity.

Step 1 is rather informal and results in somehow coarse values. However it is tricky to

define an *effective* input velocity range, as users will likely adapt their movement to the transfer function. We rather use this coarse, initial velocity range as a starting point for tuning transfer functions. A tuning process is considered over when (1) users can perform the most difficult task (largest amplitude, smaller target width) and (2) no further parameter tweaking can improve pointing performance or the users' subjective perception of smoothness and precision.

Three volunteers were asked to test the techniques in an informal iterative design process of 800 pointing trials per user on average. Table 6.1 gives the parameter values that we obtained after an extensive phase of pilot testing, for the two pointing zones that served in the experiments reported later: a large zone that fits within a tablet device in portrait mode, and a small zone that fits within a smartphone in landscape mode.

Size	Technique	CD_{max}	λ	$ratio_{inf}$
Large zone 148 × 49 (mm)	$ContHead_L$	9.9	19.9	0.4
	$ContPad_L$	6.6	17.5	0.4
	$DiscHead_L$	12.9	17.5	0.4
	$DiscPad_L$			
	RelaLarge	43.1	20.0	0.5
Small zone 75 × 26 (mm)	$ContHead_S$	6.3	26.5	0.4
	$ContPad_S$			
	$DiscHead_S$	16.4	26.5	0.7
	$DiscPad_S$			
	RelaSmall	83.0	22.5	1.0

Large zone: $V_{min} = 0.006m/s$, $V_{max} = 0.37m/s$, $CD_{min} = 0.22$
Small zone: $V_{min} = 0.003m/s$, $V_{max} = 0.19m/s$, $CD_{min} = 0.27$

Table 6.1: Transfer function parameter values.

Generally-speaking, V_{min} and V_{max} are lower for the small zone because users have less physical (motor) space at their disposal for pointing, and thus less amplitude to accelerate. Similarly, the small zone features a higher CD_{max}, allowing for faster cursor movements to compensate for the smaller operating range. *Discrete* techniques feature a higher CD_{max} than *Continuous* ones to compensate for the larger distance between the target and the point where mode switching is performed.

6.4 Comparing the Dual-Precision Techniques

We conducted two experiments to evaluate the performance of the four dual-precision pointing techniques introduced above, *ContHead*, *ContPad*, *DiscHead* and *DiscPad*, and

to assess the cost of mode switching. The two experiments followed the same design but involved two different devices and input sizes: we used a tablet with a large pointing zone in the first experiment, and a smartphone with a small pointing zone in the second one. For the sake of clarity, we use subscripts $_L$ (*Large*) and $_S$ (*Small*) when referring specifically to their tablet and smartphone implementation, e.g. $ContHead_L$ and $ContHead_S$.

Our hypotheses are as follows:

> **(H1)** *Discrete* techniques lower the cognitive (and thus overall) cost of mode switching by leaving no choice to users about when to switch mode, leading to a shorter coarse pointing phase than *Continuous* techniques.
>
> **(H2)** *Head*-based techniques make mode switching cognitively less demanding, as they use different body parts to control the two modes. *Pad*-based techniques use the fingers in both modes and for mode switching. With *Head*-based techniques, a mode switch is triggered when the finger comes into contact with the pointing zone.

Related to (H1), we expect an effect of forcing the mode switch with *Discrete* techniques on the time spent in the precise phase, since this often entails engaging *Precise* pointing mode significantly farther away from the target than with *Continuous* techniques.

We also expect to observe relative differences between the two experiments due to the smaller pointing zone used in the second one: we expect *Head*-based techniques to be at an advantage in the second experiment, as the smaller size of the pointing zone will hinder performance of *Pad*-based techniques in *Coarse* mode.

6.4.1 General Design and Procedure

6.4.1.1 Apparatus.

Both experiments were conducted in the WILD room (seen Appendix A) and developed with the jBricks [138] library. We used the VICON motion-capture system to track passive IR retroreflective markers and provide the 3D coordinates of the participant's head. Participants stood up, 2 meters away from the display. Given this position and the size of the wall, the operating range of the head was $\pi/2 \times \pi/5$ rad.

6.4.1.2 Trials.

The task consisted in acquiring circular targets of width W. Participants first had to dwell for half a second in a dedicated zone in coarse mode, and could then acquire the target, positioned at a distance A from the cursor's position. Experimental conditions combined

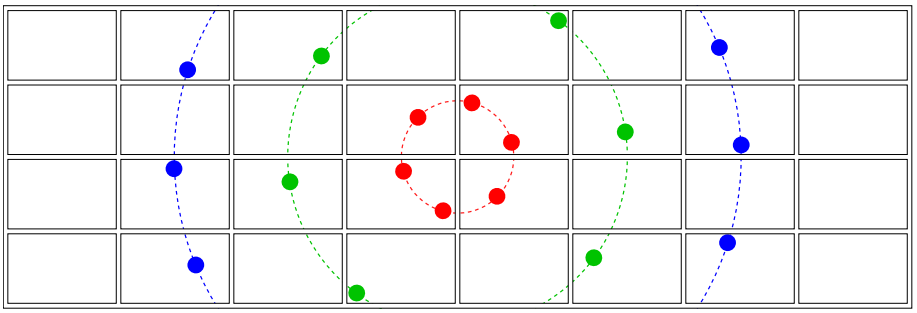


Figure 6.8: Location of the six targets in all three experiments depending on the task amplitude. Colors were changed for readability: the experiment background was black and all targets were green. Targets are also represented much larger than they were in the experiment.

one of two target widths (W): 18 pixels (4.16 mm), 80 pixels (18.48 mm), and one of three movement amplitudes (A): 2760 pixels (width of one LCD panel, 0.637 m), 8280 pixels (width of 3 LCD panels, 1.912 m), 13 800 pixels (width of 5 LCD panels, 3.187 m).

As discussed in Chapter 5, and according to Casiez *et al.*'s literature review in [43], the highest Fitts' Index of Difficulty (ID) tested prior to their article was 7.6 bits in a desktop context (A= 30 cm, W= 1.5 mm). In this same article they report a univariate (1D) Fitts' experiment with IDs as high as 9 (A= 4.5 m, W= 9 mm) on a 25-ppi projected display. The highest ID tested in the following studies is 9.54 in bivariate tasks (2D) tasks, a difficulty never evaluated to our knowledge (except in Chapter 5).

For each movement amplitude A, we used six combinations of starting and ending screens around the wall's geometrical center (see Figure 6.8) so that the dwell zone was never located in the same screen as the previous target. Targets were pseudo-randomly positioned inside a screen so that the average distance from the center of that screen to the target was about 600 pixels (150 mm). We chose this value so as to neither advantage nor disadvantage *Discrete* techniques (relative to the size of a screen), given that those techniques position the cursor at the center of the most recently visited LCD panel.

6.4.1.3 Design and Procedure.

Both experiments use a $4 \times 2 \times 3$ within-subject design with factors TECH (technique), W and A. We blocked by TECH and used a latin square to balance the presentation order of techniques among participants. At the beginning of each block of a *Head* technique, participants were asked to stand still and stare at the center of the display for 5 seconds to calibrate the center of their head's operating range, in order to balance the angular offsets described earlier. Each block started with a training session composed of two parts. In the first part, W and A were set to their largest values. The operator explained the technique

and asked participants to practice until their performance had stabilized. Performance was considered stable when they managed to perform six consecutive trials with a movement time (MT) variation below 40 % of their average, similar to Chapter 2. Participants were allowed to practice longer if they needed to. Then the second training phase started, consisting of three blocks whose purpose was to put participants in conditions closer to the actual trials to be performed next: largest W and largest A, smallest W and medium A, smallest W and smallest A. The remaining TECH blocks were measured and decomposed into six sub-blocks composed of six replications of each of the W × A conditions described earlier. For both experiments, sessions lasted 40 minutes on average. At the end of a session, participants answered a questionnaire about their preferences and were encouraged to make comments.

6.4.1.4 Participants.

The same 12 participants (2 female; 24 to 38 year-old, avg. 29.6, med. 28.5) served in both experiments. All had normal or corrected-to-normal vision and were right-handed. All were daily users of personal computers and smartphones. Only two used tablets regularly.

6.4.1.5 Measures.

We measured movement time MT – from the moment participants stop dwelling to the first successful press event on the target – and error rate. We split MT into movement time of the coarse phase, CMT, and movement time of the precise phase, PMT, according to the time of the last switch to precise mode. To evaluate the cost of mode switching, we also measured VPT, the time between the last event in coarse mode and the peak in velocity in the subsequent precise phase.

6.4.2 Tablet Experiment

In this first experiment, we used a tablet with a resolution of 768 × 1024 (Apple iPad, weight: 680 g, dimensions: $19 \times 24.3 \times 1.3$ cm, screen diagonal: 24.6cm). Participants had to hold the tablet vertically. The pointing zone used the top 768×256 pixels (148×49 mm) yielding an input resolution of 5.2 dot/mm. The transfer functions are shown in Figure 6.9.

6.4.2.1 Quantitative-objective Results.

We removed a few outliers, 0.69 % of the data (trials with an unreasonable residual/predicted ratio). These outliers were mainly due to Wi-Fi transmission problems. As expected, MT

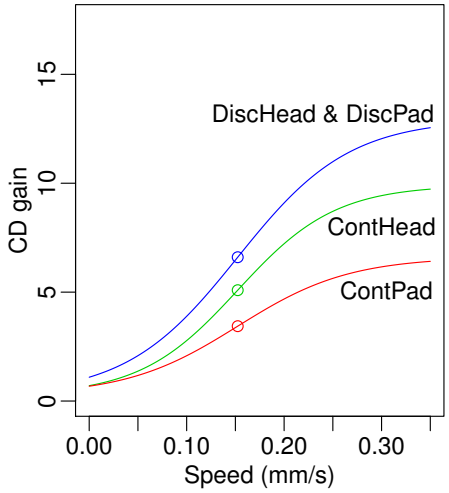


Figure 6.9: Transfer functions used for the tablet experiment (see Table 6.1 for parameter values).

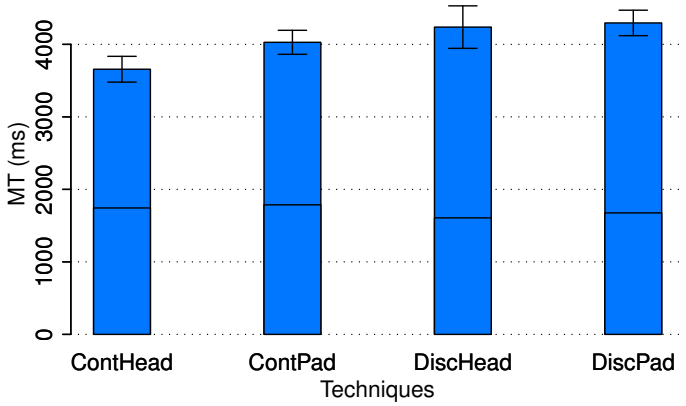


Figure 6.10: *MT* for each TECH for the tablet experiment. The black intermediate lines in the bars show the time to the last mode switch to precise mode.

distributions per condition are skewed. We thus perform our analyses using median values, per participant, on the model TECH × W × A × Rand(Participant). Figure 6.10 graphs *MT* for each technique[4].

A multiway ANOVA reveals a significant effect[5] of TECH on *MT* ($F_{3,33} = 11.4, p < 0.0001, \eta^2 = 0.14$). A post-hoc t-test with Bonferroni correction shows that $ContHead_L$ is significantly faster than all other techniques (all p's < 0.001): 9 % faster than $ContPad_L$, 15 % faster than $DiscPad_L$, and 14 % faster than $DiscHead_L$. The only other significant difference is between $ContPad_L$ and $DiscPad_L$, the former being 6 % faster.

[4]In all our barplots, the mean is taken over the medians of each experimental condition (including Participant). Error bars represent the corresponding 95 % confidence limit.

[5]We report the "generalized-partial" η^2 [16] that reflects effect size: 0.01 is a small effect size, 0.06 is medium, and 0.14 is large. Note, however, that these values should be considered with caution, see [16].

We also observe, as expected, significant effects of target width W ($F_{1,11} = 459$, $p < 0.0001$, $\eta^2 = 0.41$) and movement amplitude A ($F_{2,22} = 43.8$, $p < 0.0001$, $\eta^2 = 0.24$) on MT, participants being faster with the larger target and the smallest distances. We do not observe any interaction effect TECH × W or TECH× A.

The overall error rate is 5.9 %. A multiway ANOVA reveals no effect of TECH on the error rate ($F_{3,33} = 0.61$, $p = 0.6151$, $\eta^2 = 0.015$) and no significant interaction. Again, as expected, we find a significant effect of W on error rate ($F_{1,11} = 11.5$, $p = 0.0061$, $\eta^2 = 0.051$): 8.4 % for the small target and 3.4 % for the larger one. We also measure a significant effect of A, tough the effect size is very small ($F_{2,22} = 0.69$, $p = 0.0240$, $\eta^2 = 0.005$): 4.9 % for the largest amplitude, 6.9 % for the medium one and 5.9 % for the small distance.

As illustrated in Figure 6.10, the time spent in the coarse phase is slightly shorter with *Discrete* techniques than with *Continuous* techniques. However, this difference is not statistically significant – (H1) is not supported – and is not large enough to make *Discrete* techniques more efficient than *Continuous* ones. Indeed, the time spent in precise mode is far longer with the *Discrete* techniques. We tentatively attribute this to the fact that the last mode switch is performed 150 mm away from the target on average with *Discrete* techniques, compared to 67 mm with $ContHead_L$ and 71 mm with $ContPad_L$.

$ContHead_L$ and $ContPad_L$ feature very similar coarse pointing times (CMT) and distance-to-target at mode-switch time. $ContHead_L$'s shorter task time is mainly due to performance improvements during the precise pointing phase. We observe that peaks in velocity occur earlier with $ContHead_L$ than with $ContPad_L$ (average VPT of 462 ms vs. 735 ms for not significantly different mean velocities: 0.23 and 0.30 $m.s^{-1}$). The distances from the mode switch to the target are similar for *Head*- and *Pad*-based techniques, and their acceleration curves have the same input characteristics (V_{min}, V_{max} and $ratio_{inf}$). We thus expected the velocity peaks of $ContHead_L$ and $ContPad_L$ to occur at similar times, yet participants required more time to go "full-speed" with $ContPad_L$. This could be caused by the cognitive cost of switching between two very different control-display ratios while using the same input. This supports (H2), suggesting that the cost of the mode switch is indeed lower for *Head*-based techniques than for *Pad*-based ones.

Note also that with the *Pad* techniques, participants removed two fingers for mode switching (vs. removing only one finger and continuing pointing) in about 54.5 % of all cases (52.8 % for *ContPad* and 56.1 % for *DiscPad*). This might also explain the higher mode switch cost for *Pad*-based techniques.

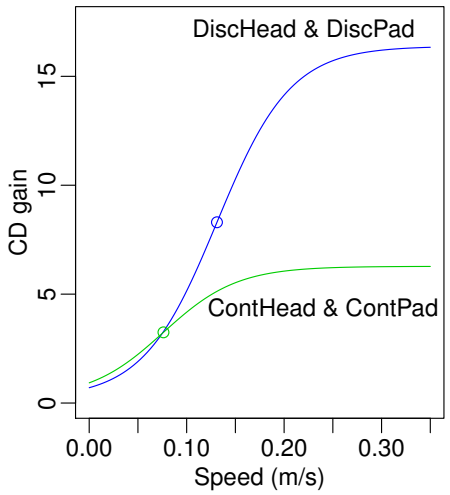


Figure 6.11: Transfer functions used for the smartphone experiment (see Table 6.1 for parameter values).

6.4.2.2 Quantitative-subjective Results.

Overall, participants preferred to use *Head*-based techniques (10 out of 12) and *Continuous* techniques (8 out of 12). 7 participants ranked $ContHead_L$ first, 4 ranked $DiscHead_L$ first and 2 ranked $ContPad_L$ first. However, there were no strong complaints about any particular technique, except for one participant who clearly stated that he disliked *Discrete* techniques.

Several participants complained about the lack of tactile feedback, that made it difficult to know when the fingers where leaving the pointing zone. They expressed the need for some sort of physical border, such as the border of a touchpad on a laptop. Only one participant answered that holding the tablet for 40 minutes was indeed a cause of fatigue when we inquired about this potential issue.

6.4.3 Smartphone Experiment

This second experiment used a smartphone with a resolution of 480 × 320 (Apple iPod Touch, 115 g, 11 × 6.2 × 0.8 cm, screen diagonal: 8.9 cm). Participants had to hold the device horizontally. The pointing zone used the top 480 × 166 pixels (75 × 26 mm), yielding an input resolution of 6.4 dot/mm. The transfer functions are shown in Figure 6.11.

We took participant feedback about the lack of tactile feedback into account, and stuck an easily-removable and slightly protruding tape delimiting the pointing zone (3 mm wide, 0.8 mm thick, see Figure 6.14). Preliminary tests showed that the tape made it much

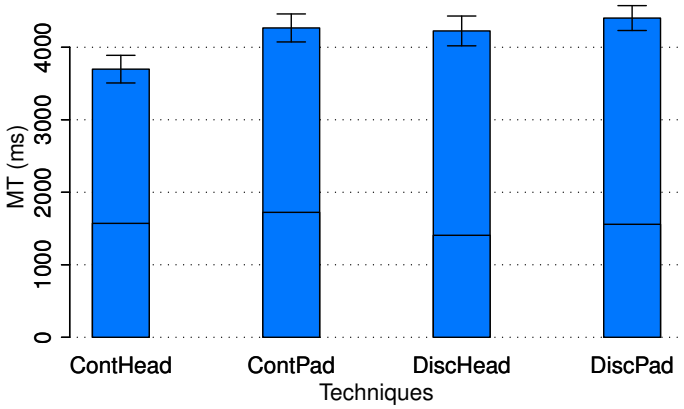


Figure 6.12: *MT* for each TECH for the smartphone experiment. The black intermediate lines in the bars show the time to the last mode switch to the precise mode.

easier to find the input zone eye-free and to know in advance when the finger is about to leave the input zone while pointing.

Results are very similar to those of the tablet experiment. We get similar error rates, and an overall subjective preference for $ContHead_S$. The ANOVA reveals a significant effect of TECH on *MT* ($F_{3,33} = 12.1$, $p < 0.0001$, $\eta^2 = 0.19$) and, as in the first experiment, no interaction of TECH with W, A and W × A. A post-hoc t-test with Bonferonni correction shows that *ContHead* is significantly faster than all the other techniques (all p's < 0.0001), with a speed up of 13 % against *ContPad*, of 12 % against *DiscHead* and of 16 % against *DiscPad*. Regarding W and A we find results similar to the smartphone experiment for *MT*. Movement time, split between the coarse and precise phases, is shown in Figure 6.12 using the same scale as in Figure 6.10.

Regarding the coarse pointing phase, this time we observe a significant effect of TECH ($F_{3,33} = 5.39$, $p = 0.0039$, $\eta^2 = 0.11$). $ContHead_S$ is significantly faster than all other techniques, and $DiscPad_S$ and $ContHead_S$ are significantly faster than $ContPad_S$ (considering *Coarse* phase only). With the smaller pointing zone used in this experiment, we do observe the hypothesized advantage for discrete techniques (H1). However, the time gained during the coarse phase is again not significant enough to make *Discrete* techniques faster than *Continuous* ones.

