FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Heuristiken zur Verbesserung der Mehrbenutzer-Anwendung an Multitouch-Tischen

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Wirschaftsinformatik

eingereicht von

Matthias Vilsecker
Matrikelnummer 0525373

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung
Betreuer: Thomas Grechenig
Mitwirkung: Christoph Wimmer

Wien, 20.09.2010

(Unterschrift Verfasser) (Unterschrift Betreuer)

Forschungsgruppe Industrial Software (INSO)
Institut für Rechnergestützte Automation (E183)
Fakultät für Informatik
Technische Universität Wien

Diplomarbeit

Heuristiken zur Verbesserung der Mehrbenutzer-Anwendung an Multitouch-Tischen

Autor:
Matthias Vilsecker
Degengasse 23/2
1160 Wien

Betreuer:
Thomas Grechenig
Mitwirkung:
Christoph Wimmer

Wien, 20.09.2010

„It's mine, don't touch!"

P. Peltonen et al. [PKS+08]
(zitiert einen Probanden einer Feldstudie)

Erklärung

Matthias Vilsecker, Degengasse 23/2, 1160 Wien.

Hiermit erkläre ich, dass ich diese Arbeit selbstständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 20.09.2010

Matthias Vilsecker

Danksagung

Mein Dank gilt meiner Familie für ihre Unterstützung; sowie Christoph Wimmer und Thomas Grechenig für die wissenschaftliche Betreuung der Arbeit.

Kurzfassung

Im Rahmen der vorliegenden Diplomarbeit werden aktuelle Trends in der Mehrbenutzer-Anwendung an Multitouch-Tischen untersucht. Der Fokus der Arbeit liegt dabei darauf, auftretende Probleme zu erheben und Möglichkeiten aufzuzeigen, um deren Auswirkungen in Form von Heuristiken und Algorithmen zu minimieren. Es wird von der These ausgegangen, dass Heuristiken in Form algorithmischer Ansätze und Design Guidelines die Usability bei Kollaborationen erhöhen können, während in verwandten Arbeiten vorwiegend der Einsatz von zusätzlicher Hardware zur Verbesserung der Mehrbenutzer-Fähigkeit evaluiert wird. Häufige Probleme in der Mehrbenutzer-Anwendung stellen beispielsweise inkorrekte Gruppierungen oder Interpretationen von Gesten dar, die durch zu etablierende Heuristiken hinsichtlich Korrektheit, Performance, Stabilität und Robustheit verbessert werden sollen.

Dabei werden zahlreiche Möglichkeiten dargestellt, wie mit Algorithmen und der Umsetzung von Design Guidelines Probleme der Mehrbenutzer-Anwendung verhindert werden können: Diese erschließen sich auf den Ebenen der gestenspezifischen, räumlichen, objektspezifischen und benutzerspezifischen Segmentierung. So wurde für jedes identifizierte Problem mindestens eine Verbesserungsmöglichkeit aufgezeigt. Im Vergleich zum Einsatz zusätzlicher Hardware, wie dies in der verwandten Literatur der Fall ist, sind die Heuristiken flexibel einsetzbar - bei der Benutzeridentifikation und den sich ergebenden zusätzlichen Interaktionsarten müssen jedoch Abstriche gemacht werden.

Unter Einsatz einer explorativen, vergleichenden Evaluierung zweier in der Arbeit entwickelten Prototypen - einem mit und einem ohne implementierten Verbesserungen - wurden die Heuristiken validiert. Zusammenfassend zeigt diese Evaluierung klar auf, dass die getesteten Benutzer zufriedener mit der durch Heuristiken verbesserten Version des Prototyps sind und in dieser Version in Summe deutlich weniger Probleme auftreten.

Keywords. *Multitouch, Multitouch-Tisch, Mehrbenutzerfähigkeit, Multi-User, Heuristiken, Gestenerkennung, Kollaborationen*

Abstract

This diploma thesis explores trends in collaborative computing on multi-touch tabletops. The thesis focuses on identifying problems, which appear in multi-user situations, and identifying possibilities to minimize their effects. In doing so, we follow the approach, that heuristics can be used to improve the usability while collaborating. In contrast, the related work mainly tries to enhance multi-user awareness with the help of additional hardware. Common problems of multi-user usage are for example caused by incorrect groupings and interpretation of gestures. These should be enhanced with heuristics concerning the characteristics of robustness, performance, correctness and stability.

In doing so, a set of possibilities to avoid problems of multi-user usage via algorithms and design guidelines is identified: These are segmented on different levels - gesture specific, object specific, user specific and spatial. Within those levels, for every identified problem at least one improvement opportunity is shown. In comparison to operating with additional hardware, as it can be found in the related work, the heuristics are flexible in use - however, the identification of the users and additional interaction techniques are limited.

With the help of an explorative, comparative evaluation of two prototypes developed in this thesis - one with, one without implemented improvements -, the heuristics were validated. To sum up, the study shows, that the users were more content with the improved version of the prototype and less problems occurred during the test.

Keywords. *Multitouch, Tabletop, Multi-User, Heuristics, Gesture Tracking, Gesture Recognition, Collaborative Computing*

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Problemstellung

Multitouch-Tische bieten ein großes und natürliches User Interface. Dieses ermöglicht die direkte Manipulation von Inhalten mit der gleichzeitigen Erkennung von mehr als einem Finger. Die Interaktion von mehreren Benutzern mit dem Gerät wird dadurch unterstützt - Multitouch-Tische erlauben Teamarbeit.
Für die direkte Manipulation von Inhalten beziehungsweise der Interaktion im Allgemeinen, spielt die Erkennung der Finger und Gesten des Benutzers eine wesentliche Rolle, da deren Qualität direkten Einfluss auf die Akzeptanz und die Usability von Multitouch-Anwendungen hat. Diese Qualität in den Ausprägungen Korrektheit, Performance, Stabilität und Robustheit der erkannten Finger und Gesten gilt es daher zu optimieren.
Vor allem hinsichtlich der Unterstützung von mehreren Benutzern ergeben sich bei der Interpretation der Berührungspunkte einige Probleme, die in dieser Arbeit untersucht werden. Die Anzahl der Touch-Punkte steigt, mehrere Gesten und Interaktionen müssen synchron erfasst und die einzelnen Benutzer müssen identifiziert werden. Zusammengefasst führt dies zu einer höheren Komplexität. In Folge kann es beispielsweise zu Fehleinschätzungen kommen, welcher Touch-Punkt zu welcher Geste beziehungsweise zu welchem Benutzer gehört.
Sofern die Problematik der Multi-User-Eingabe nicht durch zusätzliche Hardware behandelt wird, führt dies unmittelbar zu einer erhöhten Fehlerrate beziehungsweise Fehleranfälligkeit. Jedoch erscheint gerade eine fehler- und problemfreie Benutzung eines Multitouch-Tisches von mehreren Personen, auf Grund der Größe des Interfaces und der Möglichkeit der Interaktion an nur einem Gerät, wesentlich.
Die Interaktion bei dieser Art von kollaborativer Arbeit wird erst durch Multi-Touch ermöglicht, da beispielsweise eine Interaktion mit mehreren Computer-Mäusen, aufgrund von mehreren indirekten Zeigern, nicht möglich

wäre. [DL01, Seite 219] beschreibt diese Problematik treffend: *„It can be challenging for users to keep track of one pointer on a large surface with lots of activity. Keeping track of many mice is nearly impossible.“*

Jeder Berührungspunkt an einem Multitouch-Tisch stellt ein unabhängiges Ereignis dar. Um diese zu komplexen Interaktionen und Gesten zusammenzufassen gibt es Algorithmen, wie sie beispielsweise bei [WRL08] vorgestellt werden. Damit mehrere Benutzer an einem Tisch arbeiten können, müssen diese Algorithmen um Heuristiken, Design Guidelines und Schätzverfahren erweitert werden. Ein algorithmischer Ansatz dieser Form wird etwa in [DSA09] vorgestellt. Diese werden in der vorliegenden Arbeit analysiert und ausgebaut. Darauf aufbauend sollen zusätzliche Heuristiken zur Verbesserung der erwähnten Problematiken definiert werden. Des weiteren wird festgelegt in welchem Umfeld deren Einsatz sinnvoll ist.

