SVEN STROTHOFF

MULTI-TOUCH SELECTION AND INTERACTION ON AND ABOVE THE SURFACE

INFORMATIK

MULTI-TOUCH SELECTION AND INTERACTION ON AND ABOVE THE SURFACE

Inaugural-Disseration zur Erlangung des Doktorgrades der Naturwissenschaften im Fachbereich Mathematik und Informatik der Mathematisch-Naturwissenschaftlichen Fakultät der Westfälischen Wilhelms-Universität Münster

vorgelegt von

SVEN STROTHOFF

aus Herten

2015

Sven Strothoff: *Multi-touch Selection and Interaction On and Above the Surface* © 2015

DEKAN:
Prof. Dr. Martin Stein

ERSTER GUTACHTER:
Prof. Dr. Klaus Hinrichs
ZWEITER GUTACHTER:
Prof. Dr. Wolfgang Stürzlinger

TAG DER MÜNDLICHEN PRÜFUNG:
10. Juli 2015
TAG DER PROMOTION:
10. Juli 2015

ABSTRACT

Multi-touch input is the de facto standard for interaction with mobile devices. As mobile devices become more powerful users perform increasingly complex computing tasks on them, necessitating interaction techniques and interfaces that are both precise and expressive. Selection is a fundamental operation in graphical user interfaces. In a survey of existing selection techniques an overview of previous work on selection is presented. Most techniques for touch input are based on traditional selection techniques that were developed for mouse-based interfaces and only use a single touch. Using true *multi*-touch input presents an opportunity, but also a challenge, to create more powerful and expressive interaction techniques.

We present a selection technique for arbitrary regions. Using the *context* of multiple touches, i. e. their order and relative position, our technique is capable of modifying and refining selections to precisely select the desired regions of interest.

An interaction concept based on secondary touches that provide contextual information to other touch gestures called *pinning touches* is introduced. We present an interface that uses pinning touches to facilitate object grouping and tagging.

In the second part of this thesis multi-touch input is combined with a stereoscopic projection to access the third dimension above the surface. Even though objects can be visualised above the display the touch sensing is only available on the surface. We introduce *Triangle Cursor*, an indirect interaction technique for objects above the surface.

All presented interaction techniques are evaluated in user studies and include a discussion of design considerations for their integration into existing user interfaces.

PUBLICATIONS

Some ideas and figures have appeared previously in the following publications:

Strothoff, Sven, Frank Steinicke, Dirk Feldmann, Jan Roters, Klaus H. Hinrichs, Tom Vierjahn, Markus Dunkel, and Sina Mostafawy (2010). "A Virtual Reality-based Simulator for Avionic Digital Service Platforms." In: *Proceedings of Joint Virtual Reality Conference (Additional Material)*, 8 pages.

Strothoff, Sven, Dirk Feldmann, Frank Steinicke, Tom Vierjahn, and Sina Mostafawy (2011a). "Interactive Generation of Virtual Environments Using MUAVs." In: *Proceedings of International Symposium on VR Innovation*. IEEE, pp. 89–96.

Strothoff, Sven, Dimitar Valkov, and Klaus Hinrichs (2011b). "Triangle Cursor: Interactions With Objects Above the Tabletop." In: *Proceedings of ITS 2011*. ACM, pp. 111–119.

Strothoff, Sven and Klaus Hinrichs (2013a). "Adding Context to Multi-touch Region Selections." In: *Proceedings of ITS 2013*. ACM, pp. 397–400.

Strothoff, Sven and Klaus Hinrichs (2013b). "Adding Context to Multi-touch Region Selections." In: *Proceedings of MUM 2013*. ACM, 15:1–15:8.

Strothoff, Sven, Wolfgang Stuerzlinger, and Klaus Hinrichs (2015). "Pins 'n' Touches: An Interface for Tagging and Editing Complex Groups." Submitted.

ACKNOWLEDGEMENTS

First, I would like to thank my advisor Prof. Dr. Klaus Hinrichs for accompanying me through all of my time at the university—from my first lecture in computer science to my time as a PhD student.

I want to thank Prof. Dr. Wolfgang Stürzlinger for many insightful and interesting discussions and becoming the second examiner of my thesis.

I would also like to thank Prof. Dr. Frank Steinicke for introducing me into our research group. For keeping the hardware in the lab up and running—especially our capricious multi-touch *workbench*—my thanks go to Dr. Dimitar Valkov.

Two special thanks go to Edinburgh, as oddly enough, both Benjamin Risse and Michel Steuwer ended up there. Michel was always happy to discuss all kinds of strange C++ questions and introduced me to many helpful tools that have become an integral part of my workflow. Bena and I enjoyed many interesting chats.

Many thanks go to my current and former colleagues Tobias Brix, Ulrich Burgbacher, Dr. Stefan Diepenbrock, Dr. Dirk Feldmann, Dr. Florian Lindemann, Dr. Jan Roters, Dr. Jörg-Stefan Praßni, Manuel Prätorius, and Aaron Scherzinger. I will miss our lively discussions of work-related and non-work-related topics.

For proofreading and many helpful suggestions I thank Stefan Gumprich, Dr. Johannes Köster and Michel Steuwer.

I thank all volunteers that participated in the user studies.

Finally, I would like to thank Kerstin Gonschorrek for all the support she has given me, her love and her never-ending patience.

Sven Strothoff
Münster, May 2015

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

ANOVA	analysis of variance
CAD	computer-aided design
DOF	degree of freedom
FTIR	frustrated total internal reflection
GIS	geographic information system
MUAV	miniature unmanned aerial vehicle
SD	standard deviation
SE	standard error (of the mean)
UAV	unmanned aerial vehicle
VR	virtual reality

1 INTRODUCTION

Multi-touch interaction has seen a rapid growth over the past few years. Since the introduction of the iPhone by Apple in 2007, multi-touch input has become the de facto standard for smartphones and other mobile devices like tablet computers. With the availability of touch-capable notebook computers and computer displays, multi-touch input is available on an ever-increasing number of devices. While multi-touch input had been invented many years before, the introduction of the iPhone initiated its ascent to a mainstream input technology. Nowadays, billions of people around the world use multi-touch interfaces every day.

Many interfaces are called multi-touch interfaces, although only a single touch is used for interaction. Often the fact that the hardware is capable of detecting multiple simultaneous touches is the only reason for an interface to be labelled as multi-touch. Most user interfaces for touch input are heavily influenced by traditional graphical user interfaces. The computer mouse was the dominant input device for graphical interfaces for more than three decades. User interfaces that were designed for mouse-based input are often just transferred to a new touch-based environment by replacing the mouse cursor with a touch. While there are benefits of using traditional interaction schemes that users are familiar with, there is also a huge potential to improve the interaction by adapting the techniques to the novel capabilities provided by multi-touch input.

In this thesis we present three interaction techniques for multi-touch input that contribute to different areas. The common goal of all these techniques is to enable more expressive and powerful interaction by combining multiple touches. We present user studies for all interaction techniques to verify our design decisions and examine their performance compared to existing techniques. As we are designing *user interface* techniques the participants' subjective ratings and observations during the experiments are considered in addition to these performance characteristics. The discussion of each technique is concluded with an examination of design considerations for integrating our techniques into existing user interfaces. Challenges that arise from the combination with other touch gestures and possible solutions are discussed. In addition, several design variants that might be more suitable for different applications and usage scenarios are presented.

In the first part of this thesis we focus on *regular* multi-touch interaction on the surface. An extensive survey of existing selection techniques is presented in Chapter 2. Using a system of classifiers previous work is categorised and structured. A graphical overview of the examined body of related work is used to identify areas that have not been explored. This result motivates the development of two novel multi-touch selection techniques presented in the following two chapters.

Chapter 3 presents our novel multi-touch region selection technique (Strothoff and Hinrichs 2013a,b). Through the use of multiple touches and their *context*—their order and relative position—our technique can not only precisely select regions, but also modify and refine existing selections. The contextual information allows touch-based interfaces to achieve a level of control that was previously only available for mouse-based region selection techniques.

In Chapter 4 we extend the touch input vocabulary by introducing the concept of *pinning* touches. Pinning touches are stationary touches that are used in addition to other touch actions. Complementing existing touch gestures with the contextual information provided by pinning touches enables more expressive touch interaction. We present a group selection and tagging interface (Strothoff et al. 2015) based on the concept of pinning touches.

The second part of this thesis takes a leap into the third dimension above the surface. By combining touch-capable displays with a stereoscopic projection that enables the visualisation of objects above the display's surface we can access the volume above the display. As touches can only be sensed on the surface, objects displayed above the surface can only be interacted with indirectly.

Chapter 5 introduces *Triangle Cursor* (Strothoff et al. 2011b) an indirect interaction technique for multi-touch input. Using Triangle Cursor it is possible to select and interact with objects that are displayed in the volume above the surface.

Chapter 6 concludes the thesis.

In the following section we take a brief look at project AVIGLE, a research project with challenging user interface requirements. The focus of this thesis remains on multi-touch interaction. We introduce project AVIGLE to create a backdrop for the user interface requirements discovered during the project that inspired the interaction techniques presented in the following chapters.

1.1 PROJECT AVIGLE

Project AVIGLE[1] was an industrial research project in which three universities and seven members of the high-tech industry combined their domain knowledge and expertise to create a new multifunctional aerial service platform. The project was funded by the Hightech.NRW initiative of the Ministry of Innovation, Science, and Research of the German state of North Rhine-Westphalia and the European Union. University working groups focused on flight dynamics, communication networks, computer graphics, virtual reality (VR) and visualisation. The universities worked together

1 http://www.avigle.de

with industry experts for building miniature unmanned aerial vehicles (MUAVs), wireless communication technology and traditional aerial photogrammetry. The goal of the project was to create a swarm of partly autonomous flying robots to provide inexpensive alternatives to existing approaches.

The project aimed at developing a swarm of MUAVs for two main objectives.

VIRTUAL REALITY By acquiring aerial photographs 3D geometrical information of buildings and landscapes can be extracted to develop an efficient process for creating 3D models for VR and visualisation applications. Such virtual environments can be used for visualisation and simulation purposes in city planning, architecture and catastrophe management.

CELLULAR NETWORK PROVISION The temporary supplementation of cellular networks through flying relay stations was researched. At events where large crowds gather, like sport events or concerts, the amount of voice and data traffic cannot be handled by the existing stationary infrastructure. In such cases the existing cellular network can be extended by a swarm of MUAVs equipped with networking gear.

In order to address these goals, the project was divided into four work packages.

1. Design and development of autonomous MUAVs.
2. Development of team strategies and swarming algorithms for acquisition of aerial photographs or establishment of a cellular network.
3. Definition of interfaces between different subsystems and development of communication protocols.
4. Development of a process to create a virtual environment from aerial photographs.

The scope of the following discussion is limited to the fourth work package (Strothoff et al. 2010) and the virtual environment (Strothoff et al. 2011a). For more information on the other work packages please refer to Rohde et al. 2010.

1.1.1 *Virtual Environment and Interaction*

The aerial images acquired by the MUAVs are used to create a virtual environment. This virtual environment can then be explored and experienced using immersive display systems. But it is also used as feedback for the operator controlling the swarm of MUAVs. Visual representation of the MUAVs and their current state can be integrated into the virtual environment to provide an overview of the mission area and the current mission state. This visualisation is also used as a control interface allowing the operator to alter mission parameters and send new or updated commands to the swarm.

The virtual environment is constructed using two steps. In the first step a flat aerial image layer is created by combining aerial photographs sent by the MUAVs. This image layer is then extended by a 3D reconstruction in the second step. The 3D reconstruction can not start immediately, as several overlapping images have to be collected first, while the aerial image layer can be displayed as soon as the first image is available. To establish a correspondence between the mission area in the real world and the virtual environment meta data from the MUAV, like position and orientation, are used to geo-reference the images.

The aerial image layer is similar to image layers offered by most digital mapping tools, for instance Google Maps, in addition to a schematic map. These image layers are often called satellite imagery although the high-resolution images used on high zoom levels are actually orthographic photographs taken from aeroplanes equipped with special cameras.

Traditionally acquired images are captured under finely controlled conditions like precise flight planning, good weather conditions and best lighting (i. e. time of day). Even though, a lot of time-consuming manual processing is necessary to turn the captured images into one continuous image layer.

Compared to traditional aerial photography project AVIGLE faced several specific challenges. The images acquired by the MUAVs are of different quality. MUAVs flying at different altitudes results in images of varying resolution. Yet, if a region is covered by multiple photographs the photograph (or part of the photograph) that offers the best resolution should be displayed for that region.

As the MUAVs are small and light, their flight behaviour is less stable than that of a larger aircraft. Even if the camera is mounted so that it is pointing directly downwards most of the images will not be taken orthogonal to the ground. This presents another challenge for combining the images into a flat aerial image layer. Before the images can be used they have to be *rectified*, that is reprojected, to yield *orthophotos*.

All photographs are subject to perspective distortion. There is no such thing as an *orthographic* photograph. However, if the images are taken from an altitude that is far greater than the difference between the ground and the tallest structures, perspective effects become so small that the resulting images are close to an orthographic projection. Only small corrections are necessary to create orthophotos from these photographs.

Due to the comparably low altitude of the MUAVs perspective changes in the images are much more noticeable. While photographs taken from aeroplanes flying at high altitudes provide a reasonable approximation of an orthographic projection, a MUAV flying just a few metres over a building will result in very noticeable perspective effects in the captured images.

AERIAL IMAGE LAYER Images received from the MUAVs not only have to be preprocessed (e. g. geo-referencing and rectification), but also combined with overlapping images to determine and use the best image data available. This results in a high number of images that have to be considered (and reconsidered when new images arrive) and large amounts of image data. Depending on the size of the mission area even the combined aerial image layer on its own becomes huge. While it is desirable to show the highest resolution content available it is not feasible to use this resolution if the virtual camera is moved high above the landscape and large regions are displayed. A data structure that is capable of handling and displaying these amounts of image data had to be developed. For digital mapping tools it is common to pre-process map layers and split them into map *tiles* for different resolutions. Then only the visible map tiles have to be determined and an adequate resolution depending on the position of the virtual camera has to be chosen. A similar approach was chosen for project AVIGLE. Due to the interactive nature of the project a static pre-processed data structure was insufficient. New images should be integrated into the aerial image layer immediately when they arrived. Therefore, a dynamic data structure called *Flexible Clipmap* (Feldmann et al. 2011) that allows incremental updates of the image layer and is capable of efficiently displaying the most recent version was developed.

3D RECONSTRUCTION While aerial photographs taken from different angles pose a challenge to combining them into a two-dimensional image layer, they provide the opportunity to create a three-dimensional reconstruction of the environment if enough overlapping images of different angles are collected. Again, it was required that the 3D reconstruction can be extended and refined incrementally as additional images become available over time. Existing tools that generate 3D data from a collection of photographs, like Microsoft Photosynth (Snavely et al. 2006), deliver stunning results but are not suitable for project AVIGLE, because all images are required beforehand and the calculation takes many hours. If only a single image is added the 3D reconstruction has to start over and all processing done so far was wasted. Also, the 3D reconstruction should be updated with new images in a matter of seconds instead of hours. To achieve these goals a new incremental 3D reconstruction algorithm was developed. This new algorithm is capable of showing a coarse 3D reconstruction that allows the operator a broad orientation almost immediately. This, for instance, allows the operator to decide where additional data is needed and send commands to the swarm accordingly. As more images are added or more processing time is spent on the existing images the 3D reconstruction is incrementally updated and refined. This allows the information gathered in previously uncovered regions to be quickly integrated into the 3D reconstruction. If no new data is available additional time is spent on the existing data to increase the quality of the current 3D reconstruction.

VIRTUAL REALITY APPLICATIONS The creation of a three-dimensional virtual environment enables interesting VR applications. Using immersive display techniques the user can explore the virtual representation of the real environment. A head-mounted display and a redirected walking technique (Steinicke et al. 2010) allow the wearer to explore the virtual landscape as if being present in the real environment. Another option is using a CAVE, a combination of multiple display surfaces with stereoscopic projection, and head tracking to be immersed into the virtual environment. The user can step into the virtual scene and explore it by flying over it with a virtual camera or investigating structures, like buildings, from all sides.

OPERATOR INTERFACE In addition to enabling immersive exploration the virtual environment serves another important purpose as control interface for the MUAV operators. Combined with real-time information like visual representations of the MUAVs and their current location and orientation the operator can get an overview of the current state of the mission. Although this control interface was never fully realised during the project a large tabletop display could be used. The virtual environment would be displayed like a map on a table. Visual representations of the MUAVs would be displayed *above* this map layer. To visualise the positions of the MUAVs above the mission area and their relative positions to each other a stereoscopic projection with head tracking should be used. The operator could walk around the table to look at different parts of the mission area to get a good overview of the whole mission through the three-dimensional display. To interact with the swarm of MUAVs and alter mission parameters the tabletop display would be combined with multi-touch input technology. This would allow the operator to directly interact with the displayed MUAVs and command them as necessary. With hardware capable of supporting multiple users the tabletop set-up would allow the collaboration with other people involved in a mission. The operator could discuss the current state of the mission with experts to decide how to progress the mission further and then issue new or updated commands to the swarm.

RELATED PUBLICATIONS More information on the other aspects of project AVIGLE concerning the aerial image processing and creation of the virtual environment can be found in the following related publications. For more information on the creation of a combined aerial photography layer from individual aerial photographs and the data structures that enable rendering the large amount of image data please refer to the work of Dirk Feldmann (Feldmann and Hinrichs 2012; Feldmann et al. 2011). The 3D reconstruction consisted of two processing steps: creation of a 3D point cloud from the aerial images and the generation of 3D meshes from these point clouds. Finding feature points in overlapping images and calculating their 3D position to get a 3D point cloud and later refinement to get a dense point cloud is covered by the work of Jan Roters (Roters and Jiang 2013; Roters et al. 2011). Turning 3D point clouds into meshes and gathering texture information from the original aerial photographs is discussed in the work of Tom Vierjahn (Vierjahn et al. 2013a, 2012, 2013b).

1.1.2 *Inspirations Drawn from Project AVIGLE*

Project AVIGLE was very ambitious and a lot of challenges from different domains (one of the ideas behind the Hightech.NRW initiative) were tackled at the same time. The high complexity of the project resulted in several delays that eventually led to a change of focus and a revision of the original project goals.

Until the very end of the project only a single unmanned aerial vehicle (UAV) prototype was available, so the interface to steer and command a whole swarm or even multiple swarms of drones was no longer a key focus of the project. While there were ambitions to proceed with the control interface using a simulated swarm of drones, ultimately, the intended control interface was never fully realized during the course of the project.

However, several of the requirements for the operator interface that were developed and composed during the project served as inspiration for the interaction techniques discussed in the following chapters.

COMPLEX REGION SELECTION One of the goals of project AVIGLE was to operate the UAVs in a semi-autonomous mode. Capable of vertical take off and landing, equipped with advanced sensor technology and constantly connected to the base station the UAVs were supposed to control their flight parameters autonomously.

Instead of one pilot for each individual drone a single operator should be able to control all drones at once. To support such a scenario an indirect control scheme is necessary. The drones should be able to stay airborne and avoid collisions autonomously, therefore, the operator only defines the mission parameters and objectives. While it is possible to define the role of a drone (photographs, communications link, . . .), by default the swarm would assign the roles automatically based on the current state of the mission, the available drones and their current location.

Aside from some global mission parameters, the operator only sets the mission objectives and target regions in the autonomous mode. A typical mission starts out with some kind of base map layer (like OpenStreetMap) on which the general mission area is defined. To get up-to-date aerial images of the mission area a high-altitude overflight is a common next step. After the overview images are acquired and processed additional mission objectives and regions can be specified.

The target regions are defined as two-dimensional regions on a map layer—either the initial base map or the map enriched with acquired aerial photographs. Mission objectives can then be defined as a combination of commands and a selected region. For example: "collect high-resolution aerial images (i. e. a low- altitude overflight) of the specified region", "gather enough images (from different angles) to perform a 3D reconstruction of the specified region" or "provide wireless network coverage in the specified region".

Due to the interactive and iterative nature of the data acquisition the virtual representation of the mission area is constantly refined. During the course of a mission these updates might result in revised or refined mission objectives. On the one hand this can be a change of priorities, so that the scheduling of objectives is changed. On the other hand it can also mean that the target regions need to be adapted.

In an image acquisition mission an object of interest might only be partly covered by the acquired photographs, so that the target region has to be extended to capture the whole object. If the drones are used to provide wireless network coverage the network users might be moving, so that the bandwidth requirements have to be monitored and the coverage area has to be updated regularly. It might also be possible that specific regions have to be avoided by the drones altogether. Government-regulated no-fly zones (e. g. around airports) are static restrictions that can be considered during mission planning. However, if the drones are used in a disaster response mission, there might also be dynamic flight restrictions. For instance, a fire might break out or spread during the mission. If the fire is detected in the aerial photographs iteratively sent by the drones the mission target regions should be updated to avoid flying over the fire. So, in addition to defining target regions it must also be possible to restrict flight in an area by excluding it from defined target regions or setting up explicit no fly regions.

To accommodate these requirements an interface must not only allow the quick definition of regions, but also later refinement and editing of them. In Chapter 3 a region selection technique is presented that was designed to facilitate this kind of editing and refinement in a multi-touch setting.

COMPLEX GROUP SELECTION Drones are grouped into swarms to control the behaviour of multiple drones at once. Depending on the use case multiple swarms of drones, possibly overlapping or arranged in a hierarchical structure could be needed. A specific role could be assigned to each drone or a swarm of drones. For instance, one drone could be tasked to take photographs while other drones are responsible for keeping up a communication link between the photographing drone and the base station. It would also be possible that more than one role at a time would be assigned to a single drone.

During the course of a mission several parameters could change. New drones could be deployed and added to the swarm. Depleted batteries could force active drones to land. Also, as the drones constantly gather data that is relayed to the operator at the base station, the tasks assigned to the swarms are possibly refined by the operator or mission objectives might change completely.

This feedback loop results in constantly changing mission objectives and a variable number of available drones which necessitates a user interface that allows to dynamically group or regroup drones to swarms, assign drones to one or multiple swarms and assign roles to drones. Ideas originally developed to meet these requirements motivated the development of the tagging and complex group selection interface that is presented in Chapter 4.

3D INTERACTION For mission planning, observation and control during a mission a tabletop set-up resembling a virtual sand table could be used. The base map, and aerial photographs as soon as they are available, are displayed on a large tabletop display like a physical map lying on the table. This allows users to walk around the table and look at different parts of the map. Visual representations of the drones should be displayed on the map. Their current state, like battery levels, their current role or their swarm association should be visualised by adding overlays or annotations. Using a birds-eye view and adding the drones to the map layer would not be sufficient to get a good understanding of their position above the terrain and relative to each other. Therefore, a three-dimensional display is required. As the drones move in the space above the terrain their visual representations should be displayed *above* the flat map layer. To achieve this a stereoscopic tabletop display is used, so that the drone representations are actually perceived above the tabletop. If the captured aerial images are used to generate a 3D model of the environment these models can be displayed as well. The incrementally refined 3D reconstruction of the terrain and structures like buildings could be used to augment the flat map or aerial image layer to get an even better understanding of the mission environment.

While the use of a stereoscopic display is beneficial to the visualisation and spatial understanding of the operator it also introduces challenges to the user interface and interaction design. The drones are perceived to be above the tabletop, however, the physical display remains two-dimensional. One possible solution is the use of an in-air interaction technique, but there are some drawbacks when using these techniques that make them unsuitable for an operator interface. For example, holding outstretched arms in mid-air quickly becomes tiring and prevents prolonged use of such an interface. The lack of haptic feedback and occlusion caused by the arms make it difficult to perform precise actions. Furthermore, a problem often referred to as *Midas Touch* (Kjeldsen and Hartman 2001; Schwarz et al. 2014) is inherent for all in-air interaction techniques. It is hard to understand the user's intention to decide which motions are to be interpreted as interactions. For these reasons we investigate another solution for the operator interface.

An indirect approach using two-dimensional multi-touch interaction on the tabletop was developed. The result is a *cursor* that uses two touches to interact in the three-dimensional volume above the tabletop. The two touch points and the point of interaction form a triangle, which gives this technique its name: *Triangle Cursor*.