Regarding the precise (and post mode-switch) phase, we could first note that the distance to the target at the last mode switch is about 83 mm and 88 mm for *ContHead* and *ContPad*, and about 149 mm and 179 mm for *DiscHead* and *DiscPad* (with *DiscPad*, in a number of trials, the participant switched to the precise mode in a screen that does not contain the target). The difference between continuous and discrete techniques shows that the time saved in the coarse phase by the discrete technique does not lead to better

performance for the full pointing task.

In the case of *ContHead* and *ContPad* the above distances are very close, but *ContHead* is faster than *ContPad* (though using the same transfer functions), enlarging the gap that already exists in the coarse phase. As in the tablet experiment, it seems that the cost of the mode switch is higher for *ContPad* than for *ContHead*. Indeed, as in the case of the tablet experiment, the velocity peak comes earlier in the precise phase with *ContHead* than with *ContPad* (485 ms vs 773 ms for peaks of 0.33 and 0.29 $m.s^{-1}$). As in the tablet experiment (H2) is supported.

6.4.3.1 Subjective Results.

In this experiment 8 participants (out of 12) ranked the *ContHead* technique first, 3 participants ranked *DiscHead* first and one participant ranked *DiscPad* first. This is similar to the tablet experiment, however no participant preferred $ContPad_S$ in this experiment while two of them ranked $ContPad_L$ first in the tablet experiment (where *DiscPad* was not ranked first). As in the tablet experiment only one participant (not the same) found that holding the smartphone during 40 minutes caused fatigue.

Surprisingly, this experiment was considered more tiring by participants. Despite the smartphone being lighter and smaller, it required a different grip and the body could not be used to hold a part of the device's weight, as in the tablet experiment.

6.4.4 Discussion on Both Experiments

Since we did not counterbalance the order of pointing zone sizes (all participants performed the tablet experiment first), and since we added tactile feedback using tape to delimit the zone in the second experiment, we cannot formally compare overall performance across both experiments. However, we can make three informal observations. First, the movement time difference between $ContHead_S$ and $ContPad_S$ is larger for the smartphone (13 %) than for the tablet (9 %). Second, $ContPad_S$ is not significantly faster than $DiscPad_S$, as opposed to $ContPad_L$ vs. $DiscPad_L$. Third, Figures 6.10 and 6.12 suggest that *DiscHead* performed better with the smartphone than with the tablet, relative to *ContPad* and *DiscPad*. These observations suggest that as we had anticipated, *Head*-based techniques are at an advantage with smaller pointing zones.

In Chapter 5, we hypothesized that participants implicitly paid more attention to the proximity of their mode switch to the target when either (1) the *Precise* phase seemed difficult (small target), or (2) the *Coarse* phase was fast enough to spend extra time

to switch modes closer to the target. We expected this unconscious mechanism to be cognitively demanding and time consuming. We investigated *Discrete* techniques that simplify the decision of switching modes by leaving the user little or no choice about the switch location. However in both the Tablet and Smartphone experiments the effect of discretization on dual-precision pointing performance was either neutral or negative. This suggests that the "natural" movement time optimization observed in Chapter 5 is somehow more efficient than forcing the location and time of the mode switch. However, we studied only one size of discretization: participants could only point at screens in *Coarse* mode with *Discrete* techniques. Future works could include a more thorough investigation of the size of discretization similar to the theoretical analysis of L in Chapter 5.

6.5 Comparison with State-of-the-Art Techniques

Among the four techniques studied in the previous section, *ContHead* is both the fastest and the preferred technique. Having identified this technique, we wanted to compare it with state-of-the-art remote pointing techniques that do not necessarily require holding a tablet. While such techniques cannot accommodate widgets used for other purposes than pointing, our primary goal was to evaluate the potential cost of pointing with a handheld device and put our results in a more general context. There are two families of candidate techniques for this task, briefly discussed earlier: purely relative techniques that use a transfer function to provide Pointer Acceleration, and other dual-precision techniques described in the literature.

We noticed during our early pilot studies that purely relative pointing techniques can actually be viable candidates if they use a properly tuned acceleration function. We thus decided to include such a technique in our evaluation. We implemented *RelaSmall* and *RelaLarge*, two purely relative, trackpad-like techniques using different input sizes, each with its own optimized transfer function. As mentioned before, the literature on calibrating transfer functions for relative pointing techniques is scarce [44, 43]. The functions used by major operating systems do not meet the requirements of ultra-high-resolution wall-sized displays in terms of speed and precision. We therefore used the transfer function calibration method initially developed for our dual-precision techniques. We tuned two transfer functions, one for each of the two pointing zones used in previous experiments (see Figure 6.13).

Compared to the functions used in the first two experiments, these two functions feature a much higher CD_{max} that allows traveling much larger distances without too much clutching, but also drastically increases their maximum slope. To avoid introducing a con-

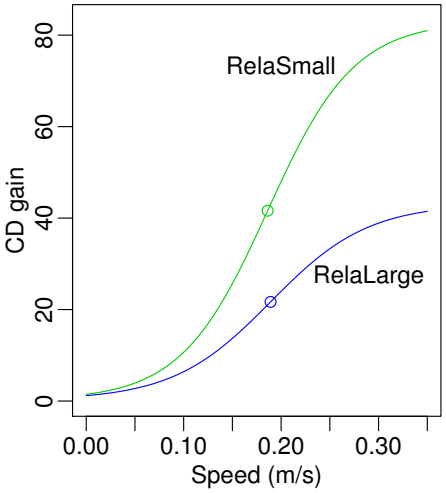


Figure 6.13: Transfer functions used for the third experiment (see Table 6.1 for parameter values).

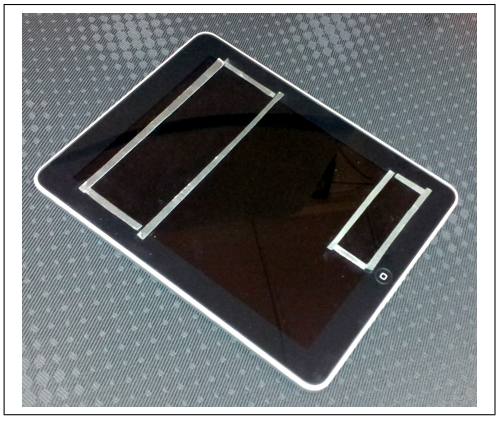


Figure 6.14: The two input zones delimited by thick removable tape.

found due to the device itself, we decided to use $ContHead_S$ on the tablet (see Fig. 6.5-A) as suggested by the very similar performance observed in the first two experiments with respect to the size of the pointing zone. This makes sense beyond this laboratory study, since using a smaller pointing zone on a tablet means that the device can accommodate more widgets for other purposes.

We compared the three above techniques, which use a handheld device at least partially for pointing, with two techniques from the literature: *LaserGyro* and *SmoothPoint*.

LaserGyro, presented earlier in this dissertation (Chapter 5), is a mid-air dual-precision technique that uses ray-casting as coarse mode and relative angular motion as precise mode [127]. This technique was originally inspired by Vogel *et al.*'s work on distant freehand pointing [163]. A difference with Chapter 5 is that we computed a transfer

function for the *Precise* mode of *LaserGyro* using the same process (Section 6.3). Its parameter values are reported in Table 6.1.

SmoothPoint [69], also combines ray-casting and relative pointing. However, the transition between the two modes is progressive, based upon a transfer function that depends on input motion velocity. The authors propose a method to tune this function, but pilot tests in our environment revealed that this method does not scale to pixel densities such as those typically encountered on ultra-high-resolution wall displays: the difference between the minimum and maximum CD gain values is too high, causing the precise mode to be far too jerky to select small targets such as those considered here. We transposed the calibration method described in [69] to our context[6] to the best of our abilities, iterating until the technique eventually enabled us to achieve the pointing tasks considered in our experiments.

In this experiment, we tested five techniques (TECH): *ContHead*$_S$, *RelaSmall*, *RelaLarge*, *SmoothPoint* and *LaserGyro*. The apparatus, design, and procedure were exactly the same as in the previous experiments. We added a physical border around the large and small pointing zones on the tablet to limit the need to look at the input device by providing tactile feedback when the fingers were about to leave the zone. We used a 5×5 latin square to balance the techniques. 15 participants served in the experiment (10 of them had participated in the previous experiments, 5 were new and assigned to the same Latin square).

6.5.0.1 Quantitative-objective Results.

As in the previous experiment we removed a few outliers (1.28%). Figure 6.15 shows movement time MT by W and A. For *ContHead*$_S$ the results are very close to those of the previous experiments. We observe that *SmoothPoint* performs poorly compared to all the other techniques (significantly so, for each W and A condition). For brevity, we do not report the results of *SmoothPoint* in post-hoc tests, even though it was of course included in those tests.

An ANOVA reveals a significant effect of TECH on MT ($F_{4,56} = 20.5$, $p < 0.0001$, $\eta^2 = 0.25$). A post-hoc t-test with Bonferonni correction shows that *ContHead*$_S$, *LaserGyro* and *RelaLarge* are all significantly faster than *RelaSmall* (with speed-ups of 5.4%, 8% and 9%, respectively). We find no significant difference between *ContHead*$_S$, *LaserGyro* and *RelaLarge*.

This time, the ANOVA reveals significant interactions TECH × W ($F_{4,56} = 5.90$, $p = 0.0005$,

[6] We did not use our logistic function because the function described in [69] is part of its contribution.

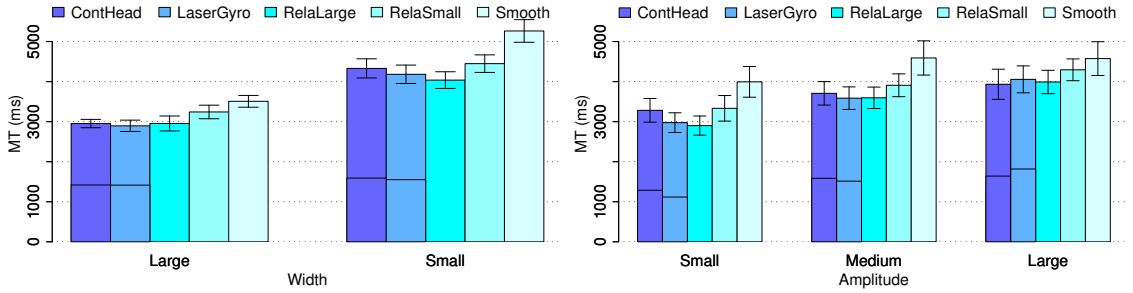


Figure 6.15: *MT* for each TECH by W and by A for the last experiment.

$\eta^2 = 0.04$) and TECH × A ($F_{8,112} = 2.60$, $p = 0.0122$, $\eta^2 = 0.03$) on *MT*. One cause for these interactions is *SmoothPoint*, which is slower for small targets than for larger ones, and faster for the largest amplitude. We can also observe on Figure 6.15 that the two relative techniques, *RelaSmall* and *RelaLarge*, are faster for the small width and the small amplitude. Indeed, post-hoc tests show that (i) for small targets, the only significant result is that *RelaLarge* is faster than *RelaSmall*, and for large targets, *ContHead*$_S$ and *LaserGyro* are also significantly faster than *RelaSmall*; (ii) for the small amplitude the only significant results are that *RelaLarge* is faster than *RelaSmall* and *ContHead*$_S$, while for the large amplitude the only significant results are that both *RelaLarge* and *ContHead*$_S$ are significantly faster than *RelaSmall*.

The overall error rate is 9.6%. *ContHead*$_S$, *RelaSmall* and *RelaLarge* feature low error rates (3.90%, 3.52% and 4.04%) with only marginal differences between large and small targets (2.5% vs 5.17%). *LaserGyro* and *SmoothPoint* also feature low error rates for large targets (0.70% and 5.60%), but the error rate rises dramatically for small targets: 25.6% and 43.0%. Based on our pilot studies, we did not expect such an increase in error rate. The problem turned out to come from the fact that clicking with the handheld wireless mouse caused small hand movements which, in turn, caused small cursor displacements that were sometimes large enough to make the cursor leave the target. This led users to click multiple times in quick succession to acquire the target. We furthered our analysis by measuring the time to first click (instead of first click on the target). This did not change the results for *LaserGyro*. However, results were slightly different for *SmoothPoint*, which remained slower than *RelaSmall* in all conditions but not significantly so for the small target + large amplitude condition.

6.5.0.2 Quantitative-subjective Results.

At the end of the experiment, we asked participants to rank the techniques on a five-point Likert scale in terms of preference, fatigue and perceived performance. Figure 6.16

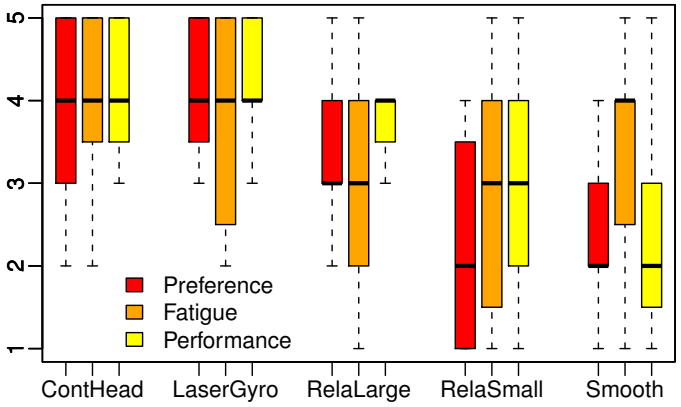


Figure 6.16: Preference, fatigue and (self-reported) perceived performance for each TECH on a five-point Likert scale (5 is best, 1 is worst). Bold lines show the median, boxes show the lower and upper quartiles and the whiskers show the 1.5 × inter-quartile range.

summarizes the results in a boxplot. Kruskal-Wallis tests revealed a significant effect of TECH on Preference ($\chi^2_4 = 28.5$, $p < 0.0001$) and Performance ($\chi^2_4 = 23.4$, $p = 0.0001$), but not on Fatigue. Post-hoc tests using Mann-Whitney tests with Bonferroni correction show that (i) *ContHead*$_S$ and *LaserGyro* were preferred to both *RelaSmall* and *SmoothPoint*; (ii) *RelaLarge* was preferred to *SmoothPoint*; (iii) *ContHead*$_S$, *LaserGyro* and *RelaLarge* were perceived as faster than *SmoothPoint*; and (iv) *LaserGyro* (and *ContHead*$_S$) was perceived as (marginally) faster than *RelaLarge*.

6.5.1 Discussion and Design Guidelines

Results show that using a handheld device for high precision pointing enables users to point at least as efficiently as state-of-the-art mid-air pointing techniques in terms of both speed and accuracy, while leaving a large portion of the handheld's surface available for additional, application-specific widgets.

The three techniques that perform best in terms of movement time and preference are *RelaLarge*, *ContHead* and *LaserGyro*. While there is no significant speed difference between them, each technique has its own strengths and limitations, making it suitable in specific contexts of use.

While relative pointing is not novel, making it work efficiently in such challenging contexts is an interesting result. Indeed, existing functions, even elaborate ones such as that of *SmoothPoint*, were designed for lower-resolution environments, and fare poorly with the high Fitts' IDs considered here. I suspect this is because *SmoothPoint* uses a sine-based function that does not allow to control the abscissa and slope of its inflexion point: for

a given input and output range, the corresponding method provides a single function that, when the output range is much larger than the input range, can be very steep. Indeed, the cursor was much jerkier with *SmoothPoint* than with any other technique evaluated in Section 6.5. *RelaLarge*, based on the simple CD transfer function tuning method introduced in this paper, provides pointing performance that matches that of more elaborate techniques. It is straightforward to implement and does not require elaborate equipment to track spatial position and orientation.

RelaSmall is also quite an achievement, considering that it provides enough precision to perform bivariate pointing tasks of difficulty seldom evaluated (up to 9.5), requiring a pointing zone of 20 cm^2 only (approximately $1/4^{th}$ of *RelaLarge*'s surface area). Given that the pointing zone dimensions of *RelaLarge* preclude its use on smartphones, *RelaSmall* can be seen as an interesting option. Indeed, the technique will only incur a 5-to-10-percent performance cost when compared to more efficient techniques, which can be considered an acceptable tradeoff when only small handheld devices are available, or when a large portion of the handheld's screen real-estate should be allocated to additional interface widgets.

This performance decrease can be avoided by using *ContHead*, which achieves the same level of performance as *RelaLarge* but on a much smaller input area, equivalent to that of *RelaSmall*. *ContHead* should be considered when the tasks and context of use require many additional interface widgets or when only smartphones are available, provided that tracking the location and orientation of the head is possible.

Finally, *LaserGyro* causes many more errors than other techniques for small values of W, as the tremor caused by pressing a physical button, even if comparatively small, is large enough to severely hinder acquiring very small targets. This problem does not happen with tablet-based techniques since their tapping mechanism is algorithmically decoupled from their pointing mechanism. Another drawback of this technique is that it cannot accommodate additional interface widgets on the input device. However, *LaserGyro* leaves the non-dominant hand free to perform other interactions, making it a relevant option when the task requires operating additional input devices, provided that pointing task IDs are lower than 9.

6.6 Conclusion

This chapter investigated the use of handheld devices for very difficult remote pointing tasks on wall displays, where only a portion of the device's touch surface is dedicated to pointing. The goal is to leave the larger part of the handheld device's screen for the

display of task-specific widgets, following the *Factorizing task allocation* strategy defined in Section 1.3. It is complementary to the techniques studied in Chapters 4 and 5 that were all single-handed, enabling other input devices in the non-dominant hand and following the *Maximizing limb usage* strategy also defined in Section 1.3.

We designed and evaluated techniques that use various input channels to improve the efficiency of pointing at very small targets across large amplitudes. Discrete *Coarse* pointing was expected to minimize the cognitive load of deciding where and when to switch modes. However we found that its effect on performance ranges from neutral to negative. Head orientation proved very efficient for *Coarse* pointing despite expected neck fatigue and lower accuracy. However it depends strongly on specific tracking capabilities: the system must know the absolute orientation and location of the user's head, which can be difficult to perform without expensive hardware or with multiple users. Satoh et al. [149] proposed a low-cost method to track users' heads in mid-air with acceptable accuracy. It could be interesting to evaluate *Head*-based techniques with such simpler sensors. Finally, we used the number of fingers touching the device to discriminate between *Coarse* and *Precise* pointing. *Head*-based coarse pointing performed better and was preferred by participants. However *Pad*-based techniques worked well and do not require additional hardware, they should thus be evaluated further.

Our most successful design, *ContHead*, lets users perform pointing actions at two levels of granularity: coarse pointing uses the natural movements of the head when moving the cursor across large distances; precise pointing uses a small pointing zone on the handheld device to perform relative pointing movements via finger gestures. The technique performs well including for indices of difficulty beyond those tested in most previous work, even when compared with state-of-the-art techniques that are not constrained by our requirements. *ContHead* was also rated as one of the best techniques overall in terms of subjective preference and perceived performance.

Purely relative techniques using only the touch surface of the handheld device performed better than we had originally anticipated. This led us to carefully investigate Control-Display transfer functions that enable both very fast pointing across large amplitudes while minimizing clutching, and precise cursor adjustments to acquire targets only a few millimeters in diameter. We showed that with a large-enough pointing zone and proper tuning, such a relative technique competes with the most efficient dual-mode techniques.