Heuristiken werden in dieser Arbeit als „Daumenregeln“ betrachtet, deren Anwendung im Allgemeinen zu einer guten Lösung eines Problems führt. Ein Beispiel für eine zusätzliche Heuristik ist die Analyse von Interaktionssequenzen. So könnte die Applikation zuerst eine Sequenz von Interaktionen abwarten, um dann zeitversetzt zu entscheiden, welche Gesten erkannt wurden. Hierbei müssten natürlich Abstriche betreffend der Flüssigkeit des Ablaufs gemacht werden.

In der Literatur findet man derzeit hauptsächlich Verfahren, die zusätzliche Hardware benutzen, um zwischen den Benutzern zu differenzieren. Beispielsweise werden mit speziellen Sensoren ausgestattete Stifte benutzt [BHH+07], zusätzliche Kameras über dem Tisch montiert [DDSP08], oder Antennen und Sensoren im Tisch und in den Stühlen der Benutzer [DL01] verwendet. Diese Verfahren sollen jedoch in dieser Arbeit nur theoretisch untersucht werden. Grund dafür ist, dass die Installation von weiterer Hardware einerseits zusätzliche Anforderungen an das Betriebsumfeld mit sich bringen würde, und andererseits zu Akzeptanzproblemen bei den Benutzern führen könnte, da diese bestimmte Hilfsmittel bei der Interaktion verwenden müssten. Durch die Notwendigkeit der Montage von zusätzlichen Kameras über den Tisch werden zusätzliche Anforderungen an das Setting gestellt: Zusätzliche, vom Tisch unabhängig angebrachte Hardware wäre notwendig. Dadurch könnte etwa der Tisch nicht mehr ohne zusätzlichen Aufwand in jedem beliebigen Raum eingesetzt werden.
Ist das Tragen von zusätzlichen Hilfsmitteln notwendig, so beschreibt [DL01, Seite 219] *„[...] relying on a separate physical device keeps us from utilizing the natural human tendencies of reaching, touching and grasping.“*. Es geht demnach die Natürlichkeit der Interaktion verloren und es treten Akzeptanzprobleme auf. So beobachten beispielsweise [LPB+09, Seite 3214] bei deren Studie, bei welcher beide Interaktionsformen - die Interaktion mit

zusätzlichen Geräten und die Interaktion mit bloßen Händen - möglich war, *„in our observations, users immediately tried to touch the interactive surface with bare hands“*.

1.2 Motivation

Multitouch-Tische erlauben, vergleicht man sie mit traditionellen Interaktionsmöglichkeiten wie Monitor und Computer-Maus, eine natürliche und direkte Interaktion. Im Gegensatz zu dieser traditionellen Interaktionsform, welche als GUI[1]-basiert bezeichnet wird, verwenden Multitouch-Tische bereits existierende Objekte der Umgebung um diese interaktiv zu gestalten. [IU97, Seite 240] führt an: *„GUIs fall short of embracing the richness of human senses and skills people have developed through a lifetime of interaction with the physical world.“*. Multitouch-Tische dagegen unterstützen die menschlichen Fähigkeiten: Sie stellen sowohl Display, also den Output, als auch Möglichkeit zum direkten Input dar und ermöglichen so natürliche Gesten um Objekte zu manipulieren. Des weiteren erlauben diese Geräte, wie bereits erwähnt, eine neue Form der kollaborativen Arbeit, bei der die Tendenz der Menschen, Tische für Meetings und Face-to-Face Kommunikation zu nützen, unterstützt wird. Denkt man beispielsweise an Besprechungen und Meetings im traditionellen Umfeld bei Verwendung eines Computers, so wäre eine gute Sicht auf einen Monitor nur beengt möglich. Auch der Einsatz eines Videoprojektors würde der menschlichen Natur nicht entgegen kommen, da Besprechungen normalerweise rund um einen Tisch statt finden. In beiden Fällen ist weiters die Eingabe von Daten beziehungsweise die Interaktion nur der einen Person an Maus und Tastatur vorbehalten. Im Rahmen kollaborativer Tätigkeiten werden weiters typische Arbeitsformen unterstützt, wie dies [TR09, Seite 2139] anführt: *„They (Anmerkung: Multitouch-Tische) afford some of the familiar work practices of collaboration around traditional tables, such as fluid transitioning between individual and group work, and coordination based on spatial partitioning.“*

Die Möglichkeiten von Multitouch sind jedoch nicht gänzlich neu. Konzepte in verschiedenen Ausprägungen existieren bereits seit den 1970er Jahren. Zahlreiche Patente unterstreichen dies und zeigen unterschiedliche Methoden zur Implementierung von Multitouch [SBD+08]. Über Geräte ähnlich zu den hier behandelten Multitouch-Tischen diskutierten bereits 1985 Buxton, Hill und Rowley [BHR85]. Es werden berührungsempfindliche, horizontal angebrachte Oberflächen, die in der Größe variieren können und dabei mehrere Touch-Punkte simultan erkennen, erwähnt und es wurden Prototypen mit diesen Ausprägungen entwickelt.
Seit der Präsentation von Multitouch durch Jefferson Han [Han05] und der

[1] Graphical User Interface, deutsch: graphische Benutzerschnittstelle.

Markteinführung des Apple iPhone[2] ist das Thema wieder aktuell. Diese Aktualität wird auch in Abbildung 1.1 ersichtlich, welche die Popularität der Suchbegriffe 'multitouch' und 'multi-touch' kumuliert und normalisiert von 2005 bis 2009 - analysiert von Google Trends[3] - darstellt. Interessant ist hierbei, dass die Präsentation von Han und die einzelnen Markteinführungen von diversen Multitouch-Produkten klar ersichtlich sind.

Abbildung 1.1 – Die Popularität der Suchbegriffe 'multitouch' und 'multi-touch' kumuliert von 2005 bis 2009 analysiert von Google Trends, adaptiert nach [SBD+08]

Die erwähnten innovativen Möglichkeiten, welche Multitouch-Tische bieten, die Wichtigkeit der Identifizierung welche Gesten zu welchem Benutzer gehören, sowie die Aktualität der Technologie im Allgemeinen stellen die Motivation für diese Arbeit dar.