Using multi-touch interaction the display surface provides haptic feedback, so there are no unintended actions. During interaction the hand can be rested on the display, resulting in lower fatigue and thus higher precision. While occlusion by the hand can not be avoided altogether, Triangle Cursor was designed to minimise the occlusion of the area of the user's focus and not to interfere with the stereoscopic perception. Triangle Cursor can be used to quickly select any point in the three-dimensional space above the tabletop. It can be used to select drones and move them to different locations (i. e. set new target positions for them). By using Triangle Cursor four degrees of freedom (DOFs) can be controlled using a single hand: the 3D position and the yaw angle. An extension of Triangle Cursor allows controlling of all six DOFs to precisely define and plan complex flight paths. In Chapter 5 Triangle Cursor is presented and discussed in more detail.

Part I

ON THE SURFACE

We start by presenting an extensive survey of existing work on selection techniques. In a graphical overview of the survey we identify areas that have not been explored yet: selecting regions using multi-touch input and the extension of the touch vocabulary through pinning touches. We present two novel interaction techniques for surfaces that are capable of multi-touch input. These interaction techniques target two areas not addressed by existing related work. Both techniques take advantage of true *multi*-touch input—using multiple simultaneous touches on the surface.

2

SURVEY OF SELECTION TECHNIQUES

To obtain an overview of existing work for group selection we first compiled a list of published papers describing relevant work: Dehmeshki and Stuerzlinger 2008, 2009a,b, 2010; Hinckley et al. 2006; Kawasaki and Igarashi 2004; Leitner and Haller 2011; Lindlbauer et al. 2013; Mizobuchi and Yasumura 2004; Moran et al. 1997; North et al. 2009; Saund et al. 2003; Seifried et al. 2012; Xu et al. 2012.

We then developed a set of classifiers based on different properties of the investigated selection techniques. Using these classifiers we categorised the body of related work. To get a better understanding of existing techniques and identify areas that are not yet explored we developed a graphical representation of our categorisation.

In the following sections we introduce the classifiers, provide a summary and classification of the related work, and, finally, present and examine the graphical overview to identify unexplored areas.

2.1 DESCRIPTION OF CLASSIFIERS

To categorise the body of selection techniques we developed a set of classifiers based on input hardware, the mode of operation of the techniques and different selection capabilities. Following are the descriptions of these classifiers.

Mouse This classifier indicates if a selection technique is designed for mouse-based interfaces. Many traditional user interfaces on desktop computers rely primarily on the computer mouse as input device. Mouse-based interaction is indirect. The mouse is used to control the position of a cursor which is then used to interact with the user interface. On notebook computers a trackpad may be used as pointing device instead of a mouse. However, the interaction scheme is the same for both mouse and trackpad.

Pen This classifier specifies if a selection technique is designed for input using a digital pen. Interfaces using digital pens come in different forms, from interactive whiteboards to tablet computers and graphics tablets. Interactive whiteboards, like the popular SMART boards produced by Smart Technologies, usually combine a projector and a projection surface that can detect input via digital pens. As the name suggests interactive whiteboards are a digitally enhanced version of whiteboards that can be found in many classrooms or meeting rooms. Some tablet computers rely on digital pens to control their user interface. Newer models usually combine touch input with a digital pen for writing or drawing applications. Pen interaction on interactive whiteboards or tablet computers is direct, i. e. the user can *draw* directly on the screen. Graphic tablets, for instance the Wacom tablets used by many digital artists, are a kind of hybrid between direct and indirect interaction. Usually the interaction area of the tablet is mapped directly to the display, so that each position on the display could be accessed directly. But as there is no way to see where an object on the display would be located on the interactive surface of the tablet a cursor is needed. While the pen's tip is hovered over the tablet a cursor displays the pen's position over the tablet. Using this cursor the pen can be positioned precisely before it is brought down to the surface to start interacting.

The classifier Pen is assigned to interfaces using any of these kinds of digital pens.

Touch This classifier applies to direct touch input. It covers selection techniques designed for all types of display devices that are capable of detecting touch input. Large projection screens or tabletop displays using touch detection techniques like rear diffuse illumination (Matsushita and Rekimoto 1997) or frustrated total internal reflection (FTIR, Han 2005) are included as well as smaller devices like tablets or smartphones that usually rely on capacitive sensing for touch detection.

Multi-touch While the classifier Touch is used for all touch-based interaction this classifier is only assigned if multiple simultaneous touches are used. Touch and multi-touch are often used synonymously. Two separate classifiers are used to make a clear distinction between them. The classifier Multi-touch also implies the classifier Touch. This is not the case for the opposite direction. Not all interaction on multi-touch-capable hardware (i. e. the hardware is capable of detecting more than one simultaneous touch) is taking advantage of simultaneous touches and should not be called multi-touch interaction.

Hold If the selection technique makes use of a hold gesture this classifier is assigned. A hold gesture is a stationary touch for a prolonged period of time. It is different from a tap gesture where the display is touched only for a short period of time or a sliding gesture with a moving touch.

Pin A pin gesture is the combination of the hold gesture with other user actions. This classifier is assigned if the selection technique uses a hold gesture and other actions at the same time, i. e. something is held down (*pinned* down) while other actions are performed. The interaction technique introduced in Chapter 4 is based on the pin gesture. Pinning touches will be discussed in more detail in Section 4.1.3.

Rectangle This classifier indicates that a selection technique allows the specification of a rectangular area to select the region or objects inside the rectangle. Many applications on desktop computers support at least some form of rectangular selection. Rectangles can be dragged in file management applications to select files. In image editing applications regions of an image can be selected using a rectangle tool. Virtually all applications for editing vector graphics or designing three-dimensional objects, like computer-aided design (CAD) applications, support selecting objects contained inside a rectangular area.

Lasso Free form selections are used by a selection technique with this classifier. Most image editing applications contain a free form selection tool—often called lasso tool. The user can draw the contour of a region to select the region inside the contour. It is also possible to select objects by drawing a closed path around them and selecting all enclosed objects.

In/out This classifier is assigned to selection techniques that differentiate between actions performed or started inside or outside of an existing selection. The term actions in this case refers to actions that are part of the selection process, not other actions of the application that are performed on the selection. If the selection technique selects a region In/out refers to the differentiation of the inside and outside of the selection region. For techniques that select objects In/out refers to the differentiation of selection actions performed on selected and de-selected objects.

Region This classifier specifies that a selection technique is used to select a region (instead of discrete objects). In image editing applications this region may be a precise area of pixels in the image. Other applications might require the user to select one ore more region(s) of interest.

Objects This classifier is assigned if a selection technique is used to select discrete objects. An *object* can be any graphical representation in the interface, like icons representing files or applications, graphical primitives in graphics applications or more complex entities in CAD applications.

There exists a larger variety of selection techniques for selecting objects (instead of regions). To better differentiate these techniques we introduce a set of additional criteria that are specific to object selection.

Auto mode If a selection technique automatically decides its mode of operation, i. e. selection or de-selection, this classifier is assigned. In desktop applications the mode is often set explicitly using modifier keys on the keyboard or buttons in the user interface to switch the mode. If only toggling the selection of a single object by repeatedly clicking or, in case of touch or pen input, tapping it this classifier is not assigned. This classifier is assigned only if the selection technique includes some form of logic that automatically decides if the user wants to perform a selection or de-selection based on the user's actions.

On/off objects For the assignment of this classifier it is required that a selection technique differentiates between actions that are performed or started on objects or on the surrounding space (background). In contrast to the classifier In/out this classifier is independent of the selection state of the objects.

Direct This classifier is assigned if objects that are to be selected are specified directly. For instance, objects are selected by tapping or clicking directly on them.

Indirect If objects are selected indirectly this classifier is used. There are different ways of indirect selection that are covered by this classifier. For instance, selection techniques specifying a region for selecting all contained objects are classified as Indirect. The classifier is also used for techniques that select objects based on the objects' properties or their layout. While it appears as if direct and indirect selection are mutually exclusive some selection techniques support different ways of selection resulting in a classification as both Direct and Indirect.

Surround This classifier is used for selection techniques that select objects by surrounding them. For instance, the objects inside a rectangular region or enclosed by the boundary of a free form selection are selected by surrounding them.

Crossing If all objects intersecting a selection gesture are selected this classifier is assigned. For instance, drawing a line through several objects to select them. This technique of intersecting the objects with a gesture is commonly referred to as *crossing*.

Path Techniques that base object selection on paths instead of regions are covered by this classifier. This applies to straight and curved paths. Most selection techniques that are based on Crossing use a path-based gesture for selection. But also techniques that select all objects along a linear path, i. e. with a linear layout, are included in the classifier Path.

2.2 CLASSIFICATION OF SELECTION TECHNIQUES

In the following, we categorise each item of related work by assigning above classifiers. A summary of each selection technique, highlighting the aspects that led to the assignment of the respective classifiers, is included.

INTELLIGENT MOUSE-BASED OBJECT GROUP SELECTION

Dehmeshki and Stuerzlinger (2008)

Mouse Objects Direct Indirect

The authors investigate alternatives to traditional mouse-based object group selection techniques such as rectangle or lasso selection. By considering the way the human perception naturally groups objects, also known as Gestalt theory (Koffka 1935), the authors present a new method for grouping objects based on their proximity or (curvi-)linearity. Single objects can be selected or de-selected directly. By double-clicking on an object the Gestalt group that contains the object can be selected. In case of ambiguities resulting from intersecting (curvi-)linear groups or multiple object clusters in close proximity, the selection technique employs clicks with modifier keys to extend or constrain the selection.

GPSEL: A GESTURAL PERCEPTUAL-BASED PATH SELECTION TECHNIQUE

Dehmeshki and Stuerzlinger (2009a)

Pen Path Crossing Surround Indirect Objects Lasso Auto mode

GPSel is a selection technique for selecting objects (nodes, edges) along a path in graphs. Using the Gestalt principles of continuity and closure, perceptually salient paths are detected. Short stroke gestures with a pen that start on an object and point in the direction of an edge are used to select a path. To constrain the path or to avoid ambiguities in the selection additional gestures can be used. By crossing an already selected path the selection can be constrained in the direction of the gesture. Selected objects that lie on the path in the opposite direction are de-selected. Narrow forks along a path can result in multiple continuous paths. In this case all paths that satisfy the continuity criteria are selected. To resolve these ambiguous selections a path selection gesture at the fork can be used to constrain the selection to one of the sub-paths. Depending on the state of the selection and the position where the gesture is performed the selection technique automatically decides if additional objects are to be selected or the current selection is to be constrained, resulting in the de-selection of some of the already selected objects. For the selection of cyclic paths the authors use a lasso gesture. If an object is surrounded by the lasso gesture all closed paths that contain that object are selected.

ICE-LASSO: AN ENHANCED FORM OF LASSO SELECTION

Dehmeshki and Stuerzlinger (2009b)

Pen Lasso Objects In/out Indirect Surround

ICE-Lasso is an extension of the lasso selection technique that is common for pen-based interfaces. While the user is drawing a lasso gesture ICE-Lasso detects clusters of objects near the gesture and highlights them. The user can select the highlighted cluster using a pigtail gesture (i. e. drawing a small loop in the path). Alternatively the user can continue drawing a regular lasso gesture, for instance to select only a part of an object cluster. As soon as the first object is selected, by surrounding it with the gesture, ICE-Lasso automatically switches to an auto-completing lasso mode (Mizobuchi and Yasumura 2004). To select multiple clusters using a single selection gesture the pigtail gesture can be repeated. It is also possible to extend a selection after it is completed. If the lasso gesture is started inside an already selected group the selected objects or clusters are added to the selection. A new selection is begun if the lasso gesture is started outside of the current selection. To de-select objects a curved line can be drawn inside a selected group. All objects that are on the concave side of the curve are de-selected. A whole cluster can be de-selected by drawing a small curve inside it that does not contain any objects.

DESIGN AND EVALUATION OF A PERCEPTUAL-BASED OBJECT GROUP SELECTION TECHNIQUE

Dehmeshki and Stuerzlinger (*2010*)

Pen Indirect Crossing On/off objects Path Auto mode

The authors present a perceptual-based selection technique for selecting whole or partial groups with (curvi-)linear or random structures. The pen-based technique uses flick gestures to select object groups. By analysing location, direction and shape of the gestures the desired groups are inferred and selected. The underlying group structure of the selection is visualised by links between the objects. Using a line or an arc gesture that crosses through an object the group of objects aligned with the gesture is selected. To restrain the selection of the group in one direction the gesture can be started on an object and be performed in the desired direction. In case of ambiguities all matching groups are selected. This can result in branching along a path of selected objects. An unwanted branch can be *cut* by drawing a flick gesture through the visual link connecting the object where the branching occurs and the first object on the unwanted sub-path. The objects on that branch are then de-selected. For the selection of arbitrary groups the authors use a series of flick gestures. Depending on the location of the gestures the selection technique automatically decides which objects are to be selected and de-selected to continue the selection process. If the gesture is performed at the end of a connected group the selection is extended in the direction of the gesture. If the gesture is performed in the middle of a connected group the objects between the location of the gesture and the end of the group are deselected and the objects in the direction of the gesture are selected. Using this technique arbitrary selections can be incrementally constructed.

PHRASING TECHNIQUES FOR MULTI-STROKE SELECTION GESTURES

Hinckley et al. (*2006*)

Pen Crossing Lasso Surround Indirect Direct Objects In/out Auto mode

This paper investigates different pen-based object selection techniques with a focus of phrasing multiple strokes to a single action. If a selection action is comprised of several strokes it is not clear if the strokes belong to an ongoing action or if the action was completed and the stroke begins a new action. The combination of elemental actions into a single command is called phrasing. The authors investigate the possibilities of phrasing multiple actions using a button or touchpad operated by the non-dominant hand. A prototype application is presented that supports different selection techniques using multiple strokes. The elemental operations include directly selecting objects by tapping them with the pen or crossing strokes through objects and indirectly selecting objects by surrounding them with a lasso gesture. By using multiple lasso gestures additional objects can be selected or already selected objects can be de-selected. The selection technique automatically decides if objects are to be selected or de-selected based on the location of multiple lasso strokes. Strokes that begin inside the current selection and extend outwards can be used to select additional objects, while strokes that begin and end outside of the current selection de-select objects.

REGIONAL UNDO FOR SPREADSHEETS

Kawasaki and Igarashi (*2004*)

Mouse Region

The authors propose a regional undo technique for spreadsheet applications. Typical undo techniques are based on the temporal order of operations. If the user spots an error in a specific cell of the spreadsheet, there is no information *when* the error occurred. To fix the error possibly a lot of actions have to be undone, losing a lot of work that has been done after the error occurred. Using the regional undo technique proposed by the authors the region containing the error can be selected and the previous state (or states) of the cells in this region can be restored. As this undo operation only affects the selected region it is independent from actions that affected other parts of the document. This avoids having to undo a lot of unrelated operations to fix an error.

HARPOON SELECTION: EFFICIENT SELECTIONS FOR UNGROUPED CONTENT ON LARGE PEN-BASED SURFACES

Leitner and Haller (2011)

Pen Crossing Surround Indirect Direct Lasso Objects Path Auto mode

Harpoon Selection is a selection technique for pen-based interfaces. Its main selection mode combines selection by crossing with an area cursor. All objects that are crossed by a selection path are selected. Instead of using just the path an area surrounding the path is also used for selection, as if the selection path was drawn using a pen with a large tip. As there is no perfect spot size for all selection cases the authors use a dynamic spot size that changes with the speed at which the selection gesture is drawn. Fast movements result in a big spot size and can be used to quickly select large groups of objects. With slow movements and a small spot size precise selections are possible. To modify existing selections single objects can be added or removed by tapping on them directly. For larger modifications additional selection strokes using the Harpoon tool can be used. Harpoon Selection automatically decides whether objects are to be added to the selection of removed from it based on the selection state of the first object that is crossed. If the first object is selected all objects crossed by the gesture will be de-selected. If the first object is not part of the selection all crossed objects will be added to the selection. The authors also suggest an improvement of Harpoon Selection by overloading it with a lasso selection tool. When the user starts to draw a selection gesture around objects the lasso mode is used to select all surrounded objects. If an object is crossed selection switches to the Harpoon mode.

PERCEPTUAL GROUPING: SELECTION ASSISTANCE FOR DIGITAL SKETCHING

Lindlbauer et al. (2013)

Pen Crossing Indirect Direct Objects Path

The authors present *Suggero*, a selection method facilitating the selection process in pen-based interfaces by identifying perceptually related objects. After an initial selection Suggero displays a linear marking menu with suggested extensions of the current selection. Perceptual features like proximity, stroke connectivity, stroke parallelism and similarity are used to dynamically group objects. These perceptual groups are added to the list of suggestions enabling the quick selection of perceptually similar objects that would have been hard to select using other methods. For the initial selection Suggero supports directly tapping objects or using Harpoon Selection (Leitner and Haller 2011) to select all objects crossed by a selection path.

TAPPING VS. CIRCLING SELECTIONS ON PEN-BASED DEVICES: EVIDENCE FOR DIFFERENT PERFORMANCE-SHAPING FACTORS

Mizobuchi and Yasumura (2004)

Pen Objects Lasso Direct Indirect Surround

As the title suggests the authors compare directly selecting objects through tapping with selecting them using a lasso selection tool. In the paper a variant of the traditional lasso selection is used. When the stroke of the lasso is not completed (i. e. closed) the endpoints of the stroke are connected with a straight line to define the lasso region. As no other stroke-based actions are used by the authors this does not lead to ambiguities. Usually closing a lasso stroke is necessary to distinguish it from other actions.

With the lasso variant used in the paper it is also not necessary to completely surround objects to select them. Surrounding the object centres is sufficient to select them. While this helps to avoid the accidental selection of objects in close proximity it might be difficult to guess the location of the centre point of objects that are more complex than the squares used in the paper. Ultimately the authors did not find a clear answer to the question if tapping or circling is a better selection technique. While they do discuss several interesting characteristics of each selection technique the overall performance is largely dependent on the layout of the objects that are to be selected.

PEN-BASED INTERACTION TECHNIQUES FOR ORGANIZING MATERIAL ON AN ELECTRONIC WHITEBOARD

Moran et al. (1997)

Pen In/out Lasso Rectangle Region Auto mode

The authors describe interaction and selection techniques for the Tivoli application (Pedersen et al. 1993) on the Xerox LiveBoard, an early version of an electronic whiteboard. A pen-based lasso gesture is used to select regions on the whiteboard. Selected regions can be modified using additional strokes that begin and end on the selection boundary. Depending on the location of the stroke, inside or outside of the current selection, the selection is extended (called a *bump* gesture) or a part of the selection is removed (called a *bite* gesture). As a bite gesture effectively cuts the current selection in half it has to be decided which half is to be removed. The authors discuss several criteria, like the arc length of the boundary or the area of the regions, for determining which half to retain. However, there is no explicit way to choose which half of the selection is to be retained. To alter rectangular selections the authors introduce an L-shaped gesture. The closest corner of the rectangular selection is then moved to the location of the L gesture. While this is an efficient way to alter the selection region it is constrained to exactly one rectangular selection.

UNDERSTANDING MULTI-TOUCH MANIPULATION FOR SURFACE COMPUTING

North et al. (2009)

Touch Objects Direct Indirect Surround On/off objects Hold Pin Multi-touch

In this paper several multi-touch interaction techniques for large tabletop displays inspired by physical interactions are explored. Previous work (Wobbrock et al. 2009) has shown that users commonly approach touch interfaces using a single hand. However, if the users interact with real objects on a tabletop they use both hands. The authors investigate different interaction techniques using multiple fingers and one or two hands. Some of the techniques interact directly with the objects by touching and moving them or by shoving multiple objects with the side of the hand—like users would move real objects. Other techniques use a more indirect approach for selecting groups of objects: All objects contained in the convex hull of multiple fingers or contained in a rectangular area defined by both hands are selected. While groups of objects are moved single objects can be pinned down by touching and holding them in place to separate them from the group. To add objects to a group they can be moved into a currently selected group. The user study presented in the paper has shown that using multiple touches for grouping tasks is very useful. However, the authors have reported a large range of different expectations users had when they first approached the system. The authors caution developers of tabletop interfaces to consider a good balance of physical metaphors and abstract gestures.

PERCEPTUALLY-SUPPORTED IMAGE EDITING OF TEXT AND GRAPHICS

Saund et al. (2003)

Mouse Rectangle Lasso On/off objects Region Objects Direct

The authors describe the program *ScanScribe*. The mouse-based interface was designed to work without the typical tool palette found in other image editing programs. Image objects can be selected by clicking on them. If the mouse button is pressed down on the background (i. e. not on any object) an overloaded version of rectangle and lasso selection is initiated. The current selection region of lasso and rectangle selection are shown simultaneously. If the mouse button is released while the rectangle is visible the rectangular region is selected. As the path has closed enough that the program thinks the user is attempting a lasso selection the rectangle is discarded. If the mouse button is released after the rectangle disappeared the region enclosed by the lasso path is selected. Using this overloaded selection technique no explicit mode switch is necessary to use the different selection techniques. Once a region has been selected by either method it becomes an image object and can later be re-selected by simply clicking on it.

REGIONAL UNDO/REDO TECHNIQUES FOR LARGE INTERACTIVE SURFACES

Seifried et al. (2012)

Pen Surround Objects Region

Different regional undo techniques for large interactive surfaces are explored in this paper. As large interactive surfaces are intended to be used by multiple users, either for collaborative or individual work, a regular undo function based on the order of operations would lead to unpredictable results for individual users. The authors describe different ways of applying local undo operations to different regions. Regions can be predefined by system parameters like different projectors or by workspaces defined by the application. Another approach uses a semi-automatic definition of regions by clustering objects and applying undo operations to clusters. In a manual approach users can explicitly select the region for the undo operation by surrounding all objects that should be affected using a pen-based gesture. The authors note, that the manual approach needs two steps, selecting the region and triggering the undo action, while the other alternatives can invoke the undo action directly.

LAZY SELECTION: A SCRIBBLE-BASED TOOL FOR SMART SHAPE ELEMENTS SELECTION

Xu et al. (2012)

Touch Crossing Objects Direct Indirect Path

Lazy Selection allows the quick selection of shape elements in drawings. The shape elements in images are closed polygons, i. e. each element is a distinct object. Objects can be selected directly by tapping them or indirectly using a scribble-based technique. To select objects a rough scribble drawn over them is often sufficient. The algorithm used by Lazy Selection uses the location and shape information of the scribbles to recognise the user's intent by shape matching between the drawing and the scribbles. Objects can either be selected by crossing them with a selection path or can be selected by painting, i. e. completely covering the desired objects.

2.3 GRAPHICAL OVERVIEW

To get an overview and a better understanding of the existing selection techniques we developed a graphical representation of our categorisation. The overview of all pair-wise combinations of classifiers is shown in Figure 1. The numbers at the intersections indicate how often the corresponding combination is present in previous work. While Figure 1 can not capture all details of the categorisation it shows that some areas are more populated

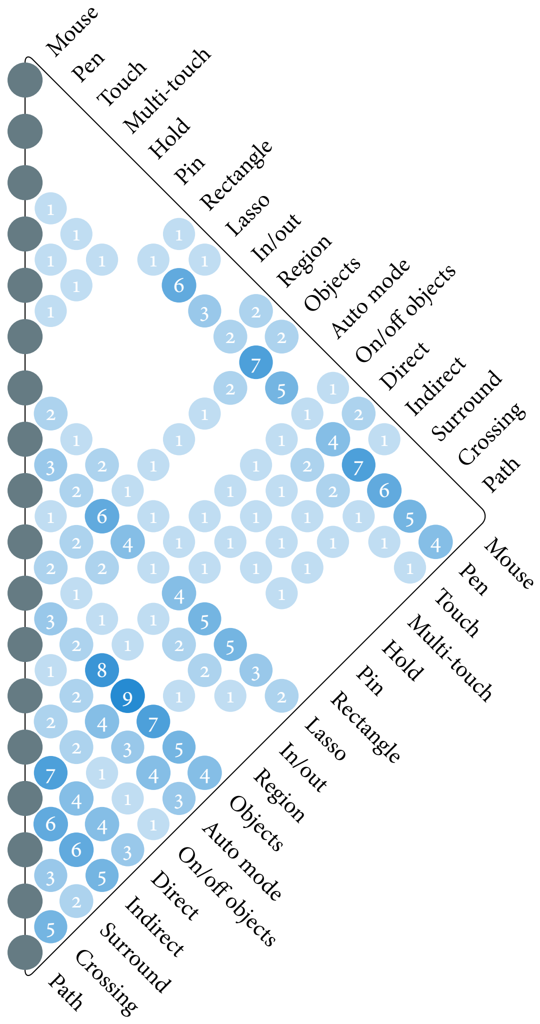


Figure 1: All pair-wise combinations of classifiers with their number of occurrences provide an overview of the examined body of previous work on group selection.

than others. For instance, a lot of previous work included object selection, while region selection is far less often considered.