Conclusion of Part II: Decision and Adaptation

In this Part, I investigated mid-air pointing techniques for location-independent interaction on ultra-high-resolution wall-sized displays. These environments enable very large datasets to be visualized and manipulated by several users at once. However their large physical size and high pixel density also create new constraints for interaction: users may have to acquire very small targets from relatively far away (Requirement 2, Section 1.4), and other interactions must remain possible with the input devices at hand (Requirement 3).

In Chapter 4 I explored the limits of existing pointing techniques and devices. I used Casiez et al.'s [43] formulae to develop a theoretical testbed for techniques with fixed CD gain. I ran a controlled experiment with "simple" pointing techniques and devices and showed that fixed CD gains indeed reach a performance limit around Fitts indexes of difficulty (IDs) of about 6.7 bits. This limit is raised with Pointer Acceleration, a pointing technique that adapts the CD gain of a (relative) technique using input movement velocity. However, to date, no precise method has been proposed to calibrate the resolution function of Pointer Acceleration: I could only assess its performance through an ad-hoc transfer function.

As discussed in Chapter 4, the core problem of fixed CD gain techniques is that the input expressiveness of either the device's or the user's movements fails to match the difficulty of possible pointing tasks. Pointing techniques thus need to provide means to vary their CD gain, as demonstrated by the improved performance of Pointer Acceleration. In Chapter 5 I described *Dual-precision techniques*, a family of techniques that provide two modes, each with a specific range of CD gains for either coarse or precise control. Switching between these modes is an explicit action so that users can fully control the movements of their cursor, as opposed to Pointer Acceleration where this transition is implicit. I provide a theoretical analysis of such techniques as well as a method to calibrate the CD gain of each mode. In Chapters 5 and 6, I implemented dual-precision techniques that work either one-handed or on a small area of a hand-held device. In the former I varied the

input channels used for the *Precise* mode, and in the latter the input channels used for the *Coarse* mode. These techniques proved equivalent or better than existing ones despite their limited input channels (single-hand or small input area), meeting Requirement 3.

I also developed a method for tuning the transfer functions of Pointer Acceleration techniques, based on the capabilities of both the user and the input device(s) as well as the pointing task's characteristics. The resulting transfer functions showed performance similar to dual-precision techniques', and even tended to be better for the "easiest" tasks, probably because of the constant mode-switch time and associated cognitive load of dual-precision techniques, as discussed below. The shape of the resulting curve is reminiscent of *SmoothPoint*'s. Yet *SmoothPoint* proved much less efficient with very difficult pointing tasks. As discussed in Chapter 6, I hypothesize this is because the inflexion point of *SmoothPoint*'s transfer function cannot be tuned, as opposed to the calibration method I introduced in the same chapter.

One could hypothesize that the explicit mode switch, which is almost systematically required when using dual-precision techniques, is a clear disadvantage in terms of attention cost: users need to know and/or choose which mode they are using all along the pointing action. Indeed, using pointer acceleration techniques is a continuous process that does not require any additional, explicit action to switch from fast to precise cursor movements. However dual-precision techniques were designed to mitigate this cost through several simultaneous mechanisms and feedbacks:

1. **The CD gain at any given time should leave little to no doubt about the current mode**: dual-precision techniques are designed for very difficult tasks, thus the CD gains of the coarse and precise modes should be very different. This difference provides an all-time, implicit indication of the current mode.

2. **The *Coarse* mode is the default mode, the *Precise* mode is a quasi-mode**: users must keep a button depressed or a physical contact to maintain the technique in *Precise* mode. This ensures that they are always aware of the current mode, possibly at a semi-conscious level. In other words, the *Precise* mode is a way to increase the precision: if the default (*Coarse*) mode is too fast to acquire a given target, users can "buy" more precision by temporarily invoking the *Precise* mode.

3. Finally, **the visual representation of the cursor indicates the current mode of the technique**, as illustrated in Figure 5.1 page 125.

Dual-precision techniques performed at least as well as pointer acceleration in the controlled studies described in Chapters 4, 5 and 6 and were marginally preferred, however they remain to be evaluated within full-fledged, feature-rich applications. From the anal-

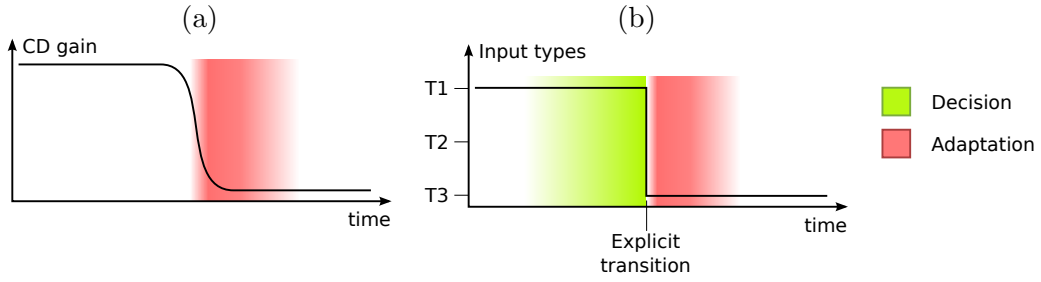


Figure 6.17: Illustration of Decision and Adaptation after a continuous transition (a) and during an explicit switch (b).

yses and studies reported in Chapters 4, 5 and 6 I also gathered general insights about high-precision pointing techniques that I summarize and discuss below.

Decision and Adaptation

The results of the experiments ran in Chapters 5 and 6 highlight two cognitive mechanisms induced by changing resolution functions during pointing: *Decision* and *Adaptation* (Figure 6.17).

Decision is the cognitive cost incurred by explicitly breaking the continuity of a pointing task, e.g. by changing its resolution function. It occurs when users have to decide *whether*, *when* and *where* they apply this change, depending on the pointing task parameters (width and amplitude) and the technique.

For example, we observed in Chapter 5 that the width and amplitude of a dual-precision pointing task had an effect on the duration of its *Coarse* phase and on the proximity of the mode switch to the target. In a few easier cases participants even tried to select the target in *Coarse* mode only, without switching mode.

My hypothesis, introduced in Chapter 5, is that the process of deciding where and when to switch modes was part of a higher-level, partly unconscious process of time optimization. Participants took more time and were more careful about the location of the mode switch (i) when the *Coarse* phase seemed easy, leaving "spare" time that could be used to ease the *Precise* phase, or (ii) when the *Precise* phase seemed difficult, i.e. when the target width was small: more time was spent in *Coarse* mode to lower the amplitude of the *Precise* phase. The no-switch behavior is an extreme case of this optimization process in which the *Coarse* mode is able to bring the cursor so close to the target that the time needed to switch modes then acquire the target with the slower *Precise* mode might appear greater

than the time needed to acquire the target with the less precise *Coarse* mode.

Similar effects have been observed in previous studies. For example, Mandryk and Lough [118] studied the effect of the intended use of the target on the performance of pointing tasks with the same ID. The task they evaluated consisted in acquiring a target (first subtask), then performing a second subtask that could be either click, click and acquire a second target, press and "throw" the target towards an edge of the screen, or drag the target onto another target. They observed that the changes in movement time can be attributed to the differences in the secondary, precise pointing phase, and not the initial ballistic phase. In accordance with our results, they hypothesize that the movement time increases with the precision of the objective, and that the motor planning and control for the second subtask is encapsulated within the second phase of the first subtask. Quinn et al. [139] observed a similar effect when studying multiple trajectory pointing methods. Participants did not perform better with methods that shorten the trajectory to a target, e.g. cursor wrapping [88] or Ninja cursors [101]. The authors suggest that selecting the best trajectory among several possibilities caused search and decision times that eventually overwhelmed any potential advantage in performance.

Adaptation is the cognitive and motor adjustment to a change of resolution function. It occurs after a sudden change of control that requires the user to adapt to a resolution function that is "different enough" from the previous one. This difference can be input-wise, e.g. switching between two input devices or modes to control cursor displacements, or scale-wise, e.g. transitioning between very different CD gains.

I formulate three hypotheses about what affects *Adaptation*:

H1 *Higher slopes in transfer functions require more cognitive and motor adjustments between their lower and higher levels* – We observed in Chapter 6 that the higher the slope, the lower the performance. Indeed, *SmoothPoint* was the worst technique for all measures, followed by *RelaSmall* for movement time.

H2 *Different resolution functions are better controlled with different limbs* – When controlled with one hand, switching from one resolution function to another constrains the starting point of the latter to the last point of the former. Users thus have a limited operating range and must adapt their movement accordingly, e.g. by clutching. In the *Tablet* experiment in Chapter 6, participants spent more time in *Precise* mode with the *Pad*-based techniques than with the *Head*-based techniques, despite the morphological similarity between the *Coarse* and *Precise* control of the *Pad*-based techniques. In both experiments of Chapter 6, participants also took more time to reach their peak velocity in *Precise* mode with *ContHead* than with *ContPad*, despite the similar

proximity of the cursor to the target at mode switch time.

H3 *When controlled with the same limb, different resolution functions are better controlled with similar input channels* – For example, switching from angular to linear control is more cognitively demanding than if the whole pointing movement is angular or linear. We observed in Chapter 5 that users spent less time in *Precise* mode with *Laser+Gyro* than with *Laser+Position* and *Laser+Track* while the average amplitudes of the *Precise* sub-task were not significantly different. Indeed, angular control (*Precise* mode of *Laser+Gyro*) is morphologically closer to ray-casting (*Coarse* mode of all three techniques) than mid-air position control and touch input (*Precise* modes of *Laser+Position* and *Laser+Track* respectively).

The controlled experiments in Chapters 5 and 6 were intended to evaluate morphological factors and input techniques, not to investigate cognition and dynamic motor control. My hypotheses are thus backed by indirect proofs and should be evaluated specifically in future work.

Conclusion and Perspectives

Large display environments feature one or more large display(s) and rich input capabilities to visualize and manipulate very large datasets. However they also constrain the design of interaction techniques: users can walk around the visualization and need similarly rich interaction capabilities; they can see very small visual elements in very large visualizations and need to interact with them; they can work collaboratively on a shared display and must not disturb each other; they can carry only a limited number of input devices and can become tired, but may need very large vocabularies of possible commands and actions. Existing research on large display environments usually focused on one or two of these constraints, seldom more than that.

The goals of this thesis were (1) to better understand the constraints of large display platforms and how to design interaction for these environments, and (2) to contribute a set of efficient and combinable techniques for "classical" interaction in mid-air: virtual navigation, command selection and pointing.

Contributions

The contributions of this dissertation are of several types:

Design of interaction techniques for large display environments: In Chapter 1, I analyzed the fundamental differences between desktop and large display environments. From these differences, and the corresponding constraints, I introduced a design space for *Feedback Location* that provides insights in the collaboration capabilities of interaction techniques and their usability. The design space emphasizes the compromise between occlusion and attention switch: (1) displaying feedback on a shared display can distract other users and occlude underlying data; (2) separating feedback from visualized data makes users switch their attention from one to the other. Based on the points of this design space I also introduced a taxonomy of limb allocation. This taxonomy explores

how multiple tasks and interaction techniques can be assigned to the limited input capabilities available when standing in front of a wall-sized display. Based upon both these classifications I introduced a set of requirements for interaction techniques in large display environments to ensure that a set of interaction techniques can be used (i) at a distance, (ii) with other interaction techniques and (iii) when collaborating with other users.

New input channels: The input capabilities of users being limited when standing or walking in a large display environment, I explored which unused input channels could be used alone or in combination to improve users' expressiveness. This dissertation thus features analyses and morphological design spaces about mid-air input (Chapter 2), on-body touch (Chapter 3) and head orientation (Chapter 6). All these input channels were evaluated in controlled experiments. The results of these experiments provided insights into how to use these channels for efficient mid-air techniques. As discussed, e.g. in Chapter 3, some of these input channels and design spaces could be explored further in future work.

New interaction techniques: This dissertation presents the design and evaluation of a set of new interaction techniques for large display environments. Chapter 2 features six efficient pan-and-zoom techniques adapted to different input requirements: one- or two-handed interaction, level of physical guidance, and the path of input movements. Chapter 3 introduces several designs for discrete selection techniques that can be used while pointing. These techniques can theoretically accommodate a large number of items, e.g. 84 items in a simple example using on-body touch and the number of contact fingers (see page 79), but remain to be formally tested.

In Part II, I formally defined dual-precision pointing techniques, a family of target-agnostic techniques that can accommodate pointing tasks with very high difficulties. Such tasks are typically found when interacting with ultra-high-resolution wall-sized displays. These techniques solicit the user's sensory-motor system sequentially by allowing two (or more) phases of different velocities and precision depending on the proximity and size of the target. I implemented and evaluated a set of dual-precision techniques with two usability constraints: being one-handed or usable on small touch areas. From these analyses and studies I propose three high-precision techniques that can perform tasks of difficulties as high as 9.5 bits: *Laser+Gyro*, *ContHead* and *DiscHead*. This level of difficulty was, to my knowledge, never evaluated in bivariate pointing studies.

Finally, in Chapter 6 I introduced a novel method to calibrate velocity-based transfer functions for Pointer Acceleration pointing techniques. Techniques using this method

showed performance levels as good as the best existing pointing techniques, including those introduced in this dissertation.

Human performance in constrained situations: From the results of seven controlled experiments (Chapters 2, 3, 4, 5 and 6), I introduced two higher level considerations about human performance in constrained environments.

The conclusion of Part I introduced a trade-off between performance, simultaneity and input requirement of combined interactions. Simultaneity usually is an asset for performance, and techniques should require as few limbs as possible in order to be combined in environments with limited input expressiveness. However I observed that optimizing two of these factors usually impairs the third. Application designers should thus organize their requirements hierarchically in order to provide the best compromise to their users, depending on the interaction needs and on the input capabilities of the platform.

Second, as discussed in the conclusion of Part II, varying the resolution function of a pointing technique is necessary for difficult pointing tasks. However it adds cognitive costs that impair performance. I observed empirical evidence of two phenomena: (1) explicit changes in the resolution function causes partially unconscious adjustments of cursor velocity and precision, that seem intended to minimize pointing time and effort; (2) sudden, noticeable changes in the resolution function require mental and motor adaptation that in turn may cause longer task completion times.

The hypotheses about human performance were inferred from indirect results and require further exploration and more specific studies. It would also be interesting to investigate if these phenomena apply to other interactions.

Perspectives

The work presented in this dissertation had two goals: understanding the constraints of large display environments, and designing interaction techniques with the best performance possible. The corresponding results have two limitations: (1) they were gathered in only one type of environment, while some of them can probably be generalized to other environments, and (2) they were gathered during controlled experiments only, which may limit their external validity, e.g. regarding learning and expert behavior.

Applicability to other environments

Part I features general considerations about (1) the effect of feedback location on usability and collaboration (Chapter 1) and (2) how to physically combine interaction techniques with limited input capabilities (Chapters 1, 2 and 3). These conclusions were drawn from results gathered in large display environments, but could probably be applied to other situations such as virtual and augmented reality platforms. This would probably cause a shift in the constraints when compared with those described in Chapter 1. For example, virtual and augmented reality are less sensitive to occlusion and distraction caused by collocated users since parts of the interaction feedback can be displayed only to the user on head-mounted personal displays. Augmented reality also features one or more shared visual elements in the real world and are probably more prone to occlusion caused by collocated users' bodies than when the only shared display is a flat surface, as in large display environments. Finally, augmented and virtual reality can be used while sitting at a desktop with head-mounted displays, which provides an interesting combination of limb and device support (see Section 1.1) as well as very large visual environments. Studying environments such as these would help generalize our conclusions about feedback location and combined interactions to other platforms and contexts. This could in turn lead to more robust and general design guidelines for interaction in multi-surface and multi-display environments.

Similarly, the *Decision* and *Adaptation* effects observed with high-precision pointing in large display platforms (Part II) describe general phenomena caused by changing the resolution function of an interaction technique during its use. As such, they are likely applicable to other environments and tasks, which could be interesting to investigate. For example, zooming with two (or more) levels of precision could enable faster completion times in pan-and-zoom tasks, with very large ranges of zoom levels and regardless of the environment.

Novice and expert performance

All controlled experiments reported in this dissertation featured practice phases intended to train the participants for the tested tasks and techniques. While this training can provide insights about expert usage, it is not always equivalent to an actual, extensive use of a given technique in a particular environment. Also, while gaining interest [48, 38, 85], large display platforms remain new environments for which novice use and novice-to-expert transitions have been much less studied than in desktop environments. A development of this work could be a more thorough analysis of these aspects: How do standing and

walking affect the novice-to-expert transition of a technique, as opposed to sitting at a fixed location? How precise can highly trained users become at controlling their angular precision (Chapter 6) and stability (Chapter 3)? Such work would improve our understanding of novice and expert behavior in constrained environments, and could result in more precise guidelines for designing and combining interaction techniques in large display environments.

More generally, I discuss briefly in Section 6.3 (page 155) how users will likely adapt their movement to the resolution function of a given technique. This phenomenon could be interesting to study in more detail: How does the resolution function affect the dynamics of users' movements, i.e. their velocity and acceleration, for difficult tasks? How does the resolution function affect users' performance ceiling? Such a study, along with a deeper understanding of the *Adaptation* effect described in the conclusion of Part II, could lead to the design of co-adaptive algorithms that would smoothly adapt the resolution functions to the users' movement and level of performance.