1.3 Zielsetzung

Ziel der Arbeit ist es zum einen Probleme bei der Interaktion von mehreren Personen mit einem Multitouch-Tisch aufzuzeigen. Zum anderen sollen heuristische Methoden zur Lösung dieser Probleme entwickelt werden, welche die Qualität hinsichtlich Richtigkeit, Stabilität, Performance und Robustheit der erkannten Finger und Gesten bei der Verwendung eines Multitouch-Tisches durch mehrere Personen verbessern.
In Bezug auf die in Kapitel 1.1 definierte Problemstellung, soll folgende Forschungsfrage beantwortet werden:

Wie lässt sich die Qualität von Gesten- und Finger-Tracking an einem Multitouch-Tisch bei einer Multi-User-Anwendung mit Algorithmen und Heuristiken verbessern?

Es soll die Hypothese, es gäbe Algorithmen und Design Guidelines zur Op-

[2] http://www.apple.com/iphone/, zuletzt abgerufen am 14.01.2010.
[3] http://www.google.de/intl/en/trends/about.html, zuletzt abgerufen am 14.01.2010.

timierung der Multitouch-Interaktion mit Applikationen, die mehrere Benutzer unterstützen, bestätigt werden. Dabei wird erwartet, dass als Bestätigung eine Liste dient, die aufzeigt, welche evaluierten Algorithmen und Heuristiken sich zur Anwendung in welchem Umfeld eignen.

Die Beantwortung der Fragestellung beziehungsweise die Bestätigung der aufgestellten Hypothese findet in drei Phasen statt. In der ersten Phase werden durch eine Evaluierung von Mehrbenutzer-Applikationen konkrete Probleme und kritische Designaspekte bei der Interaktion aufgezeigt. Weiters wird in dieser Phase ein Prototyp entwickelt der diese Designentscheidungen mangelhaft berücksichtigt, indem gegen diese Aspekte bewusst verstoßen wird. Im nächsten Schritt werden Heuristiken zur Verbesserung der Interaktion entwickelt, beziehungsweise bestehende Algorithmen analysiert. Diese werden auf die Applikation aus der ersten Phase angewandt und implementiert. Um die Heuristiken zu verifizieren und Anwendungsgebiete zu definieren, wird in der dritten Phase schließlich die Ausgangsapplikation und die verbesserte Applikation einer vergleichenden Evaluierung unterzogen.

1.4 Aufbau der Arbeit

Kapitel 2 behandelt sowohl Grundlagen als auch Definitionen und bietet einen Überblick über die derzeitigen Entwicklungen im Umfeld von Multitouch und Multitouch-Tischen. Dabei wird auf die grundlegenden Aspekte von Multitouch hinsichtlich deren Relevanz und deren Auswirkungen auf das Forschungsgebiet der Mensch-Computer Interaktion eingegangen. Zusätzlich finden auch die physikalischen Aspekte der optischen Sensorik, welche für den eingesetzten Multitouch-Tisch wesentlich sind, Erwähnung. Diese Grundlagen dienen als Ausgangspunkt für den implementierten, in Kapitel 7 vorgestellten Prototypen. Auf welche Technologien bei dieser Implementierung zurückgegriffen wurde beziehungsweise Details zum verwendeten Multitouch-Tisch, werden in Kapitel 3 erläutert. Kapitel 4 fasst verwandte Literatur zum Thema zusammen und beschreibt zusätzliche Methoden, die anders als im praktischen Teil dieser Arbeit, zusätzliche Hardware benutzen, um zwischen den Benutzern zu differenzieren.

Im darauf folgenden Teil der Arbeit, in Kapitel 5, werden die Problemgebiete der Multi-User-Anwendung sowie die eingesetzte Methodik diese zu identifizieren vorgestellt. Basierend auf diesen Problemen wird in Kapitel 6 auf Heuristiken und Algorithmen, sowohl existierende als auch im Zuge dieser Arbeit neu entwickelte, eingegangen. Dabei wurde versucht die Heuristiken zu kategorisieren und Einsatzfelder zu definieren. Die durchgeführte Evaluierung dieser entwickelten Heuristiken beziehungsweise deren Ergebnisse werden in Kapitel 8 vorgestellt.

Stärken und Schwächen der Ergebnisse werden in Kapitel 9 diskutiert und mit Ergebnissen verwandter Arbeiten verglichen. Abschließend fasst Kapitel 10 die wesentlichen Erkenntnisse zusammen und gibt einen Ausblick.

Kapitel 2

Technische und inhaltliche Grundlagen

Dieses Kapitel behandelt grundlegende Aspekte dieser Arbeit - aus dem Umfeld Multitouch und aus dem Umfeld des Software-Engineerings. So werden Entwicklungen hinsichtlich der Technologie Multitouch besprochen; auch Mehrbenutzer-Situationen an Multitouch-Tischen finden Erwähnung. Um einen Überblick über die bei Multitouch-Bildschirmen eingesetzte Hardware zu verschaffen, werden ebenfalls die wichtigsten zugrunde liegenden Techniken der optischen Sensorik besprochen. Zusätzlich werden wichtige Aspekte der Software-Qualität erläutert, da in dieser Arbeit eine Verbesserung der Korrektheit, Performance, Stabilität und Robustheit einer Software - Ausprägungen der Qualität von Software - erreicht werden soll.

2.1 Multitouch

Bill Buxton et al. [BHR85, Seite 215-216] bezeichnet 1985 Touch-Oberflächen als *„flat surface [...] that can sense the location of a finger pressing on it. That is, it is a tablet that can sense that it is being touched, and where it is being touched.“*. Multitouch bezeichnet er als eine Erweiterung dieser „einfachen“ Form der touch-sensitiven Eingabe indem es mehrere Berührungspunkte gleichzeitig erkennt: *„In this case, the location of several points of contact would be reported.“*[BHR85, Seite 216].
Aufgrund dieser Definition lässt sich feststellen, dass ein Touchscreen durch das Berühren des Monitors einen Computer direkt steuern kann und somit ein Eingabegerät darstellt, bei dem Eingabe und Ausgabe (sprich das Display) direkt gekoppelt sind. Es wird also durch Berühren des Bildschirms - üblicherweise mit der Hand oder mit Fingern, der Programmzustand beeinflusst. Kann bei einem solchen Gerät nicht nur eine einzige X-/Y-Koordinate aufgezeichnet werden, sondern mehrere gleichzeitig, spricht man von Multitouch-Fähigkeit.

2.1.1 Touch versus Klick

Betrachtet man die erwähnten Vorteile von Multitouch-Umgebungen, die Natürlichkeit der Interaktion und die direkte Kopplung von Ein- und Ausgabegerät, so könnte gefolgert werden, dass Multitouch-Oberflächen traditionelle Interaktionsformen - wie Tastatur und Maus - ablösen könnten. Zusätzlich zeigen Studien [KAD09] sowie [FWSB07], dass die Selektion von Zielen mit Touch-Oberflächen durch die angeführten Vorteile grundsätzlich schneller ist, als die Selektion mit der Maus. Die Studie, welche in [KAD09] vorgestellt wird, zeigt folgende Ergebnisse für Erstbenutzer von Multitouch:

- Die schnellste Multitouch-Interaktion ist circa zweimal so schnell wie Maus-Interaktion - unabhängig von der Anzahl der Ziele.
- Den Hauptanteil der reduzierten Selektionszeit trägt der direkten Input und die Natürlichkeit von Touch-Interaktion.