Looking at Figure 1 one can identify two big gaps: multi-touch techniques for selecting regions and the combination of hold or pin touches with crossing-based techniques. In this thesis we address these two untouched areas with interaction techniques that fill the empty spots in Figure 1. Our own contribution to multi-touch region selection is presented in Chapter 3. The use of pinning touches in combination with crossing or path-based selection techniques offers the potential for more expressive interaction. In Chapter 4 we introduce a group selection and tagging technique that is based on pinning touches.

3

ADDING CONTEXT TO MULTI-TOUCH REGION SELECTIONS

Region selection has been around since the mouse became an indispensable part of the human-computer interface. Virtually all graphics editing programs have a selection tool that can be used to select rectangular regions. Many other programs from highly specialised CAD applications to everyday word processors include some form of rectangular selection; either as region selection or for selecting all objects within a specified region. It comes as no surprise that this popular interaction metaphor found its way into modern multi-touch-enabled applications on devices ranging from smartphones and tablets to large tabletop displays. However, the interaction metaphor that is well established for mouse-based interaction has just been transferred to the new touch-based environment. We argue that instead of just transferring the existing region selection technique it should be better adjusted to the new environment and take advantage of the additional capabilities of multi-touch input.

3.1 INTRODUCTION

In this chapter we introduce our novel multi-touch-based approach to region selection that can not only define, but also easily extend, modify and refine selection regions using the *context* of the user's touches, that is the order of the touches and their posi-

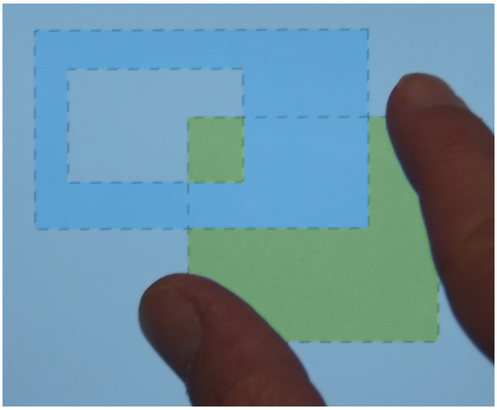


Figure 2: Our suggested multi-touch region selection technique.

tion relative to an existing selection. Using two fingers, as shown in Figure 2, the initial selection region can quickly be defined and then adjusted until the fingers are lifted. Additional selection regions can be combined with the currently selected area to create complex selections. Complex selection regions are achieved by combining several *simple* rectangular selections, therefore we investigate how our technique compares to existing region selection techniques, that are only capable of simple rectangular selections. We present a user study that compares the speed and accuracy of the presented techniques as well as the users' subjective perceptions of them. After empirically exploring our technique we propose possible modifications to further increase the selection accuracy and discuss several design considerations for integrating our selection technique into applications with richer user interfaces.

To conclude this chapter we present an adaptation of the concepts introduced for our rectangular selection technique to free form selections.

3.1.1 *Related Work*

Many aspects of interaction with multi-touch devices have been studied. Interaction metaphors originating from the physical world or from desktop computing have been examined in a new touch-based environment. Forlines et al. (2007) compared mouse input to direct touch input. Their results show that there is a benefit of touch interaction if multiple touches are used. For interfaces that are solely based on single point interaction a mouse might be a better choice. Esenther and Ryall (2006) stated that touch-based interactions can not easily be mapped to mouse-based interactions and have proposed an interaction scheme that allows a better control of mouse-based interfaces using direct touch input. Bi et al. (2011) have even proposed the re-integration of interaction schemes developed for multi-touch devices into desktop applications.

A lot of work studied the selection of objects on multi-touch devices. Moscovich and Hughes (2006) used a multi-finger technique to provide an adjustable cursor area for object selections. An interesting examination of object selection techniques inspired by interactions on physical tabletops has been provided by North et al. (2009). However, multi-touch region selections have not been fully explored. Casalta et al. (1999) proposed the idea of specifying rectangles and other geometric primitives through two points, but have not considered using it beyond the specification of geometric primitives. Latulipe et al. (2006) have used an interaction scheme to specify a rectangular region similar to our proposed technique. In contrast to our work that uses direct touch interaction they used an indirect approach using two computer mice.

Interaction schemes that use several fingers or both hands simultaneously and control multiple degrees of freedom have been examined in previous work. Hancock et al. (2007) have evaluated input using two or more fingers. The differences and advantages of uni-manual and bi-manual input have been examined by Brandl et al. (2008), Kin et al. (2009), and Moscovich and Hughes (2008). The idea of using the contextual information of the user's gestures can also be found in an object manipulation technique introduced by Wigdor et al. (2011).

The size of the human finger causes occlusions and inaccuracies in the actual touch position on the screen, directly touching positions on the screen is considered inherently inaccurate (Vogel and Baudisch 2007), and there has been a series of studies investigating the causes of these inaccuracies, including the fat finger problem (Vogel and Baudisch 2007) and the perceived input point model (Holz and Baudisch 2010). Many approaches tried to address this issue by using a cursor and adding a level of indirectness to the interaction by using an adjustable (Benko et al. 2006) or fixed (Potter et al. 1988) cursor offset, or by scaling the cursor motion (Benko et al. 2006).

3.1.2 *Contributions*

We propose a novel multi-touch region selection technique. By using multiple touches and the *context* these touches are placed in our technique facilitates the subsequent modification and refinement of selected regions. In addition, we present an extension of our technique that allows the selection of arbitrary regions. By applying the same basic concepts of our rectangle selection technique to free form selections we achieve the same level of possibilities for editing and refining selections.

The results of our extensive survey of related work presented in Chapter 2 is summarised in Figure 3. An overview of all pairwise combinations of classifiers used in the survey is shown. The numbers at the intersections indicate how often the corresponding combination is present in previous work. The classification of our novel region selection technique is shaded in green and hatched areas represent new contributions. A large gap in Figure 3 shows

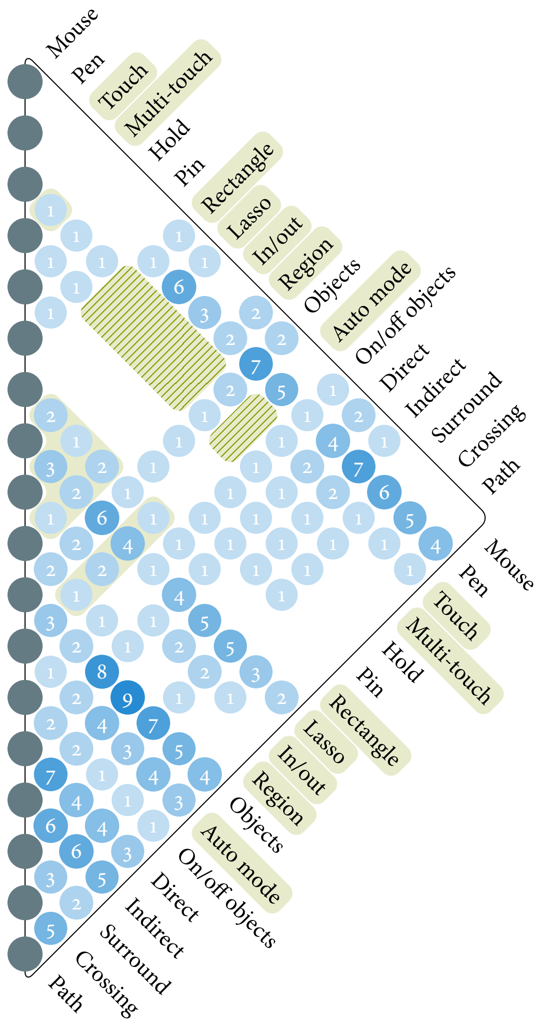


Figure 3: All pair-wise combinations of classifiers with their number of occurrences provide an overview of the examined body of previous work on group selection in Chapter 2. The classification of our novel region selection technique (shaded in green) and the new contributions (hatched area) are also shown. The extension to arbitrary regions presented in Section 3.7 is already included.

the absence of precise region selection techniques for multi-touch input. Our technique aims to fill this gap by providing the same level of control familiar from region selection tools in mouse-based interfaces in novel multi-touch settings.

With the classifiers introduced in Section 2.1 we classify our own technique as:
Touch Multi-touch Rectangle Lasso In/out Region Auto mode.

The classifier Lasso refers to the extension of our technique to arbitrary regions introduced in Section 3.7.

3.2 TOWARDS MULTI-TOUCH

On desktop computers with a mouse the traditional selection technique for selecting rectangular regions works by clicking to place the first corner of the selection region and then dragging the cursor to place the second corner. While it is certainly possible to use the same technique on a multi-touch device by replacing the mouse cursor by the user's finger, we think this is not a very good approach. By taking advantage of the possibilities of multi-touch input it is possible to modify this touch and drag technique so that the whole selection area can be specified at once. Instead of first touching and then dragging, two touch points are used to specify the whole selection region in a single step, as shown in Figure 4.

This has two advantages. On the one hand it is possible to adjust the whole selection region instead of just one *side* of it as with the touch and drag approach. This is especially important in a multi-touch environment, as it is a lot harder to hit an exact location on the screen with a finger than with the cursor of a mouse. On the other hand the users can pre-shape their hands before actually touching the surface and approximate the region to be selected. Once the users touch the screen and get a visual feedback of the selection region only a small correction is necessary.

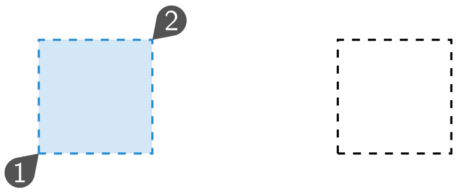


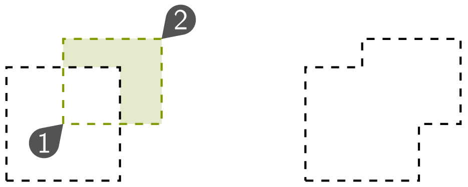
Figure 4: Positions of the first and second touch on the left and the resulting selection on the right.

3.3 REFINING SELECTIONS

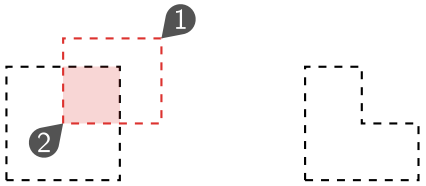
Applications often allow users to refine selections by adding or subtracting a newly defined region from the current selection. This makes it possible to select complex regions or to correct an earlier selection (Figure 5). On desktop computers this is often implemented using modifier keys. Holding down a specific modifier key toggles the selection mode to add or subtract. While it would be desirable to have that kind of control on a multi-touch device, an implementation using modifier keys is obviously not an option. It would certainly be possible to add some kind of mode switches to the user interface that set the mode for the next selection. However, we think that this would unnecessarily clutter the interface and, therefore, we propose a much cleaner solution using multiple touches and their *context*.

In contrast to its traditional counterpart on the desktop our technique uses two touch points to define a selection. We refer to the placement of the touch points inside or outside of an existing selection and the order in which they are placed as their context. Then we can map the four different context combinations listed in Table 1 to different selection modes.

If the first touch is located inside the existing selection and the second touch is placed outside of the selected region this indicates, that the newly defined region should be added to the selection (Figure 5a). If the order of the touch points is reversed,



(a) Extending an existing selection.



(b) Subtracting from an existing selection.

Figure 5: The mode of selection is decided by the order of the touches and their position relative to the existing selection, their *context*.

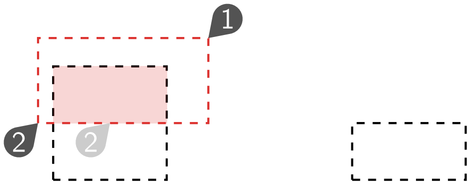


Figure 6: The mode of selection is only determined by the initial touch positions. The initial position of the second touch is shown in light grey.

		1st touch	
		inside	outside
2nd touch	inside	—	subtract from selection
	outside	add to selection	start new selection

Table 1: Decide what mode of selection to use based on the *context* of the two touches that define the selection region.

first outside, then inside, the defined region will be subtracted from the current selection (Figure 5b). If both touches are placed outside of the existing selection it is cleared and a new selection is begun. The colour of the selection region is used to provide feedback on the selection mode to the user: green for addition, red for subtraction. The case for both touches inside the selected region has been intentionally left blank in Table 1, because it is free to be mapped to some other operation. Useful examples might be moving the selected area using a two finger pan gesture or using a pinch gesture to scale the selected region.

What may at first look like an arbitrary assignment in Table 1 is actually based on a simple mental model. The first touch determines what should be extended. If it is inside the selected region the selection is extended into the unselected region. If on the other hand the first touch point is outside of the current selection this can be viewed as *extending* the unselected region into the selection, thereby making the selection smaller.

It is important to note that only the context of the touch when it is placed on the surface matters for the selection mode. As soon as the selection mode is decided the touches can be moved around freely, as shown in Figure 6. The initial position of the second touch is shown in light grey. By moving the touch points after the selection mode has been decided, complex selections containing disjoint regions or *holes* in the selected region can be achieved. In Figure 7a the user placed the first touch inside the already selected region to begin an extension and then moves his touch point outside to add a disjoint region to the current selection. If the first touch is placed outside the selected region to signal a subtraction and later both touch points are moved inside an already selected region it is possible to cut a hole in the selection region, as shown in Figure 7b. By repeating the different selection modes arbitrary regions can be selected using our selection technique.

3.4 USER STUDY

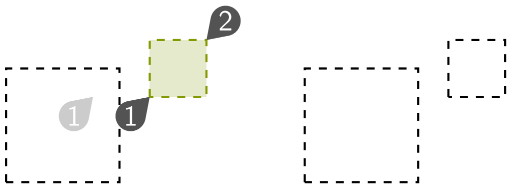
Our multi-touch selection technique requires the user to control more degrees of freedom, compared to existing techniques, by using multiple fingers simultaneously. As we were concerned that this could have a negative impact on its performance we evaluated our technique in a user study. As complex selection shapes are constructed as a series of simple rectangular selections we investigated how our technique performs for simple selections, in comparison to existing approaches.

3.4.1 *Existing Approaches*

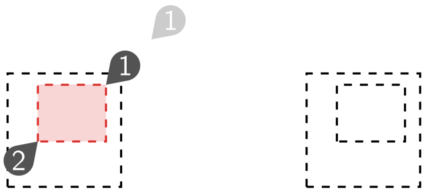
To validate our design (MULTI) and evaluate how it compares with the existing approaches in a formal user study we compare it with two selection techniques which are frequently used in multi-touch environments: Touch and Drag (DRAG) and Selection Widget (WIDGET), both shown in Figure 8.

3.4.1.1 *Touch and Drag*

The traditional selection technique that has been used on desktop computers with a mouse for many years is illustrated in Figure 8a.

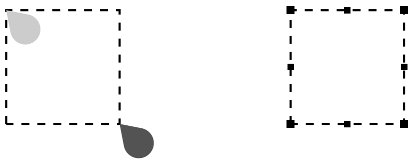


(a) Extending the selection with a disjoint region.



(b) Creating a *hole* in the selection.

Figure 7: If the touch points are moved after the selection mode has been chosen complex selections can be achieved.



(a) (b)

Figure 8: Currently used selection techniques. (a) The user touches (light grey marker) and drags (dark grey marker) to select a region. (b) Selection widget with handles to transform the selection area.

The user touches to place the first corner of the selection region and than *drags* the touch point to place the second corner. This effectively defines a diagonal of the axis-aligned selection rectangle.

3.4.1.2 *Selection Widget*

Another popular technique is using a kind of selection widget. Handles are placed around the actual selection region, as shown in Figure 8b. Usually four handles, one for each corner, are used. Often these are extended by four handles in the middle of each edge. Sometimes the edges themselves can act as handles. Handles like these are typically used for resizing objects that have already been selected, but graphics tools like GIMP or Adobe Photoshop can transform the selection area in a similar manner.

In multi-touch applications usually the rectangular selection widget is initially positioned in the centre of the screen and then adjusted by the user to fit the region of interest, rather than initially placed by the user. A prominent example of this behaviour is the Photos app that is pre-installed on all Apple iOS devices (since iOS version 6). To crop photos the user can specify a rectangular portion of the photo using a selection widget.

3.4.2 *Participants and Set-up*

21 participants (20 male, 1 female), ages 24 to 37 (M = 27.71, SD = 3.01), participated in the study. All but one participant were undergraduate, graduate or PhD students in computer science. All participants reported to have at least some experience with multi-touch devices, most of them used smartphones or tablets on a regular basis. The right hand was used to perform the experiment tasks by all users and all users reported to be right-handed. Completing the experiment, including instructions, training and debriefing, took the participants approximately half an hour. The participants were allowed to take breaks at any time.

The experiment was performed using an Apple iPad (first generation) that was attached to a desk in front of the participants. The participants were provided with an adjustable chair.

3.4.3 *Procedure*

With each technique participants performed a number of region selections. The order of techniques was counterbalanced using a Latin square. After the experiment participants were asked to complete a short questionnaire to provide feedback about the experiment and the different selection techniques.

The dependent measures were task completion time and selection error. The final position of the selection and the completion time for each trial were recorded. For WIDGET the number of steps to modify the selection region was additionally recorded.

3.4.3.1 *Experiment Task*

In each trial a target region that had to be selected was displayed. Participants were asked to select the target regions as precisely as possible. The target regions were distributed over the entire screen and differed in size. On a mobile device the typical usage scenario is using the dominant hand to interact, while holding the device in the non-dominant hand. To study this scenario the size of each region was chosen so that it could easily be selected using a single hand by all participants. Both rectangular and elliptical target regions were used to investigate the effect of fingertips partly occluding the target shape. For the elliptical regions participants were asked to select the closest fitting bounding box.

For DRAG and MULTI only one selection was allowed, that means after the participants lifted their fingers from the surface their selection was final. As this limitation is not possible for WIDGET the participants were able to modify the selection area until they were satisfied. After finishing the selection with any of the techniques it had to be confirmed by pressing a button in the top-right corner of the display. While this would not have been necessary for DRAG and MULTI it was required to provide the same conditions for all techniques. This automatically started the next trial, while also ensuring identical starting positions of the users' hands.

For each technique ten target positions, both as a rectangular and elliptical area, were presented to the participants. Two repetitions of each combination of position and shape were performed, resulting in a total number of 40 trials per technique and a total number of 120 trials per participant. The order of trials was randomised for each participant and technique. Before switching to a different technique participants were provided with short written instructions and illustrations explaining the selection technique. After reading the instructions participants could try the technique and ask questions about it if necessary. When they signalled that they had understood the technique the trials were started. To eliminate the effect of switching selection techniques four training trials were added before each set of trials and discarded from the analysis. All interactions had to be performed using a single hand.

3.4.4 *Results*

Data from the experiment task were analysed using a within-subjects, repeated measures analysis of variance (ANOVA) for the following factors:

- *Technique*: DRAG, MULTI, WIDGET
- *Position*: 10 target positions
- *Shape*: rectangle, ellipse

3.4.4.1 *Task completion times*

The task completion times were measured from the first interaction, i. e. the first touch, in each trial until the confirmation button was pressed.

Mauchly's test indicated that the assumption of sphericity had been violated for the main effects of technique ($\chi^2(2) = 18.96$, $p < .001$) and position ($\chi^2(44) = 68.49$, $p = .014$), therefore the degrees of freedom were corrected using the Greenhouse-Geisser estimates of sphericity ($\varepsilon = .61$ and $\varepsilon = .53$).

A significant main effect of technique, $F(1.23, 24.52) = 68.74$, $p < .001$ was found, with mean completion times increasing from 4.15 s (SE = 0.28) with DRAG, through 6.39 s (SE = 0.54) with MULTI, to 11.19 s (SE = 0.97) with WIDGET; more than a twofold increase across the three conditions. These results are summarised in Figure 9. Post-hoc pairwise comparisons showed significant differences between all techniques ($p < .001$).

A significant main effect of position, $F(4.80, 95.97) = 4.65$, $p = .001$ was found. Post-hoc analysis showed that the difference in task completion times was significant only for the selection target in the centre of the screen and the two targets farthest away from the centre. This was likely caused by the selection widget's initial position in the centre of the screen. This is confirmed by a significant interaction between the technique used and the target position $F(5.19, 103.83) = 4.96$, $p < .001$.

There was a significant main effect of shape, $F(1, 20) = 8.20$, $p = .010$, with mean times increasing from 7.04 s (SE = 0.56) for rectangles to 7.45 s (SE = 0.57) for ellipses and there was a significant interaction of position and shape ($F(9, 180) = 2.17$, $p = .026$). There was no significant interaction between technique and shape ($F(1.41, 28.27) = 0.73$, $p = .446$).

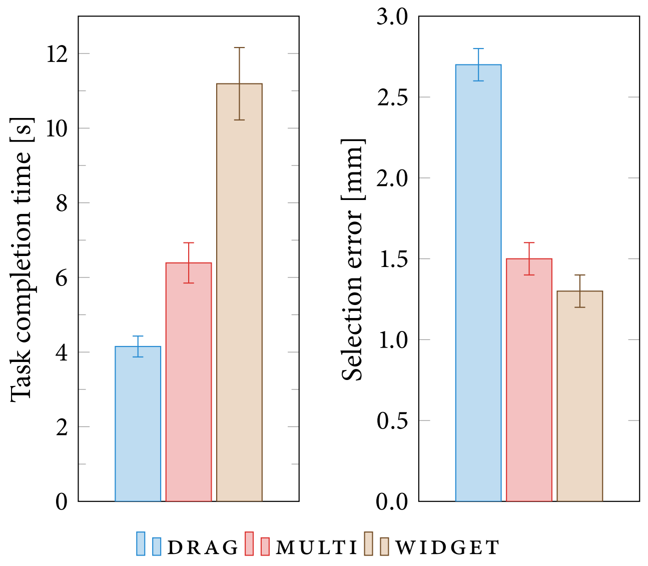


Figure 9: Mean task completion times and selection errors show an inverse trend for the three selection techniques. (Error bars = ± SE)

3.4.4.2 *Selection error*

To measure the selection error the sum of the distances between the lower-left and upper-right corner of the selection and target region, respectively, was calculated. The results are summarised in Figure 9.

Mauchly's test indicated that the assumption of sphericity had been violated for the main effect of position ($\chi^2(44) = 84.18$, $p < .001$), therefore degrees of freedom were corrected using the Greenhouse-Geisser estimates of sphericity ($\varepsilon = .48$).

A significant main effect of technique, $F(2, 40) = 80.07$, $p < .001$, was found with mean selection error decreasing from 2.7 mm (SE = 0.1) for DRAG, through 1.5 mm (SE = 0.1) for MULTI, to 1.3 mm (SE = 0.1) for WIDGET. This trend is the inverse of the trend observed for the task completion time, showing a trade-off between speed and accuracy. These results are summarised in Figure 9. Post-hoc analysis showed pairwise differences ($p < .001$) between DRAG and the other techniques. There was only a difference with borderline significance ($p = .046$) between MULTI and WIDGET.

No significant main effects of position, $F(4.28, 85.66) = 1.58$, $p = .185$, or shape, $F(1, 20) = 0.70$, $p = .414$, could be found suggesting that selection accuracy is not substantially influenced by the geometry of the target.

There were no significant interactions between technique and position ($F(7.65, 153.02) = 1.67$, $p = .114$), technique and shape ($F(1.43, 28.63) = 0.55$, $p = .526$) or between position and shape ($F(9, 180) = 1.57$, $p = .126$).

3.4.4.3 *Touches*

Through observation of the participants during the experiment and examining the collected data we found a common pattern in the participants' behaviour.

For DRAG all participants started their selection in the top left corner and extended their selection towards the lower right corner. Several users tried other directions during the training trials, but eventually settled on the top-left to bottom-right motion. All participants used their thumb and index finger for MULTI. The thumb was used to define the lower left corner of the selection, the index finger defined the upper right corner.

For WIDGET we additionally recorded the number of adjustment steps or handle motions for each trial. The mean number of adjustments was 3.09 (SD = 1.37). This is corroborated by our observation that users generally adjusted two diagonally located handles, sometimes followed by additional adjustment steps.