Bibliography

Bibliography

[1] H. Akaike. A new look at the statistical model identification. *Automatic Control, IEEE Transactions on*, 19(6):716 – 723, dec 1974. ISSN 0018-9286. doi: 10.1109/TAC.1974.1100705. 136

[2] Christopher Andrews, Alex Endert, and Chris North. Space to think: large high-resolution displays for sensemaking. In *Proc. CHI '10*, pages 55–64. ACM, 2010. 19

[3] J. Ängeslevä, I. Oakley, S. Hughes, and S. O'Modhrain. Body mnemonics: portable device interaction design concept. In *Adjunct Proc. UIST*, 2003. 76

[4] Georg Apitz, François Guimbretière, and Shumin Zhai. Foundations for designing and evaluating user interfaces based on the crossing paradigm. *ACM Trans. Comput.-Hum. Interact.*, 17(2):9:1–9:42, May 2008. ISSN 1073-0516. doi: 10.1145/1746259.1746263. URL `http://doi.acm.org/10.1145/1746259.1746263`. 143

[5] Caroline Appert and Jean-Daniel Fekete. Orthozoom scroller: 1d multi-scale navigation. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, CHI '06, pages 21–30, New York, NY, USA, 2006. ACM. ISBN 1-59593-372-7. doi: 10.1145/1124772.1124776. URL `http://doi.acm.org/10.1145/1124772.1124776`. 30

[6] Caroline Appert, Olivier Chapuis, and Michel Beaudouin-Lafon. Evaluation of pointing performance on screen edges. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '08, pages 119–126, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-141-5. doi: 10.1145/1385569.1385590. URL `http://doi.acm.org/10.1145/1385569.1385590`. 105, 118

[7] Caroline Appert, Olivier Chapuis, and Emmanuel Pietriga. High-precision magnification lenses. In *CHI '10: Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 273–282. ACM, 2010. doi: http://doi.acm.org/10.1145/1753326.1753366. 111, 112

[8] Mark Ashdown, Kenji Oka, and Yoichi Sato. Combining head tracking and mouse input for a gui on multiple monitors. In *CHI '05 extended abstracts on Human factors in computing systems*, CHI EA '05, pages 1188–1191, New York, NY, USA, 2005. ACM. ISBN 1-59593-002-7. doi: 10.1145/1056808.1056873. URL `http://doi.acm.org/10.1145/1056808.1056873`. 31, 150

[9] Behrooz Ashtiani and I. Scott MacKenzie. Blinkwrite2: an improved text entry method using eye blinks. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*, ETRA '10, pages 339–345, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-994-7. doi: 10.1145/1743666.1743742. URL `http://doi.acm.org/10.1145/1743666.1743742`. 40

[10] Thomas Augsten, Konstantin Kaefer, René Meusel, Caroline Fetzer, Dorian Kanitz, Thomas Stoff, Torsten Becker, Christian Holz, and Patrick Baudisch. Multitoe: high-precision interaction with back-projected floors based on high-resolution multi-touch input. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, UIST '10, pages 209–218, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0271-5. doi: 10.1145/1866029.1866064. URL `http://doi.acm.org/10.1145/1866029.1866064`. 20

[11] Gilles Bailly. *Techniques de menus : Caractérisation, Conception et Evaluation*. PhD thesis, 2009. Thèse de doctorat Informatique préparée au Laboratoire d'Informatique de Grenoble (LIG), Université Joseph Fourier. 280 pages. 71

[12] Gilles Bailly, Alexandre Demeure, Eric Lecolinet, and Laurence Nigay. Multitouch menu (mtm). In *Proceedings of the 20th International Conference of the Association Francophone d'Interaction Homme-Machine*, IHM '08, pages 165–168, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-285-6. doi: 10.1145/1512714.1512746. URL `http://doi.acm.org/10.1145/1512714.1512746`. 41

[13] Gilles Bailly, Eric Lecolinet, and Laurence Nigay. Flower menus: a new type of marking menu with large menu breadth, within groups and efficient expert mode memorization. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '08, pages 15–22, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-141-5. doi: 10.1145/1385569.1385575. URL `http://doi.acm.org/10.1145/1385569.1385575`. 71, 79

[14] Gilles Bailly, Anne Roudaut, Eric Lecolinet, and Laurence Nigay. Menus leaf: enrichir les menus linéaires par des gestes. In *Proceedings of the 20th International Conference of the Association Francophone d'Interaction Homme-Machine*, IHM '08, pages 169–172, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-285-6. doi:

10.1145/1512714.1512747. URL http://doi.acm.org/10.1145/1512714.1512747. 74, 75

[15] Gilles Bailly, Robert Walter, Jörg Müller, Tongyan Ning, and Eric Lecolinet. Comparing free hand menu techniques for distant displays using linear, marking and finger-count menus. In *Proceedings of the 13th IFIP TC 13 international conference on Human-computer interaction - Volume Part II*, INTERACT'11, pages 248–262, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 978-3-642-23770-6. URL http://dl.acm.org/citation.cfm?id=2042118.2042143. 73, 75, 78, 80

[16] R. Bakeman. Recommended effect size statistics for repeated measures designs. *Behav. Res. Meth.*, 37(3):379–384, 2005. 163

[17] Ravin Balakrishnan. "Beating" Fitts' law: virtual enhancements for pointing facilitation. *IJHCS*, 61(6):857–874, 2004. 145

[18] Ravin Balakrishnan and I. Scott MacKenzie. Performance differences in the fingers, wrist, and forearm in computer input control. In *Proc. CHI '97*, pages 303–310. ACM, 1997. 46, 55

[19] Robert Ball, Chris North, and Doug Bowman. Move to improve: promoting physical navigation to increase user performance with large displays. In *Proc. CHI '07*, pages 191–200. ACM, 2007. 19, 29, 30, 31, 43, 45

[20] Olivier Bau, Emilien Ghomi, and Wendy Mackay. Arpege: Design and Learning of multi-finger chord gestures. article 1533, "LRI, Février 2010. URL http://www.lri.fr/~bibli/Rapports-internes/2010/RR1533.pdf. 23 pages. 78

[21] T. Baudel and M. Beaudouin-Lafon. Charade: remote control of objects using free-hand gestures. *CACM*, 36:28–35, July 1993. ISSN 0001-0782. doi: http://doi.acm.org/10.1145/159544.159562. URL http://doi.acm.org/10.1145/159544.159562. 77, 143

[22] Patrick Baudisch, Edward Cutrell, Mary Czerwinski, Daniel C. Robbins, Peter Tandler, Benjamin B. Bederson, and A. Zierlinger. Drag-and-pop and drag-and-pick: Techniques for accessing remote screen content on touch- and pen-operated systems. In *INTERACT*. IOS Press, 2003. 20, 72, 145

[23] Patrick Baudisch, Mike Sinclair, and Andrew Wilson. Soap: a pointing and gaming device for the living room and anywhere else. In *ACM SIGGRAPH 2007 emerging technologies*, SIGGRAPH '07, New York, NY, USA, 2007. ACM. ISBN 978-1-4503-1824-2. doi: 10.1145/1278280.1278298. URL http://doi.acm.org/10.1145/1278280.1278298. 52, 113

[24] M. Beaudouin-Lafon. Lessons learned from the wild room, a multisurface interactive environment. In *23rd French Speaking Conference on Human-Computer Interaction*, page 18. ACM, 2011. 27, 213

[25] M. Beaudouin-Lafon, O. Chapuis, J. Eagan, T. Gjerlufsen, S. Huot, C. Klokmose, W. Mackay, M. Nancel, E. Pietriga, C. Pillias, et al. Multi-surface interaction in the wild room. *Computer*, (99):1–1, 2012. 27, 36, 213

[26] Michel Beaudouin-Lafon. Interaction instrumentale : de la manipulation directe à la réalité augmentée. In *"Actes Neuvièmes journées francophones sur l'Interaction Homme Machine (IHM'97)*, septembre 1997. URL `http://www-ihm.lri.fr/~mbl/INSTR/ihm97-paper/ihm97.ps.gz`. 71

[27] Michel Beaudouin-Lafon. Instrumental interaction: an interaction model for designing post-wimp user interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '00, pages 446–453, New York, NY, USA, 2000. ACM. ISBN 1-58113-216-6. doi: 10.1145/332040.332473. URL `http://doi.acm.org/10.1145/332040.332473`. 33, 48

[28] Hrvoje Benko and Andrew D. Wilson. Multi-point interactions with immersive omnidirectional visualizations in a dome. In *ACM International Conference on Interactive Tabletops and Surfaces*, ITS '10, pages 19–28, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0399-6. doi: 10.1145/1936652.1936657. URL `http://doi.acm.org/10.1145/1936652.1936657`. 30

[29] Anastasia Bezerianos and Ravin Balakrishnan. The vacuum: facilitating the manipulation of distant objects. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '05, pages 361–370, New York, NY, USA, 2005. ACM. ISBN 1-58113-998-5. doi: 10.1145/1054972.1055023. URL `http://doi.acm.org/10.1145/1054972.1055023`. 20, 72, 145

[30] Xiaojun Bi, Seok-Hyung Bae, and Ravin Balakrishnan. Effects of interior bezels of tiled-monitor large displays on visual search, tunnel steering, and target selection. In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 65–74, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-929-9. doi: 10.1145/1753326.1753337. URL `http://doi.acm.org/10.1145/1753326.1753337`. 27, 116

[31] Renaud Blanch and Michaël Ortega. Rake cursor: improving pointing performance with concurrent input channels. In *Proc. CHI '09*, pages 1415–1418. ACM, 2009. ISBN 978-1-60558-246-7. doi: 10.1145/1518701.1518914. URL `http://doi.acm.org/10.1145/1518701.1518914`. 153

[32] Renaud Blanch, Yves Guiard, and Michel Beaudouin-Lafon. Semantic pointing: improving target acquisition with control-display ratio adaptation. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '04, pages 519–526, New York, NY, USA, 2004. ACM. ISBN 1-58113-702-8. doi: 10.1145/985692.985758. URL `http://doi.acm.org/10.1145/985692.985758`. 49

[33] Sebastian Boring, Dominikus Baur, Andreas Butz, Sean Gustafson, and Patrick Baudisch. Touch projector: mobile interaction through video. In *Proc. CHI '10*, pages 2287–2296. ACM, 2010. 108, 147

[34] Frédréric Bourgeois and Yves Guiard. Multiscale pointing: facilitating pan-zoom coordination. In *CHI '02 EA*, pages 758–759. ACM, 2002. 50, 52, 55

[35] D.A. Bowman, Doug A. Bowman, Larry, and L. Houges. Formalizing the design, evaluation, and application of interaction techniques for immersive virtual environments. *Journal of Visual Languages and Computing*, 10:37–53, 1999. 25

[36] Doug A. Bowman and Chadwick A. Wingrave. Design and evaluation of menu systems for immersive virtual environments. In *VR '01: Proceedings of the Virtual Reality 2001 Conference (VR'01)*, pages 149+, Washington, DC, USA, 2001. IEEE Computer Society. ISBN 0-7695-0948-7. URL `http://portal.acm.org/citation.cfm?id=835855`. 73, 75, 80

[37] W. Buxton and B. Myers. A study in two-handed input. *SIGCHI Bull.*, 17(4): 321–326, 1986. 46, 50, 52, 55

[38] William Buxton, George Fitzmaurice, Ravin Balakrishnan, and Gordon Kurtenbach. Large displays in automotive design. *IEEE CG&A*, 20(4):68–75, 2000. ISSN 0272-1716. doi: http://dx.doi.org/10.1109/38.851753. 19, 107, 184

[39] J. Callahan, D. Hopkins, M. Weiser, and B. Shneiderman. An empirical comparison of pie vs. linear menus. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '88, pages 95–100, New York, NY, USA, 1988. ACM. ISBN 0-201-14237-6. doi: 10.1145/57167.57182. URL `http://doi.acm.org/10.1145/57167.57182`. 73

[40] Bryan A. Campbell, Katharine R. O'Brien, Michael D. Byrne, and Benjamin J. Bachman. Fitts' law predictions with an alternative pointing device (wiimote®). *Human Factors and Ergonomics Society Annual Meeting Proceedings*, 52(19):1321–1325, 2008. 20, 108, 109

[41] Xiang Cao and Ravin Balakrishnan. Visionwand: interaction techniques for large displays using a passive wand tracked in 3d. pages 173–182, 2003. doi: 10.1145/

964696.964716. URL http://doi.acm.org/10.1145/964696.964716. 20, 48, 52, 74, 108, 142

[42] Stuart K. Card, Jock D. Mackinlay, and George G. Robertson. A morphological analysis of the design space of input devices. *ACM Trans. Inf. Syst.*, 9(2):99–122, April 1991. doi: 10.1145/123078.128726. URL http://doi.acm.org/10.1145/123078.128726. 109, 151, 232

[43] G. Casiez, D. Vogel, R. Balakrishnan, and A. Cockburn. The impact of control-display gain on user performance in pointing tasks. *HCI*, 23(3):215–250, 2008. URL http://dx.doi.org/10.1080/07370020802278163. 110, 111, 112, 126, 128, 132, 156, 161, 168, 175

[44] Géry Casiez and Nicolas Roussel. No more bricolage!: methods and tools to characterize, replicate and compare pointing transfer functions. In *Proc. UIST '11*, pages 603–614. ACM, 2011. ISBN 978-1-4503-0716-1. doi: http://doi.acm.org/10.1145/2047196.2047276. URL http://doi.acm.org/10.1145/2047196.2047276. 49, 109, 155, 156, 168

[45] Géry Casiez and Daniel Vogel. The effect of spring stiffness and control gain with an elastic rate control pointing device. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '08, pages 1709–1718, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-011-1. doi: 10.1145/1357054.1357321. URL http://doi.acm.org/10.1145/1357054.1357321. 109

[46] Jessica R. Cauchard, Markus Löchtefeld, Pourang Irani, Johannes Schoening, Antonio Krüger, Mike Fraser, and Sriram Subramanian. Visual separation in mobile multi-display environments. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, UIST '11, pages 451–460, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0716-1. doi: 10.1145/2047196.2047256. URL http://doi.acm.org/10.1145/2047196.2047256. 35

[47] Olivier Chapuis, Jean-Baptiste Labrune, and Emmanuel Pietriga. Dynaspot: speed-dependent area cursor. In *Proc. CHI '09*, pages 1391–1400. ACM, 2009. ISBN 978-1-60558-246-7. doi: http://doi.acm.org/10.1145/1518701.1518911. 145

[48] Fang Chen, Eric H. C. Choi, Natalie Ruiz, Yu Shi, and Ronnie Taib. User interface design and evaluation for control room. In *Proceedings of the 17th Australia conference on Computer-Human Interaction: Citizens Online: Considerations for Today and the Future*, OZCHI '05, pages 1–4, Narrabundah, Australia, Australia, 2005. Computer-Human Interaction Special Interest Group (CHISIG) of Australia. ISBN

1-59593-222-4. URL http://dl.acm.org/citation.cfm?id=1108368.1108439. 19, 184

[49] Xing Chen and James Davis. Lumipoint: Multi-user laser-based interaction on large tiled displays. *Displays*, 23(5):205–211, 2000. 107

[50] Dustin B. Chertoff, Ross W. Byers, and Joseph J. LaViola, Jr. An exploration of menu techniques using a 3d game input device. In *Proceedings of the 4th International Conference on Foundations of Digital Games*, FDG '09, pages 256–262, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-437-9. doi: 10.1145/1536513.1536559. URL http://doi.acm.org/10.1145/1536513.1536559. 73, 75

[51] Luca Chittaro and Riccardo Sioni. An electromyographic study of a laser pointer-style device vs. mouse and keyboard in an object arrangement task on a large screen. *Int. J. Hum.-Comput. Stud.*, 70(3):234–255, March 2012. ISSN 1071-5819. doi: 10.1016/j.ijhcs.2011.11.005. URL http://dx.doi.org/10.1016/j.ijhcs.2011.11.005. 31, 32, 38, 41

[52] A. Choumane, G. Casiez, and L. Grisoni. Buttonless clicking: Intuitive select and pick-release through gesture analysis. In *Virtual Reality Conference (VR), 2010 IEEE*, pages 67 –70, march 2010. doi: 10.1109/VR.2010.5444810. 143

[53] A. Cockburn, P. Quinn, C. Gutwin, G. Ramos, and J. Looser. Air pointing: Design and evaluation of spatial target acquisition with and without visual feedback. *Int. J. Hum.-Comput. Stud.*, 69(6):401–414, June 2011. ISSN 1071-5819. doi: 10.1016/j.ijhcs.2011.02.005. URL http://dx.doi.org/10.1016/j.ijhcs.2011.02.005. 32, 34

[54] Andy Cockburn, Amy Karlson, and Benjamin B. Bederson. A review of overview+detail, zooming, and focus+context interfaces. *ACM CSUR*, 41(1):1–31, 2008. doi: http://doi.acm.org/10.1145/1456650.1456652. 45

[55] Mary Czerwinski, George Robertson, Brian Meyers, Greg Smith, Daniel Robbins, and Desney Tan. Large display research overview. In *CHI '06 extended abstracts on Human factors in computing systems*, CHI EA '06, pages 69–74, New York, NY, USA, 2006. ACM. ISBN 1-59593-298-4. doi: 10.1145/1125451.1125471. URL http://doi.acm.org/10.1145/1125451.1125471. 25

[56] Kaushik Das and Christoph W. Borst. Vr menus: investigation of distance, size, auto-scale, and ray casting vs. pointer-attached-to-menu. In *Proceedings of the 6th international conference on Advances in visual computing - Volume Part I*, ISVC'10,

pages 719–728, Berlin, Heidelberg, 2010. Springer-Verlag. ISBN 3-642-17288-1, 978-3-642-17288-5. URL http://dl.acm.org/citation.cfm?id=1939921.1940002. 73

[57] Rodrigo A. de Almeida, Clément Pillias, Emmanuel Pietriga, and Pierre Cubaud. Looking behind bezels: french windows for wall displays. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 124–131, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1287-5. doi: 10.1145/2254556.2254581. URL http://doi.acm.org/10.1145/2254556.2254581. 31

[58] H. Debarba, L. Nedel, and A. Maciel. Lop-cursor: Fast and precise interaction with tiled displays using one hand and levels of precision. In *3D User Interfaces (3DUI), 2012 IEEE Symposium on*, pages 125 –132, march 2012. doi: 10.1109/3DUI.2012.6184196. 131

[59] A. Ebert, S. Thelen, P.-S. Olech, J. Meyer, and H. Hagen. Tiled++: An enhanced tiled hi-res display wall. *Visualization and Computer Graphics, IEEE Transactions on*, 16(1):120 –132, jan.-feb. 2010. ISSN 1077-2626. doi: 10.1109/TVCG.2009.57. 27

[60] Kenneth Evans, Peter Tanner, and Marceli Wein. Tablet-based valuators that provide one, two, or three degrees of freedom. In *Proc. SIGGRAPH '81*, pages 91–97. ACM, 1981. 51

[61] Stephanie Foehrenbach, Werner A. König, Jens Gerken, and Harald Reiterer. Tactile feedback enhanced hand gesture interaction at large, high-resolution displays. *JVLC*, 20(5):341–351, 2009. ISSN 1045-926X. doi: http://dx.doi.org/10.1016/j.jvlc.2009.07.005. 145

[62] Clifton Forlines, Ravin Balakrishnan, Paul Beardsley, Jeroen van Baar, and Ramesh Raskar. Zoom-and-pick: facilitating visual zooming and precision pointing with interactive handheld projectors. In *Proc. UIST '05*, pages 73–82. ACM, 2005. 145

[63] Clifton Forlines, Daniel Vogel, and Ravin Balakrishnan. Hybridpointing: fluid switching between absolute and relative pointing with a direct input device. In *Proc. UIST '06*, pages 211–220. ACM, 2006. ISBN 1-59593-313-1. doi: http://doi.acm.org/10.1145/1166253.1166286. 124, 220

[64] Greg Foster and Terence Foxcroft. Barrel menu: a new mobile phone menu for feature rich devices. In *Proceedings of the South African Institute of Computer Scientists and Information Technologists Conference on Knowledge, Innovation and Leadership in a Diverse, Multidisciplinary Environment*, SAICSIT '11, pages 97–105, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0878-6. doi: 10.1145/2072221.2072233. URL http://doi.acm.org/10.1145/2072221.2072233. 74, 75

[65] Jeremie Francone, Gilles Bailly, Laurence Nigay, and Eric Lecolinet. Wavelet menus: a stacking metaphor for adapting marking menus to mobile devices. In *Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '09, pages 49:1–49:4, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-281-8. doi: 10.1145/1613858.1613919. URL `http://doi.acm.org/10.1145/1613858.1613919`. 74, 75

[66] Edward G. Freedman and David L. Sparks. Coordination of the eyes and head: movement kinematics. *Exp. Brain Res.*, 131:22–32, 2000. doi: 10.1007/s002219900296. 150, 151, 230

[67] Scott Frees, G. Drew Kessler, and Edwin Kay. Prism interaction for enhancing control in immersive virtual environments. *ACM Trans. Comput.-Hum. Interact.*, 14(1):2, 2007. ISSN 1073-0516. doi: http://doi.acm.org/10.1145/1229855.1229857. 124, 145, 156