Beide Studien erwähnen jedoch auch, dass Touch-Screens Nachteile mit sich bringen und deshalb nicht in allen Umgebungen sinnvoll einsetzbar sind - beispielsweise als kompletter Ersatz der Maus in Desktop-Computern. Einige dieser Nachteile betreffend Multitouch-Tischen werden etwa in [WR09, Seite 1063] erwähnt: Ermüdungserscheinungen in den Armen, das Verdecken eines Teil des Screens mit dem Arm beziehungsweise der Hand, die fehlende Präzision der Selektion, sowie das fehlende taktile Feedback bei der Eingabe von Texten.
[FWSB07] zeigt dies ebenfalls und folgert, Multitouch-Eingabe bei Tischen sei nur für Aufgaben die zwei Hände erfordern besser geeignet als Computer-Mäuse: *„[...] Experiments 1 and 2 indicate that users may be better off using a mouse for unimanual input and their fingers for bimanual input when working on a large, horizontal display.“* [FWSB07, Seite 655].

2.1.2 Direkter versus Indirekter Touchinput

Bisher wurden in dieser Arbeit nur direkte Toucheingaben, bei denen Eingabe- und Ausgabegerät gleich ist, vorgestellt. Wie bei Touchpads von Notebooks ersichtlich, ist die Toucheingabe auch indirekt vorstellbar und möglich. Diese Touchpads simulieren die Mausinteraktion und sind als Singletouch- und als Multitouch-Geräte erhältlich. Bei Multi-Touchpads ist auch Input via Gesten - wie etwa Scrollen, Zoom oder Vor und Zurück im Browser - möglich. Interessant ist hierbei, dass diese Gesten unter Umständen anders umgesetzt werden müssen, um Intuition zu gewährleisten. Ein Beispiel stellt hier das Scrollen dar: Während beim Scrollen bei direktem Touchinput - etwa bei einem Apple iPhone - die Bewegung des Fingers von unten nach oben verläuft (die Seite wird nach unten „geschoben“), verhält sich diese Geste bei indirekter Toucheingabe - etwa bei einem Trackpad eines Apple Macbooks - genau umgekehrt (die Bildlaufleiste wird mit den Fingern „geführt“).

2.1.3 Gesten bei Multitouch

[Saf09, Seite 2] definiert Gesten im Zusammenhang mit Computern als *„any physical movement that a digital system can sense and respond to without the aid of a traditional pointing device such as a mouse or stylus. A wave, a head nod, a touch, a toe tap, and even a raised eyebrow can be a gesture“*. Dies definiert Gesten für jegliche Interfaces, die auf Interaktionen mit Gesten basieren. Die Definition kann daher auch auf Gesten im Multitouch-Bereich angewandt werden, was auch durch das angeführte Bespiel - etwas Berühren als Geste - dargestellt wird. In diesem Bereich findet weiters eine gewisse Einschränkung der Möglichkeiten der Gesten statt, da die Benutzer das Gerät berühren müssen um mit diesem zu interagieren.
Folgt man der Definition von Gesten, stellt bereits das bloße Berühren einer Multitouch-Oberfläche eine Geste dar. Eine solche Berührung wird vom System als Touch-Punkt erkannt. Komplexere Gesten können durch die Interpretation der Charakteristika des Touchpunktes beziehungsweise der Interpretation mehrerer gleichzeitiger oder aufeinander folgender Touch-Punkte gebildet werden. Dabei weist ein Touch-Punkt, abhängig von der verwendeten Multitouch-Technologie und den verwendeten Sensoren, unterschiedliche Attribute auf. [WR09, Seote 1064] teilt diese Eigenschaften in vier Kategorien ein: Position, Bewegung, physische Eigenschaften und Eigenschaften des Events. Einige Ausprägungen dieser Kategorien werden in Tabelle 2.1 dargestellt.

Position	**Bewegung**	**physische Eigenschaften**	**event-spezifische Eigenschaften**
• Wert des Koordinatensystems (x,y)	• Geschwindigkeit • Beschleunigung	• Größe (Länge und Breite) • Form • Orientierung (Richtung des Fingers) • Druck	• tippen • streifen

Tabelle 2.1 – Klassifizierung der Berührungspunkte, adaptiert nach [WR09, Seite 1064]

Ein weiteres, für die Interpretation der Gesten wesentliches Attribut ist die Anzahl der aufgezeichneten Touch-Punkte, da durch deren Kombination

komplexere Gesten abgebildet werden können.
Nachfolgend werden Gesten, die sich als de-fakto Standard etabliert haben, vorgestellt. Diese werden deshalb als de-fakto Standard angesehen, da sie in zahlreichen existierenden Produkten implementiert sind und etwa in den Apple iPhone Human Interface Guidelines[4] sowie in den Microsoft Windows Touch Gestures[5] Erwähnung finden.

- **Tippen um etwas zu öffnen, „Tipp-Geste“**
  Abbildung 2.1-1a - [Saf09, Seite 46] beschreibt, dass diese Geste immer eingesetzt werden soll, wenn Objekte aktiviert beziehungsweise Buttons gedrückt werden sollen, da diese die einfachste manuelle Geste ist und den natürlichen Ersatz des Mausklicks darstellt.

- **Ziehen um ein Objekt zu bewegen, „Zieh-Geste“**
  Abbildung 2.1-1b - Diese Geste soll immer eingesetzt werden, wenn Slider bedient oder Objekte auf eine andere Stelle am Screen bewegt werden. [Saf09, Seite 50].

- **Zwei Finger drehen, „Rotier-Geste“**
  Abbildung 2.1-1c - Diese Geste wird etwa in den Microsoft Windows Touch Gestures angeführt und kann sowohl mit zwei Fingern einer Hand als auch mit zwei Fingern verschiedener Hände ausgeführt werden.

- **Zusammenziehen/Auseinanderziehen, „Skalier-Geste“/„Zoom-Geste“**
  Abbildung 2.1-1d (Skalier-Geste), Abbildung 2.1-2a (Zoom-Geste) - Je nachdem ob bei dieser Geste die beiden Finger ein Objekt oder den Screen berühren, soll dieses Interaktionsverhalten laut [Saf09, Seite 64] verwendet werden um die Größe eines Objekts beziehungsweise den Zoom-Faktor der gesamten Applikation zu verändern.

- **Einen Finger am Hintergrund bewegen[6], “‘Pan-Geste“**
  Abbildung 2.1-2b - Durch diese Geste wird der Hintergrund geschwenkt beziehungsweise am Hintergrund „gescrollt“. Diese Geste wird wiederum in den Microsoft Windows Touch Gestures angeführt. Weiters führt [Saf09, Seite 52] diese Geste an, um den sichtbaren Arbeitsbereich zu verändern.

[4] http://developer.apple.com/iphone/library/documentation/userexperience/ conceptual/mobilehig/DesigningNativeApp/DesigningNativeApp.html#//apple_ref/doc/uid/ TP40006556-CH4-SW4, zuletzt abgerufen am 25.08.2010.

[5] http://msdn.microsoft.com/en-us/library/cc872774.aspx#gestures, zuletzt abgerufen am 14.08.2010.

[6] Dies wird im englischen Sprachgebrauch als „Pan“-Geste bezeichnet. Auf Grund des Fehlens einer passenden Übersetzung wird in der Folge dieser Anglizismus verwendet.

[7] Nach http://www.mt4j.org/mediawiki/index.php/Multitouch_gestures, zuletzt abgerufen am 12.03.2010.