Figure 10 shows the touches recorded during the experiment relative to the target rectangle scaled to a unit square. As expected for DRAG the top-left corner is the most inaccurate one, because

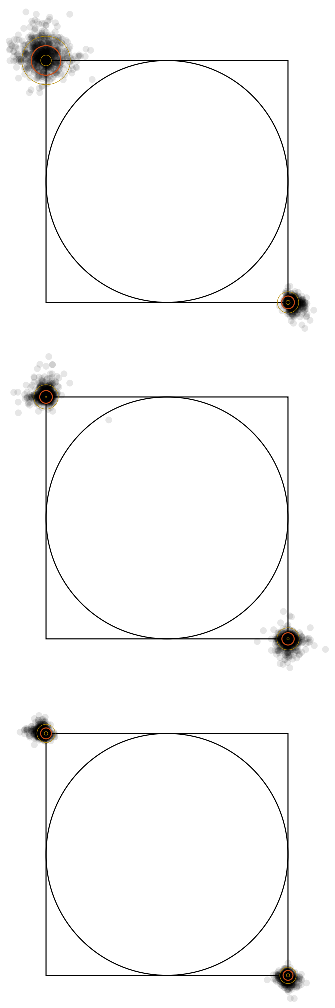


Figure 10: Touch positions relative to the target rectangle scaled to a unit square. Actual touches of DRAG (top) and MULTI (middle) are shown. The final position of WIDGET (bottom) is shown for comparison. Mean distances from the target corner are shown in orange. Yellow circles represent the standard deviation from these means.

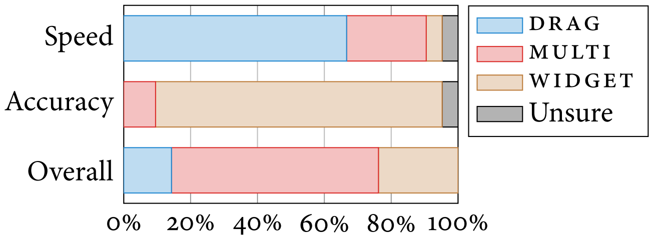


Figure 11: After completing the experiment participants were asked to state their preferred techniques regarding speed, accuracy and their overall preference.

it is the one that could not be adjusted once it was placed. The adjustable corner of DRAG shows a similar precision to both adjustable corners of MULTI. In Figure 10 the touches of MULTI have been mirrored horizontally to allow a better comparison to the other techniques. WIDGET has the highest precision although it is closely followed by MULTI. Note that for WIDGET the positions of two corners defining the selection region were plotted to allow a comparison to the other techniques for which actual touch positions were plotted.

3.4.4.4 *Subjective ratings*

After the experiment participants were asked which they felt was the fastest and the most precise selection technique. The participants' preferences are summarised in Figure 11. For speed 14 participants preferred DRAG, 5 preferred MULTI, 1 preferred WIDGET and 1 had no preference. For accuracy the participants clearly preferred WIDGET with 18 votes (MULTI: 2, DRAG: 0, unsure: 1). So the participants' impressions matched the results recorded during the experiment. Additionally, participants were asked which of the selection techniques they preferred to use. MULTI was preferred by 13 participants, 5 preferred WIDGET and 3 preferred DRAG. Participants that preferred DRAG stated that the higher selection speed was more important to them than accuracy. Accuracy of selection was reported as the single most important property of a selection technique by participants that preferred to use WIDGET. MULTI was chosen by users that liked the compromise of precision and speed. However, the participants' preference of MULTI might be biased due to the novelty of the technique.

3.5 DISCUSSION

As our study has shown our concerns that the increased complexity of controlling more degrees of freedom would negatively impact its performance was unsubstantiated. The increased possibilities that our technique offers do not come at the cost of a performance loss for simple selections.

There is a clear trade-off between speed and accuracy in the three simple selection techniques compared. As WIDGET was the only technique that allowed multiple adjustments it was to be expected that it would be the slowest of the techniques. Evaluating task completion times for WIDGET against the other techniques is not a fair comparison. However, we decided to include WIDGET in our study as it served as a good upper bound for the accuracy that can be achieved for direct selections on multi-touch devices. Due to the fat finger problem DRAG is not a good choice for precise region selections in touch-based applications. For object selections where it is possible to use a tolerance threshold to aid the user it might be a good alternative as it was the fastest and easiest to use technique. This was reflected in the participants' perception of the techniques.

When our technique is compared to DRAG as expected the accuracy for the first corner is much better, because using DRAG it is initially placed and cannot be corrected afterwards. But the interesting result is that the accuracy of both touches of MULTI is similar to the single movable corner of DRAG. Even compared to WIDGET for which also only one corner of the selection area could be moved at a time the accuracy of MULTI is only slightly lower.

A shortcoming of MULTI compared to the other techniques is that the maximum size of the selectable area is restricted by the span of the user's hand. However, for devices with smaller screens that are usually held in one hand this should not be a problem. For larger devices the user can use both hands to perform selections of large areas. Moscovich and Hughes (2008) have shown that bi-manual input even has a slight advantage for controlling two positions on the screen.

3.5.1 *Increasing accuracy*

Complex selections, as supported by our technique, are a combination of several simple rectangular selections, therefore, it is of interest to increase the accuracy of simple selections and thereby also increase the overall accuracy of complex selections.

A common problem for all multi-touch applications is the occlusion caused by the users' fingers and hands. In our experiment at least one or two of the corners of the selection rectangle were occluded by the users' fingertips. When designing the experiment we expected this occlusion to have an effect on the selection accuracy when selecting rectangles or ellipses. However, the target shape did not have a significant effect on the accuracy. The selection rectangle itself acts as a visual guide when aligning the selection region, therefore we suspect that occlusion of the target shape by the fingertips was not an issue.

However, the occlusion caused by the users' hands is still a problem. Depending on the relative position of the user and the selection region on the screen this occlusion might be worse when using a multi-finger technique instead of using a single finger. While it is possible to reduce the occlusion by using the *other* diagonal when specifying the selection rectangle, depending on the position on the screen, this would often result in unnatural positions of the hand. A possible solution to avoid most of the occlusions might be the integration of a cursor offset technique as proposed by Benko et al. (2006). For selection tasks it is even possible to always select a suitable offset direction by adding the offset inwards from each finger towards the selected region.

All tested techniques suffered from inaccuracies that resulted from accidental movement of the touch points while the fingers were lifted from the surface. To avoid this we considered *confirming* selections with the tap of an additional finger (either of the same or the other hand) anywhere on the screen. As we thought this would feel unnatural and unnecessarily complicate the otherwise natural interaction, we did not include it in our study. However, we think that extending the techniques with the ability to confirm selections before lifting the fingers might provide a benefit in certain scenarios and should be explored further.

Another approach to improve selection accuracy would be the usage of hybrid techniques that combine elements of the presented techniques. Our results show that the selection widget was most accurate. As we wanted to evaluate the widget on its own, instead of mixing it with the other techniques and because most multi-touch applications implement it like this, we decided to use a static initial position for the widget. However, the widget is complementary to the other techniques and it would be easy to combine our suggested multi-touch technique with the handles of the selection widget. After the selection is initially placed using our two finger technique selection handles could be shown to modify the selection if necessary.

3.6 DESIGN CONSIDERATIONS

In this section we discuss several design considerations for using our selection technique within more complex interfaces and briefly consider possible variations suitable for different application requirements.

3.6.1 *Conflicts with Other Touch Gestures*

Our selection technique might conflict with other touch gestures that use two touch points as input. For instance, image editing applications usually support a pinch gesture to perform a scale transformation or zoom in and out. When our selection technique is integrated into an application it has to be ensured, that the actions performed by the user are clearly distinguishable. However, there is not a single solution that is perfect for all use cases, so we merely discuss several possible options. What works best has to be decided for each specific application.

A possible solution would be to distinguish different states of the application: a navigation state that allows panning and zooming and an interaction state that allows selection and other actions. While this allows a clear separation of the possible user actions frequent mode switches might interrupt the work flow.

Another possibility is the use of an additional finger to distinguish different actions. The pinch gesture could be modified to be a three finger pinch. We do not suggest to add a third finger to our selection technique, because it is based on the two touch locations, while a pinch gesture is defined by the general motion of the fingers. However, this solution might just shift the problem if the application uses other three finger gestures. Also, it might be unintuitive for the user if the well known pinch gesture is altered.

Additionally, it would be possible to use different timings to determine which action the user wants to perform. A pinch gesture is usually performed right after the user touches the screen. Using a pre-defined time-out it would be possible to trigger a selection, that is if the user touches with two fingers and does not move them more than a small threshold in a specified amount of time a selection is begun. Alternatively, the placement of the fingers could be used to distinguish actions. To modify an existing selection the user has to perform the two touches in a specific order in any case. If the same is applied to the initial selection, that is intentionally placing the fingers down one after the other, it would be possible to distinguish it from a pinch gesture where both fingers touch at roughly the same time. However, for all the suggestions based on timing finding the right time-out is key. If the timings are not adequate the user might accidentally trigger an unintended action. Also, timings that allow a fluid work flow are probably heavily dependent on the user, so it might be difficult to find a reasonable default value that suits all users.

3.6.2 *Selecting Large Regions*

Depending on the device and the size of the user's hand selecting large regions might not always be possible using a single hand. If the user's non dominant hand is not holding the device it is always possible to perform selections using both hands. However, the selection of regions that are larger than the screen remains a problem that should be considered. For the touch and drag approach only a single finger is used to define the selection region. When the finger is moved near the screen's edge the view can automatically be scrolled at a fixed rate (similar to what happens on a desktop computer). However, if two touches are used to define the selection region scrolling is not an option as the whole selection region has to stay on screen. Instead of scrolling when the edge of the screen is approached the view could be zoomed out so that both touch points remain visible. The only drawback is that for very large selections the selection accuracy will decrease the further the view is zoomed out.

3.7 SELECTION OF ARBITRARY REGIONS

So far we investigated selection of rectangular regions. Rectangular selections are the most useful tool for quickly selecting areas of interest. However, they lack the flexibility to precisely select regions with arbitrary shapes. Using our multi-touch selection technique it is possible to refine the selection to get closer to the desired selection shape. For target regions that have a smooth border, a circular region for instance, many refinement steps would be necessary and still the target region would only be approximated by using rectangles. To enable the precise selection of arbitrary regions another tool is necessary. This is the reason why virtually all graphics editing programs offer a free form selection—often called *lasso*—tool. Using this tool the user traces the boundary of the selection region.

In this section we present our ideas how the refinement and selection editing capabilities introduced for rectangular selections can be applied to free form selections. The selection of a new region works identically to the free form selection in mouse or pen-based interfaces. To start a new selection the user begins tracing the selection boundary with a single touch, as shown in Figure 12. When the selection region is closed by crossing the boundary the inside of the traced outline is the selected region.

To edit or refine an already selected region we again draw on the idea of using secondary touches and the relative position of touches for defining the context of the operation. Extending a selected region can be done by simply starting inside the selected region, tracing the boundary of the region that is to be added to the selection and ending the trace inside the current selection. See Figure 13 for an example. If the path does not end inside the selected region the operation is aborted and nothing is changed.

For removing a part of the selected region a secondary touch is necessary. The first touch is placed inside the selected region and a second touch is used to trace a cut line through the selected region, as shown in Figure 14. This cut line has to begin and end outside the selected region, thereby effectively splitting the region into two parts. The part of the selected region that contains the first, stationary, touch is kept and the other part is removed from the selection. In the two examples shown in Figure 14 the same cut is traced by the second touch. As can be seen by comparing Figure 14a and Figure 14b the additional stationary touch decides which of the two regions is kept. It is also possible to first trace the cut line or change the region that is to be kept while the cut line is being traced. As long as a part of the split selection region is indicated by a touch when the finger tracing the cut line is lifted the operation succeeds. If the cut line is not traced all the way through the selection region, i. e. not starting and ending outside the region, or no part of the split region is pointed at the operation is cancelled. With complexly shaped selection regions or curved cut lines it is possible to split the selection region into more than two parts. In this case more than one additional touch can be used to indicate all parts of the split region that are to be kept.

These operations are inspired by physical interactions that can aid in remembering how the different modes, i. e. extension and subtraction, work. For the extension operation imagine the selected region is a blob of paint. To extend the selected region simply dip your finger into the paint and smear the paint to extend the selection. For the subtraction operation imagine you are using scissors or a knife. You hold onto the part that you want to keep (the stationary touch) and then cut away the excess parts (the moving touch, defining the cut line).

Using a series of extension and cut operations the selected region can be continuously edited and refined. To selecte even more complex regions it is also possible to extend the selection by disjoint regions and to cut holes into the selected region. To extend a selection by a disjoint region, first a stationary touch must be placed inside the selected region. Using a second touch the outline of the additional selection region can be traced, as shown

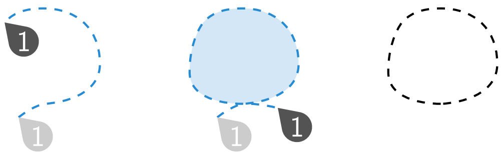


Figure 12: Select a region by tracing its outline. The selection is finished when the outline is closed by crossing it. The initial position of the touch is shown in light grey, its current position is shown in dark grey.

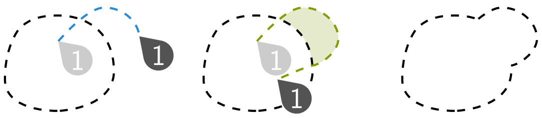
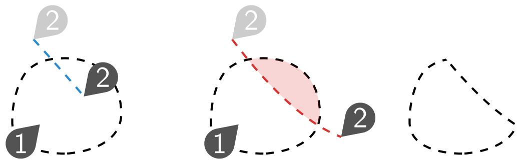
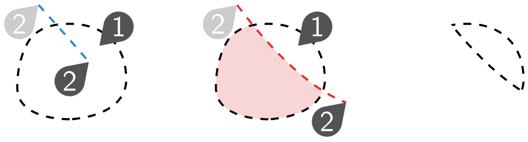


Figure 13: Extending an existing selection by tracing a path that starts end ends inside the selected region.

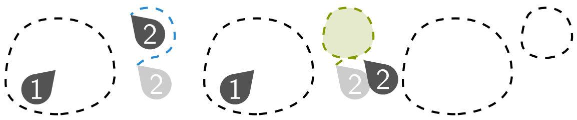


(a) The additional touch (1) indicates that the left part is kept.

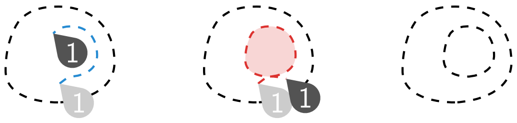


(b) Using the same cut line as in (a) the other part can be kept by moving the additional touch.

Figure 14: To cut away a part of the selection trace a cut line that starts and ends outside the selection. An additional stationary touch inside the selected region indicates which part to keep.



(a) Tracing a closed path outside the selection while placing a secondary touch inside extends the selection by a disjoint region.



(b) Tracing a closed path inside the selection cuts a hole into it.

Figure 15: Creating complex selection regions.

in Figure 15a. By closing the outline the operation is completed. The additional touch is necessary to distinguish the extension by a disjoint region from starting a new selection. If the traced outline is not closed when the touch is lifted the operation is cancelled. To cut a hole into the selected region a closed outline that is completely contained inside the selected region can be traced. See Figure 15b for an example. Before the outline is closed the operation can be cancelled by ending the trace outside of the selection.

Hinckley et al. (2006) and Moran et al. (1997) have used similar gestures for pen-based interfaces. They refer to the extension gesture as *bump*. Removing a part of the selection is called a *bite* gesture. Moran et al. (1997) discuss several heuristics for determining which part of the selection is to be retained after a bite gesture. They have experimented with variants that examine the arc length, the area or the content of the parts resulting from a cut. However, with their technique there is no possibility to explicitly choose which part to keep, which leads to some compromise on the flexibility of the cut gesture. With our proposed technique the parts that are to be kept are explicitly selected by the user, so the shape of the cut gesture is not subject to constraints (except actually *cutting* the selected area). A gesture similar to the hole gesture, shown in Figure 15b, called circles of inclusion has been used by Kurtenbach and Buxton (1991).

3.7.1 *Visual Feedback*

The outline of a region that is being traced is visible as a dashed blue line. As soon as the mode of operation can be deduced from the current state of the trace line its colour is changed to green for extensions or red for cutting operations. As can be seen in Figures 13, 14, and 15 a preview of the current operation is also shown by shading the affected region in green or red. If the user starts one of the editing operations and the visual feedback does not show the desired outcome each operation can be cancelled as described above.

3.7.2 *Collisions with Other Touch Gestures*

As mentioned earlier for the rectangle selection the design space for multi touch input metaphors is constricted by common input metaphors using one or two touches, like panning and zooming. For most applications input metaphors are bound to collide with these common basic operations. Tracing a path with a single touch can not be distinguished from a panning gesture using a single finger. To overcome this ambiguity an application has either to support different modes for viewing and editing that have to be switched explicitly or needs contextual information to decide if the user wants to pan the view or trace a path.

With the exception of the selection of the first region all operations of our free form selection technique fall into two categories: either the traced path starts inside the selected region or an additional touch inside the selected region is necessary. So all editing operations begin with a stationary touch or a traced path *inside* the selected region. This can be used as contextual information of the touch operations by an application. Touches that are placed inside the selected region are interpreted as selection operations. Touches that are placed outside the currently selected region can be interpreted as view operations like panning or zooming or any other operation the application requires.

While this is a straightforward solution to automatically identify selection editing operations this is not possible for beginning a new selection. As there is no contextual information that can be used by the application some other way to signal the selection operation has to be used. Depending on the application a button on the user interface or some other means to trigger a new selection is required.

4 PINS ’N’ TOUCHES: AN INTERFACE FOR TAGGING AND EDITING COMPLEX GROUPS

Selection of one or more objects is a fundamental operation in user interfaces, and usually done with the mouse or pen. Such interfaces rely on a single interaction point (cursor). Touch interfaces have (largely) adopted the single-touch convention, especially for tagging objects. Yet, this design does not leverage the unique capabilities of multi-touch, such as multiple simultaneous touches, which enable the user to effectively control multiple cursors simultaneously.

4.1 INTRODUCTION

In this chapter we explore the idea of using multiple touches for tagging of groups of objects to enable more efficient tag editing by adding or removing *multiple* objects to/from existing groups. We discuss the relationship between tagging and grouping and present use cases for tagging of complex groups. Then we introduce our new multi-touch tagging technique and describe how it enables the editing of tags of complex groups. To evaluate our interaction technique we present two user studies that identify the performance benefits of our method.

4.1.1 *Motivation*

While examining existing tagging techniques for multi-touch input in Chapter 2 we noticed a lack of techniques that enable tagging of complex groups. Looking at the design space for such a technique, we developed the following research questions:

- How can we support tagging of more than a single object, which resembles a grouping task?
- For tagging with multiple concurrent tags, how is overlapping handled, i. e. the fact that objects belong to more than one group?
- How do we show that an object belongs to multiple groups?
- How can we use multiple touches to support efficient editing of multiple concurrent tags?

4.1.2 *Contributions*

In this chapter we propose a novel interaction technique based on pinning touches that addresses the raised research questions and facilitates the definition of complex groups through tagging.

The results of the survey of related work presented in Chapter 2 is summarised in Figure 16. An overview of all pair-wise combinations of classifiers used in the survey is shown. The numbers at the intersections indicate how often the corresponding combination is present in previous work. The classification of our novel tagging interface is shaded in green. Hatched areas represent new contributions. Looking at the gaps in Figure 16 one can identify that true *multi*-touch interaction together with common group selection techniques like path or rectangular region has not been explored. The combination of pin and hold gestures with other selection techniques yields new possibilities for editing of tags or

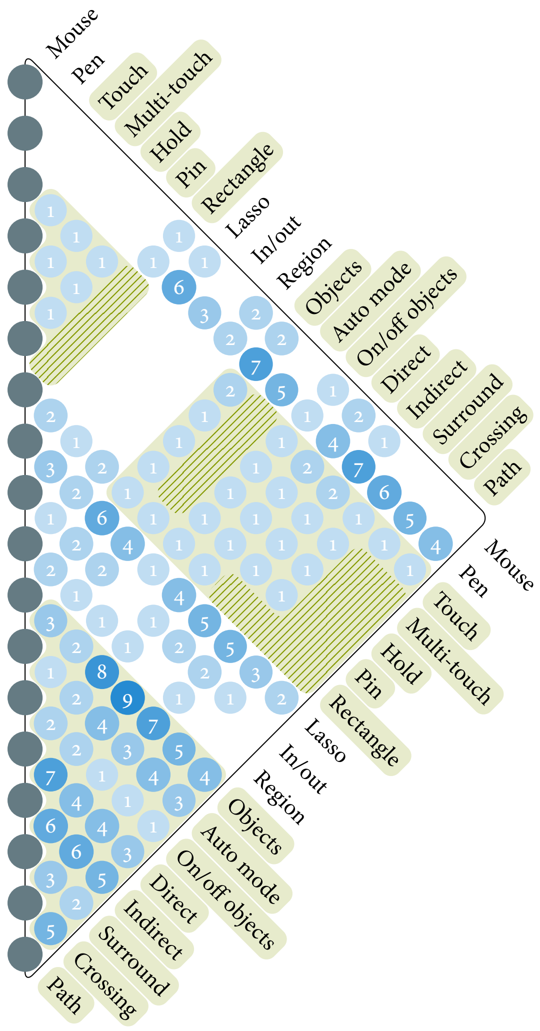


Figure 16: All pair-wise combinations of classifiers with their number of occurrences provide an overview of the examined body of previous work on group selection in Chapter 2. The classification of our novel tagging interface (shaded in green) and the new contributions (hatched area) are also shown.

selections. Wobbrock et al. (2009) identified a gesture similar to our *pin* gesture, but did not explore it further.

Using the classifiers introduced in Section 2.1 we classify our technique as:
Touch Multi-touch Hold Pin Rectangle Objects Auto mode On/off objects Direct Indirect Surround Crossing Path.

More specifically, we introduce the following contributions:

- A multi-touch user interface technique for tagging of complex groups;
- support for multiple concurrent and overlapping groups;
- pin gestures for efficient editing of existing groups;
- an evaluation of our new techniques in two user studies.

4.1.3 *Pinning Touches*

Multi-touch interaction is often constrained to a single touch. There are two reasons for that. Most multi-touch devices are small hand-held devices. As one hand is used to hold the device, only a single hand is available for interaction. A second reason is that most user interface development was influenced by traditional mouse- or pen-based interfaces, which implies a single cursor or touch. The only multi-touch interactions that really use multiple touches are common gestures such as two finger scrolling or pinching. As large tablets, notebooks and desktop computers with multi-touch capable displays become available true multi-touch input becomes more desirable.

Controlling two (or more) independent points of interaction is difficult as shown by Moscovich and Hughes (2008). Yet, a single moving touch can be augmented by additional stationary ones. These additional touches can be used to define contextual information for the action performed by the primary touch. In

Chapter 3 we used a similar idea to select rectangular regions. Here we present a generalization of this work. We extend the touch input vocabulary by *pinning touches*, i. e. stationary touches that are held during *other* touch actions. These pinning touches *pin down* objects or user interface elements. Standard touch interaction uses actions such as tapping, swiping, tracing paths, flicking and two finger swipe or pinch gestures. By using pinning touches we can associate additional contextual information with standard actions. Thus, operations that typically require a mode switch can be performed directly with pinning touches. For example, deleting images in a photo application usually requires a temporary switch to a special delete mode and then tapping on individual images to perform the deletion. With pinning touches the explicit mode switch is unnecessary. Images can be pinned down to specify them as context for the delete action: The user can touch and hold, i. e. pin, several images and then tap a button to delete them. Alternatively, the delete button can be pinned down to implicitly switch to a delete mode. All images that are then selected, by tapping or by some other selection method, will be deleted. As two simultaneous touches (one of them on the delete button) are required, accidental deletion is unlikely.

Similarly text attributes can be applied by pinning down an attribute in text processing, for instance to *paint* a bold font attribute over a passage. Additional pinning touches can even apply multiple attributes at once. Also, using pinning touches styles can be copied by pinning a word and painting over text to change its style to match that of the pinned word.

Pinning touches can also extend other multi-touch gestures. For instance, a scaling gesture (pinch) can be constrained to horizontal, vertical or uniform scaling by pinning down respective modifier buttons. Pinning touches can also enable functionality similar to a context menu. When a context menu is invoked, the properties of the object at the cursor are used to decide what actions to display in the context menu. Similarly, pinning touches can also supply context to actions performed with the primary touch. Gutwin et al. (2014) present a command selection technique that combines a pinning touch to access a menu with taps to select commands from this menu. Yet, no context was derived from the pinning touch in their work.