[68] George W. Furnas and Benjamin B. Bederson. Space-scale diagrams: understanding multiscale interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '95, pages 234–241, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co. ISBN 0-201-84705-1. doi: 10.1145/223904.223934. URL `http://dx.doi.org/10.1145/223904.223934`. 30, 47

[69] L. Gallo and A. Minutolo. Design and comparative evaluation of smoothed pointing: A velocity-oriented remote pointing enhancement technique. *IJHCS*, 70(4):287–300, 2012. 124, 156, 170

[70] Emilien Ghomi, Guillaume Faure, Stéphane Huot, Olivier Chapuis, and Michel Beaudouin-Lafon. Using rhythmic patterns as an input method. In *CHI '12: Proceedings of the 30th international conference on Human factors in computing systems*, pages 1253–1262. ACM, 2012. URL `http://doi.acm.org/10.1145/2207676.2208579`. 78, 80

[71] Tovi Grossman and Ravin Balakrishnan. The bubble cursor: enhancing target acquisition by dynamic resizing of the cursor's activation area. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '05, pages 281–290, New York, NY, USA, 2005. ACM. ISBN 1-58113-998-5. doi: 10.1145/1054972.1055012. URL `http://doi.acm.org/10.1145/1054972.1055012`. 49, 145

[72] Tiago Guerreiro, Ricardo Gamboa, and Joaquim Jorge. Gesture-based human-computer interaction and simulation. chapter Mnemonical Body Shortcuts for Interacting with Mobile Devices, pages 261–271. Springer-Verlag, Berlin, Heidel-

berg, 2009. ISBN 978-3-540-92864-5. doi: 10.1007/978-3-540-92865-2_29. URL http://dx.doi.org/10.1007/978-3-540-92865-2_29. 80

[73] Y. Guiard. Asymmetric division of labor in human skilled bimanual action: the kinematic chain as a model. *Journal of motor behavior*, 19(4):486–517, December 1987. ISSN 0022-2895. URL http://view.ncbi.nlm.nih.gov/pubmed/15136274. 39, 46, 51, 52

[74] Y. Guiard and M. Beaudouin-Lafon. Target acquisition in multiscale electronic worlds. *International Journal of Human-Computer Studies*, 61(6):875–905, 2004. 47, 49, 55, 58

[75] François Guimbretière, Maureen Stone, and Terry Winograd. Fluid interaction with high-resolution wall-size displays. In *Proc. UIST '01*, pages 21–30. ACM, 2001. ISBN 1-58113-438-X. doi: http://doi.acm.org/10.1145/502348.502353. 107

[76] S. Gustafson, C. Holz, and P. Baudisch. Imaginary Phone: learning imaginary interfaces by transferring spatial memory from a familiar dvice. In *Proc. UIST*, pages 283–292, 2011. 76

[77] Beverly L. Harrison and Kim J. Vicente. An experimental evaluation of transparent menu usage. In *Proceedings of the SIGCHI conference on Human factors in computing systems: common ground*, CHI '96, pages 391–398, New York, NY, USA, 1996. ACM. ISBN 0-89791-777-4. doi: 10.1145/238386.238583. URL http://doi.acm.org/10.1145/238386.238583. 35

[78] C. Harrison, S. Ramamurthy, and S Hudson. On-body interaction: armed and dangerous. In *Proc. TEI*, pages 69–76, 2012. 76

[79] Chris Harrison and Anind K. Dey. Lean and zoom: proximity-aware user interface and content magnification. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 507–510, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-011-1. doi: 10.1145/1357054.1357135. URL http://doi.acm.org/10.1145/1357054.1357135. 30, 40, 41

[80] Chris Harrison, Desney Tan, and Dan Morris. Skinput: appropriating the body as an input surface. In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 453–462, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-929-9. doi: 10.1145/1753326.1753394. URL http://doi.acm.org/10.1145/1753326.1753394. 20, 41, 76

[81] Kirstie Hawkey, Melanie Kellar, Derek Reilly, Tara Whalen, and Kori M. Inkpen. The proximity factor: impact of distance on co-located collaboration. In *Proceedings*

of the 2005 international ACM SIGGROUP conference on Supporting group work, GROUP '05, pages 31–40, New York, NY, USA, 2005. ACM. ISBN 1-59593-223-2. doi: 10.1145/1099203.1099209. URL http://doi.acm.org/10.1145/1099203.1099209. 21, 36

[82] Jason Hill and Carl Gutwin. The maui toolkit: Groupware widgets for group awareness. *Comput. Supported Coop. Work*, 13(5-6):539–571, December 2004. ISSN 0925-9724. doi: 10.1007/s10606-004-5063-7. URL http://dx.doi.org/10.1007/s10606-004-5063-7. 34, 35

[83] Ken Hinckley, Randy Pausch, John Goble, and Neal Kassell. A survey of design issues in spatial input. In *Proc. UIST '94*, pages 213–222. ACM, 1994. 48

[84] Ken Hinckley, Patrick Baudisch, Gonzalo Ramos, and Francois Guimbretiere. Design and analysis of delimiters for selection-action pen gesture phrases in scriboli. In *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 451–460. ACM Press, 2005. 143

[85] Elaine M. Huang, Elizabeth D. Mynatt, and Jay P. Trimble. When design just isn't enough: the unanticipated challenges of the real world for large collaborative displays. *Personal Ubiquitous Comput.*, 11(7):537–547, October 2007. ISSN 1617-4909. doi: 10.1007/s00779-006-0114-3. URL http://dx.doi.org/10.1007/s00779-006-0114-3. 19, 38, 184

[86] Stéphane Huot and Eric Lecolinet. Spiralist: a compact visualization technique for one-handed interaction with large lists on mobile devices. In *Proceedings of the 4th Nordic conference on Human-computer interaction: changing roles*, NordiCHI '06, pages 445–448, New York, NY, USA, 2006. ACM. ISBN 1-59593-325-5. doi: 10.1145/1182475.1182533. URL http://doi.acm.org/10.1145/1182475.1182533. 74, 75

[87] Stéphane Huot and Eric Lecolinet. Archmenu et thumbmenu: contrôler son dispositif mobile "sur le pouce". In *Proceedings of the 19th International Conference of the Association Francophone d'Interaction Homme-Machine*, IHM '07, pages 107–110, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-791-9. doi: 10.1145/1541436.1541457. URL http://doi.acm.org/10.1145/1541436.1541457. 74, 75

[88] Stéphane Huot, Olivier Chapuis, and Pierre Dragicevic. Torusdesktop: pointing via the backdoor is sometimes shorter. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '11, pages 829–838, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0228-9. doi: 10.1145/1978942.1979064. URL http://doi.acm.org/10.1145/1978942.1979064. 178

[89] Petra Isenberg, Anastasia Bezerianos, Pierre Dragicevic, and Jean-Daniel Fekete. A study on dual-scale data charts. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2469–2478, December 2011. ISSN 1077-2626. doi: 10.1109/TVCG.2011.160. URL `http://dx.doi.org/10.1109/TVCG.2011.160`. 21

[90] ISO. 9241-9 Ergonomic requirements for office work with visual display terminals (VDTs)-Part 9: Requirements for non-keyboard input devices. *ISO*, 2000. 216

[91] J. A. Jacko and G. Slavendy. Hierarchical menu design: breadth, depth and task complexity. In *Perceptual and Motor Skills*, volume 82, pages 1187–1201, 1996. 71

[92] R. Jacob. The use of eye movements in human-computer interaction techniques: what you look at is what you get. *ACM TOIS*, 9:152–169, 1991. doi: 10.1145/123078.128728. 150

[93] Robert Jacob and Linda Sibert. The perceptual structure of multidimensional input device selection. In *Proc. CHI '92*, pages 211–218. ACM, 1992. 46, 50, 55

[94] Robert J. K. Jacob, Linda E. Sibert, Daniel C. McFarlane, and M. Preston Mullen, Jr. Integrality and separability of input devices. *ACM Trans. Comput.-Hum. Interact.*, 1(1):3–26, March 1994. ISSN 1073-0516. doi: 10.1145/174630.174631. URL `http://doi.acm.org/10.1145/174630.174631`. 41

[95] Paul Kabbash and William 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, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co. ISBN 0-201-84705-1. doi: 10.1145/223904.223939. URL `http://dx.doi.org/10.1145/223904.223939`. 125

[96] T. Karrer, M. Wittenhagen, L. Lichtschlag, F. Heller, and J. Borchers. Pinstripe: eyes-free continuous input on interactive clothing. In *Proc. CHI*, pages 1313–1322, 2011. 20, 76, 81, 94

[97] Azam Khan, George Fitzmaurice, Don Almeida, Nicolas Burtnyk, and Gordon Kurtenbach. A remote control interface for large displays. In *Proceedings of the 17th annual ACM symposium on User interface software and technology*, UIST '04, pages 127–136, New York, NY, USA, 2004. ACM. ISBN 1-58113-957-8. doi: 10.1145/1029632.1029655. URL `http://doi.acm.org/10.1145/1029632.1029655`. 72

[98] John I. Kiger. The depth/breadth trade-off in the design of menu-driven user interfaces. *Int. J. Man-Mach. Stud.*, 20(2):201–213, March 1984. ISSN 0020-

7373. doi: 10.1016/S0020-7373(84)80018-8. URL http://dx.doi.org/10.1016/S0020-7373(84)80018-8. 71

[99] Kenrick Kin, Tom Miller, Björn Bollensdorff, Tony DeRose, Björn Hartmann, and Maneesh Agrawala. Eden: a professional multitouch tool for constructing virtual organic environments. In *Proc. CHI '11*, pages 1343–1352. ACM, 2011. ISBN 978-1-4503-0228-9. doi: 10.1145/1978942.1979141. URL http://doi.acm.org.gate6.inist.fr/10.1145/1978942.1979141. 149

[100] Kotaro Kitajima, Yoichi Sato, and Hideki Koike. Vision-based face tracking system for window interface: prototype application and empirical studies. In *Proc. CHI '01 EA*, pages 359–360. ACM, 2001. doi: 10.1145/634067.634279. 150

[101] Masatomo Kobayashi and Takeo Igarashi. Ninja cursors: using multiple cursors to assist target acquisition on large screens. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '08, pages 949–958, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-011-1. doi: 10.1145/1357054.1357201. URL http://doi.acm.org/10.1145/1357054.1357201. 178

[102] David R. Kollerl, Mark R. Mine, and Scott E. Hudson. Head-tracked orbital viewing: an interaction technique for immersive virtual environments. In *Proc. UIST '96*, pages 81–82. ACM, 1996. doi: 10.1145/237091.237103. 150

[103] W.A. König, J. Gerken, S. Dierdorf, and H. Reiterer. Adaptive pointing - design and evaluation of a precision enhancing technique for absolute pointing devices. In *Proc. Interact'09*, pages 658–671. Springer, 2009. 124, 145

[104] Werner A. König, Jens Gerken, Stefan Dierdorf, and Harald Reiterer. Adaptive pointing: implicit gain adaptation for absolute pointing devices. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, CHI EA '09, pages 4171–4176, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-247-4. doi: 10.1145/1520340.1520635. URL http://doi.acm.org/10.1145/1520340.1520635. 156

[105] M. Krueger, T. Gionfriddo, and K. Hinrichsen. VIDEOPLACE—an artificial reality. In *Proc. CHI*, pages 35–40, 1985. ISBN 0-89791-149-0. doi: 10.1145/317456.317463. URL http://doi.acm.org/10.1145/317456.317463. 76

[106] Gordon Kurtenbach and William Buxton. The limits of expert performance using hierarchic marking menus. In *Proceedings of the INTERCHI '93 conference on Human factors in computing systems*, INTERCHI '93, pages 482–487, Amsterdam,

The Netherlands, The Netherlands, 1993. IOS Press. ISBN 90-5199-133-9. URL http://dl.acm.org/citation.cfm?id=164632.164977. 73

[107] Gordon Kurtenbach, George W. Fitzmaurice, Russell N. Owen, and Thomas Baudel. The hotbox: Efficient access to a large number of menu-items. In *CHI'99*, pages 231–237, 1999. 38, 71

[108] T. K. Landauer and D. W. Nachbar. Selection from alphabetic and numeric menu trees using a touch screen: breadth, depth, and width. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '85, pages 73–78, New York, NY, USA, 1985. ACM. ISBN 0-89791-149-0. doi: 10.1145/317456.317470. URL http://doi.acm.org/10.1145/317456.317470. 71

[109] Andrea Leganchuk, Shumin Zhai, and William Buxton. Manual and cognitive benefits of two-handed input: an experimental study. *ACM ToCHI*, 5(4):326–359, 1998. 46, 51, 55

[110] Kevin A. Li, Patrick Baudisch, and Ken Hinckley. Blindsight: eyes-free access to mobile phones. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 1389–1398, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-011-1. doi: 10.1145/1357054.1357273. URL http://doi.acm.org/10.1145/1357054.1357273. 34, 48

[111] Shu-Yang Lin, Chao-Huai Su, Kai-Yin Cheng, Rong-Hao Liang, Tzu-Hao Kuo, and Bing-Yu Chen. Pub - point upon body: exploring eyes-free interaction and methods on an arm. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, UIST '11, pages 481–488, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0716-1. doi: 10.1145/2047196.2047259. URL http://doi.acm.org/10.1145/2047196.2047259. 41, 76

[112] Robert W. Lindeman, John L. Sibert, and James K. Hahn. Hand-held windows: Towards effective 2d interaction in immersive virtual environments. In *Proceedings of the IEEE Virtual Reality*, VR '99, pages 205–, Washington, DC, USA, 1999. IEEE Computer Society. ISBN 0-7695-0093-5. URL http://dl.acm.org/citation.cfm?id=554230.835715. 36, 52, 73, 74, 75

[113] Wendy E. Mackay, Caroline Appert, Michel Beaudouin-Lafon, Olivier Chapuis, Yangzhou Du, Jean-Daniel Fekete, and Yves Guiard. Touchstone: exploratory design of experiments. In *Proc. CHI '07*, pages 1425–1434. ACM, 2007. 56, 216

[114] I. Scott MacKenzie. Fitts' law as a research and design tool in human-computer interaction. *Hum.-Comput. Interact.*, 7(1):91–139, March 1992. ISSN 0737-

0024. doi: 10.1207/s15327051hci0701_3. URL http://dx.doi.org/10.1207/s15327051hci0701_3. 26, 231

[115] I. Scott MacKenzie and Shaidah Jusoh. An evaluation of two input devices for remote pointing. In *Proc. EHCI '01*, pages 235–250. Springer, 2001. ISBN 3-540-43044-X. 109

[116] Sylvain Malacria, Eric Lecolinet, and Yves Guiard. Clutch-free panning and integrated pan-zoom control on touch-sensitive surfaces: the cyclostar approach. In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 2615–2624, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-929-9. doi: 10.1145/1753326.1753724. URL http://doi.acm.org/10.1145/1753326.1753724. 30, 45, 51, 63

[117] Shahzad Malik, Abhishek Ranjan, and Ravin Balakrishnan. Interacting with large displays from a distance with vision-tracked multi-finger gestural input. In *ACM SIGGRAPH 2006 Sketches*, SIGGRAPH '06, New York, NY, USA, 2006. ACM. ISBN 1-59593-364-6. doi: 10.1145/1179849.1179856. URL http://doi.acm.org/10.1145/1179849.1179856. 20, 43, 45, 107

[118] Regan L. Mandryk and Calvin Lough. The effects of intended use on target acquisition. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '11, pages 1649–1652, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0228-9. doi: 10.1145/1978942.1979182. URL http://doi.acm.org/10.1145/1978942.1979182. 178

[119] David C. McCallum and Pourang 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*, UIST '09, pages 153–156, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-745-5. doi: 10.1145/1622176.1622205. URL http://doi.acm.org/10.1145/1622176.1622205. 21, 52, 81, 106, 124, 147, 149

[120] Michael J. McGuffin and Ravin Balakrishnan. Fitts' law and expanding targets: Experimental studies and designs for user interfaces. *ACM ToCHI*, 12(4):388–422, 2005. 145

[121] David E. Meyer, Richard A. Abrams, Sylvan Kornblum, Charles E. Wright, and J. E. Keith Smith. Optimality in human motor performance: ideal control of rapid aimed movements. *Psychological Review*, 95:340–370, 1988. 123

[122] Mark R. Mine, Frederick P. Brooks, Jr., and Carlo H. Sequin. Moving objects

in space: exploiting proprioception in virtual-environment interaction. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '97, pages 19–26, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co. ISBN 0-89791-896-7. doi: 10.1145/258734.258747. URL `http://dx.doi.org/10.1145/258734.258747`. 52, 72

[123] L.-P. Morency and Trevor Darrell. Head gesture recognition in intelligent interfaces: the role of context in improving recognition. In *Proc. IUI '06*, pages 32–38. ACM, 2006. doi: 10.1145/1111449.1111464. 150

[124] Tomer Moscovich and John F. Hughes. Navigating documents with the virtual scroll ring. In *Proc. UIST '04*, pages 57–60. ACM, 2004. 46, 51, 55

[125] Christian Müller-Tomfelde. Dwell-based pointing in applications of human computer interaction. In *Proceedings of the 11th IFIP TC 13 international conference on Human-computer interaction*, INTERACT'07, pages 560–573, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 3-540-74794-X, 978-3-540-74794-9. URL `http://dl.acm.org/citation.cfm?id=1776994.1777067`. 143

[126] Brad A. Myers, Rishi Bhatnagar, Jeffrey Nichols, Choon Hong Peck, Dave Kong, Robert Miller, and A. Chris Long. Interacting at a distance: measuring the performance of laser pointers and other devices. In *Proc. CHI '02*, pages 33–40. ACM, 2002. ISBN 1-58113-453-3. doi: http://doi.acm.org/10.1145/503376.503383. 26, 49, 107

[127] Mathieu Nancel, Emmanuel Pietriga, and Michel Beaudouin-Lafon. Precision pointing for ultra-high-resolution wall displays. Technical Report RR-7624, INRIA, 2011. 26, 39, 105, 123, 169

[128] Mathieu Nancel, Julie Wagner, Emmanuel Pietriga, Olivier Chapuis, and Wendy Mackay. Mid-air pan-and-zoom on wall-sized displays. In *Proc. CHI '11*, pages 177–186. ACM, 2011. 20, 21, 30, 41, 45

[129] Daniel Natapov, Steven J. Castellucci, and I. Scott MacKenzie. Iso 9241-9 evaluation of video game controllers. In *Proc. GI '09*, pages 223–230. Canadian Inf. Proc. Soc., 2009. 108, 109

[130] Kai Nickel and Rainer Stiefelhagen. Pointing gesture recognition based on 3d-tracking of face, hands and head orientation. In *Proc. ICMI '03*, pages 140–146. ACM, 2003. ISBN 1-58113-621-8. doi: http://doi.acm.org/10.1145/958432.958460. 108, 150

[131] Ian Oakley and Jun-Seok Park. Designing eyes-free interaction. In Ian Oakley and

Stephen Brewster, editors, *Haptic and Audio Interaction Design*, volume 4813 of *Lecture Notes in Computer Science*, pages 121–132. Springer Berlin / Heidelberg, 2007. 34, 48

[132] Ji-Young Oh and Wolfgang Stürzlinger. Laser pointers as collaborative pointing devices. In *Proc. GI'02*, pages 141–150, 2002. 107