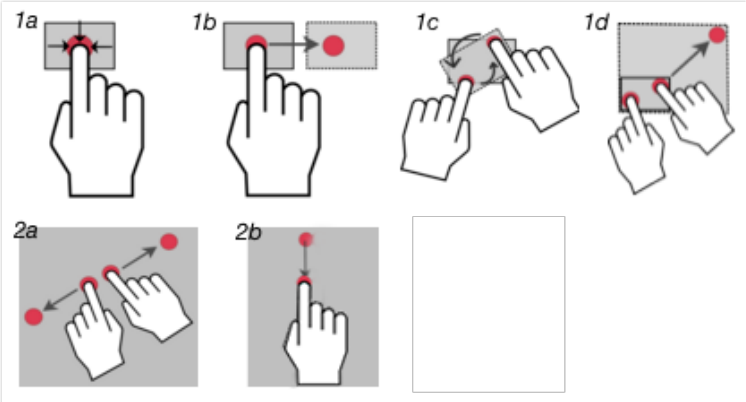


Abbildung 2.1 – Gesten auf einem Multitouch Gerät[7].

2.1.4 Multitouch-Tische

Multitouch-Tische finden bereits im Jahr 1985 bei Bill Buxton et al. Erwähnung, als *„[...] flat surface, usually mounted horizontally or nearly horizontally, that can sense the location of a finger pressing on it. [...] Touch tablets can vary greatly in size, from a few inches on a side to several feet on a side.“* [BHR85, Seiten 215-216]. Weiters wird angeführt, dass diese Tablets mehrere Kontaktpunkte gleichzeitig unterstützen sollten.

Singleuser versus Multiuser. Da im Falle von Multi-User-Anwendungen mehrere Finger-Inputs sowie mehrere Gesten simultan erkannt werden müssen, ergeben sich erweiterte Anforderungen an die Umgebung. [DL01, Seite 220] definiert sechs Eigenschaften, die eine mehrbenutzerfähige Touch-Umgebung aufweisen sollte. Die kursiv dargestellten Anmerkungen wurden ergänzt.

- **Multitouch**
  Es müssen mehrere Finger gleichzeitig erkannt werden, das bedeutet es muss sich um einen Multitouch-Tisch handeln.
  Erforderlich für Multi-User Umgebungen, bedingt erforderlich für Single-User Umgebungen.

- **Identifizierung**
  Es muss erkannt werden, welcher Finger und welche Geste zu welchem Benutzer gehört.
  Erforderlich für Multi-User Umgebungen, nicht erforderlich für Single-User Umgebungen.

- **Tolerant gegenüber weiteren Einflüssen**
  Objekte, die auf dem Tisch liegen gelassen werden, sollten die Umge-

bung nicht beeinflussen.
Erforderlich für Single- und Multi-User Umgebungen.

- **Dauerhaftigkeit**
  Wiederkehrende Kalibrierungen oder Reparaturen sollten nicht notwendig sein.
  Erforderlich für Single- und Multi-User Umgebungen.

- **Keine zusätzlichen Belastungen**
  Es sollten keine weiteren Geräte zur Interaktion benötigt (zB Stifte, Kontakte die am Körper getragen werden) werden.
  Erforderlich für Single- und Multi-User Umgebungen.

- **Billige Herstellung**
  Die Herstellung des Geräts sollte preiswert sein.
  Erforderlich für Single- und Multi-User Umgebungen.

Anwendungen. Wie man aus Abbildung 1.1 aus Kapitel 1.2 entnehmen kann, wächst die Faszination für die Multitouch-Technologie gerade durch die Einführung kommerzieller Produkte. Als Beispiel für einen Multitouch-Tisch dient der Microsoft Surface Table[8]. Microsoft präsentierte 2007 dieses Produkt für den Massenmarkt, welches vergleichbar mit einem Kaffeetisch mit einer interaktiven Oberfläche ist. Der Tisch beruht auf Diffuse Illumination und erlaubt daher Interaktionen zwischen Mensch und Computer, aber auch zwischen Computer und anderen Geräten sowie Alltagsgegenständen. Ein weiteres kommerzielles Produkt stellt der DiamondTouch Tisch dar.

Anwendung können Multitouch-Tische weiters in jeglicher Form von kollaborativen Tätigkeiten, bei der Personen mit einem einzigen Gerät arbeiten können, finden. Es wird dabei die menschliche Tendenz unterstützt, sich für Face-to-Face Interaktionen rund um Tische zu sammeln. Durch die größe des Interfaces und die simultane Erkennung von mehreren Interaktionen wird dadurch Teamarbeit unterstützt und es werden Gruppendynamiken gesteigert. [SRF+06, Seite 33].

2.2 Kollaborationen an Multitouch-Tischen

Horizontale, interaktive Displays eignen sich sehr gut für Kollaborationen. Auf der einen Seite werden durch die Größe des Displays kollaborative Tätigkeiten bereits Nahe gelegt, während auf der anderen Seite die Metapher eines Tisches benutzt wird: *„The conventional way of working in a co-located group, is to sit around tables, talking with one another and interacting with a bricolage of technological and paper-based artefacts placed on top*

[8] http://www.microsoft.com/surface/, zuletzt abgerufen am 19.02.2010.

of the table.“ [RL03, Seite 2]. Die im Zitat angesprochenen technologischen Artefakte sind in einem Multitouch-Tisch bereits involviert.

[RL03] vergleicht in einer Studie Kollaborationen an großen horizontalen und vertikalen Displays. Ebenfalls wird ein Vergleich mit der Zusammenarbeit an einem traditionellen Computer-Monitor angestellt. Es wird gezeigt, dass die meiste und engste Zusammenarbeit an horizontalen Displays - also an Tischen - statt findet, da an diesen unter anderem am meisten Vorschläge von verschiedenen Personen geäußert wurden: *„The findings from the [...] study suggest that the physical affordances of the horizontal display, compared with the vertical or PC monitor display, encourage group members to change more roles, explore more ideas and follow more closely what each other is doing. Moreover, the mediating actions of swapping, sharing and indexing were physically more comfortable to accomplish in this condition and, in so doing, made the interaction socially more comfortable. The net effect was that the groups working around the horizontal table were the most socially cohesive [...].“* [RL03, Seite 13].
Hinsichtlich des Vergleichs von horizontalen Displays und traditioneller Zusammenarbeit vor einem einzelnen Computer-Monitor, wird ebenfalls erwähnt, dass bei Computer-Monitoren zwar durch die enge physische Nähe ebenso ein hohes Level an Team-Bewusstsein erreicht werden kann, jedoch eben diese Nähe zu Unbehagen und zu sozialer „Verlegenheit“ führen kann. An einem Multitouch-Tisch etwa existiert dieses Problem nicht und wurde von den Teilnehmern der Studie auch nicht als solches empfunden. [RL03, Seite 10].