4.1.4 *Using Pinning and Tags to Define Groups*

Combining objects into groups is a common user interface operation in graphic design or layout software. Usually objects are first selected and then a widget or a shortcut is used to group these objects. Through pinning touches, groups can be defined without using a menu or shortcut. For this, a single object is first pinned down and then the other objects to be grouped with this object are selected. If the pinned object is already part of a group, the additional objects are added to its group. Note that the pinning touch *implicitly* selects the target group as part of the interaction.

In photo management software grouping is also frequently used to structure and sort photos, such as photos of an event or a subject. For example, a user wants to group all photos taken while visiting Münster. In another group all photos of the user's dog are collected. When the user brought his dog along to a trip to Münster, there are photos that belong to more than one group. To represent such group associations, tags can be added to the objects. Then, all photos from Münster share one tag, and those of the user's dog another one.

In this chapter we present an interaction technique that combines pinning touches and tags to enable the definition and editing of complex group associations.

4.1.5 *Use Cases for Tag-based Groups*

In recent versions of OS X multiple tags can be assigned to files for assisting with categorisation. Most modern photo applications

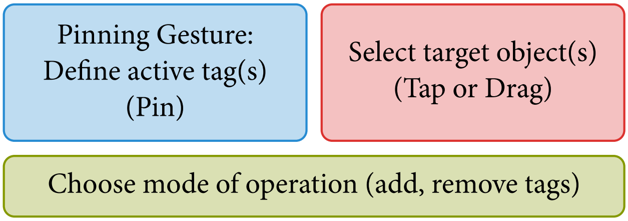


Figure 17: Our technique consists of three components: A technique to define a set of active tags (top left) and another for selecting one or more target objects (top right). Additionally, we require a method for choosing the mode of operation, such as adding or removing objects to the group (bottom).

permit users to assign tags based on faces in a picture or the location the photo was taken at, supported by (semi-)automatic classification. Many e-mail clients support tagging of messages with different colours. Often only a single tag per message is supported, but Gmail permits multiple tags per e-mail. All these use cases require multiple overlapping tag groups, such as an e-mail that is both *work related* and *important* or a photo of *my dog* taken in *Münster*.

4.2 OUR PIN-BASED INTERFACE FOR TAGGING

We propose a new interaction technique, which enables the tagging of one or more objects. It can be used to assign multiple tags concurrently to each object, permitting objects to belong to multiple groups at the same time. The association is visualised through different coloured tags. Once tags have been assigned, our technique enables easy modification of these tags.

Three main components are required in our interface: A way to define a set of active tags, a technique for selecting one or more target objects, and a way to specify the mode of operation: active tags can either be added to target objects or removed from them. The three components are visualised in Figure 17.

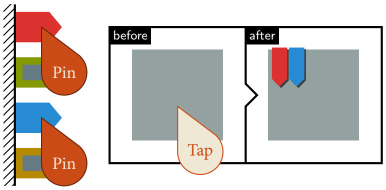


Figure 18: A set of tag widgets is visible at the screen border. Pinning down widgets (on the left) defines a set of tags for selection operations. Alternatively, the user can pin existing objects with desired tags, as shown in Figure 19.

4.2.1 *Defining a Set of Tags*

One of the main design ideas behind our tagging technique is the use of *pinning* touches. If the user touches and holds a single or multiple tagging widgets or if already tagged objects are held while performing other actions, we call those *pinned* down. Pinned tags/objects are used in our technique to define the set of active tags for subsequent operations.

4.2.1.1 *Tag Widgets*

We display a set of widgets, one for each available tag, at the border of the screen, as shown in Figure 18. Pinned widgets activate the corresponding tags, which are applied on selection. Through pinning multiple widgets the user can compose any combination of tags, as illustrated in Figure 18, where the red tag is combined with the blue tag.

4.2.1.2 *Pinning Already Tagged Objects*

When the user pins down an object, this adds the tags assigned to this object to the set of active tags (Figure 19). Pinning down multiple objects at the same time defines the union of the objects' individual tags as active (Figure 20). The tag widgets provide visual feedback of the set of active tags. Assuming that similar objects are in close proximity, such as sequences of photos in a camera roll, applying the same set of tags to similar objects is then a *local* operation: First, pin an already tagged object down and then select the other (neighbouring) objects. For example, while tagging photos, this permits the user to easily specify that this is *also* a photo of my dog taken in Münster.

4.2.2 *Selecting Target Objects*

With a set of active tags chosen, one or more target objects have to be selected. Our tagging technique supports tapping for single objects and path-based operations for selecting multiple ones.

4.2.2.1 *Selecting Single Objects By Tapping*

Selecting single objects via tapping is very common in touch interfaces (e. g. Leitner and Haller 2011; Xu et al. 2012). If a single tag is pinned down with our technique, this tag is toggled on any (other) tapped object. For example, in the top sequence in Figure 19, the yellow tag is pinned down. It is added to the tapped target object, as it has not yet been assigned the yellow tag. In the bottom sequence the pinned blue tag is already assigned to the target object, so it is removed. When multiple tags are pinned, the result depends on the tags currently assigned to the tapped object. Figure 20a shows two sequences with a pinned set of yellow and blue tags. In both cases the tapped object has not been assigned both tags, so the pinned tags are added to it. If, on the other hand, the target object has already been assigned all active tags, as shown in Figure 20b, the pinned tags are removed from the target object. *Pinning the void*, i. e. the background area between objects, and tapping on an object removes all tags from said object.

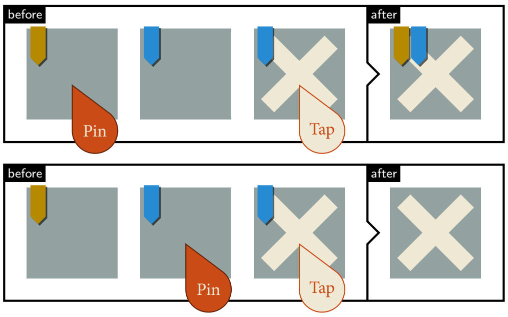
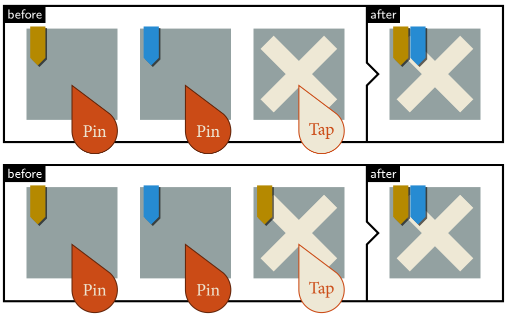
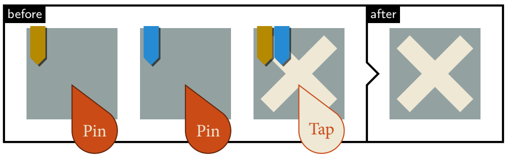


Figure 19: Pinning down a single tag: tapping an object toggles the tag on that object.



(a) If multiple objects are pinned down, the union of their tags is applied when tapping.



(b) Tags are removed if the tapped object already has all pinned tags.

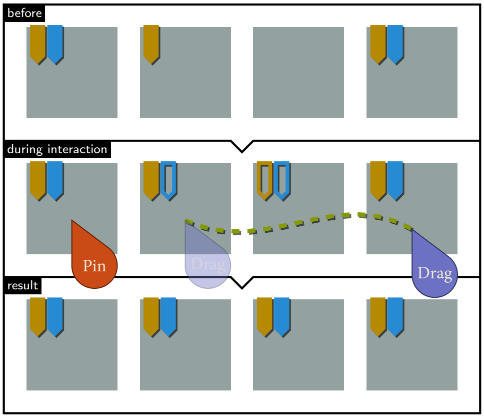
Figure 20: Semantics for multiple pinned tags.

4.2.2.2 *Tagging Multiple Objects*

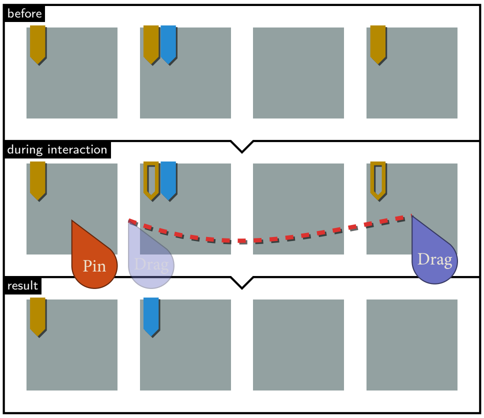
The user can trace a path to operate on, i. e. tag, multiple targets. The operation of adding or removing tags is applied to all objects intersecting the path. The mode of operation, i. e. if the tags are to be added or removed, is defined by the position of the first touch defining the path. If the touch is inside an object the current set of pinned tags will be added. In Figure 21a the pinned tags (yellow and blue) are added to the objects intersecting the path. If the first touch is in the *void*, i. e. on the background, the pinned set of tags will be removed from the objects, as shown in Figure 21b where the yellow tag is removed from the target objects. This is an extension of the idea introduced for our region selection technique in Chapter 3.

4.2.3 *Visual Feedback and Order of Operations*

The colour of the traced path shows the mode of operation: green for adding tags, red for removing them. While the path is traced, the system shows a preview of the resulting tag changes. Solid tags remain unchanged by the operation, while outlined tags illustrate the changes. This is illustrated in Figure 21. As shown in Figure 17 our system is comprised of three components. The two main components define a set of active tags and operate on one or more target objects. The order of these steps is not predefined and our technique supports all combinations. For instance, if the preview does not show the desired result, it is possible to change the set of pinned tags while tracing a path and see an updated preview. Even tagging target objects by first tracing a path and afterwards pinning down tags, before the dragged finger defining the path is lifted, is possible. As long as the pinning touches are not removed, they continue to define the set of active tags. Therefore, it is possible to use the same set of tags for multiple operations.



(a) If the path begins inside an object tags are added.



(b) If the path begins in the *void* (i. e. on the background, outside of all objects) tags are removed.

Figure 21: The starting point of the path decides if the pinned tags will be added (green path) or removed (red path) from objects intersecting the path. Outlined markers preview the changes that will be applied.

4.3 USER STUDY 1: INITIALLY ASSIGNING TAGS

To validate and evaluate our new multi-touch tagging technique we performed two user studies. In the first one we compared our path-based technique (PATH), which enables users to use a path gesture to tag multiple objects, with the more *traditional* single-touch tapping approach (TAP), where objects are tagged by tapping them one by one. We presented participants with a task similar to tagging photos in a photo application.

4.3.1 *Participants and Set-up*

21 participants (20 male, 1 female), ages 20 to 34 (median 28), participated in the study. All but two were students from the local university. All participants reported to have at least some experience with multi-touch devices. Most of them use smartphones or tablets regularly. The experiment, including instructions, training and debriefing, took about 45 minutes per participant. Breaks were permitted at any time. One participant had to be dropped due to a series of interruptions. The experiment was performed using an Apple iPad attached to a desk in front of the sitting participant.

4.3.2 *Procedure*

For both techniques participants were asked to assign tags to split the presented objects into groups, based on their content. The order of techniques was counterbalanced using a 2×2 Latin square. After finishing the experiment participants completed a short questionnaire.

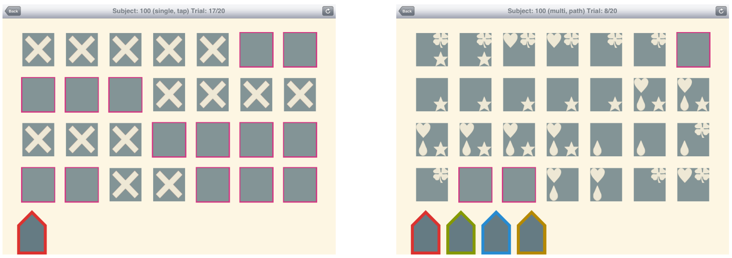

Back
Subject: 100 (single, tap) Trial: 17/20
Back
Subject: 100 (multi, path) Trial: 8/20

Figure 22: Screenshots of the two phases of the first study. (left) In the first phase all objects marked with an X had to be tagged. (right) In the second phase four different tags had to be assigned: heart → red, clover → green, drop → blue, star → yellow.

4.3.3 *Experimental Task*

The experiment was divided into two phases. The first one used a single tag and participants were asked to tag all objects marked by an X, see Figure 22 (left). The second phase used four tag categories: red, green, blue and yellow. Objects were marked with zero to four symbols representing the target tag set to be applied to each object, see Figure 22 (right). Tags had to be assigned as follows: heart → red, clover → green, drop → blue, star → yellow.

Initially, we contemplated using photos. However, the lack of a suitable photo set in the public domain led us to choose a different approach. Moreover, we did not want to deal with issues of users not being able to differentiate between similar faces. Thus, and also because we are only interested in analysing tagging performance, we chose to simplify the task to recognising symbols. We first randomized the symbol positions on the objects to more closely resemble features in a photo. However, a pilot study revealed that the cognitive load for this was too high and resulted in both very high selection variability and overall slow performance. As we are looking for reliable measurements of selection task performance we needed to eliminate the potential confound of image recognition. Therefore, we made the task easier and assigned symbols to fixed locations on the objects.

In both phases no tags were assigned initially. When all tags were correctly assigned, the trial was automatically completed. After a notification, the experiment automatically advanced to the next trial. A coloured border was added to objects that had all tags correctly assigned. This helped participants identify single missing tags towards the end of each trial.

4.3.4 *Experiment Design*

In each trial a sequence of 28 objects (7×4) was displayed. For each phase 10 object sequences were generated and shown twice to the participants, resulting in a total of 80 trials per participant (2 phases, 2 techniques, 10 sequences, 2 repetitions). The order of sequences was randomized for each user and technique. To minimize the effect of switching selection techniques four training trials were added before each trial set, but discarded from the analysis.

Photo organization applications usually display photos in the order they were taken (like on a camera roll). This results in sequences of similar photos that would require similar tags. Therefore, we generated symbol sequences using an exponential distribution with an expected sequence length of 3 ($\lambda = 1/3$) for the trials in our study. For the second phase, sequences of each symbol were generated independently of each other, resulting in zero to four symbols per object.

4.3.5 *Results*

To analyse the data of the experiment we performed a 2×2 within-subjects, repeated measures ANOVA ($\alpha = .05$) for the used selection technique and the presented sequence. The dependent

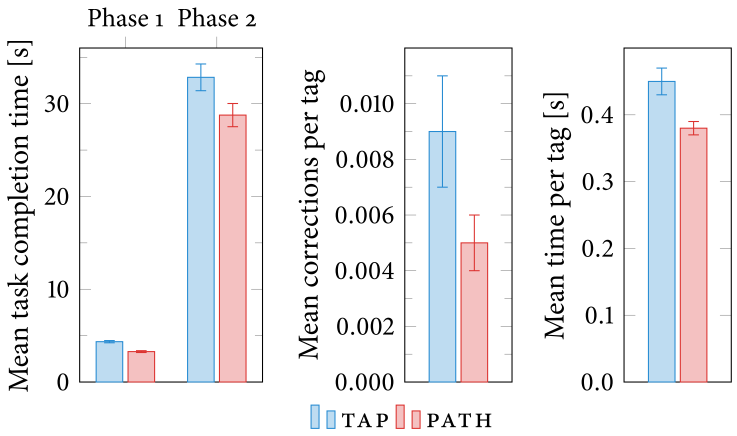


Figure 23: Comparison of TAP and PATH for: (left) mean task completion time in both phases of the first study, (centre) mean number of corrections for each placed tag, (right) mean time for each placed tag.

measure was the task completion time, measured from the first touch in each trial until all tags were correctly assigned. The results are summarized in Figure 23.

4.3.5.1 *Task Completion Time*

For the first phase (single tag) a significant main effect of technique, $F(1,19) = 100.79, p < .001$, was found, with mean times of 4.35 s (SE = 0.12) for TAP and 3.28 s for PATH (SE = 0.10). A significant main effect of the presented sequence, $F(9,171) = 70.56, p < .001$, was found, with mean times ranging from 2.25 s to 5.12 s. For the second phase (up to four tags) a significant main effect of technique, $F(1,19) = 6.93, p = .016$, was found, with mean times of 32.84 s (SE = 1.44) for TAP and 28.77 s for PATH (SE = 1.25). A significant main effect of the presented sequence, $F(9,171) = 20.42, p < .001$, was found, with mean times ranging from 24.40 s to 36.17 s. Post-hoc analysis for both phases showed that the differences in completion time were caused by the sequences that required the least/most tags to be completed. A linear trend between the number of target tags and the completion time could be observed: task completion time = $-3.41 + (0.59 \times \text{number of target tags})$, $R^2 = 0.8$.

4.3.5.2 *Error Rate*

To measure the error rate, we looked at the number of corrections a user performed in relation to the total number of tags that had to be placed. There are two types of such *errors*: Assigning a wrong tag to an object, which has to be removed again later on, and removing a tag that was correctly assigned. We combined the error rates for both phases and performed a 2×2 within-subjects, repeated measures ANOVA using experiment phase and technique as factors. A significant main effect of phase, $F(1,19) = 48.33, p < .001$ was found, with mean error rates of 0.002 (SE = 0.001) for the first and 0.012 (SE = 0.002) for the second phase. Overall, a significant main effect of selection technique, $F(1,19) = 5.97, p = .024$ was found, with mean error rates of 0.009 (SE = 0.002) for TAP and 0.005 (SE = 0.001) for PATH.

4.3.5.3 *Time per Tag*

To further investigate how the two techniques compare to each other and to illuminate the difference between the two phases, we looked at the mean time to place a tag. We calculated the time per tag for each trial by dividing the task completion time by the number of tags that had to be assigned to complete the sequence. Any wrong tags that were assigned and had to be removed were not counted, thus yielding the mean time it took to correctly place a tag. To analyse the time per tag we performed an ANOVA of the combined values of both phases. A significant main effect of phase, $F(1,19) = 261.77, p < .001$, was found, with mean times of 0.29 s (SE = 0.01) for the first phase and 0.55 s (SE = 0.02) for the second phase. Overall, a significant main effect of selection technique, $F(1, 19) = 22.41$, $p < .001$, was found, with mean times of 0.45 s (SE = 0.02) for TAP and 0.38 s (SE = 0.01) for PATH.

4.3.5.4 *Observations*

While it was possible to assign multiple tags at the same time and participants were shown how to do so, we noticed that after the first few operations virtually all tags were placed one after another during the training. We also observed a change in the usage of the path gesture between the two phases of the experiment. Thus we took a closer look at the recorded paths. Figure 24 shows the distribution of path lengths, measured as the number of affected objects. In the first phase a length of five was most frequent, but considerably longer paths occurred as well. For the second phase the peak shifts towards a path length of 2, followed by a steep fall-off. An overlay of all recorded paths for a representative sequence of each phase is shown in Figure 25. In the first phase, many long winding paths were used, in most cases to tag each connected component of marked objects via a single path gesture. During the second phase, most tags were applied in a *scanline* pattern. Rows of objects were considered one after another, resulting in predominantly horizontal linear paths, as in Figure 25 (right). Users selecting a whole row caused the secondary peak for paths of length 7 in phase two.

4.3.5.5 *Subjective Ratings*

After the experiment participants were asked which tagging technique they perceived as faster. All but two thought they were faster with the path technique. The remaining rated both techniques as equal. Additionally, participants were asked for their preferred technique. All but one preferred the path technique to tapping. Most participants found individually tapping objects one by one to be tedious. One participant preferred tapping and

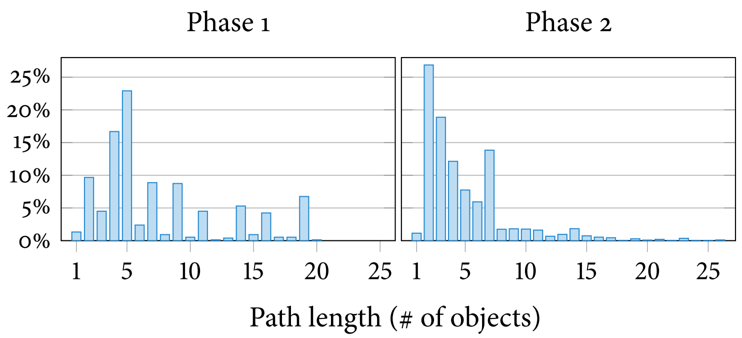


Figure 24: Distribution of path lengths, i. e. the number of affected objects, for both phases of the experiment. A clear shift towards shorter paths is visible for the second, more complex, phase.

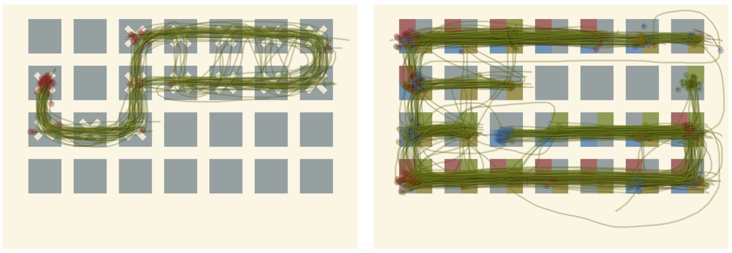


Figure 25: Illustrations overlaying paths of representative tag operations for each phase of the experiment. (left) In the first phase longer, curved paths were used. (right) The paths of the second phase were shorter and show a distinct *scanline* pattern.

stated that path had a higher risk of selecting more objects than intended. However, that participant also stated that the path technique would still be faster overall. Some participants that started the study with tapping inquired if there was a way to affect multiple objects at once—some even proposing tracing a path through them.

4.4 USER STUDY 2: MODIFYING TAGS

The first user study showed that our path-based selection technique is well suited for assigning tags to an untagged set of objects. Although the participants were introduced to the mechanism for addition or removal of tags, they rarely used it, likely because it was only needed for the removal of individual accidentally placed tags. Therefore, the first study yielded no insight on how users respond to the mechanism for deciding addition or removal of tags. Thus, we designed a second experiment that forced participants to use both add and removal operations.

4.4.1 *Participants and Set-up*

16 participants (15 male, 1 female), ages 26 to 38 (median 29), participated in the study. All but one were students recruited from the local university department. All participants reported to have at least some experience with multi-touch devices. Most of them use smartphones or tablets on a regular basis. Participants took about 20 minutes to complete the study and were permitted breaks. The set-up was identical to the first user study.

4.4.2 *Procedure*

Sequences of objects with already assigned tags were presented to the users. With each technique participants were asked to correct the tag assignment, so that only the marked objects were tagged—adding and removing tags as necessary. The order of techniques was counterbalanced using a 2×2 Latin square. After the experiment participants were asked to complete a short questionnaire to provide their subjective ratings and preference of both techniques.

4.4.3 *Experimental Task*

The experimental task was similar to the task in phase one of the first study. However, this time there were already some tags assigned to objects. Only a single tag category was available. Participants were asked to edit the presented tag assignment, so that only the objects marked by an X were tagged. In contrast to the first study, users not only had to add but also to remove wrongly assigned tags. When all tags were correct, the trial was automatically completed and the experiment advanced to the next trial. To help participants locate the final missing or misplaced tags a coloured border was added to correctly tagged objects.

4.4.4 *Experiment Design*

In each trial a sequence of 28 objects (7×4) was displayed. Ten object sequences were generated and shown twice to each participant, for a total of 40 trials (2 techniques, 10 sequences, 2 repetitions). The order of sequences was randomized for each user. Four training trials were added before the trials and discarded from analysis. The sequences were generated as in the first study. For each object sequence a sequence of tags was independently generated using the same exponential distribution. Overlaying the tag and object sequences resulted in groups of correct and wrong tag assignments. In a first pilot for this experiment tags were assigned completely randomly. While randomly distributed errors may occur, we aimed to simulate a pre-tagged data set with some kind of systematic issue. For instance, if a face detection

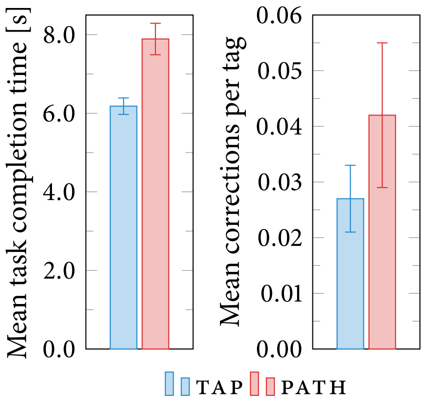


Figure 26: Comparison of TAP and PATH for: (left) mean task completion time, (right) mean number of corrections for each changed tag.

algorithm fails for one image, it is likely that it will also fail for very similar images in a sequence.

4.4.5 *Results*

The collected data was analysed using a within-subjects, repeated measures ANOVA ($\alpha = .05$) using technique and the presented sequence as factors. If the assumption of sphericity was violated degrees of freedom were adjusted via Greenhouse-Geisser. The results are summarized in Figure 26.