[133] Dan R. Olsen, Jr. and Travis Nielsen. Laser pointer interaction. In *Proc. CHI '01*, pages 17–22. ACM, 2001. ISBN 1-58113-327-8. doi: http://doi.acm.org/10.1145/365024.365030. 107

[134] Alex Olwal, Steven Feiner, and Susanna Heyman. Rubbing and tapping for precise and rapid selection on touch-screen displays. In *Proc. CHI '08*, pages 295–304. ACM, 2008. 54

[135] Kay Owens. The work of glendon lean on the counting systems of papua new guinea and oceania. In *Mathematics Education Research*, volume 13, pages 47–71, 2001. URL `http://www.uog.ac.pg/glec/thesis/thesis.htm`. 72, 76, 78

[136] Kay Owens. Indigenous mathematics: A rich diversity. In *Mathematics: Shaping Australia. Proceedings of Australian Association of Mathematics Teachers*, pages 157–167, 2001. URL `http://www.uog.ac.pg/glec/Key/Kay/diversity.htm`. 72, 76, 77, 78

[137] Jeffrey S. Pierce, Andrew S. Forsberg, Matthew J. Conway, Seung Hong, Robert C. Zeleznik, and Mark R. Mine. Image plane interaction techniques in 3d immersive environments. In *Proc. I3D '97*, pages 39–42. ACM, 1997. doi: http://doi.acm.org/10.1145/253284.253303. 108

[138] Emmanuel Pietriga, Stéphane Huot, Mathieu Nancel, and Romain Primet. Rapid development of user interfaces on cluster-driven wall displays with jBricks. In *Proc. EICS '11*, pages 185–190. ACM, 2011. 56, 116, 160, 214, 216

[139] Philip Quinn, Andy Cockburn, Kari-Jouko Räihä, and Jérôme Delamarche. On the costs of multiple trajectory pointing methods. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '11, pages 859–862, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0228-9. doi: 10.1145/1978942.1979067. URL `http://doi.acm.org/10.1145/1978942.1979067`. 178

[140] Mahfuz Rahman, Sean Gustafson, Pourang Irani, and Sriram Subramanian. Tilt techniques: Investigating the dexterity of wrist-based input. In *Proc. CHI*, pages 1943–1952. ACM, 2009. doi: 10.1145/1518701.1518997. 147

[141] Kari-Jouko Räihä and Oleg Špakov. Disambiguating ninja cursors with eye gaze. In

Proc. CHI, pages 1411–1414. ACM, 2009. ISBN 978-1-60558-246-7. doi: 10.1145/1518701.1518913. URL http://doi.acm.org.gate6.inist.fr/10.1145/1518701.1518913. 153

[142] Gonzalo A. Ramos and Ravin Balakrishnan. Pressure marks. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '07, pages 1375–1384, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-593-9. doi: 10.1145/1240624.1240834. URL http://doi.acm.org/10.1145/1240624.1240834. 79, 80

[143] Umar Rashid, Miguel A. Nacenta, and Aaron Quigley. The cost of display switching: a comparison of mobile, large display and hybrid ui configurations. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 99–106, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1287-5. doi: 10.1145/2254556.2254577. URL http://doi.acm.org/10.1145/2254556.2254577. 35, 37, 41

[144] Umar Rashid, Miguel A. Nacenta, and Aaron Quigley. Factors influencing visual attention switch in multi-display user interfaces: a survey. In *Proceedings of the 2012 International Symposium on Pervasive Displays*, PerDis '12, pages 1:1–1:6, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1414-5. doi: 10.1145/2307798.2307799. URL http://doi.acm.org/10.1145/2307798.2307799. 35

[145] Jun Rekimoto. A multiple device approach for supporting whiteboard-based interactions. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '98, pages 344–351, New York, NY, USA, 1998. ACM Press/Addison-Wesley Publishing Co. ISBN 0-201-30987-4. doi: 10.1145/274644.274692. URL http://dx.doi.org/10.1145/274644.274692. 36, 41, 74, 75

[146] Xiangshi Ren, Jibin Yin, Shengdong Zhao, and Yang Li. The adaptive hybrid cursor: a pressure-based target selection technique for pen-based user interfaces. In *Proceedings of the 11th IFIP TC 13 international conference on Human-computer interaction*, INTERACT'07, pages 310–323, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 3-540-74794-X, 978-3-540-74794-9. URL http://dl.acm.org/citation.cfm?id=1776994.1777033. 145

[147] Anne Roudaut, Eric Lecolinet, and Yves Guiard. Microrolls: expanding touch-screen input vocabulary by distinguishing rolls vs. slides of the thumb. In *Proceedings of the 27th international conference on Human factors in computing systems*, CHI '09, pages 927–936, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-246-7. doi: 10.1145/1518701.1518843. URL http://doi.acm.org/10.1145/1518701.1518843. 32, 54, 62, 74

[148] Nicolas Roussel, Géry Casiez, Jonathan Aceituno, and Daniel Vogel. Giving a hand to the eyes: leveraging input accuracy for subpixel interaction. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*, UIST '12, pages 351–358, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1580-7. doi: 10.1145/2380116.2380162. URL `http://doi.acm.org/10.1145/2380116.2380162`. 115, 155

[149] Kiyohide Satoh, Shinji Uchiyama, and Hiroyuki Yamamoto. A head tracking method using bird's-eye view camera and gyroscope. In *Proceedings of the 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality*, ISMAR '04, pages 202–211, Washington, DC, USA, 2004. IEEE Computer Society. ISBN 0-7695-2191-6. doi: 10.1109/ISMAR.2004.3. URL `http://dx.doi.org/10.1109/ISMAR.2004.3`. 174

[150] Joey Scarr, Andy Cockburn, Carl Gutwin, and Philip Quinn. Dips and ceilings: understanding and supporting transitions to expertise in user interfaces. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 2741–2750, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0228-9. doi: 10.1145/1978942.1979348. URL `http://doi.acm.org/10.1145/1978942.1979348`. 34

[151] Steven C. Seow. Information theoretic models of hci: a comparison of the hick-hyman law and fitts' law. *Hum.-Comput. Interact.*, 20(3):315–352, September 2005. ISSN 0737-0024. doi: 10.1207/s15327051hci2003_3. URL `http://dx.doi.org/10.1207/s15327051hci2003_3`. 73

[152] Garth Shoemaker, Anthony Tang, and Kellogg S. Booth. Shadow reaching: a new perspective on interaction for large displays. In *Proc. UIST '07*, pages 53–56. ACM, 2007. ISBN 978-1-59593-679-2. doi: http://doi.acm.org/10.1145/1294211.1294221. 108

[153] Garth Shoemaker, Leah Findlater, Jessica Q. Dawson, and Kellogg S. Booth. Mid-air text input techniques for very large wall displays. In *Proceedings of Graphics Interface 2009*, GI '09, pages 231–238, Toronto, Ont., Canada, Canada, 2009. Canadian Information Processing Society. ISBN 978-1-56881-470-4. URL `http://dl.acm.org/citation.cfm?id=1555880.1555931`. 21, 33, 34, 36, 43, 48

[154] Garth Shoemaker, Takayuki Tsukitani, Yoshifumi Kitamura, and Kellogg S. Booth. Body-centric interaction techniques for very large wall displays. In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries*, NordiCHI '10, pages 463–472, New York, NY, USA, 2010. ACM. ISBN 978-1-

60558-934-3. doi: 10.1145/1868914.1868967. URL http://doi.acm.org/10.1145/1868914.1868967. 49, 76

[155] Lauren Shupp, Robert Ball, Beth Yost, John Booker, and Chris North. Evaluation of viewport size and curvature of large, high-resolution displays. In *Proceedings of Graphics Interface 2006*, GI '06, pages 123–130, Toronto, Ont., Canada, Canada, 2006. Canadian Information Processing Society. ISBN 1-56881-308-2. URL http://dl.acm.org/citation.cfm?id=1143079.1143100. 30

[156] K. Snowberry, S. R. Parkinson, and N Sisson. Computer display menus. In *Ergonomics*, volume 26, pages 699–712, 1983. 71

[157] R. W. Soukoreff and I. S. MacKenzie. Towards a standard for pointing device evaluation: Perspectives on 27 years of Fitts' law research in HCI. *IJHCS*, 61(6):751–789, 2004. 112

[158] J. S. Stahl. Amplitude of human head movements associated with horizontal saccades. *Exp. Brain Res.*, 126:41–54, 1999. doi: 10.1007/s002210050715. 150

[159] Sophie Stellmach and Raimund Dachselt. Look & touch: gaze-supported target acquisition. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, CHI '12, pages 2981–2990, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1015-4. doi: 10.1145/2208636.2208709. URL http://doi.acm.org/10.1145/2208636.2208709. 150

[160] Norbert A. Streitz, Jörg Geissler, Torsten Holmer, Shin'ichi Konomi, Christian Müller-Tomfelde, Wolfgang Reischl, Petra Rexroth, Peter Seitz, and Ralf Steinmetz. i-land: an interactive landscape for creativity and innovation. In *Proc. CHI '99*, pages 120–127. ACM, 1999. ISBN 0-201-48559-1. doi: http://doi.acm.org/10.1145/302979.303010. 107

[161] Desney S. Tan, Darren Gergle, Peter Scupelli, and Randy Pausch. Physically large displays improve performance on spatial tasks. *ACM Trans. Comput.-Hum. Interact.*, 13(1):71–99, March 2006. ISSN 1073-0516. doi: 10.1145/1143518.1143521. URL http://doi.acm.org/10.1145/1143518.1143521. 30

[162] Nattapong Tongrod, Adisorn Tuantranont, and Teerakiat Kerdcharoen. Design and development of data-glove based on printed polymeric sensors and zigbee networks for human computer interface. In *Proceedings of the 4th International Convention on Rehabilitation Engineering & Assistive Technology*, iCREATe '10, pages 47:1–47:4, Kaki Bukit TechPark II,, Singapore, 2010. Singapore Therapeutic, Assistive

& Rehabilitative Technologies (START) Centre. ISBN 978-981-08-6199-5. URL http://dl.acm.org/citation.cfm?id=1926058.1926105. 79, 80

[163] Daniel Vogel and Ravin Balakrishnan. Distant freehand pointing and clicking on very large, high resolution displays. In *Proc. UIST '05*, pages 33–42. ACM, 2005. ISBN 1-59593-271-2. doi: http://doi.acm.org/10.1145/1095034.1095041. 52, 81, 117, 124, 128, 143, 169

[164] Daniel Vogel and Ravin Balakrishnan. Occlusion-aware interfaces. In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 263–272, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-929-9. doi: 10.1145/1753326.1753365. URL http://doi.acm.org/10.1145/1753326.1753365. 35

[165] Daniel Vogel and Géry Casiez. Hand occlusion on a multi-touch tabletop. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '12, pages 2307–2316, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1015-4. doi: 10.1145/2207676.2208390. URL http://doi.acm.org/10.1145/2207676.2208390. 35

[166] J. Voisin, Y. Lamarre, and C. Chapman. Haptic discrimination of object shape in humans: contribution of cutaneous and proprioceptive inputs. *Experimental Brain Research*, 145(2):251–260, May 2002. ISSN 0014-4819. doi: 10.1007/s00221-002-1118-5. URL http://www.springerlink.com/index/10.1007/s00221-002-1118-5. 76

[167] Julie Wagner, Stéphane Huot, and Wendy Mackay. Bitouch and bipad: designing bimanual interaction for hand-held tablets. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, CHI '12, pages 2317–2326, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1015-4. doi: 10.1145/2208276.2208391. URL http://doi.acm.org/10.1145/2208276.2208391. 31, 39, 40, 41, 74, 75, 148

[168] James R. Wallace, Regan L. Mandryk, and Kori M. Inkpen. Comparing content and input redirection in mdes. In *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, CSCW '08, pages 157–166, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-007-4. doi: 10.1145/1460563.1460588. URL http://doi.acm.org/10.1145/1460563.1460588. 37

[169] Robert Y. Wang and Jovan Popović. Real-time hand-tracking with a color glove. *ACM Trans. Graph.*, 28(3):63:1–63:8, July 2009. ISSN 0730-0301. doi: 10.1145/1531326.1531369. URL http://doi.acm.org/10.1145/1531326.1531369. 80

[170] Colin Ware. *Information visualization: perception for design.* Morgan Kaufmann Publishers Inc., 2004. ISBN 1-55860-511-8. 28, 117

[171] Gerold Wesche. The toolfinger: supporting complex direct manipulation in virtual environments. In *Proceedings of the workshop on Virtual environments 2003*, EGVE '03, pages 39–45, New York, NY, USA, 2003. ACM. ISBN 1-58113-686-2. doi: 10.1145/769953.769958. URL `http://doi.acm.org/10.1145/769953.769958`. 73, 75

[172] Elaine Wherry. Scroll ring performance evaluation. In *CHI '03 EA*, pages 758–759. ACM, 2003. 46, 51, 55

[173] Daniel Wigdor, Chia Shen, Clifton Forlines, and Ravin Balakrishnan. Table-centric interactive spaces for real-time collaboration. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '06, pages 103–107, New York, NY, USA, 2006. ACM. ISBN 1-59593-353-0. doi: 10.1145/1133265.1133286. URL `http://doi.acm.org/10.1145/1133265.1133286`. 20

[174] Andrew Wilson and Hubert Pham. Pointing in intelligent environments with the worldcursor. In *Proc. Interact '03*, pages 495–502, 2003. 145

[175] Graham Wilson, Craig Stewart, and Stephen A. Brewster. Pressure-based menu selection for mobile devices. In *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services*, MobileHCI '10, pages 181–190, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-835-3. doi: 10.1145/1851600.1851631. URL `http://doi.acm.org/10.1145/1851600.1851631`. 75, 79, 80

[176] Beth Yost, Yonca Haciahmetoglu, and Chris North. Beyond visual acuity: the perceptual scalability of information visualizations for large displays. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '07, pages 101–110, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-593-9. doi: 10.1145/1240624.1240639. URL `http://doi.acm.org/10.1145/1240624.1240639`. 19, 21, 30, 45

[177] Shumin Zhai, Paul Milgram, and William Buxton. The influence of muscle groups on performance of multiple degree-of-freedom input. In *Proc. CHI '96*, pages 308–315. ACM, 1996. 46, 47, 55

[178] Shumin Zhai, Carlos Morimoto, and Steven Ihde. Manual and gaze input cascaded (magic) pointing. In *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, CHI '99, pages 246–253, New York, NY,

USA, 1999. ACM. ISBN 0-201-48559-1. doi: 10.1145/302979.303053. URL http://doi.acm.org/10.1145/302979.303053. 40, 41, 150

[179] Shengdong Zhao and Ravin Balakrishnan. Simple vs. compound mark hierarchical marking menus. In *Proceedings of the 17th annual ACM symposium on User interface software and technology*, UIST '04, pages 33–42, New York, NY, USA, 2004. ACM. ISBN 1-58113-957-8. doi: 10.1145/1029632.1029639. URL http://doi.acm.org/10.1145/1029632.1029639. 34, 73, 80

[180] Shengdong Zhao, Pierre Dragicevic, Mark Chignell, Ravin Balakrishnan, and Patrick Baudisch. Earpod: eyes-free menu selection using touch input and reactive audio feedback. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '07, pages 1395–1404, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-593-9. doi: 10.1145/1240624.1240836. URL http://doi.acm.org/10.1145/1240624.1240836. 34, 48

[181] Jamie Zigelbaum, Alan Browning, Daniel Leithinger, Olivier Bau, and Hiroshi Ishii. g-stalt: a chirocentric, spatiotemporal, and telekinetic gestural interface. In *Proc. TEI '10*, pages 261–264. ACM, 2010. 48, 52

Appendices

Appendix A

General apparatus: the WILD platform

All development and experimental work described in this document was conducted in the WILD[1] room [24, 25], an experimental high-resolution, interactive platform for conducting research on collaborative human-computer interaction and the visualization of large datasets. It is located in the PCRI building of Université Paris-Sud in Orsay, France. The WILD room is composed of:

(a)

An ultra-high-resolution 5.5×1.8-meter display made of 8×4 tiled 30" desktop screens at approximately 100 ppi; it provides a unified 20480 × 6400 pixels display, or 22080 × 7360 pixels if we consider "virtual" pixels under the screen bezels.

(b)

A VICON[2] motion-capture system with 8 to 10 200 Hz cameras that track passive IR retro-reflective markers with sub-millimeter accuracy; the location and orientation of physical objects mounted with these markers can thus be tracked very precisely.

(c)

A cluster of 16 computers that communicate via a dedicated high-speed network through a front-end computer; each has two high-end nVidia 8800GT graphics cards and drives two screens of the wall display.

[1] http://insitu.lri.fr/Projects/WILD
[2] http://www.vicon.com

(d)

A 2 × 1 meter interactive table with FTIR technology that can track up to 32 simultaneous contact points with an output 1920 × 1080 resolution.

All the interaction techniques and corresponding experiments presented in this document were implemented in Java (1.4 to 1.7) using the jBricks [138] framework that enables multi-scale vector graphics and smooth integration of virtually any input source. Input devices other than the ones listed above, such as smartphones and tactile tablets, were used thanks to this library.

WILD is supported by a Région Île-de-France / Digiteo grant and by Université Paris-Sud, INRIA, CNRS, ANR and the INRIA-Microsoft joint laboratory.

Appendix B

Preliminary Experiment with Single-Mode Pointing Techniques

In this experiment we compare the performance and ease of use of three single-mode techniques (Figure B.1):

- *RayCasting* uses the location and orientation of a hand-held device;
- *Eye-Tip* also uses ray-casting but between the location of the user's eyes (estimated with a tracked hat) and the tip of a hand-held device;
- *GyroMouse* is a relative technique that maps angular movements of a hand-held device into cursor displacements.

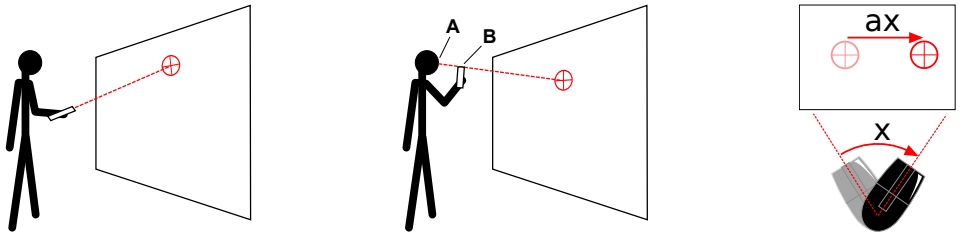


Figure B.1: The *RayCasting*, *Eye-Tip* and *GyroMouse* techniques evaluated in this experiment.

B.1 Participants

Twelve unpaid volunteers (11 male, 1 female), age 24 to 35 (mean 26.9, std deviation 2.23), all right-handed but one, participated in the experiment. The left-handed participant uses his right hand for pointing and used our techniques accordingly. Most participants were

familiar with the concept of remote interaction thanks to the WiiMote[1]. Two of them had already used computer-performed *RayCasting* techniques and one had already tried *GyroMouse*.

B.2 Apparatus

The experiment was conducted on the WILD platform (Appendix A). The graphics and behavior of the tasks and techniques were developed with jBricks [138]. The design and runtime of the experiment were developed with Touchstone [113].