In [TTP+06] wird eine Studie vorgestellt, welche Zusammenarbeit von mehreren, gleichzeitig anwesenden Personen auf horizontalen Displays, wie sie etwa Multitouch-Tisch darstellen, fokussiert. Es wird dabei gezeigt, dass Kollaboration an horizontalen Displays genau wie an traditionellen Tischen aus Phasen individueller Tätigkeiten und Phasen, in denen aktive Teamarbeit vorherrschend ist, besteht und es fließende Übergänge zwischen diesen Phasen gibt: *„Our results indicate that individuals frequently and fluidly engage and disengage with group activity through several distinct, recognizable states with unique characteristics.“* [TTP+06, Seite 1181]. Die Charakteristika einiger dieser Phasen werden wie folgt beschrieben ([TTP+06, Seite 1187]):

- Gleiches Problem, gleicher Arbeitsbereich: Die Teilnehmer arbeiten aktiv miteinander am gleichen Problem.
- Gleiches Problem, unterschiedlicher Arbeitsbereich: Die Teilnehmer arbeiten gleichzeitig am selben Teilproblem, konzentrieren sich jedoch auf unterschiedliche Teile des Tisches.

- Einer arbeitet, der Andere sieht zu: Ein Teilnehmer interagiert aktiv, der andere sieht dabei nur zu. In einigen Fällen sind sie auch aktiv (in Form von mündlich geäußerten Ratschlägen) am Geschehen beteiligt.
- Verschiedene Probleme: Die Teilnehmer arbeiten völlig unabhängig voneinander an unabhängigen Teilproblemen.

Aus diesen verschiedenen Phasen lässt sich ableiten, dass - wie auch bei kollaborativen Arbeiten an traditionellen Tischen - Phasen der aktiven Zusammenarbeit als auch der individuellen Tätigkeit auftreten. Betreffend der individuellen Tätigkeiten kann sowohl das gleiche Problem als auch eine völlig unabhängige Tätigkeit - beispielsweise im Rahmen eines Teilproblems - den Fokus der Aufgabe bilden. Wie auch in [RL03] deutlich wird, sind die Übergänge zwischen diesen Phasen wesentlich, wie aus obiger Zitation (*„the mediating actions of swapping, sharing and indexing“* beziehungsweise die erwähnten Wechsel in den Rollen) deutlich wird. Dies sieht auch [RMRS+04, Seite 1441] so: *„Group work frequently involves transitions between periods of active collaborations and periods of individual activity.“*

Neben dieser Untersuchung der Kollaboration auf horizontalen Displays sowie der verschiedenen Phasen während der Zusammenarbeit, untersuchen [Sco05] sowie [SCH05] die Koordination basierend auf der räumlichen Aufteilung der Arbeitsfläche. [Sco05] zeigt bei einer Studie basierend auf einem traditionellen Tisch mit Papierzettel, dass diese Blätter für individuelle Aufgaben - beziehungsweise in Phasen in denen unabhängig gearbeitet wird - in eigene Teile des Tisches bewegt werden. Dadurch wurden diese für die Person reserviert. Nach Beendigung der Tätigkeit wurden die Zettel wieder auf den Ausgangsort beziehungsweise auch in die Mitte des Tisches zurück gelegt. Dadurch wurden diese wieder für alle Teilnehmenden frei verfügbar. [SCH05] untersucht die gleiche Problematik anhand einer Fotoapplikation und unterstreicht dabei, dass diese Vorgehensweise wie bei traditionellen Tischen auch bei Multitouch-Tischen möglich ist und auch von den Teilnehmern vollzogen wurde.

2.3 Optische Sensorik bei Multitouch-Screens

Multitouch-Bildschirme basierend auf optischer Sensorik ermöglichen eine einfache, kostengünstige Herstellung und sind beliebig skalierbar. So sind die Hauptbestandteile an Hardware einer optischen Multitouchlösung lediglich ein Videoprojektor, eine Kamera (beispielsweise eine Webcam) und Infrarot-Leuchten. Zwei solcher Methoden werden in den nächsten beiden Kapiteln vorgestellt.

2.3.1 Frustrated Total Internal Reflection (FTIR)

Jeff Han rief 2005 die Technologie Multitouch wieder ins Gedächtnis, indem er die optische Methodik Frustrated Total Internal Refelction (FTIR)[9] vorstellte. 2005 kann deshalb als Startpunkt für optische Touchsysteme gesehen werden. FTIR verwendet ein bei Licht beobachtbares Wellen-Phänomen, welches an der Grenzfläche zweier Medien (beispielsweise Luft-Wasser) auftritt, die einen unterschiedlichen Brechungsindex aufweisen. Das Licht wird dabei an der Grenzfläche nicht gebrochen, sondern vollständig reflektiert, wenngleich diese nicht beschichtet ist. Es basiert demnach auf dem Prinzip der Lichtbrechung beziehungsweise der Lichtreflektion.[Han05].

Wie in Abbildung 2.2 ersichtlich, besitzen Multitouch-Screens mit FTIR eine transparente Acryl-Oberfläche, in welche auf den Seiten Infrarot-Licht mit LEDs injiziert wird. Dies hat den Vorteil, dass die Oberflächen kostengünstig und beliebig skalierbar herzustellen sind. Wenn der Benutzer die Oberfläche berührt, kann dieser Punkt durch die Eigenschaften der Totalreflexion wahrgenommen werden. Die Veränderungen im Infrarot-Licht werden dabei von einer infrarotempfindlichen Kamera aufgezeichnet. Durch Computer-Vision-Algorithmen wird die Position des Touch-Punktes berechnet.[SBD+08].

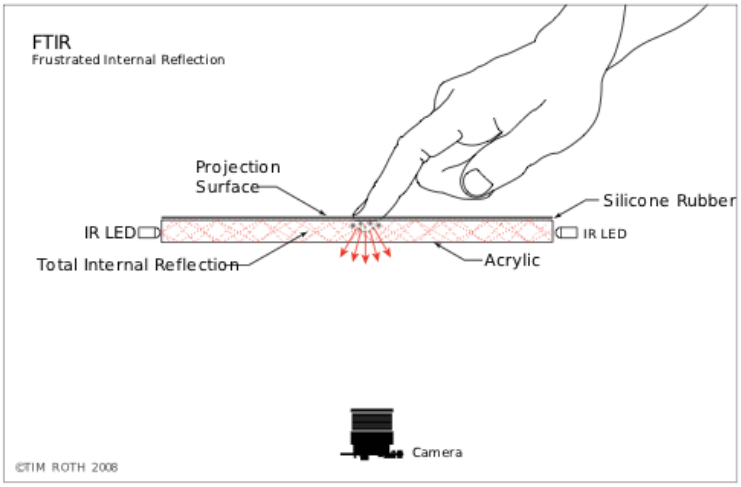


Abbildung 2.2 – Setting eines Multitouch-Tisches basierend auf Frustrated Total Internal Reflection, [SBD+08]

[9] deutsch: Totalreflexion.

2.3.2 Diffuse Illumination (DI)

Bei Systemen basierend auf Diffuse Illumination (DI)[10] ist die Hardware sehr ähnlich derer die FTIR verwenden. Bei Diffuse Illumination werden jedoch die Infrarot-Lichtquellen unter der Projektionsfläche platziert. Die Kamera kann so die Objekte in der von den Infrarot-Quellen beleuchteten Fläche durch entstehende Reflexionen wahrnehmen. Dies inkludiert nicht nur Objekte, die die Oberfläche berühren, sondern ebenfalls Objekte knapp über der Oberfläche. Im Gegensatz zu FTIR können bei Diffuse Illumination daher auch andere Objekte als Finger wahrgenommen werden. [SBD+08]. Dies ermöglicht die Verwendung von Diffuse Illumination für Tangible User Interfaces, bei welchen - zusätzlich zur Interaktion mit der Hand - die Mensch-Computer Interaktion durch physikalische Objekte unterstützt wird. [IU97, Seite 235] etwa definiert Tangible User Interfaces (TUI) als Benutzerschnittstellen, welche die Reale Welt durch die Kopplung von Alltagsgegenstände und Umgebungen mit digitalen Informationen erweitern. Die Identifikation dieser Alltagsgegenstände geschieht dabei über deren Form und über sogenannte „Fiducials“ (Markierungen die leicht erkannt werden können). Eine konkrete Anwendung von TUI stellt der „reacTable“, welcher in [JGAK07] vorgestellt wird, dar. Dieser ist eine Anwendung eines tangiblen Multitouch-Tisch basierend auf Diffuse Illumination als Musikinstrument, bei dem die physikalischen Objekte einen modularen Synthesizer nachempfinden.