4.4.5.1 *Task Completion Time*

The task completion time for each trial was measured from the first touch until the trial was completed successfully. We were able to find significant main effects for technique, $F(1,15) = 40.19$, $p < .001$, and presented sequence, $F(9,135) = 29.41, p < .001$. The mean task completion times were 6.18 s (SE = 0.21) for TAP and 7.89 s (SE = 0.40) for PATH. As in the first study, post-hoc analysis showed that the significant differences in completion time were caused by the sequences that required the least/most tags to be changed.

4.4.5.2 *Number of Corrections per Tag*

We analysed the number of superfluous tag changes a user performed in relation to the minimum number of tag changes required to complete a trial, i. e. the sum of tags that had to be added and removed. There was no effect for technique, $F(1, 15) = 1.334$, $p = .266$, nor for sequence, $F(3.76, 56.35) = 1.64$, $p = .180$.

4.4.5.3 *Subjective Ratings*

After each set of trials participants were asked to rate each technique concerning ease of learning, speed and error rate. Each category was rated using a 7-point Likert scale (1 = very good to 7 = poor). The results are summarized in Figure 27. After the trials, participants were asked which technique they preferred for each category and their overall preference. The results are shown in Figure 28. The collected data suggests that the participants' overall preference was influenced mostly by the perceived speed. All participants that preferred PATH for its speed also preferred it overall.

4.5 DISCUSSION

The results of the first study demonstrate that tracing a path through groups of objects was faster than tapping each object individually. This is not unexpected (Baudisch 1998; Luo and Vogel 2014), but current tagging techniques rarely support selection by path. While trials in the second phase of the first study required placing four times as many tags on average, it took users nearly ten times longer to complete the trials. This shows that the second ex-

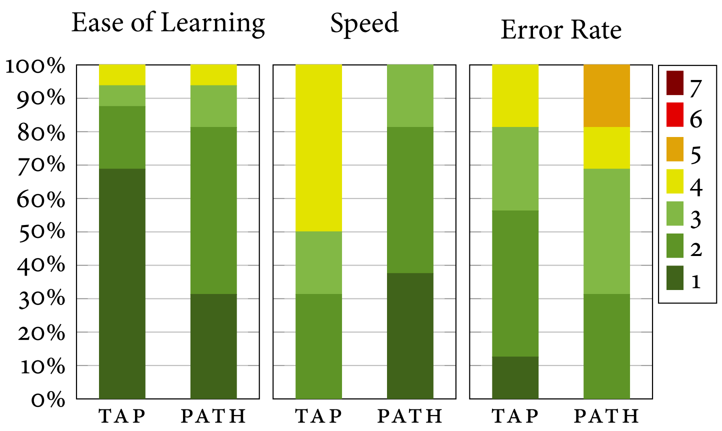


Figure 27: Ratings on ease of learning, speed and error rate based on a 7-point Likert scale (1 = very good).

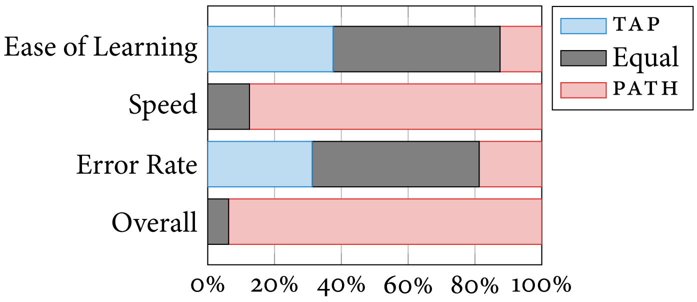


Figure 28: After completing the experiment participants were asked to state their preferred techniques regarding ease of learning, speed, error rate and their overall preference.

periment was more complex, due to the different symbols and the four different tag categories. Although the data shows a significant difference in the task completion times between both techniques, the time is dependent on the number of tags that needed to be assigned to complete a trial. To better understand the interaction of the different techniques and the complexity of the task in both phases, we analysed the time and corrections required for each correctly placed tag in both phases. Using a path gesture to tag multiple objects is more complex than tapping each single object individually, as multiple objects have to be considered while planning the gesture. Our results show that the benefit of affecting multiple objects outweighs the seemingly higher complexity of the path gesture. PATH was significantly faster than TAP with a mean time per tag that is about 15% lower. This is a very promising result as it opens the way for more efficient touch interaction compared to tapping.

In the second phase of the first study it took users nearly twice as long to correctly assign a tag. The mean error rate was six times as high as in the first phase, which is to be expected due to the higher task difficulty. Still, the error rate remained surprisingly low for the large amount of fast tagging operations: one error per 83 modifications. Although the time required for tagging as well as the error rate increased when dealing with multiple tags, we believe that the results show that our interface still works well under such circumstances.

Some users modified multiple tags for final corrections at the end of a trial. We believe that this is due to the nature of our experimental task. One potential cause is that our experiment started with an *empty* object sequence. Consequently, it was somewhat easier to finish setting all tags of one category before moving to another tag, compared to constantly switching the set of active tags. The increased complexity of the second phase also caused users to consider one row after another, as it was harder to see larger connected components at a glance. The resulting *scanline* pattern is illustrated in Figure 25. This change in the usage of the path gesture is the reason for the shorter paths observed during the experiment’s second phase, as shown in Figure 24.

While the first study evaluated our interface for placing tags, it provided little insight into our method for changing the mode of operation, i. e. if tags are to be added or removed, for the path technique. In the second study users were required to add and remove tags to complete the trials. After a short explanation of the principle and a bit of experimentation during the training, participants quickly grasped the concept and were able to complete the experiment task successfully. When we compared the performance of the path technique to individual tapping of objects we were surprised to find that, in contrast to the first study, individual tapping was faster. Upon taking a closer look at the sequences generated for the experiment, these results were not unexpected. Baudisch (1998) identified that several objects need to be manipulated in one mouse drag to see a speed-up over single mouse clicks. Although Baudisch’s statement relates to a mouse-based user interface, the same insight holds true here. In our task, the sequences of consecutive missing or misplaced tags that had to be corrected were often no longer than two or three objects. For such short paths the speed benefit of the path selection technique does not apply. So there is an obvious minimum to the number of objects before a speed-up can be realized with the path technique. Even though the quantitative measurements show that tapping was significantly faster, all but two participants rated the path technique as faster than tapping. The qualitative results further show that tapping was rated as both easier to learn and less error-prone. The perceived speed appears to dominate the overall rating.

The measured number of corrections shown in Figure 26 should not be interpreted as an error rate. It is a combination of accidental errors and deliberately superfluous actions. With the path technique a few users added more tags than necessary and later removed them again. During debriefing three participants highlighted that they did this to chain smaller actions together, by cleverly combining add and remove operations. This behaviour was not observed during the first study.

As the work of Kin et al. (2009) has shown, experienced users can perform better by using more than one finger and tapping multiple objects at once. While this is supported by our implementation, during the experiment all users performed taps using a single finger. Some users discovered the option to tap multiple objects at once, but reverted back to single tapping for the study. Although we only applied our interface to objects in a regular grid layout, previous work (Dehmeshki and Stuerzlinger 2009a) suggests that a path-based selection technique should work well for node networks with non-rectangular layouts.

4.6 DESIGN CONSIDERATIONS

In this section we outline several design considerations for integrating our proposed technique into an application's GUI. Depending on the application scenario, the object layout can vary from a grid to a random distribution. This has a strong influence for picking the "best" method to select multiple objects. We also discuss the need to adapt to different devices: not only for different screen sizes, but also uni- vs. bi-manual input.

It is not possible to present the single best variation of our technique, which fits all scenarios and devices. Thus, we present variations for different components of our interaction technique. However, all variants that we discuss here use the same underlying principle. The two main components, i. e. defining a set of tags and selecting target objects, are independent of each other. This provides the flexibility to pick the best match for the target application and device combination.

4.6.1 *Necessity of Tagging Widgets*

Pinning objects that already have the desired set of tags works best if related objects are close to each other. However, if there is no object nearby that can be used to *gather* the desired tags by pinning, an appropriately tagged object has to be found first. Assuming such an object exists, such a search could take time. But, if a tag has not been used yet, finding said tag among the objects is even impossible. Another example is the removal of a single specific tag. The user would have to find and pin first an object that has only the desired tag. Thus another mechanism for dealing with these situations is necessary. The user can always use the tagging widgets at the side of the screen when there is no object with the desired set of tags available (or it would simply take too long to find it). This specifically targets the removal of a single tag and the initial placement of tags that have not been used yet.

4.6.2 *Copying vs. Toggling vs. Combining Tags*

There are several possibilities for applying a set of tags to one or more target objects. The simplest possibility is to copy the tags to all target objects, overwriting any previous tags—effectively *painting* the tags on the objects. However, for complex tag assignments this may need more steps than ideal. Adding another tag to differently tagged objects would become tedious, as existing tags would have to be re-selected, so that they are not overwritten. We decided to use the more expressive combination of tags, as presented in this chapter and used in the user studies. During early development we played with toggling all selected tags on target objects. Yet, toggling all tags simultaneously results in hard to predict changes. Thus we decided that the explicit choice of either adding or removing tags is a better alternative.

4.6.3 *Adaptation to Smaller Devices*

Depending on the used device, the presented tagging widgets might not be the best possible interface for accessing single tags. The two main factors are display size and if the device is operated using a single or both hands. While pinning objects and performing selections can theoretically be performed with a single hand, it is much more limiting than bi-manual input. Yet, the behaviour of the tagging widgets at the display border can be adjusted for uni-manual input by making them *also* toggle-able.

On larger devices, such as tabletop displays, there is enough space for tag widgets at the border of the screen. This way any desired combination of tags can be activated at all times, independent of the tags currently assigned to the visible objects. Pinning tagged objects that are close to the objects being modified also remains very useful. Large screens benefit from this *local* operation, as the attention does not need to be divided between the target area and the tagging widgets, that are potentially far away.

For smaller multi-touch devices, such as smartphones or small tablets, most often a single hand is used for interaction, while the other hand is holding the device. Wagner et al. (2012) provided an interesting discussion of design implications for controls at the border of the screens of hand-held tablets. Small devices have little screen space to spare making the tagging widgets a suboptimal design choice. In this case, a pop-up menu with selectable tags is a good alternative—it can be operated using a single hand and does not consume screen real estate, unless it is invoked. Such a menu can be activated by a single hold gesture. To resolve the ambiguity with a pin gesture, we cancel the menu if the system registers a second touch outside of the menu.

4.6.4 *Pop-up Menu*

If the user holds a touch on a single object for two seconds a pop-up menu is displayed, as shown in Figure 29. The segments of the circular menu represent the available tags. Filled segments are tags currently assigned to the object. Tapping a segment toggles the corresponding tag, and thus allows the user to directly specify which tags should be assigned to the object. The menu is closed if the user touches anywhere outside the menu.

There is a potential overlap of the hold gesture to open the menu and pinning a single object. To deal with this ambiguity we proceed as follows. If the user intends to pin a single object, but takes too long to start selecting other objects, the menu opens. If the user is still pinning the object and starts selecting other objects with a second touch the menu is automatically dismissed and the user can perform the desired operation as if the menu never had appeared.

While the pop-up menu may occlude nearby objects, this will only happen if the user does not intend to use the menu but accidentally triggers it by spending too much time between pinning a single object and selecting nearby objects. Choosing a reasonable threshold for opening the pop-up menu should minimise this potential problem. In any case it is also possible to first select the desired objects and pin an object later, thereby avoiding accidental triggering of the menu altogether.

4.6.5 *Selecting Rectangular Groups of Objects*

In addition to selecting objects crossed by a path, our technique can also support the selection of objects inside rectangular regions, as shown in Figure 30. Which selection method is more useful depends on the spatial layout of the objects themselves, the usage scenario, and the specific application.

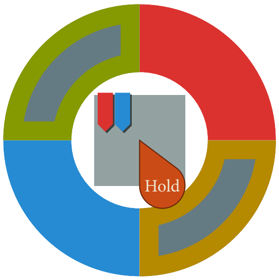


Figure 29: Holding a single object for two seconds displays a circular pop-up menu. Tapping a segment toggles the corresponding tag on the object. A touch outside the menu dismisses it.

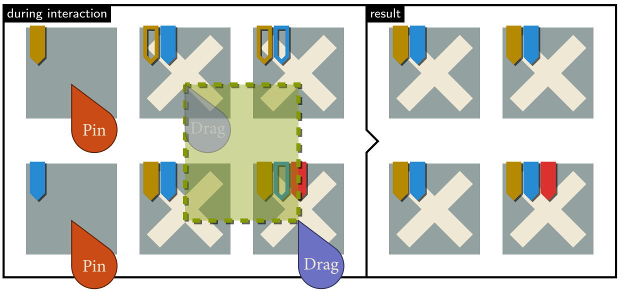


Figure 30: Dragging a rectangle selects all objects that are contained or intersecting it. As with the path technique the initial touch position of the dragging operation defines if the pinned tags are added (as shown here) or removed. Outlined tags preview impending changes.

To select a rectangular group of objects a rectangular area can be defined by dragging the diagonal of the rectangle. All objects intersecting or completely inside the rectangle are then selected. The mechanism for choosing the mode of operation, i. e. if tags are to be added or removed from the selected objects, is the same as with path selection. The position of the first touch, in this case defining one end of the diagonal, determines if the tags are to be added or removed from the selected objects. While the rectangle is dragged a preview of the resulting tag changes is shown, as illustrated in Figure 30.

Depending on the spatial layout and application context, rectangular or path selection is more useful. Occasionally both variants are necessary, but supporting both simultaneously is difficult. Then, a way is needed to switch between the two selection methods. In our system the rectangle and path selection methods are mostly compatible, as they use the same logic for choosing the operation mode (addition/removal). Thus, a combination similar to the overloaded version of lasso and rectangle selection by Saund et al. (2003) could be used here.

4.6.6 *Simplification to a Single Group*

Many existing applications support only selection of a single group. These can still benefit from the selection *editing* capabilities of our interface. In this scenario the interaction technique of pinning a selected object is still very useful for adding *additional* objects to an existing selection with a rectangle or path gesture. Yet, highlighting is likely more appropriate than a tagging interface. This idea effectively enhances existing multi-touch group selection approaches, which typically do not support editing of a selection.

4.6.7 *Selecting Overlapping Objects*

If objects overlap each other it might not be possible to select the desired objects using a single selection operation. One of the benefits of our technique is its capability to easily extend and modify selections, and we can compose the desired set of objects through multiple selection steps. *Harpoon Selection* (Leitner and Haller 2011) already demonstrated that *partially* overlapping objects can often still be selected using path-based techniques. For objects that completely overlap, a technique such as *Tumble! Splat!* (Ramos et al. 2006) can be combined with our technique to access occluded objects. Intelligent selection assistance systems such as *Suggero* (Lindlbauer et al. 2013) might also be helpful for selection of Gestalt configurations. If the background is completely covered by an object, for instance while the view is zoomed in, it is impossible to signal a subtraction by starting a path gesture on the background. In this case the editing gesture could be started on the bezel, which can equally signal subtraction.

Part II

ABOVE THE SURFACE

So far we have explored interaction on touch-capable surfaces. In the second part we extend the tabletop display into the third dimension using a stereoscopic projection. Multi-touch input is constrained to the surface, as touches cannot be detected in mid-air. To access the third dimension we developed *Triangle Cursor*, an indirect multi-touch metaphor that enlarges the interaction space to include the volume above the surface.

5

TRIANGLE CURSOR: INTERACTIONS WITH OBJECTS ABOVE THE TABLETOP

Multi-touch enabled tabletop surfaces have shown significant potential for exploring complex content in an easy and natural manner, supporting expressive interaction without any instrumentation. In particular dense environments, such as physical simulations or geographic information systems (GISs), in which multiple small objects are displayed close to each other could benefit from horizontal interactive display set-ups, which closely resemble the way humans are interacting with their real-world surrounding. Furthermore, collaborative interaction that is naturally supported on a multi-touch tabletop is a common requirement in these domains. Due to the complexity of the environment such visualisations often benefit from additional perceptional cues, most noticeably from binocular disparity and motion parallax, as provided by tracked stereoscopic projections.

5.1 INTRODUCTION

Although multi-touch surfaces could exhibit limitations in the context of stereoscopically rendered projections, because the input is inherently constrained to the 2D surface, these two technologies have recently been combined in different set-ups (Schöning et al. 2009; Valkov et al. 2011, 2010), and even first commercial products are already available. Furthermore, interdisciplinary research projects like iMUTS[1] or InSTInCT[2] have addressed the question of touch interaction with stereoscopic content on a two-dimensional surface. However, until now such systems are mainly used for navigation purposes whereas interaction with stereoscopically displayed objects is supported only rather rudimentarily.

With stereoscopic displays, objects can be displayed with different parallaxes, i. e. negative, zero, or positive, resulting in different perceived object positions. Objects rendered with *zero* or *positive parallax* appear attached to the projection screen or beyond it and are well suited for touch interaction, either by directly touching the virtual objects or using indirect selection and manipulation techniques (Hancock et al. 2007; Pierce et al. 1997; Reisman et al. 2009). However, such an approach cannot easily be extended to access objects exhibiting *negative parallax*. In this case an object appears above the surface and the users have to penetrate it in order to interact with it, since most touch sensitive screens capture only direct contacts or hover gestures close to the screen. This is usually considered unnatural by users, and also leads to some specific visual artefacts, such as accommodation-convergence problems, unordered occlusions, misaligned touch targets, etc. (Schöning et al. 2009; Valkov et al. 2011). Hence, selection and manipulation of objects above the interactive surface pose a major challenge in this context. Camera-based hand and gesture detection algorithms could be applied to allow free space interaction above the surface (Hilliges et al. 2009), but such techniques are usually considered more imprecise and exhausting compared to the constrained touch interaction (Benko and Feiner 2007; Hinckley et al. 1994).

In this chapter we present *Triangle Cursor*, a novel indirect selection and manipulation metaphor for objects rendered stereoscopically above an interactive tabletop's surface. Figure 31 shows Triangle Cursor in action. The metaphor is designed to overcome

1 http://imuts.uni-muenster.de
2 http://anr-instinct.cap-sciences.net

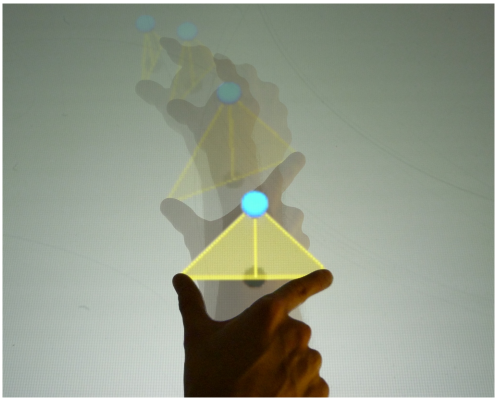


Figure 31: Illustration of Triangle Cursor in action. The user controls the 3D cursor position and height above the surface using a single hand.

the occlusion artefacts and fat finger problem (Holz and Baudisch 2010; Potter et al. 1988) usually accompanying similar shadow interaction techniques (Benko and Feiner 2007; Coffey and Keefe 2010). In its basic implementation 4 DOFs are supported, i. e.3D position and yaw rotation. Since the metaphor is usually controlled with only the dominant hand it could easily be extended to support further actions. For instance, pitch and roll rotations could be mapped to a trackball metaphor controlled with the non-dominant hand.

We will investigate the applicability of the metaphor for some common tasks in an initial usability study and measure its quantitative performance against the currently most related technique, *Balloon Selection* (Benko and Feiner 2007), in a formal experiment. Our results confirm our initial expectations, that users consider the metaphor to be intuitive and adequate for the tested tasks, and they perform the tasks up to 20% faster without a loss in precision.

5.1.1 *Related Work*

An extensive review of the existing work on interactive tabletops has been presented by Grossman and Wigdor (2007) who also developed a taxonomy for classification of this area of research. Their framework takes into account the perceived and the actual display space, the input space and the physical properties of an interactive surface. As shown in their paper, 3D volumetric visualisations are rarely being considered in combination with 2D direct surface input. Currently, only few approaches have addressed this problem which introduces new challenges. For instance, Schöning et al. (2009) have investigated stereo visualisation on a multi-touch enabled wall and discussed approaches based on mobile devices for addressing the formulated parallax problems. Furthermore, Valkov et al. (2011, 2010) investigated humans' sensibility to stereoscopic depth in touch environments and how the parallax changes the touch behaviour and precision.

Extending the interaction space above the touch surface has been addressed mainly by two different approaches: free-hand interaction above the surface (Hilliges et al. 2009), or indirect interaction with the objects' shadows, possibly using physical props (Ishii and Ullmer 1997) or widgets (Benko and Feiner 2007; Coffey and Keefe 2010; Cohé et al. 2011). Following the first approach, Hilliges et al. (2009) have tested two depth sensing techniques to enrich the multi-touch interaction on a tabletop set-up with monoscopic projection. Wilson and Benko (2010) have used multiple cameras and projectors to enable interaction on, above and between multiple surfaces. Indirect interaction with the shadows of stereoscopically rendered objects on a multi-touch tabletop set-up has been addressed by Coffey and Keefe (2010). They have combined a monoscopic tabletop projection with a large scale horizontal stereo display to explore complex medical data sets with an extended world-in-miniature metaphor.

Benko and Feiner (2007) have proposed the *Balloon Selection* metaphor, which supports precise object selection and manipulation for augmented reality set-ups. The metaphor imitates a small helium balloon, which the user is holding on a string above the surface. By pulling the string with a second finger, the height of the balloon is controlled. Although such indirect techniques usually provide better precision and lower levels of fatigue compared to free-hand interaction (Hinckley et al. 1994), they could suffer from occlusion artefacts and the fat finger problem.

Because of the size of the human fingers, directly touching a small object is considered inherently inaccurate (Vogel and Baudisch 2007), and there exists a series of studies investigating the factors responsible for this inaccuracy, including the fat finger problem (Vogel and Baudisch 2007) and the perceived input point model (Holz and Baudisch 2010). Many approaches have addressed this issue, for example by providing adjustable (Benko et al. 2006) or fixed cursor offset (Potter et al. 1988), by scaling the cursor motion (Benko et al. 2006), or by extracting the orientation of the user's finger (Holz and Baudisch 2010).

In this chapter we propose a shadow interaction technique which was initially inspired by the Balloon Selection, and which we designed to support more fluid motions and to overcome some of the problems by using an adjustable cursor offset.

5.2 TRIANGLE CURSOR TECHNIQUE

We came up with the idea for our Triangle Cursor technique when we examined how well existing selection techniques for multi-touch surfaces work combined with displays using a stereoscopic projection.

Triangle Cursor uses an indirect approach to specify a 3D position above the tabletop display. Instead of specifying the 3D position directly the user only interacts with the zero parallax plane, i. e. the table surface. This essentially splits the 3 DOFs positioning task into a 2 DOFs positioning task on the table surface and a 1 DOF task to select the desired height above the table. Even though Triangle Cursor uses this indirect approach, it allows the user to combine the position specification with the height specification into a single 3 DOFs task.

A spherical cursor is used to represent the currently selected 3D position. A triangle is displayed between two touch points and perpendicular to the table surface with the 3D cursor attached to the top vertex. This set-up is shown in Figure 31.

5.2.1 *Interaction Technique*

When the user touches the surface at two points an isosceles triangle is displayed with the two base vertices at the touch positions. The triangle's altitude is displayed to provide additional visual feedback to the user. The altitude's base point represents the 2D position on the surface and is located at the midpoint between the user's fingers. The altitude's length is equal to the height above the surface. The use of the midpoint between the two fingers has two benefits for accurately specifying the position on the surface: first, the point of interest is not occluded by the user's fingers; second, the movement speed of the midpoint can be reduced by a factor of up to 2 by only moving a single finger (Benko et al. 2006).

The triangle's position can be controlled by moving the fingers on the surface (2 DOFs). The height above the surface (1 DOF) is controlled by the distance between the two fingers and independent of their absolute positions. When the fingers are moved the triangle is scaled according to the distance between the fingers. This behaviour resembles the usual scaling gesture that is used in many multi-touch applications, and the user can effectively scale the triangle and accordingly the height above the surface. It is possible to use Triangle Cursor with two hands or with two fingers of the same hand, in most cases the index finger and the thumb of the dominant hand. When using a single hand the height of the interaction space is limited by the amount the user's fingers can be spread apart. As shown by Hancock et al. (2007), similar to real tables rich interactions with digital tables can be implemented by limiting the interaction to a shallow area above the table. The use of a stereoscopic projection already limits the maximum height at which objects above the table can be visualised, depending on the user's point of view. Initial tests have shown that mapping the distance between the fingers to the altitude of the triangle using a quadratic function allows users to cover the interaction space required by most applications. Close to the table surface the users have fine control over the height changes, while they are still able to access the upper boundary of the interaction space.