The *RayCasting* and *Eye-Tip* techniques were implemented by attaching reflective markers tracked by the VICON system onto passive devices. The *GyroMouse* was the Logitech MX Air described earlier. The bias of the tracking system causes some twitching. We measured the standard deviation of the cursor position with the *RayCasting* technique when the device is held stationary 3 meters away from the display and found a value of 3.57 pixels horizontally 9.12 pixels vertically. We lowered these values to 1.66 and 3.87 pixels using a low-pass filter with a window size of 10 samples. Clicks were performed using a regular mouse in the non-dominant hand for all three techniques.

B.3 Task

The task was based on the ISO9241-9 standard [90]. Targets were laid out in a circular pattern. The order of appearance forced participants to perform pointing tasks in every direction. The target was presented as a bright green circle on a black background, and was always surrounded by an additional concentric circle of the same color to reduce visual search time. The target was highlighted in white when the cursor was over it. If the target was not selected after ten seconds, the trial was considered a miss (time-out).

The main factors were Technique and Gap, the distance between the participant and the display. The values of Gap were 75 (*"Close"*), 187.5 (*"Medium"*) and 300 cm (*"Far"*). Secondary factors were the target's Width and the screen distance between consecutive targets: movement Amplitude. The values of Width were 80, 236 and 514 pixels. The values of Amplitude were 1024, 3000 and 6500 pixels. We chose these values for Width and Amplitude so that we could have more than one occurrence of Fitts' ID values. A few 514-pixel-wide targets were partially overlapped by the bezels, but in such cases at least 80% of their surface was visible.

[1] `http://www.nintendo.com/wii/what/controllers`

We used a $3 \times 3 \times 3 \times 3$ repeated within-subject measures design with four independent variables. For each participant, we grouped trials into 9 blocks, one per TECHNIQUE and distance to the wall (GAP), so as not to disturb them with too many changes between techniques. The presentation order for TECHNIQUE and GAP was counterbalanced across participants using a Latin square. Within each block, WIDTH and AMPLITUDE were also counterbalanced using a Latin square. Each combination of TECHNIQUE × GAP was preceded by two training blocks of 12 trials. To summarize, we collected 3 TECHNIQUE × 3 GAP × 3 WIDTH × 3 AMPLITUDE × 12 replications × 12 participants = 11664 trials for analysis. The experiment lasted approximately one hour. For each trial we logged the movement time (*MT*) and the number of clicks outside of a target (*Outside Clicks*). At the end of the experiment, each participant was asked to rank the three techniques according to four criteria: perceived efficiency, ease of use, fatigue and overall preference.

B.4 Predictions

Since *GyroMouse* is a relative technique, we expect it to be more precise, i.e., to cause fewer errors with small targets (1), and to be slower for large AMPLITUDES (2) because of the need to clutch. We also expect *Eye-Tip* to be more precise than *RayCasting* (3) as users can increase angular precision by extending their arm. Finally we expect a negative effect of GAP on accuracy for the two absolute techniques, because the precision is angle-dependent (4).

B.5 Results

We analyzed the collected data with one-way ANOVAs, and performed Tukey HSD post-hoc tests for pairwise comparisons, accounting for repeated measures using JMP's REML procedure. We checked that TECHNIQUE presentation order did not have any significant effect. The only *TimeOuts* occurred in the *Eye-Tip* × *Far* condition (0.08% of all trials).

B.5.1 Movement Time

For this analysis we removed the trials that timed-out. We considered that *Outside Clicks* were a part of the pointing adjustment and included them in the analyses.

As expected for a pointing task we found significant effects on *MT* for WIDTH ($F_{2,22} = 819.52$, $p < 0.0001$), AMPLITUDE ($F_{2,22} = 847.65$, $p < 0.0001$) and WIDTH × AMPLITUDE ($F_{4,44.01} = 10.28$, $p < 0.0001$). We also found a significant effect of GAP on *MT* ($F_{2,22} = 29.83$,

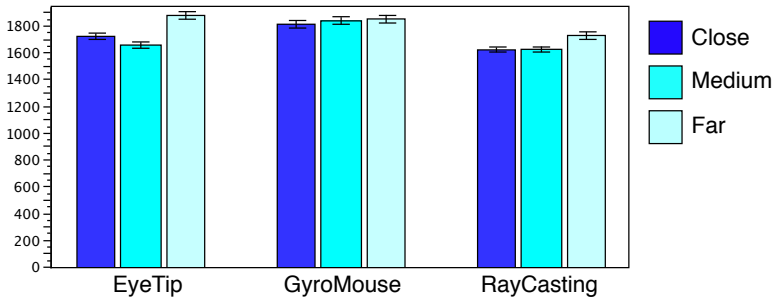


Figure B.2: Movement time (*MT*) per GAP × TECHNIQUE.

$p < 0.0001$). Selections were significantly slower when the participants were *Far* (mean 1977ms) than when they were at *Medium* distance (1805ms) or *Close* (1791ms) to the display (Fig. B.2).

We found a significant effect on *MT* for TECHNIQUE ($F_{2,22} = 3.58$, $p = 0.045$), but a Tukey post-hoc test showed no difference among the techniques. This prompts for a further analysis of interaction effects.

The following interaction effects were significant for *MT*: TECHNIQUE × GAP ($F_{4,44} = 4.19$, $p = 0.0058$), TECHNIQUE × WIDTH ($F_{4,44} = 7.7$, $p < 0.0001$) and TECHNIQUE × AMPLITUDE ($F_{4,43.99} = 26.56$, $p < 0.0001$). *RayCasting* is faster than *Eye-Tip* for any GAP, while *GyroMouse* is the slowest for *Close* and *Medium* but fastest for *Far*. There is no significant difference between the three techniques for small targets and *GyroMouse* is significantly slower than *RayCasting* for medium and large targets. Finally there is no difference among techniques for AMPLITUDE = 1024 and 3000 pixels while *GyroMouse* is significantly slower than *RayCasting* and *Eye-Tip* for AMPLITUDE = 6500 pixels, supporting prediction (2).

B.5.2 Error Rate

We now consider *Outside Clicks* as errors. We found a significant effect on error rate for TECHNIQUE ($F_{2,21.99} = 24.33$, $p < 0.0001$): *RayCasting* (mean 44.5%) and *Eye-Tip* (38.1%) generate more errors than *GyroMouse* (24.0%), supporting prediction (1).

We found a significant effect for GAP ($F_{2,22} = 60.92$, $p < 0.0001$), WIDTH ($F_{2,22} = 177.51$, $p < 0.0001$) and AMPLITUDE ($F_{2,22} = 52.36$, $p < 0.0001$). The number of errors significantly decreases with GAP (means 46.9, 33.6 and 26.1%), when target WIDTH increases (means 77, 21.2 and 8.4%) and when AMPLITUDE decreases (means 48.1, 32.6 and 25.9%).

We found a significant interaction effect for TECHNIQUE × GAP ($F_{4,43.97} = 7.56$, $p < 0.0001$) (Fig. B.3). There is no difference between techniques for GAP = *Close*. The only significant differences are that *RayCasting* is less precise for GAP = *Far* than *Medium* and

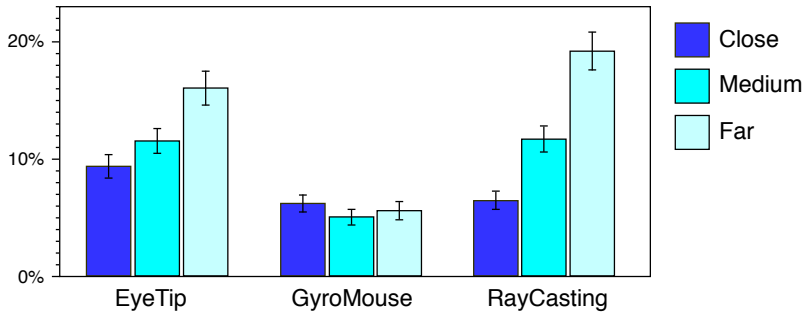


Figure B.3: Error rate (*Outside Clicks*) per GAP × TECHNIQUE.

Close; and that *Eye-Tip* is less precise for GAP = *Far* than *Close*. These results partially support prediction (4). There is no significant evidence that *Eye-Tip* causes less errors than *RayCasting*. Prediction(3) is thus not validated.

We found a significant interaction effect for TECHNIQUE × WIDTH ($F_{4,43.99} = 21.64$, $p < 0.0001$). There is no difference between the three techniques for large and medium targets. For small targets, participants made significantly more errors with *RayCasting* (mean 99.7%) and *Eye-Tip* (84.8%) than with *GyroMouse* (46.5%). These very high rates confirm that these techniques cannot be used as is for high-precision distant pointing.

Finally we found a significant interaction effect for TECHNIQUE × AMPLITUDE ($F_{4,44.02} = 14.23$, $p < 0.0001$). There is no difference between the three values of AMPLITUDE for the *GyroMouse*. Means for *GyroMouse* are always below those of the two other techniques. Participants made significantly more errors with the large AMPLITUDE (64.2% for *RayCasting* and 53.8% for *Eye-Tip*) than in any other condition.

B.5.3 Fitts' law

We computed the Fitts' Index of Difficulty (ID) of the task for each trial with no *Outside Clicks* (92%) and performed linear fits for *MT*. The resulting r^2 for *GyroMouse*, *RayCasting* and *Eye-Tip* are respectively .95, .94 and .97, supporting the fact that distant pointing with these techniques follows Fitts' law. In particular, we note that clutching did not seem to affect the fit for *GyroMouse*.

The equations of the fitting lines show very similar slopes, with a small advantage for *RayCasting*:

$$\begin{aligned} \textit{Eye-Tip}: &\quad MT(ms) = -327 + 451 * ID \\ \textit{GyroMouse}: &\quad MT(ms) = -489 + 498 * ID \\ \textit{RayCasting}: &\quad MT(ms) = -310 + 430 * ID \end{aligned}$$

B.5.4 Subjective results

We found no effect for TECHNIQUE on *Perceived efficiency* and *Overall preference*. *Eye-Tip* was perceived as the most tiring technique, followed by *RayCasting* and then *GyroMouse*. *RayCasting* was perceived as the easiest technique and *GyroMouse* the hardest one ; *Eye-Tip* was in-between but not significantly different from either of them.

B.6 Summary

In summary, we found that the gyroscopic mouse is more precise, making it possible to acquire smaller targets, but also slower for large amplitudes. Eye-Tip is consistently slower than ray-casting but is also more precise. These results support Forlines et al.'s findings [63] that absolute techniques are faster for large movements and that relative techniques are better for small targets.

Appendix C

Head and Chest Orientations

This study was a preliminary work for the project I describe in Chapter 6. The goal of this study was to observe how head movements in a pointing task on a high resolution wall display correlate with pointing. To assess whether we can exploit head position as a natural trait for coarse pointing, we sought to answer the following questions:

Q1 **How close is the head's position relative to the user's area of interest (or target position) before selection occurs?**
This will inform us about the viability of head and chest orientations as input channels for a pointing technique, as well as give us an appropriate size for the *Coarse* mode cursor (parameter L in our framework).

Q2 **How quickly does head movement take place with respect to upper limb movement in a target pointing task?**
Answering this question can assist us in determining whether switching from head positioning to another hand-based pointing mode results in a significant time overhead. This can assist us in producing the appropriate visual feedback to inform participants of the cursor movements if we observe a high difference between head and hand movements.

Q3 **How precise are users with their head orientation without feedback?**
Answering this question can help us design techniques using head orientation without adding visual disturbance about this head orientation.

Participants were asked to perform a reciprocal pointing task with two simple pointing techniques: *RayCasting* emulated laser pointing by putting the cursor at the intersection between the ray-casting of the device and the display, and *Pointer Acceleration* transformed angular movements of the device into relative cursor displacements using a transfer

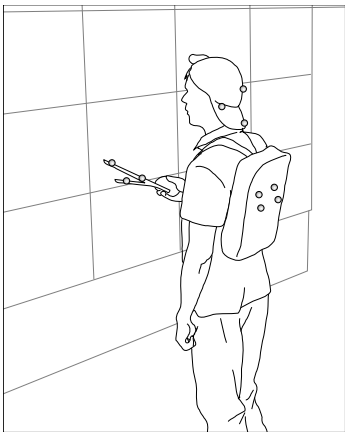


Figure C.1: Apparatus of the experiment: participants wore a hat and a backpack mounted with passive infrared markers. A handheld device was also tracked for pointing.

function. We chose a linear transfer function for simplicity:

$$CD(v) = 264 \times v + 1754$$

with CD expressed in pixels per radians and v in radians per seconds. *Pointer Acceleration* is a relative technique so a difference may appear between the hand-held device orientation and the cursor position. In order to prevent this difference to accumulate over trials, each trial started with a calibration phase where users were asked to physically reorient their device towards the location of the previous target.

In the first half of the experiment, users were free to move their head and body as they felt comfortable and we only asked them not to move their feet. In the second half, we asked them to move their head during the pointing task as if they were also pointing with it, but without additional feedback. The instruction was to have their head oriented towards the target when they clicked; whether their head movement preceded their hand movement or not was up to them.

C.1 Participants

We recruited 12 right-handed participants, 10 male and 2 female, aging from 25 to 36 (mean 28, stdev 1.23) from the IT field. All the participants that needed eye-correction (8) were using it.

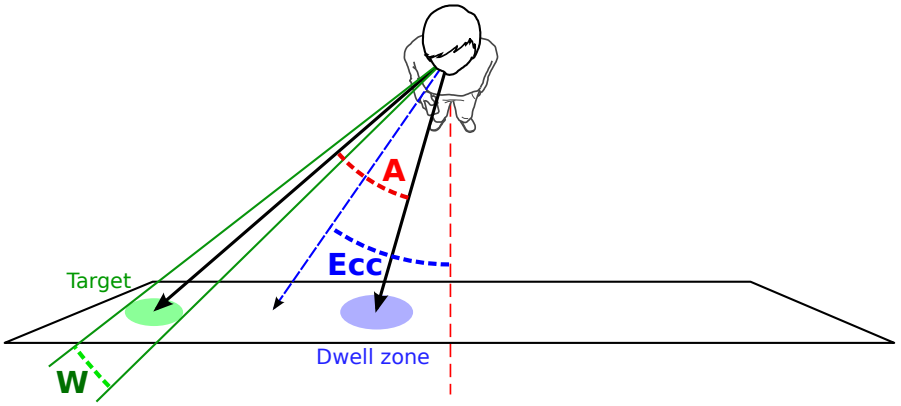


Figure C.2: The task parameters we varied in the experiment: A is the angular distance between the dwell zone and the target, Ecc is the angular distance between the sagittal plane of the user and the angular middle of the dwell zone and the target, and W is the angular width of the target. W was constant during the experiment.

C.2 Apparatus

Participants wore a hat and a backpack (Figure C.1) which were both tracked with a VICON tracking system to understand the physiological mechanisms involved with head movement while pointing. Participants also held a wireless mouse in their dominant hand which was tracked with the same system (the events from the optical system of the mouse were discarded). We used the display of the WILD room (Appendix A).

In order to have constant visual sizes during the trials, the location and dimensions of the targets were defined in angular units, as shown in Fig. C.2. Participants were displayed ellipses whose width and height were visually equal, depending on the eccentricity of the target from the center of the screen, in order for them to always perceive circular targets[1] (see Figure C.3). The same visual transformation was applied to the cursor.

C.3 Task

At the beginning of a trial, a blue circle of visual diameter $\pi/48$ radians (the dwell zone) was displayed. Participants were asked to move their cursor inside it using simulated laser pointing and stay within for half a second. Then this dwell zone disappeared and the actual target, a green disc of visual diameter $\pi/64$ radians, was displayed. Users were then to acquire this disc using the current technique and click it using the left button of the wireless mouse. When the cursor was over the target, the target became white and

[1] We limited our perspective correction to the width and height of the ellipse and did not distort the ellipse itself.

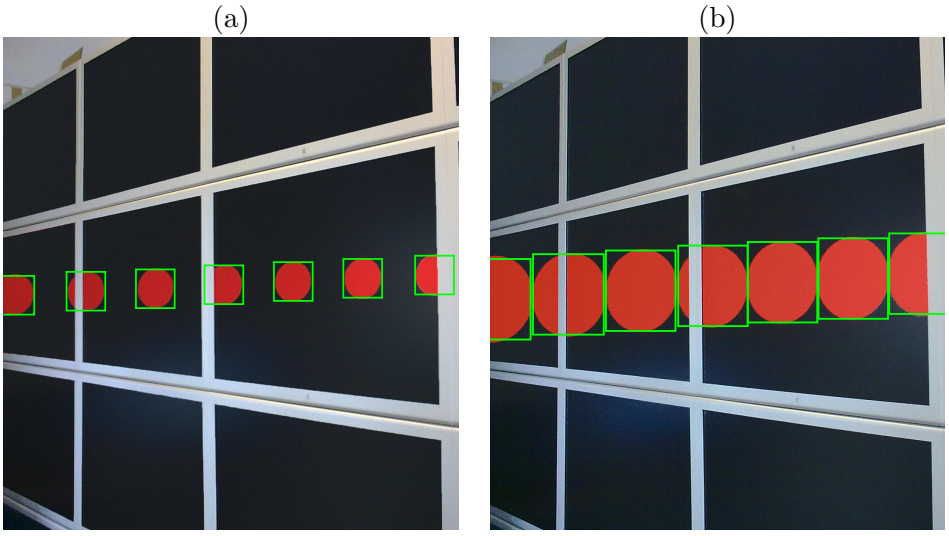


Figure C.3: The targets were displayed so that participants saw circular (rather than elliptic) targets when standing precisely where instructed to (a). However we checked that slight changes in head location did not change the perspective effect too much: image (b) was taken 20 cm to the left of the instructed head location. We added the green squares and rectangles in this figure to account for the perspective effect induced by the screen bezels; all squares (a) and rectangles (b) are the same size in their respective image.

and additional green circle was displayed around it. The trials ended until the participant clicked the target. Then the dwell zone and the target switched positions, the target was hidden and the dwell zone revealed.

C.4 Design

We varied the TECHNIQUE, INSTRUC, A and ECC. ECC is the horizontal angle of the point located between the dwell zone and the target (see Figure C.2).

We used two techniques (TECHNIQUE $\in$ {*RayCasting* ; *Pointer Acceleration*}), two instruction conditions (INSTRUC $\in$ {*Instruction* ; *NoInstruction*}), three amplitudes (A): *Large* ($\frac{3\pi}{10}$ rad, 54°), *Medium* ($\frac{3\pi}{20}$ rad, 27°) and *Small* ($\frac{3\pi}{40}$ rad, 13.5°), three eccentricities (ECC): *Center* (0 rad), *Left* ($-.13\pi$ rad, $-23.4°$) and *Right* ($.13\pi$ rad, 23.4°) and two pointing directions (POINTDIR): right to left and left to right. Fig. C.2 shows the ECC and A factors.

INSTRUC was always in the order *NoInstruction* then *Instruction*, TECHNIQUE and ECC were counterbalanced using latin square designs and A was randomized.

C.5 Measures

We measured the time needed to complete a trial (movement time *MT*) and whether a trial contained clicks outside the target (*ErrorRate*). At the end of each trial we also logged the angle of the cursor (*CursorPos*), the orientation of the head (*HeadPos*) and of the torso (*BodyPos*).

At the end of the experiment we asked the participant to rate the two techniques for both *Instruction* and *NoInstruction* on five-level Likert scales for comfort, speed and accuracy.

C.6 Results

C.6.1 Subjective Results

The differences in ages, gender and eye correction made no significant difference in the subjective results.