Abbildung 2.3 illustriert schematisch das vorgestellte Setting basierend auf Diffuse Illumination.

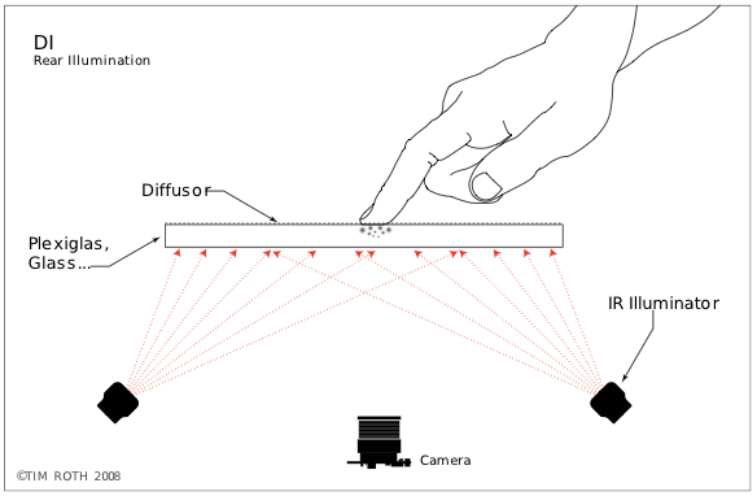


Abbildung 2.3 – Setting eines Multitouch-Tisches basierend auf Diffuse Illumination, [SBD+08]

[10] deutsch, in etwa: gestreute Beleuchtung.

Dieses Verfahren wird bei dem Multitouch-Tisch, der für Prototyp und Evaluierung verwendet wird (mehr dazu in Kapitel 3.3), eingesetzt.

Diffused Surface Illumination (DSI). Ein Problem in Diffuse Illumination ist, dass die Infrarot-Quellen gleichmäßig verteilt werden müssen, was sich in der Praxis oftmals als schwierig darstellt. Diese Problematik wird durch eine Erweiterung von Diffuse Illumination[11] - durch Diffused Surface Illumination - gelöst. Ausgangsbasis bildet dabei ein FTIR-Setup, wie es in Abbildung 2.2 dargestellt ist. Dieses wird verändert, indem eine Acryl-Oberfläche verwendet wird, welche spezielle Partikel beinhaltet, die als winzige Spiegel dienen. Wenn nun die Kanten mit Infrarot-Licht durch LEDs beleuchtet werden, so wird das Licht gleichmäßig verteilt.[Rot08].
Ein Nachteil dieser Erweiterung ist jedoch, dass diese nur einen geringeren Kontrast ermöglicht.

2.3.3 Weitere Methoden

Optische Touch-Umgebungen arbeiten mit Infrarot-Licht. Das kann in Kombination mit Umgebungslicht Probleme verursachen, da im Sonnenlicht ebenfalls Infrarot-Licht enthalten ist. Aufgrund dieses Nachteils seien an dieser Stelle auch weitere Methoden zur Implementierung eines Multitouch-Tisches angeführt.

Resistive Verfahren. Diese Verfahren basieren auf Widerstände und werden beispielsweise im Nintendo DS, PDAs und Digitalkameras eingesetzt. Die Oberfläche kann mit Finger und Stift bedient werden. Sie bestehen aus zwei übereinander liegenden Schichten aus Indiumzinnoxid. Diese sind fast durchsichtig und leiten Strom zwar, leisten aber einen gleichbleibenden, messbaren Widerstand. Berührt man die Oberfläche, verbinden sich diese Schichten und der Widerstand ändert sich. Dadurch kann einer oder mehrere Berührungspunkte gemessen werden.
Diese Methodik hat jedoch den Nachteil, dass die Oberfläche generell eine schlechte Robustheit aufweist und ein zusätzlicher Schutz dieser nicht angebracht werden kann ohne die Funktionalität zu beeinflussen. [SBD+08].

Kapazitive Verfahren. Diese Methodik wurde prinzipiell für Singletouch-Interaktionen entwickelt und hat deshalb einen Genauigkeitsverlust beim Einsatz in Multitouch. Die Technologie ermöglicht sowohl einen platzsparenden Einsatz als auch ein sehr genaues Tracking und eignet sich deshalb für den Einsatz in kleinen Geräten, wie beispielsweise Mobiltelefonen (wird etwa auch im Apple iPhone eingesetzt). Geräte basierend auf kapazitiven Verfahren sind jedoch relativ teuer zu produzieren und schlecht skalierbar.

[11] deutsch, in etwa: gestreute Umgebungsbeleuchtung.

Die Grundlegende Idee dieser Methodik ist die Erzeugung eines elektrischen Felds auf der Oberfläche. Da Menschen Ladungen in sich tragen, ändert sich beim Berühren der Oberfläche die Spannung.
Eine konkrete Ausprägung dieses Verfahren stellen projezierte kapazitive Touch-Umgebungen dar. Hierbei wird ein feines Array an Antennen auf der Unterseite des Panels gebündelt. Zusätzlich befindet sich auf der Oberseite des Panels ein Array an Antennen, welche die gesendeten Signale empfangen. Um die Position von Druckpunkten zu bestimmen, wird dieses Array ausgeweitet. Abhängig von der Anzahl der Antennen und der Signalverarbeitung sind dadurch mehrere Druckpunkte und die Stärke des Drucks messbar.
Multitouch-Tische dieser Art besitzen eine hohe Widerstandsfähigkeit und Belastbarkeit, sind jedoch in der Herstellung das teuerste Verfahren. Ein Beispiel stellt der DiamondTouch Tisch dar, welcher in Kapitel 4.1 näher vorgestellt wird. [SBD+08].