To accommodate differences in hand size or applications that require a fine level of control in a deeper interaction space the metaphor could be extended to allow a change of the base-to-altitude mapping or adding a height offset while using it.

Moscovich and Hughes (2008) have shown that positioning and rotating the hand and adjusting the span of the fingers are compatible and can be combined into a unimanual manipulation task. A yaw rotation around an axis perpendicular to the table surface can be applied using the relative position of the touch points and midpoint. A rotation of the fingers around the midpoint is mapped to a rotation around the axis defined by the triangle's altitude.

To select an object the spherical cursor has to intersect the object. When the user triggers the selection the object is attached to the spherical cursor and is moved and rotated with it until it is deselected. To trigger the selection a technique like *SimPress* clicking (Benko et al. 2006) or a simple tap with a finger of either the dominant or non-dominant hand can be used.

5.2.2 *Extension to Six Degrees of Freedom*

With the possibility to control 3 DOFs position and yaw orientation with two fingers of a single hand, we explored an extension of the technique to simultaneously control the other 2 DOFs of the orientation.

In the case of placing a new object into the virtual scene it is required to define a *forward* facing direction, i. e. the initial rotation. We chose the direction perpendicular to the segment connecting the two touch points and facing away from the user's hand. Our initial approach required the fingers to be placed in a specific order to determine the *forward* direction. Early testing has shown that this is not a satisfying solution. Remembering the touching order was awkward and dependent on whether the left or right hand was used. A touch surface that supports a registration of the user's fingers (e. g. Dang et al. 2009) could be used to avoid the touch order issues, but would also force the user to use predefined fingers to interact. For a better solution a hand detection algorithm (e. g. Dohse et al. 2008) could be added to always choose a predefined direction relative to the position of the user's hand.

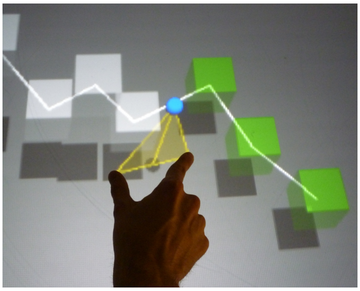
To control the pitch and roll rotation a trackball metaphor could be used. When the user touches the surface with the free hand a trackball is displayed at the touch point. The movement of the touch point is mapped to the rotation of the trackball and accordingly to the orientation of the selected object. The combination of Triangle Cursor with orientation and a trackball results in a bimanual 6 DOFs interaction technique.

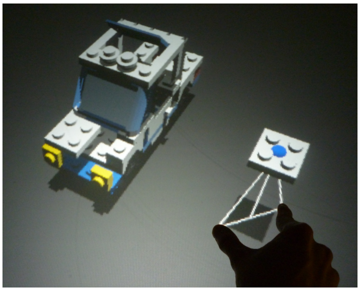
5.3 FIRST USER STUDY

We have tested the usability of Triangle Cursor in an initial user study with 18 (15 male and 3 female) members of our working group and students.

We chose Balloon Selection (Benko and Feiner 2007) as a baseline comparison technique. Balloon Selection offers a similar selection mechanism that decomposes the 3 DOFs positioning task into a 2 DOFs positioning and a 1 DOF height adjustment task. A spherical cursor (*balloon*) is displayed to show the currently selected 3D position. The user can specify the position on the surface with the first touch. Using a second touch point the user can change the cursor's height by varying the distance between the two touch points. As the original version of Balloon Selection does not offer support for an additional specification of the orientation, we expanded the technique. If the user rotates the second touch point around the primary touch point the rotation is applied to the currently selected object. This extension allows us to evaluate the two interaction metaphors against each other. In both metaphors selection and deselection were triggered by tapping with a finger of the non-dominant hand.

The subjects had to alternately use Triangle Cursor and Balloon Selection to complete three common tasks, illustrated in Figure 32. The first task (Figure 32a) was to follow a 3D path represented by a set of waypoints. At each waypoint a small box illustrated the tolerance area. If the cursor was moved completely into a box, the colour of the box was changed to provide feedback to the user. In the second task (Figure 32b) the user had to complete a small model using LEGO bricks. The users were allowed to spend 20 minutes with each metaphor to complete the task and were supported by a supervisor during this time. In the last task (Figure 32c) the users had to move a small plane to a predefined position and adjust its orientation using the 6 DOFs extension of the Triangle Cursor technique. This task was only performed with Triangle Cursor. After the three tasks were completed, the users were asked about their subjective impression of Triangle Cursor (Figure 33) and their preference of the two techniques (Figure 34). In an informal interview users were asked about their experience with the two techniques.


(a) Following a predefined path.


(b) Completing a small model with LEGO bricks.

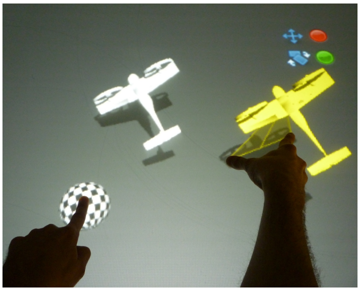

(c) Setting the position and orientation of an aircraft.

Figure 32: The three tasks of the initial user study using: (a, b) Triangle Cursor, (c) 6 DOFs extension of Triangle Cursor for controlling position and orientation. *(A monoscopic projection was used while these images were captured.)*

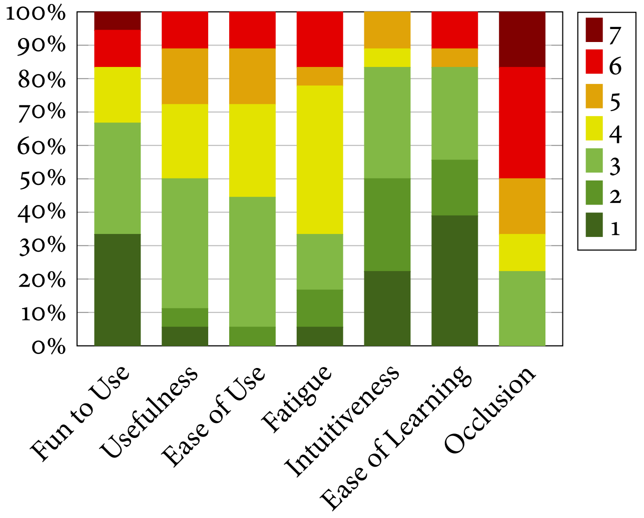


Figure 33: Users' rating of different aspects of Triangle Cursor on a 7-point Likert scale (1 = very good).

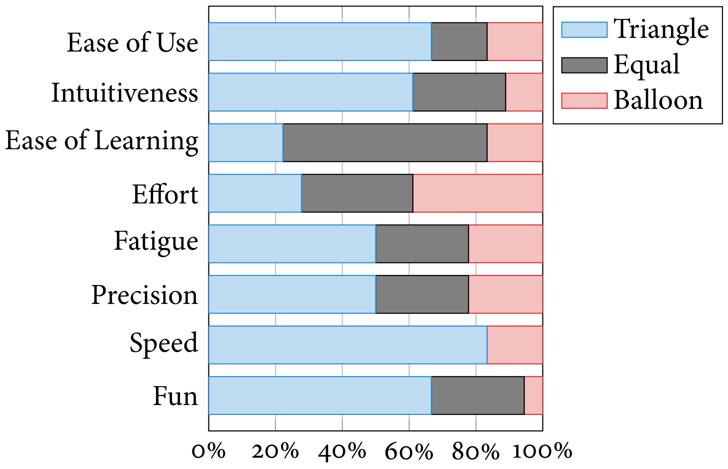


Figure 34: After completing the experiment tasks users were asked to state their preferred technique.

Most of the users had the subjective feeling of being faster with Triangle Cursor than with Balloon Selection and have rated the positioning precision higher or as equal. All users have appreciated the fact that Triangle Cursor provides a smooth single motion to control all degrees of freedom with a single hand, and they have rated it as more appropriate for the path following task. Nevertheless, two of the users have described the Balloon Selection as having a simpler mental model, clearly separating surface positioning from changing the height, and one of them commented that there may be a difference if the path would have been specified as a smooth curve. Our extension of Balloon Selection to support yaw rotation leads to difficulties in separating the yaw rotation from height changes. None of the subjects rated this as a problem, and most of them intuitively overcame this by first adjusting the object's orientation and then its height.

The users were asked to comment on whether they were fatigued by the tasks. Most of the users reported that they found the touch interaction somewhat tiring and that they did not experience a large difference between the techniques regarding fatigue. Most users were unfamiliar with an optical touch detection system that behaves differently than a capacitive touch screen; touches can be lost if the surface is only barely touched, and fingers that are hovering close to the surface can be detected as false touches. We observed that the users were applying more pressure on the touch surface to counter such detection problems, contributing to the experienced fatigue.

Because of the promising results of this initial study, a formal experiment was conducted. In this experiment (described in detail in the following sections) objective, quantitative values were measured and evaluated.

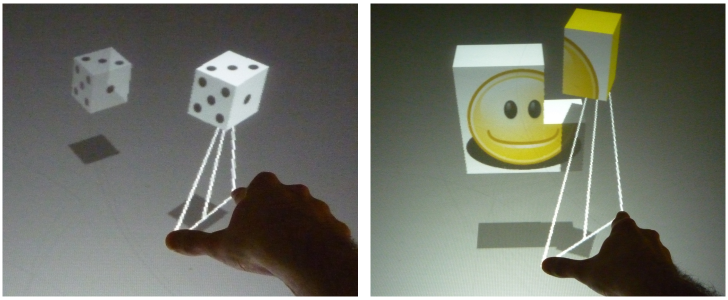
5.4 SECOND USER STUDY

We performed a second, formal user study to quantitatively evaluate the performance of Triangle Cursor and how it compares with similar selection techniques. Again, Triangle Cursor (TRIANGLE) was tested against its closest counterpart—an extended version of Balloon Selection (BALLOON) that supports yaw rotation, as described in the previous section.

17 participants (14 male, 3 female), ages 23–35 (median 26), participated in the experiment. All but two participants were undergraduate, graduate or PhD students in computer science. All participants were frequent computer users and all reported to have experience with stereoscopic projections and multi-touch devices. The right hand was used as the dominant hand by all users, and all users reported themselves as being right-handed. All participants had normal or corrected to normal vision, none reported amblyopia or another stereopsis disruption. Six participants wore glasses or contact lenses. Completing the experiment, including training, instructions and debriefing, took the participants 45–60 minutes. The participants were allowed to take breaks at any time. The results of one participant were discarded from the statistical analysis as the participant had obviously misunderstood the experiment task.

5.4.1 *Experimental Set-up*

The experiment was performed using our prototype stereoscopic multi-touch tabletop set-up. A PC running Windows 7 with an Nvidia Quadro FX 4800 graphics card was used for the experiment. For the back projection on the surface an Optoma EX785 DLP projector with resolution 1024×768, providing a frame-sequential stereoscopic projection at 120 Hz, was used. The physical dimensions of the touch surface were $136\text{ cm} \times 102\text{ cm}$, resulting in a pixel size of approximately 1.3 mm.



(a) Task one: moving a die. (b) Task two: puzzle.

Figure 35: The two tasks of the user study, performed using Triangle Cursor. *(A monoscopic projection was used while these images were captured.)*

To enable touch detection on our table set-up we used the rear diffuse illumination principle (Matsushita and Rekimoto 1997). Infrared lights were placed under the table surface to provide a uniform illumination of the projection surface. A Point Grey Research Dragonfly 2 camera with an appropriate filter was used to capture the infrared light reflected by the user's fingers and detect the touch points. The touch surface was captured at 30 frames per second at a resolution of 1024×768, resulting in a touch detection resolution of approximately 1.3 mm in each dimension.

For this experiment we did not use head tracking and asked users to stand at a marked position. The virtual camera's position was adjusted to match the user's height to provide correctly rendered images for the user's point of view.

5.4.2 *Task One: Moving a Die*

Participants were asked to use both interaction metaphors to select a die displayed above the table and *drag* it to a target position, indicated by a semitransparent die (Figure 35a). In addition to the position the participants were asked to match the target die's orientation. Again, selection and deselection was triggered by a tap on the display with the non-dominant hand.

We encouraged users to control position, orientation and height above the surface at the same time by intentionally placing the object die and the target die at different heights and in different orientations, although it was not necessary to do so in order to successfully complete this task.

5.4.2.1 *Procedure*

A within-subjects, repeated measures design was used consisting of 2 techniques and 9 positions in 3 different heights. A random orientation (yaw angle; $0°$–$360°$ in $30°$ increments) was assigned to each combination of height and position. All trials were repeated twice. The positions were aligned on a 3×3 grid near the centre of the screen and could easily be reached by all participants. For the target a different position and height were chosen. When the participants were satisfied with the alignment of the die at the target they had to complete the trial by pressing a button on the border of the display with their dominant hand. After completing a trial the next trial was automatically started. The use of the button ensured the same starting position of the user's dominant hand in all trials. The order in which the techniques were tested was randomized to avoid ordering effects. To eliminate the effects of switching selection techniques the first two trials were marked as practice trials and discarded from the results for each trial set, resulting in a total number of 108 trials per subject. To see how the two techniques differ in a translation task, where no rotation is necessary, twelve of these trials (six for each technique) had identical orientations for the object and target dice. The completion time for each trial, i. e. the time between the button presses at the beginning and the end, was recorded. Additionally, the distance of the die to the target position and the difference in the actual and target orientation were recorded.

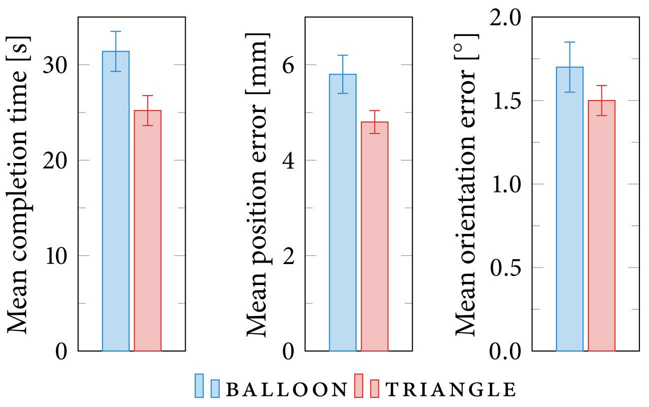


Figure 36: Comparison of BALLOON and TRIANGLE for task 1 (moving a die). (Error bars = ± SE)

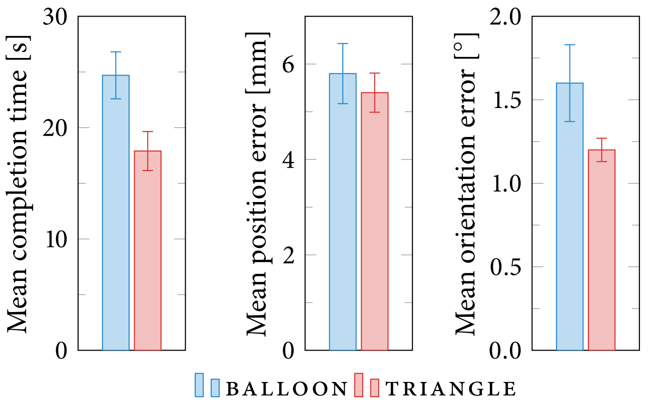


Figure 37: Comparison of BALLOON and TRIANGLE for trials of task 1 (moving a die) that required only a translation and no change in die orientation. (Error bars = ± SE)

5.4.2.2 *Results*

The mean completion time, position error and orientation error were calculated for each subject and selection technique and are shown in Figure 36. With a two-sided t-test over subjects' means we found that TRIANGLE ($M = 25.2$ s, $SE = 1.6$) was significantly faster ($t_{(30)} = 2.353$, $p = 0.025$) compared to BALLOON ($M = 31.4$ s, $SE = 2.1$). TRIANGLE also performed better with respect to the position error with borderline significance ($t_{(30)} = 2.124$, $p = 0.042$). The mean position error was approximately 1 mm smaller with TRIANGLE ($M = 4.8$ mm, $SE = 0.2$) than with BALLOON ($M = 5.8$ mm, $SE = 0.4$). With an average difference of just $0.2°$ the orientation error was nearly identical with TRIANGLE ($M = 1.5°$, $SE = 0.1$) and BALLOON ($M = 1.7°$, $SE = 0.1$). No significant difference could be found for the orientation error ($p = 0.258$).

The results for the subset of trials with equal orientations of the object and target dice, i. e. the trials that required only a translation and no change in orientation, are shown in Figure 37. In these trials TRIANGLE ($M = 17.9$ s, $SE = 1.8$) was significantly faster ($t_{(30)} = 2.471$, $p = 0.019$) than BALLOON ($M = 24.7$ s, $SE = 2.1$). The difference of only 0.4 mm between the mean position errors for TRIANGLE ($M = 5.4$ mm, $SE = 0.4$) and BALLOON ($M = 5.8$ mm, $SE = 0.6$) was not found to be significant ($p = 0.564$). For the orientation error no significant difference ($p = 0.097$) between TRIANGLE ($M = 1.2°$, $SE = 0.1$) and BALLOON ($M = 1.6°$, $SE = 0.2$) could be found.

5.4.3 *Task Two: Puzzle*

Six boxes with a part of an image attached to their front side were displayed and the participants were asked to assemble the complete image by aligning the boxes (Figure 35b). One box was placed in the centre as reference point. The other five boxes were distributed on a half circle around the reference box with random orientations (yaw angle, $0°$–$360°$ in $30°$ increments). The participants had to push a button on the border of the display to start a trial. Once they were satisfied with the arrangement of the boxes the trial was ended by pressing the button again.

Each box had to be selected, moved and rotated to the target position to complete the picture. There was no predefined order in which the boxes had to be moved, and the users could align the boxes freely. Selection and deselection was triggered by a tap on the display with the non-dominant hand.

5.4.3.1 *Procedure*

We used a within-subjects repeated measures design. The participants were asked to perform 2 trials with each technique, resulting in a total of 4 trials per user. To eliminate the effects of switching techniques a practice trial was added for each technique that was excluded from the results. The order in which the techniques were tested was randomized to avoid introducing a bias for one technique. The task completion time, the distance to the perfect target position and the difference between the actual and target orientation for each box were recorded.

5.4.3.2 *Results*

The mean completion time, accumulated position error and orientation error of all 5 movable boxes were calculated for each subject and selection technique and are shown in Figure 38. The task completion time with TRIANGLE (M = 126 s, SE = 10) was on average 31 s shorter than with BALLOON (M = 157 s, SE = 10). A two-sided t-test over the subjects' means has shown this difference to be significant ($t_{(30)} = 2.181$, $p = 0.037$). While the position error was on average 5 mm smaller with TRIANGLE (M = 26.1 mm, SE = 1.5) than with BALLOON (M = 31.1 mm, SE = 2.4), no significant difference could be found ($p = 0.094$). Also no significant difference could be found for the orientation error with an av-

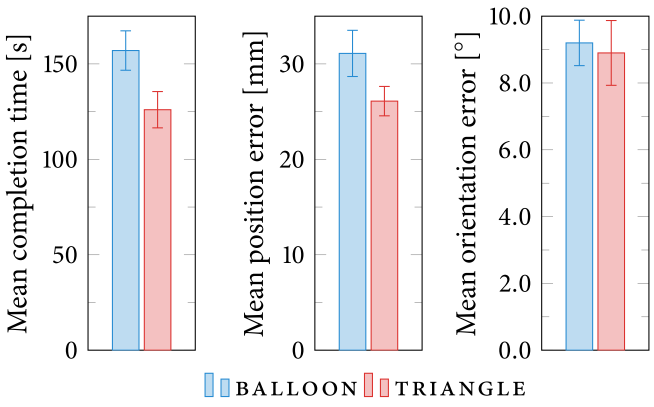


Figure 38: Comparison of BALLOON and TRIANGLE for task 2 (puzzle). (Error bars = ± SE)

erage difference of just 0.3°; the mean value of TRIANGLE was M = 8.9° with SE = 1.0 and for BALLOON it was M = 9.2° with SE = 0.7.

5.5 DISCUSSION

Our formal study has shown that users were able to perform the synthetic die moving task about 20% faster with our Triangle Cursor than with the Balloon Selection technique. The increase in speed did not come at the expense of precision, as the positioning and orientation errors were either nearly identical for both techniques or slightly in favour of Triangle Cursor. In the more complex puzzle task, that more closely resembles what a real-world task could be like, Triangle Cursor is again about 20% faster while maintaining a similar precision.

As Triangle Cursor is more complex and multiple degrees of freedom are controlled at the same time using a single hand one could expect it to be less precise than Balloon Selection. The results show that this is not the case. On average, Triangle Cursor even outperformed Balloon Selection in terms of precision by a small margin. We believe that this is the result of using the midpoint of two touches that allow a more stable control of the cursor position on the surface—one of Triangle Cursor's initial design ideas.

Nevertheless, it might be possible that our extension of Balloon Selection to support orientation changes has changed its mental model, that was focused on separating the 2D positioning task from the 1D height changing task. We added another 1D task for setting the orientation; however, this was not clearly separable from the height changing task. This alteration might have had a negative impact on Balloon Selection's performance. The examination of those trials, for which no change of orientation was necessary, has shown that Triangle Cursor also performed better in these cases. However, the change of orientation was coupled with height and position changes for both techniques, so that accidental orientation changes that had to be corrected later on could have occurred during these trials. To determine how the original Balloon Selection compares to Triangle Cursor as a pure selection technique further investigations are required.

One of the advantages of Triangle Cursor is the ability to directly approximate the desired height by spreading the two fingers the right amount apart before touching the surface. We observed that all participants took advantage of this to some extent. Most users were able to approximately *guess* the right height to perform a selection after they had completed several trials. Thus, often only a small correction of the height was necessary. We believe this is one of the main reasons why Triangle Cursor outperformed Balloon Selection. After the initial guess only a small correction phase is required for Triangle Cursor, whereas a larger height change for the cursor of Balloon Selection was required.

Furthermore, we were able to observe that most users utilized the time while they were moving an object from the starting position to the target with Triangle Cursor. The users were adjusting their fingers to approximately match the target height and orientation while moving their hand to the target position, so that only a small adjustment at the target position was necessary. In contrast, with Balloon Selection most users had more difficulties to adjust the height and orientation while moving, so that either a larger correction at the target position was necessary or the users performed the task with discrete phases for moving and adjusting the height and orientation, leading to higher times to complete the tasks.

Although Triangle Cursor is an indirect interaction technique, the users did not perceive it as such. The combination of orientation, position on and height above the surface in a single-handed technique resulted in fluid motions. This was particularly observable during the more complex puzzle task. The users quickly moved the pieces around in a natural looking way. Several participants noted that it was awkward to use Triangle Cursor for rotations with large angles. Indeed, the users had to stop during the rotation, reposition their hand and select the object again to resume the rotation. We observed that users who seemed more comfortable with touch interaction quickly learned to plan ahead and oriented their hand to *grab* the object so that no or only one reselection was necessary. Other users always grabbed the objects in the same way and were forced to reselect the object more often. While some users reported the reselection step as slightly disturbing, their results show that they still performed faster than with Balloon Selection, that usually does not need the additional reselection step.

5.6 DESIGN CONSIDERATIONS

In this section we discuss several design considerations for using Triangle Cursor within more complex interfaces and briefly consider possible variations suitable for different application requirements.

5.6.1 *Triangle Cursor as a Widget*

In an application, where the user manipulates a single or just a few objects over a longer period of time or a continuing sequence of manipulations is necessary, it might be beneficial to modify Triangle Cursor and use it as a manipulation widget. When the user removes his fingers from the surface while an object is selected, Triangle Cursor stays visible. Thus it acts as a handle, and the user can instantly *grab* it to continue manipulating the object, without needing to reselect the object first. A separate gesture, like tapping on the Triangle Cursor widget, could be used to deselect the object and dismiss the widget, when it is no longer used. However, in a dense visualisation environment, where a large number of selectable objects or two objects that are very close to each other are manipulated, the widgets might occlude other widgets or could be hit accidentally by the user while trying to perform a new selection.