Ordinal logistic ANOVA's for the model TECHNIQUE×INSTRUC reveal a significant effect of TECHNIQUE for each ordinal measure comfort, speed and accuracy ($\chi^2_{1,1} = 11.7$ $p = 0.0006$, $\chi^2_{1,1} = 10.6$ $p = 0.0011$, $\chi^2_{1,1} = 6.14$ $p = 0.0132$). *RayCasting* was evaluated more comfortable, faster and more accurate than *Pointer Acceleration*. The ANOVA reveals no significant effects of INSTRUC.

The above confirm the hypothesis that *RayCasting* is preferred over *Pointer Acceleration* in front of a wall display for acquisition of large targets. Moreover, adding a constraint on head accuracy does not seem to affect user preference.

C.6.2 Movement Time & Errors

Figure C.4 shows the movement time and the error rate for each technique with and without instruction. We analyzed these measures in the full factorial model:

$$\text{INSTRUC}\times\text{A}\times\text{ECC}\times\text{Rand}(\text{PARTICIPANT})$$

for each techniques (we did not wanted to compare the techniques at this level).

Without surprise, the ANOVA reveals an effects of A on *MT* for both *RayCasting* and *Pointer Acceleration* ($F_{2,22} = 344$, $p < 0.0001$ and $F_{2,22} = 185$, $p < 0.0001$) – *MT* grows with the amplitude. These are the only simple significant effects on *MT*, and there is only one significant interaction: for *RayCasting*, there is a significant interaction INSTRUC×A ($F_{2,22} = 3.45$, $p = 0.049$). This interaction comes from the fact that participants were 54

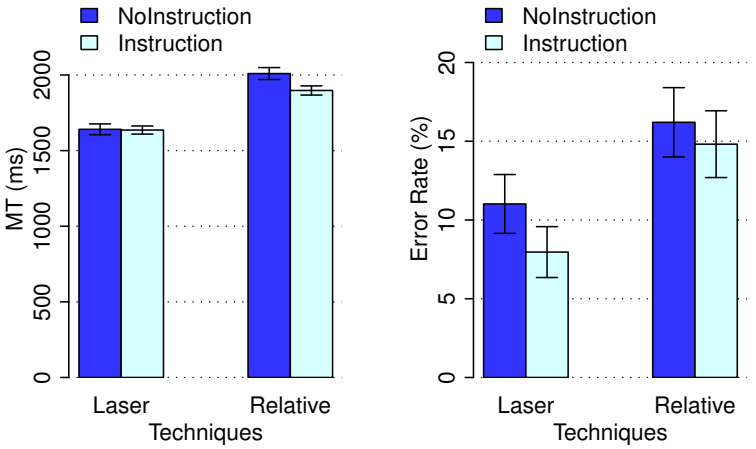


Figure C.4: Movement time (right) and error rate (left) for each technique, with and without instruction.

ms faster with instruction for large distance, but 40 ms slower for small distance. For Laser, participants were significantly faster with instruction for large A and slower for small A. This might be caused by the fact that for small distance participants had to force themselves to move their head, while this might be more natural for large distances.

Regarding errors rate, for *RayCasting* we have a significant effect of A ($F_{2,22} = 5.14$, $p = 0.015$) and a significant interaction A×INSTRUC ($F_{2,22} = 3.92$, $p = 0.035$). Indeed we measured a significantly worse error rate (17%) for large distance without instruction than for all other A×INSTRUC conditions (about 8%). For *Pointer Acceleration* the only significant result is an effect of ECC ($F_{2,22} = 5.14$, $p = 0.020$). There are fewer errors when the targets are on the left of the screen.

Overall, we did not measure any significant results showing that instruction degrades performance or accuracy. Also we did not measure any effect of eccentricity.

C.6.3 Head position vs Cursor Position

We are interested in comparing the position of where the head looks at on the screen and the position of the cursor. For simplicity we restrict ourselves to the x axis and we consider the angle from the participant to the point of the screen in front of him and the cursor, or the point where the head points on the screen.

Table C.1 provides the correlation $CursorPos = a.HeadPos + b$ for (i) all the data; (ii) by participant (see Figure C.5 for an example) and then averaged; and (iii) by participant with a direction correction term and then averaged.

Instruc	*RayCasting*			*Pointer Acceleration*		
No/Yes	a	b	r^2	a	b	r^2
ALL DATA						
No	0.862	-0.021	0.599	0.929	-0.045	0.597
Yes	0.888	-0.031	0.800	0.940	-0.061	0.781
BY PARTICIPANT AND THEN AVERAGE						
No	1.173	-0.065	0.797	1.246	-0.082	0.782
Yes	1.079	-0.059	0.926	1.100	-0.089	0.892
DIRECTION CORRECTION, BY PARTICIPANT AND THEN AVERAGE						
No	1.034	-0.199	0.888	1.102	-0.242	0.916
Yes	1.024	-0.097	0.944	0.997	-0.179	0.945

Table C.1: $CursorPos = a.HeadPos + b$ correlation for all the data, by participant and then averaged and by participant with a direction correction term and then averaged

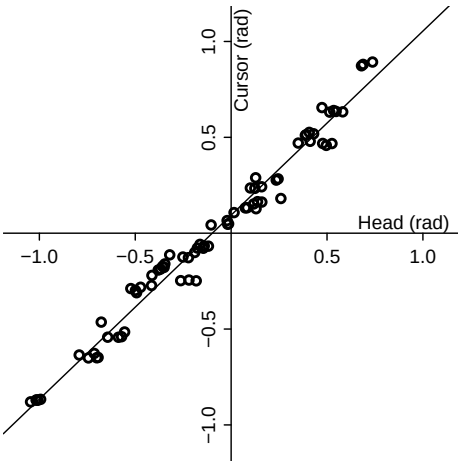


Figure C.5: An example of a correlation between the Head Screen Position (radian) and the cursor screen position at click time (participant 10, *RayCasting* with instruction, $a = 0.960, b = 0.094, r^2 = 0.977$.

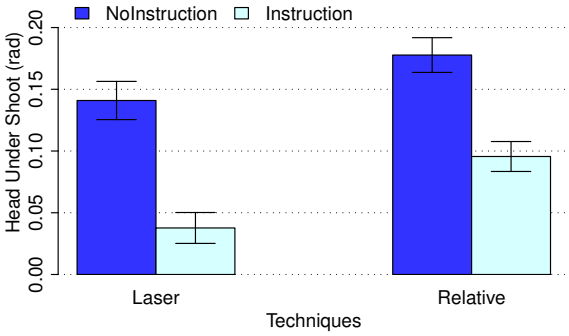


Figure C.6: Head undershoot for each technique, with and without instruction.

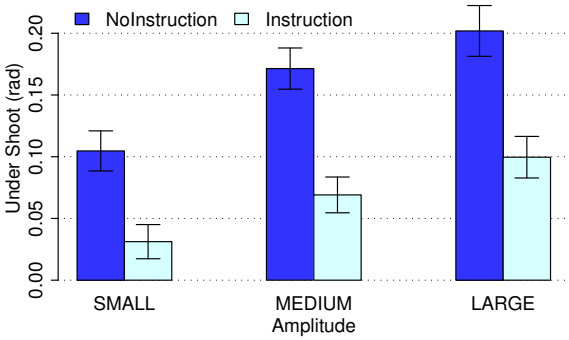


Figure C.7: Head undershoot by A × INSTRUC

Note that whichever the method, the b parameter is always negative, indicating that *HeadPos* is smaller than *CursorPos* on average. We computed the head *Undershoot* as the horizontal, angular difference between the head orientation and the target angle (*Undershoot* = *TargetAngle* − *HeadPos*), POINTDIR-wise. *Undershoot* is significantly affected by INSTRUC ($F_{1,11} = 7.43$, $p = 0.0197$), TECHNIQUE ($F_{1,11} = 4.91$, $p = 0.0487$) and AMPLITUDE ($F_{2,22} = 27.48$, $p < 0.0001$), but not by ECC not POINTDIR. Post-hoc tests revealed that head *Undershoot* was higher (i) without instruction and (ii) with *Pointer Acceleration*, as shown in Figure C.6. *Undershoot* also increased significantly with A (Figure C.7).

We want to compute a natural "size" for a head area cursor. For this propose we compute the error (difference) from the real data and the prediction given by the correlation (participant by participant). This leads to distributions (TECHNIQUE × INSTRUC) as shown in Figure C.8. Then, we want 95 % of the clicks to be inside the head area cursor. This gives the following size in radian (degree):

INSTRUC	*RayCasting*	*Pointer Acceleration*
No	0.542 (31.1)	0.613 (35.1)
Yes	0.364 (20.9)	0.461 (26.4)

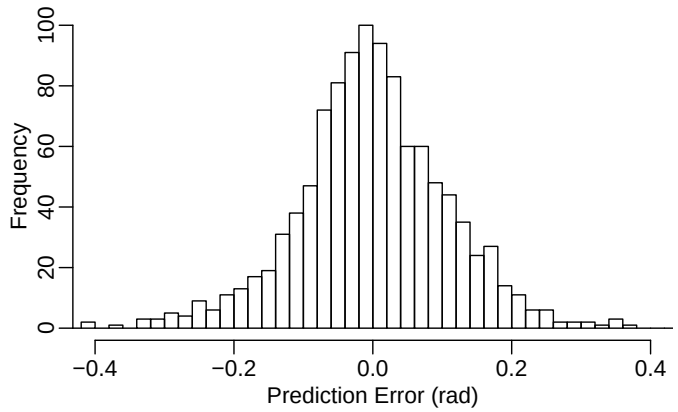


Figure C.8: Distribution of the prediction errors for *RayCasting* with instruction.

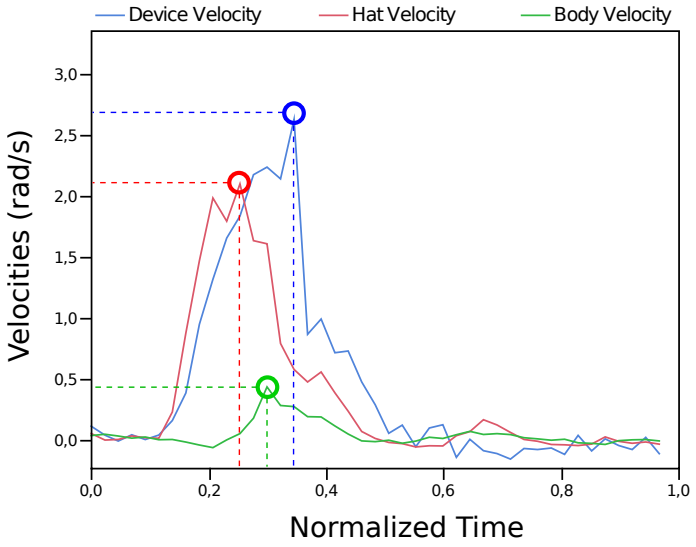


Figure C.9: Example of device, head and body velocities during a trial.

These rather large angles show that head orientation is not very precise without visual feedback, at least with a precision criterion as high as 95 %. This size remains to be computed with visual feedback.

C.6.4 Kinematics

For each of the tracked body parts we identified the maximum angular velocity value of each trial: *Device Velocity Peak*, *Head Velocity Peak* and *Body Velocity Peak*, and the normalized time when they occurred: *Device Peak Time*, *Head Peak Time* and *Body Peak Time* as shown in Fig. C.9.

The angular movement amplitude of the cursor during a trial (*Cursor Movement*) is sig-

nificantly affected by *Device Velocity Peak* ($F_{1,11.33} = 29.64$, $p = 0.0002$) and *Head Velocity Peak* ($F_{1,12.23} = 54.54$, $p < 0.0001$).

We also observed when the head velocity peak occurred respectively to the device velocity peak. We call *Head Earliness* the difference *Device Peak Time*$-$*Head Peak Time*. Positive values indicate that the head main movement occurred before the device main movement. We first observe that head is globally as often in advance as it is late: 51.23% of the trials have a negative *Head Earliness*. *Head Earliness* was significantly lower in trials with *NoInstruction*: -0.066 in average against -0.034 for *Instruction* trials ($F_{1,11} = 8.65$, $p = 0,0134$). *Head Earliness* also decreased significantly ($F_{2,22} = 80.98$, $p < 0.0001$) with A: 0 for *Large* A, -0.052 for *Medium* and $-0,1$ for *Small* in average.

C.7 Summary

In this preliminary experiment we showed that:

1. In accordance with Freedman and Sparks' [66] results, head orientation always undershoots the actual (horizontal) orientation of the target;
2. Head orientation can be used to infer target location, although with low accuracy;
3. This accuracy increases if participants are considered independently, which implies that calibration could improve the precision of the estimate of the target location;
4. Head rotation occurs roughly at the same time as the equivalent pointing gesture.

All these results apply only when no feedback is given about the orientation of the head.

Glossary

Cave Automatic Virtual Environment (CAVE) immersive virtual reality environment where projectors are directed to three, four, five or six of the walls of a room-sized cube.
27, 35, 230

clutching when users reach the limits of their motor operating range, they can clutch, i.e., temporarily decouple the input device from the cursor position so as to reposition the body part (finger, arm, ...) actuating the device in a more suitable configuration.
46, 51, 53–55, 62, 64, 66, 67, 109, 111–114, 116, 117, 121, 124, 126–132, 135, 139, 140, 149, 154, 156, 158, 168, 174, 178, 217, 219, 230

Control-to-Display (CD) gain multiplicative gain between input variation (Control) and cursor movement (Display); multiplicative inverse of CD ratio.
108–117, 120, 121, 123, 126–128, 132, 133, 142, 146–148, 154–156, 159, 170, 172, 175, 176, 178, 230–232

Fitts' Index of Difficulty (ID) continuous measure of the difficulty of a pointing task; it has several mathematical expression, the most common and only one used in this dissertation being the Shannon [114] expression: $ID = \log_2\left(1 + \frac{A}{W}\right)$, with A being the amplitude of the task and W the width of the target.
58, 105, 112, 120, 121, 123, 125, 132, 161, 172, 173, 175, 178, 216, 219, 230

pixel density number of pixels in a given surface unit; expressed in Points Per Inches (ppi) or Points Per Millimeters (ppm).
20, 26–31, 103, 105, 106, 110, 150, 170, 175, 230, 231

Pointer Acceleration (PA) specific resolution function where the CD gain varies with the input velocity, on the basis that slow input are meant for precise movements and fast input for fast movements.
109, 116, 120, 146, 148, 155, 156, 168, 175, 176, 182, 230

Points Per Inch (ppi) unit of pixel density; number of points per inch.
26, 28, 105, 110, 113, 114, 116, 121, 132, 139, 161, 213, 230, 231

Region Of Interest (ROI) portion of a visualization that a user is currently focused on.
30, 34–36, 46–49, 52, 230

resolution function mapping from an input domain set, e.g. mouse, touch or 3D events, to an output domain set, e.g. cursor movements [42]; the most common resolution function used for pointing is Control-to-Display (CD) gains.
109, 123, 124, 128, 151, 175, 177–179, 183–185, 230, 231

WILD Wall-sized Interaction with Large Datasets, see Appendix A.
26, 49, 54, 56, 82, 105, 116, 147, 155, 160, 216, 222, 230

WIMP Windows, Icons, Menus and Pointing: style of interaction using these elements.
34, 36, 230

Mathieu Nancel

Designing and Combining Interaction Techniques in Large Display Environments

Abstract :
Large display environments (LDEs) are interactive physical workspaces featuring one or more static large displays as well as rich interaction capabilities, and are meant to visualize and manipulate very large datasets. Research about mid-air interactions in such environments has emerged over the past decade, and a number of interaction techniques are now available for most elementary tasks such as pointing, navigating and command selection. However these techniques are often designed and evaluated separately on specific platforms and for specific use-cases or operationalizations, which makes it hard to choose, compare and combine them.
In this dissertation I propose a framework and a set of guidelines for analyzing and combining the input and output channels available in LDEs. I analyze the characteristics of LDEs in terms of (1) visual output and how it affects usability and collaboration and (2) input channels and how to combine them in rich sets of mid-air interaction techniques. These analyses lead to four design requirements intended to ensure that a set of interaction techniques can be used (i) at a distance, (ii) together with other interaction techniques and (iii) when collaborating with other users. In accordance with these requirements, I designed and evaluated a set of mid-air interaction techniques for panning and zooming, for invoking commands while pointing and for performing difficult pointing tasks with limited input requirements. For the latter I also developed two methods, one for calibrating high-precision techniques with two levels of precision and one for tuning velocity-based transfer functions. Finally, I introduce two higher-level design considerations for combining interaction techniques in input-constrained environments. Designers should take into account (1) the trade-off between minimizing limb usage and performing actions in parallel that affects overall performance, and (2) the decision and adaptation costs incurred by changing the resolution function of a pointing technique during a pointing task.

Keywords : large display environments, LDE, ultra-high-resolution displays, feedback location, task allocation, pan-and-zoom, on-body touch, dual-precision pointing, head orientation, pointer acceleration, decision, adaptation

Résumé :
Les environnements à grands écrans (Large Display Environments, LDE) sont des espaces de travail interactifs contenant un ou plusieurs grands écrans fixes et divers dispositifs d'entrée ayant pour but de permettre la visualisation et la manipulation de très grands jeux de données. La recherche s'est de plus en plus intéressé à ces environnements durant ces dix dernières années, et il existe d'ores-et-déjà un certain nombre de techniques d'interaction correspondant à la plupart des tâches élémentaires comme le pointage, la navigation et la sélection de commandes. Cependant, ces techniques sont souvent conçues et évaluées séparément, dans des environnements et des cas d'utilisations spécifiques. Il est donc difficile de les comparer et de les combiner.
Dans ce manuscrit, je propose un ensemble de guides pour l'analyse et la combinaison des canaux d'entrée et de sortie disponibles dans les LDEs. Je présente d'abord une étude de leurs caractéristiques selon deux axes: (1) le retour visuel, et la manière dont il affecte l'utilisabilité des techniques d'interaction et la collaboration co-localisée, et (2) les canaux d'entrée, et comment les combiner en d'efficaces ensembles de techniques d'interaction. Grâce à ces analyses, j'ai développé quatre pré-requis de conception destinés à assurer que des techniques d'interaction peuvent être utilisées (i) à distance, (ii) en même temps que d'autres techniques et (iii) avec d'autres utilisateurs. Suivant ces pré-requis, j'ai conçu et évalué un ensemble de techniques de navigation, d'invocation de commandes tout en pointant, et de pointage haute-précision avec des moyens d'entrée limités. J'ai également développé deux méthodes de calibration de techniques de pointage, l'une spécifique aux techniques ayant deux niveaux de précision et l'autre adaptée aux fonctions d'accélération. En conclusion, j'introduis deux considérations de plus haut niveau sur la combinaison de techniques d'interaction dans des environnements aux canaux d'entrée limités : (1) il existe un compromis entre le fait de minimiser l'utilisation des membres de l'utilisateur et celui d'effectuer des actions en parallèle qui affecte les performances de l'ensemble ; (2) changer la fonction de transfert d'une technique de pointage durant son utilisation peut avoir un effet négatif sur les performances.

Mots clés : environnements à grands écrans, LDE, écrans ultra-haute-résolution, retour visuel, assignation de tâches, pointage haute-précision, orientation de la tête, fonctions d'accélération, décision, adaptation