5.6.2 *Supporting Multiple Users*

We believe that Triangle Cursor can easily be extended to support multiple users. Using an existing technique that is able to identify to which user each touch point belongs, multiple Triangle Cursors can be active at the same time—one for each pair of touches of the same user that are in close proximity. The fact that Triangle Cursor is typically operated using a single hand can aid in the assignment of touches to the users. For example, the touch detection for a tabletop based on the rear diffuse illumination principle can be extended to also detect the users' hands above the surface (Dohse et al. 2008). While this information does not provide a complete user identification, it is sufficient to decide which touches belong together, so that multiple users can use multiple Triangle Cursors at the same time.

The widget variant of Triangle Cursor described above could be especially useful for tasks that require multiple users to collaborate. For instance, a selected object could be moved and then passed on to another user.

While Triangle Cursor could be extended to multiple users, special considerations to extend stereoscopic displays to multiple users have to be made. A possible solution using a combination of shuttering and polarization has been proposed by Fröhlich et al. (2005).

5.6.3 *Conflicts with Other Touch Gestures*

Triangle Cursor might conflict with other touch gestures, for instance the most common camera navigation gestures pan, pinch and rotate. This is especially apparent for the pinch gesture that is commonly used to adjust the camera zoom or scale objects, as Triangle Cursor was designed to resemble the scale gesture for height changes.

In applications where there is a clear separation between objects that can be manipulated and a static environment this could provide a context to decide whether the user wants to select an object or perform a camera navigation action. For instance, if the shadows of selectable objects are displayed below them they provide a visual cue for the user where to initiate a Triangle Cursor gesture. When two touches are registered by the system and the midpoint between the touches lies inside an object's shadow, Triangle Cursor is initiated. When the user touches at an *empty* spot, a camera navigation action can be performed. A good example for this is a GIS application with a predefined landscape and movable building models. When the user touches next to a building Triangle Cursor is used, and when the user touches on the landscape a pinch gesture can be used to scale the landscape, respectively zoom in or out.

5.6.4 *Different Height Mappings*

During the experiments our set-up used a quadratic function to map the distance of the user's fingers on the surface to the height of the cursor above the surface. This was adequate for our experiment tasks, as we had the highest level of precision close to the surface and it was still possible to precisely reach the *highest* points necessary in the experiments. We believe that for most applications it makes sense to have the highest level of precision at the zero parallax plane, i. e. close to the surface, because it makes sense to also place the main objects of interest on the surface. There are, however, applications where there is another *reference height*. Imagine an application in which the user controls a group of aerial vehicles that are displayed above a map. It would make sense to place the map in the zero parallax plane and show the aerial vehicles actually *above* the surface. If all vehicles operate at a common flying altitude the range with the most precise height control should be around that altitude and the height mapping function should be adapted accordingly.

In some applications there might be more than one reference height. In this case a switching mechanism to select different height mappings could be added to the application. Another possibility is to extend Triangle Cursor by another gesture to change the height of an object while it is selected. A third finger could be used to perform a sliding gesture to move the selected object up or down without changing the distance between the two fingers creating the triangle. Changing the distance of the fingers would then result in a manipulation relative to the new height.

Nevertheless, one has to consider that a changing height mapping might cancel out the benefit of being able to *guess* the right finger placement to get close to the desired selection height.

6 CONCLUSION

This thesis presented three novel multi-touch selection and interaction techniques. A characteristic shared by all these techniques is the use of true *multi*-touch input. We showed that multiple touches allow for more expressive interaction than interfaces based on a single point of interaction. We presented user studies for the interaction techniques to verify our design decisions and evaluate their performance. In addition to quantitative results, we also reported users' subjective ratings and observations during the experiments. For each interaction technique we included several design considerations that discuss different variations of our techniques. These discussions are intended as a sort of guideline for integrating our techniques into existing user interfaces and *real* applications.

Chapter 2 presented a survey of selection techniques. We compiled a list of related work on selection techniques and group selection and developed a system of classifiers to structure and categorise the body of previous work. Using a graphical representation we created an overview of existing selection techniques and identified areas that had not been examined. While exploring the design space that lies hidden in these *gaps* we developed the interaction techniques presented in the first part of this thesis.

In Chapter 3 we have shown that adapting rectangular selections to the new multi-touch environment instead of just transferring the traditional mouse-based metaphor results in a better suited interaction technique. A user study has shown that using our multi-touch selection technique, and thus the capabilities of multi-touch input, results in a good compromise of speed and accuracy compared to existing approaches. The increased expressiveness of multi-touch interaction allows our presented technique to be naturally extended to support refinements of selections through the use of context information. We have presented the adaptation of the concept of context information, introduced for selecting rectangular regions, to free form selections. Similarly to the rectangular selections we have introduced interaction metaphors to modify and refine existing selections.

Chapter 4 introduced pinning touches and our new multi-touch tagging interface. Our interface relies heavily on pin gestures, i.e., holding touches, to provide a context for other actions performed at the same time. It employs tag widgets to define multiple overlapping tag groups. One of its key features is the ability to easily edit existing tags by adding and removing items, even just for a single group. We validated our design in two user studies and demonstrated that our tagging interface performs better than alternative interfaces that rely on tapping of individual objects.

In the second part of this thesis we combined multi-touch input with a stereoscopic projection to extend the interaction to the third dimension. Triangle Cursor, a new 3D interaction technique that allows users to operate in the space above a tabletop display, was introduced in Chapter 5. We performed an initial user study to see whether the users liked our technique and were able to complete different tasks with it. The positive feedback from the users encouraged us to evaluate our technique's performance in a second, formal user study. The results of our studies show that Triangle Cursor is significantly faster than the similar Balloon Selection technique and that the increased speed does not result in a loss of precision. As Triangle Cursor can be operated with a single hand we presented a possible extension to a full 6 DOFs interaction metaphor using the non-dominant hand.

As mobile devices become more powerful users tend to perform increasingly complex computing tasks on them. There is a strong demand on user interfaces to keep up with these technological advances. Providing users with a level of control they are familiar with from desktop computers will be an ongoing challenge for interface and interaction designers. Through the availability of notebook computers that include touch-capable displays the division between small mobile devices and fully-featured computers is blurred even further. It will be interesting to see what user interfaces will emerge from the fusion of different input technologies and if there will be a single dominant interaction scheme, like the mouse was for over three decades.

BIBLIOGRAPHY

Baudisch, Patrick (1998). "Don't Click, Paint! Using Toggle Maps to Manipulate Sets of Toggle Switches." In: *Proceedings of UIST '98*. ACM, pp. 65–66.

Benko, Hrvoje and Steven Feiner (2007). "Balloon Selection: A Multi-Finger Technique for Accurate Low-Fatigue 3D Selection." In: *Proceedings of 3DUI '07*. IEEE, pp. 79–86.

Benko, Hrvoje, Andrew D. Wilson, and Patrick Baudisch (2006). "Precise Selection Techniques for Multi-Touch Screens." In: *Proceedings of CHI '06*. ACM, pp. 1263–1272.

Bi, Xiaojun, Tovi Grossman, Justin Matejka, and George Fitzmaurice (2011). "Magic Desk: Bringing Multi-Touch Surfaces into Desktop Work." In: *Proceedings of CHI '11*. ACM, pp. 2511–2520.

Brandl, Peter, Clifton Forlines, Daniel Wigdor, Michael Haller, and Chia Shen (2008). "Combining and Measuring the Benefits of Bimanual Pen and Direct-Touch Interaction on Horizontal Interfaces." In: *Proceedings of AVI '08*. ACM, pp. 154–161.

Casalta, Didier, Yves Guiard, and Michel Beaudouin-Lafon (1999). "Evaluating Two-Handed Input Techniques: Rectangle Editing and Navigation." In: *Extended Abstracts of CHI '99*. ACM, pp. 236–237.

Coffey, Dane M. and Daniel F. Keefe (2010). "Shadow WIM: A Multi-Touch, Dynamic World-In-Miniature Interface for Exploring Biomedical Data." In: *SIGGRAPH '10 Posters*. ACM, 96:1–96:1.

Cohé, Aurélie, Fabrice Dècle, and Martin Hachet (2011). "tBox: A 3D Transformation Widget designed for Touch-screens." In: *Proceedings of CHI '11*. ACM, pp. 3005–3008.

Dang, Chi Tai, Martin Straub, and Elisabeth André (2009). "Hand Distinction for Multi-Touch Tabletop Interaction." In: *Proceedings of ITS '09*. ACM, pp. 101–108.

Dehmeshki, Hoda and Wolfgang Stuerzlinger (2008). "Intelligent Mouse-Based Object Group Selection." In: *Proceedings of SG '08*. Springer, pp. 33–44.

Dehmeshki, Hoda and Wolfgang Stuerzlinger (2009a). "GPSel: A Gestural Perceptual-Based Path Selection Technique." In: *Proceedings of SG '09*. Springer, pp. 243–252.

Dehmeshki, Hoda and Wolfgang Stuerzlinger (2009b). "ICE-Lasso: An Enhanced Form Of Lasso Selection." In: *Proceedings of TIC-STH '09*. IEEE, pp. 630–635.

Dehmeshki, Hoda and Wolfgang Stuerzlinger (2010). "Design and Evaluation of a Perceptual-Based Object Group Selection Technique." In: *Proceedings of BCS '10*. BCS, pp. 365–373.

Dohse, K. C., Thomas Dohse, Jeremiah D. Still, and Derrick J. Parkhurst (2008). "Enhancing Multi-user Interaction with Multi-touch Tabletop Displays using Hand Tracking." In: *Proceedings of ACHI '08*. IEEE, pp. 297–302.

Esenther, Alan and Kathy Ryall (2006). "Fluid DTMouse: Better Mouse Support for Touch-Based Interactions." In: *Proceedings of AVI '06*. ACM, pp. 112–115.

Feldmann, Dirk and Klaus Hinrichs (2012). "GPU based Single-Pass Ray Casting of Large Heightfields Using Clipmaps." In: *Proceedings of Computer Graphics International (CGI)*.

Feldmann, Dirk, Frank Steinicke, and Klaus H. Hinrichs (2011). "Flexible Clipmaps for Managing Growing Textures." In: *International Conference on Computer Graphics Theory and Applications (GRAPP)*. Ed. by Paul Richard and José Braz. SciTePress, pp. 173–180.

Forlines, Clifton, Daniel Wigdor, Chia Shen, and Ravin Balakrishnan (2007). "Direct-Touch vs. Mouse Input for Tabletop Displays." In: *Proceedings of CHI '07*. ACM, pp. 647–656.

Fröhlich, Bernd, Roland Blach, Oliver Stefani, Jan Hochstrate, Jörg Hoffmann, Karsten Klüger, and Matthias Bues (2005). "Implementing Multi-Viewer Stereo Displays." In: *Proceedings of WSCG '05*, pp. 139–146.

Grossman, Tovi and Daniel Wigdor (2007). "Going Deeper: a Taxonomy of 3D on the Tabletop." In: *Proceedings of TABLETOP '07*. IEEE, pp. 137–144.

Gutwin, Carl, Andy Cockburn, Joey Scarr, Sylvain Malacria, and Scott Olson (2014). "Faster Command Selection on Tablets with FastTap." In: *Proceedings of CHI '14*. ACM, pp. 2617–2626.

Han, Jefferson Y. (2005). "Low-Cost Multi-Touch Sensing through Frustrated Total Internal Reflection." In: *Proceedings of UIST '05*. ACM, pp. 115–118.

Hancock, Mark, Sheelagh Carpendale, and Andy Cockburn (2007). "Shallow-Depth 3D Interaction: Design and Evaluation of One-, Two- and Three-Touch Techniques." In: *Proceedings of CHI '07*. ACM, pp. 1147–1156.

Hilliges, Otmar, Shahram Izadi, Andrew D. Wilson, Steve Hodges, Armando Garcia-Mendoza, and Andreas Butz (2009). "Interactions in the Air: Adding Further Depth to Interactive Tabletops." In: *Proceedings of UIST '09*. ACM, pp. 139–148.

Hinckley, Ken, Francois Guimbretiere, Maneesh Agrawala, Georg Apitz, and Nicholas Chen (2006). "Phrasing Techniques for Multi-Stroke Selection Gestures." In: *Proceedings of GI '06*. CIPS, pp. 147–154.

Hinckley, Ken, Randy Pausch, John C. Goble, and Neal F. Kassell (1994). "A Survey of Design Issues in Spatial Input." In: *Proceedings of UIST '94*. ACM, pp. 213–222.

Holz, Christian and Patrick Baudisch (2010). "The Generalized Perceived Input Point Model and How to Double Touch Accuracy by Extracting Fingerprints." In: *Proceedings of CHI '10*. ACM, pp. 581–590.

Ishii, Hiroshi and Brygg Ullmer (1997). “Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms.” In: *Proceedings of CHI ’97*. ACM, pp. 234–241.

Kawasaki, Yoshinori and Takeo Igarashi (2004). “Regional Undo for Spreadsheets.” In: *Adjunct Proceedings of UIST ’04*. ACM, 2 pages.

Kin, Kenrick, Maneesh Agrawala, and Tony DeRose (2009). “Determining The Benefits of Direct-Touch, Bimanual, and Multifinger Input on a Multitouch Workstation.” In: *Proceedings of GI ’09*. CIPS, pp. 119–124.

Kjeldsen, Rick and Jacob Hartman (2001). “Design Issues for Vision-based Computer Interaction Systems.” In: *Proceedings of the PUI ’01*. ACM, pp. 1–8.

Koffka, Kurt (1935). *Principles of Gestalt psychology*. Routledge.

Kurtenbach, Gordon and William Buxton (1991). “Issues in Combining Marking and Direct Manipulation Techniques.” In: *Proceedings of UIST ’91*. ACM, pp. 137–144.

Latulipe, Celine, Ian Bell, Charles L. A. Clarke, and Craig S. Kaplan (2006). “symTone: Two-Handed Manipulation of Tone Reproduction Curves.” In: *Proceedings of GI ’06*. CIPS, pp. 9–16.

Leitner, Jakob and Michael Haller (2011). “Harpoon Selection: Efficient Selections for Ungrouped Content on Large Pen-based Surfaces.” In: *Proceedings of UIST ’11*. ACM, pp. 593–602.

Lindlbauer, David, Michael Haller, Mark Hancock, Stacey D. Scott, and Wolfgang Stuerzlinger (2013). “Perceptual Grouping: Selection Assistance for Digital Sketching.” In: *Proceedings of ITS ’13*. ACM, pp. 51–60.

Luo, Yuexing and Daniel Vogel (2014). “Crossing-Based Selection with Direct Touch Input.” In: *Proceedings of CHI ’14*. ACM, pp. 2627–2636.

Matsushita, Nobuyuki and Jun Rekimoto (1997). “HoloWall: Designing a Finger, Hand, Body, and Object Sensitive Wall.” In: *Proceedings of UIST ’97*. ACM, pp. 209–210.

Mizobuchi, Sachi and Michiaki Yasumura (2004). “Tapping vs. Circling Selections on Pen-based Devices: Evidence for Different Performance-Shaping Factors.” In: *Proceedings of CHI ’04*. ACM, pp. 607–614.

Moran, Thomas P., Patrick Chiu, and William van Melle (1997). “Pen-Based Interaction Techniques for Organizing Material on an Electronic Whiteboard.” In: *Proceedings of UIST ’97*. ACM, pp. 45–54.

Moscovich, Tomer and John F. Hughes (2006). “Multi-finger Cursor Techniques.” In: *Proceedings of GI ’06*. CIPS, pp. 1–7.

Moscovich, Tomer and John F. Hughes (2008). “Indirect Mappings of Multi-touch Input Using One and Two Hands.” In: *Proceedings of CHI ’08*. ACM, pp. 1275–1284.

North, Chris, Tim Dwyer, Bongshin Lee, Danyel Fisher, Petra Isenberg, George Robertson, and Kori Inkpen (2009). “Understanding Multi-touch Manipulation for Surface Computing.” In: *Proceedings of INTERACT ’09*. Ed. by Tom Gross, Jan Gulliksen, Paula Kotzé, Lars Oestreicher, Philippe Palanque, Raquel Oliveira Prates, and Marco Winckler. Vol. 5727. Lecture Notes in Computer Science. Springer, pp. 236–249.

Pedersen, Elin Rønby, Kim McCall, Thomas P. Moran, and Frank G. Halasz (1993). “Tivoli: An Electronic Whiteboard for Informal Workgroup Meetings.” In: *Proceedings of INTERCHI ’93*. ACM, pp. 391–398.

Pierce, Jeffrey S., Andrew Forsberg, Matthew J. Conway, Seung Hong, Robert Zeleznik, and Mark R. Mine (1997). “Image Plane Interaction Techniques In 3D Immersive Environments.” In: *I3D ’97*. ACM, pp. 39–44.

Potter, Richard L., Linda J. Weldon, and Ben Shneiderman (1988). “Improving the Accuracy of Touchscreens: An Experimental

Evaluation of Three Strategies." In: *Proceedings of CHI '88*. ACM, pp. 27–32.

Ramos, Gonzalo, George Robertson, Mary Czerwinski, Desney Tan, Patrick Baudisch, Ken Hinckley, and Maneesh Agrawala (2006). "Tumble! Splat! Helping Users Access and Manipulate Occluded Content in 2D Drawings." In: *Proceedings of AVI '06*. ACM, pp. 428–435.

Reisman, Jason L., Philip L. Davidson, and Jefferson Y. Han (2009). "A Screen-Space Formulation for 2D and 3D Direct Manipulation." In: *Proceedings of UIST '09*. ACM, pp. 69–78.

Rohde, Sebastian, Niklas Goddemeier, Christian Wietfeld, Frank Steinicke, Klaus Hinrichs, Tobias Ostermann, Johanna Holsten, and Dieter Moormann (2010). "AVIGLE: A System of Systems Concept for an Avionic Digital Service Platform Based on Micro Unmanned Aerial Vehicles." In: *Proceedings of SMC '10*. IEEE, pp. 459–466.

Roters, Jan and Xiaoyi Jiang (2013). "Incremental Dense Reconstruction from Sparse 3D Points with an Integrated Level-of-Detail Concept." In: *Advances in Depth Image Analysis and Applications*. Ed. by Xiaoyi Jiang, Olga Regina Pereira Bellon, Dmitry Goldgof, and Takeshi Oishi. Vol. 7854. Lecture Notes in Computer Science. Springer, pp. 116–125.

Roters, Jan, Frank Steinicke, and Klaus Hinrichs (2011). "Quasi-Real-Time 3D Reconstruction from Low-Altitude Aerial Images." In: *Proceedings of UDMS '11*. Ed. by Sisi Zlatanova, Hugo Ledoux, Elfriede Fendel, and Massimo Rumor. Vol. 40. CRC Press/Balkema, pp. 231–241.

Saund, Eric, David Fleet, Daniel Larner, and James Mahoney (2003). "Perceptually-Supported Image Editing of Text and Graphics." In: *Proceedings of UIST '03*. ACM, pp. 183–192.

Schöning, Johannes, Frank Steinicke, Dimitar Valkov, Antonio Krüger, and Klaus Hinrichs (2009). "Bimanual Interaction with Interscopic Multi-Touch Surfaces." In: *Proceedings of INTERACT '09*. Springer, pp. 40–53.

Schwarz, Julia, Charles Marais, Tommer Leyvand, Scott E. Hudson, and Jennifer Mankoff (2014). “Combining Body Pose, Gaze, and Gesture to Determine Intention to Interact in Vision-Based Interfaces.” In: *Proceedings of CHI ’14*. ACM, pp. 3443–3452.

Seifried, Thomas, Christian Rendl, Michael Haller, and Stacey D. Scott (2012). “Regional Undo/Redo Techniques for Large Interactive Surfaces.” In: *Proceedings of CHI ’12*. ACM, pp. 2855–2864.

Snavely, Noah, Steven M. Seitz, and Richard Szeliski (2006). “Photo Tourism: Exploring Photo Collections in 3D.” In: *Proceedings of SIGGRAPH ’06*. ACM, pp. 835–846.

Steinicke, Frank, Gerd Bruder, Jason Jerald, Harald Frenz, and Markus Lappe (2010). “Estimation of Detection Thresholds for Redirected Walking Techniques.” In: *IEEE Transactions on Visualization and Computer Graphics* 16.1, pp. 17–27.

Strothoff, Sven, Dirk Feldmann, Frank Steinicke, Tom Vierjahn, and Sina Mostafawy (2011a). “Interactive Generation of Virtual Environments Using MUAVs.” In: *Proceedings of International Symposium on VR Innovation*. IEEE, pp. 89–96.

Strothoff, Sven and Klaus Hinrichs (2013a). “Adding Context to Multi-touch Region Selections.” In: *Proceedings of ITS 2013*. ACM, pp. 397–400.

Strothoff, Sven and Klaus Hinrichs (2013b). “Adding Context to Multi-touch Region Selections.” In: *Proceedings of MUM 2013*. ACM, 15:1–15:8.

Strothoff, Sven, Frank Steinicke, Dirk Feldmann, Jan Roters, Klaus H. Hinrichs, Tom Vierjahn, Markus Dunkel, and Sina Mostafawy (2010). “A Virtual Reality-based Simulator for Avionic Digital Service Platforms.” In: *Proceedings of Joint Virtual Reality Conference (Additional Material)*, 8 pages.

Strothoff, Sven, Wolfgang Stuerzlinger, and Klaus Hinrichs (2015). “Pins ’n’ Touches: An Interface for Tagging and Editing Complex Groups.” Submitted.

Strothoff, Sven, Dimitar Valkov, and Klaus Hinrichs (2011b). "Triangle Cursor: Interactions With Objects Above the Tabletop." In: *Proceedings of ITS 2011*. ACM, pp. 111–119.

Valkov, Dimitar, Frank Steinicke, Gerd Bruder, and Klaus Hinrichs (2011). "2D Touching of 3D Stereoscopic Objects." In: *Proceedings of CHI '11*. ACM, pp. 1353–1362.

Valkov, Dimitar, Frank Steinicke, Gerd Bruder, Klaus Hinrichs, Johannes Schöning, Florian Daiber, and Antonio Krüger (2010). "Touching Floating Objects in Projection-based Virtual Reality Environments." In: *Proceedings of Joint Virtual Reality Conference*, pp. 17–24.

Vierjahn, Tom, Niklas Henrich, Klaus Hinrichs, and Sina Mostafawy (2013a). *sGNG: Online Surface Reconstruction Based on Growing Neural Gas*. Tech. rep. Deptartment of Computer Science, University of Münster.

Vierjahn, Tom, Guido Lorenz, Sina Mostafawy, and Klaus Hinrichs (2012). "Growing Cell Structures Learning a Progressive Mesh During Surface Reconstruction – A Top-Down Approach." In: *EG '12 - Short Papers*. Ed. by Carlos Andujar and Enrico Puppo. Eurographics Association, pp. 29–32.

Vierjahn, Tom, Jan Roters, Manuel Moser, Klaus Hinrichs, and Sina Mostafawy (2013b). "Online Reconstruction of Textured Triangle Meshes from Aerial Images." In: *Eurographics Workshop on Urban Data Modelling and Visualisation*. Ed. by Vincent Tourre and Gonzalo Besuievsky. Eurographics Association, pp. 1–4.

Vogel, Daniel and Patrick Baudisch (2007). "Shift: A Technique for Operating Pen-Based Interfaces Using Touch." In: *Proceedings of CHI '07*. ACM, pp. 657–666.

Wagner, Julie, Stéphane Huot, and Wendy E. Mackay (2012). "BiTouch and BiPad: Designing Bimanual Interaction for Hand-held Tablets." In: *Proceedings of CHI '12*. ACM, pp. 2317–2326.

Wigdor, Daniel, Hrvoje Benko, John Pella, Jarrod Lombardo, and Sarah Williams (2011). "Rock & Rails: Extending Multi-touch Interactions with Shape Gestures to Enable Precise Spatial Manipulations." In: *Proceedings of CHI '11*. ACM, pp. 1581–1590.

Wilson, Andrew D. and Hrvoje Benko (2010). "Combining Multiple Depth Cameras and Projectors for Interactions On, Above, and Between Surfaces." In: *Proceedings of UIST '10*. ACM, pp. 273–282.

Wobbrock, Jacob O., Meredith Ringel Morris, and Andrew D. Wilson (2009). "User-Defined Gestures for Surface Computing." In: *Proceedings of CHI '09*. ACM, pp. 1083–1092.

Xu, Pengfei, Hongbo Fu, Oscar Kin-Chung Au, and Chiew-Lan Tai (2012). "Lazy Selection: A Scribble-based Tool for Smart Shape Elements Selection." In: *ACM Transactions on Graphics* 31.6, 142:1–142:9.

COLOPHON

This document was typeset using open-source software and the typographical look-and-feel `classicthesis` developed by André Miede. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*".