2.4 Software Qualität

Laut ANSI-Norm (ANSI/ASQC A3-1978)[12], ISO 8402[13] und DIN 55350 (Teil 11)[14] ist Qualität *„die Gesamtheit von Eigenschaften und Merkmalen eines Produkts oder einer Tätigkeit, die sich auf deren Eignung zur Erfüllung gegebener Erfordernisse bezieht.“* [ZGK04, führt dies aus, Seite 142].
Bezieht man dies auf Software, so entsprächen in dieser Definition Produkte den Software-Produkten und gegebene Erfordernisse beispielsweise bestimmten Spezifikationen[15]. [ZGK04, Seite 140-141] erwähnt jedoch zusätzlich, dass Qualität für beteiligte Gruppen eine unterschiedliche Bedeutung aufweist und dadurch nicht eindeutig und kontextabhängig ist. [ZGK04, Seite 142] definiert Eigenschaften und Merkmale aus Benutzersicht einer Software, die Auswirkungen auf die Qualität dieser haben. Einige davon sind: Abdeckung der Funktionen, Stabilität, Korrektheit, Genauigkeit, Robustheit, Zeitbedarf (unter anderem Performance), Einfachheit und Einheitlichkeit. ISO 9126 beschreibt ebenfalls diese und darüber hinaus zusätzliche Qualitätskriterien. Einige dieser Qualitätskriterien werden durch die Behebung von Problemen in der Multi-User Interaktion bei Multitouch-Tischen beeinflusst und sollen durch die Anwendung der in dieser Arbeit vorgeschlagenen Heuristiken verbessert werden. Diese werden in den nächsten Absätzen definiert und näher vorgestellt:

Robustheit. Unter der Robustheit einer Software versteht man im Allgemeinen die Fähigkeit einer Software unter außergewöhnlichen Bedingungen

[12] American National Standards Institute.
[13] International Organization for Standardization. Die von der Organisation definierten Standards werden in diesem Kapitel ohne weitere Zitation verwendet.
[14] Deutsche Industrie-Norm.
[15] siehe dazu ISO 9126, welcher Software Qualität definiert.

- Bedingungen die nicht dem geplanten Programmfluss entsprechen - zu funktionieren. Die Robustheit beschäftigt sich daher mit der Zuverlässigkeit von Software. Software soll fehlerhaftes Verhalten erkennen und entsprechend darauf reagieren. In dieser Arbeit wird Robustheit jedoch konkreter, bezogen auf User Interfaces im Sinne des ISO 9241-110, verstanden. In diesem Standard wird die Robustheit als Fähigkeit des interaktiven Systems bezeichnet, sich dem Benutzer des Systems als robust beziehungsweise tolerant gegenüber Fehlern zu beweisen. Ein Beispiel für diese Robustheit in Bezug auf Interfaces basierend auf Multitouch wäre, bei Touch-Objekten ein größeres Ziel für die Berührung als eigentlich sichtbar ist zu definieren. Ist das Touch-Objekt beispielsweise ein Button, dann ist die Interaktionsfläche des Buttons eigentlich größer als der sichtbare Teil. Dieses Konzept wird als „Iceberg Tips“ bezeichnet und wird in Kapitel 6.1.1 näher erläutert. Setzt man dieses um, würde sich das System etwa toleranter gegenüber unpräziser Selektion des Benutzers erweisen.

Performance. Die Performance beschäftigt sich mit der Effizienz einer Software und bezieht sich auf die Schnelligkeit und das Zeitverhalten von Antwort- und Verarbeitungszeiten. Umgelegt auf die Erkennung von Multitouch-Gesten wird in dieser Arbeit einerseits die Flüssigkeit der Interaktion im Allgemeinen, beispielsweise die effiziente Erkennung von Gesten, und andererseits die generelle Flüssigkeit im Programmablauf verstanden.

Korrektheit. Die Korrektheit befasst sich mit der Funktionalität einer Software; also inwieweit die Software die geforderten Funktionen erfüllt und die richtigen Ergebnisse liefert. In dieser Arbeit wird dieses Kriterium wiederum in Bezug auf das Multitouch User Interface interpretiert: Werden alle Eingaben richtig von der Software interpretiert und können somit die Funktionen ausgeführt werden? Daher ist es notwendig die von Benutzern ausgeführten Gesten korrekt zu interpretieren. Dabei kann es zu Problemen kommen, die durch die Multi-User Anwendung hervorgerufen werden (mehr dazu in Kapitel 5.2.2).

Stabilität. Laut ISO 9126 versteht man unter der Stabilität von Software, die Wahrscheinlichkeit des Auftretens unerwarteter Auswirkungen von Veränderungen. In qualitativ hochwertiger Software sollte diese Wahrscheinlichkeit gering gehalten werden. In Bezug auf die Erkennung von Gesten, wird dieses Kriterium folgendermaßen interpretiert: Gesten sollen stabil und kontinuierlich über die Zeit hinweg korrekt erkannt werden. Wird außerdem eine zusätzliche Geste in das Repertoire aufgenommen, sollen die vorhandenen Gesten noch immer korrekt interpretiert werden können.

Diese Kriterien sind jedoch nicht als gänzlich voneinander unabhängig zu

betrachten. Wie [ZGK04, Seite 141] erwähnt sind *„verschiedene Qualitätskriterien [...] ineinander verwoben und können damit zu Konflikten führen“*. Dies bedeutet, dass die Verbesserung eines Qualitätskriteriums Auswirkungen auf das andere haben kann. Würde man beispielsweise eine bessere Performance erreichen wollen und stimmt deshalb die Software mit einem bestimmten Betriebssystem ab, so würde darunter die Anpassbarkeit der Software auf verschiedene Umgebungen beziehungsweise die Portabilität leiden. Plant man die Qualität hinsichtlich eines Kriteriums zu stärken - wie dies in der vorliegenden Arbeit der Fall ist -, sollten daher immer die Auswirkungen auf andere Kriterien überprüft werden. In dieser Arbeit wird dies im Rahmen einer Evaluierung der Heuristiken durchgeführt.

Kapitel 3

Methodik

Die Studie zu dieser Arbeit wurde in drei Phasen durchgeführt. Anhand dieser Phasen wird nachfolgend erläutert auf welche Methodiken zurückgegriffen wurde. Der theoretische Hintergrund der Methodiken wird in den darauf folgenden Unterkapiteln näher erläutert.

- **Phase I: Problemidentifikation**
  In dieser Phase wurde, nach einer eingehenden Literaturstudie, eine Evaluierung vorhandener Multitouch-Applikationen durchgeführt, um anhand dieser Interaktionsprobleme und kritische Designentscheidungen bei Multi-User Anwendung zu identifizieren. Im Rahmen dieser Phase wurde ein high-fidelity Protoyp einer Multiuser-Applikation (siehe Kapitel 7) für einen Multitouch-Tisch implementiert. Bei der Implementierung dieses Prototyps wurden die identifizierten Problembereiche und kritischen Designentscheidungen bewusst mangelhaft berücksichtigt.

  Die Applikation wurde durch Rapid Prototyping in Java im Rahmen des TUIO-Frameworks entwickelt. Die in [Saf09] vorgestellten Pattern bildeten dabei die Grundlage für das Design des User Interface.

- **Phase II: Entwicklung/Analyse von Algorithmen und Heuristiken**
  In der zweiten Phase wurden aufbauend auf die identifizierten Probleme Heuristiken und Algorithmen entwickelt. Diese sollten zur Verbesserung zum Beispiel der Robustheit der Applikation beitragen. Weiters wurden bestehende Heuristiken, welche bei der Literaturrecherche gefunden wurden, adaptiert. Die Algorithmen wurden auf den entwickelten Prototyp angewandt.

- **Phase III: Evaluierung der Heuristiken**
  In dieser Phase wurde im Rahmen eines User-Tests evaluiert. Es galt dabei die Sinnhaftigkeit des Einsatzes der Heuristiken zu ermitteln

beziehungsweise die Effektivität und das Ausmaß der Besserung in Erfahrung zu bringen. Weiters sollten durch die Evaluierung Einsatzgebiete definiert werden.

3.1 Prototyping

Prototypen sind Repräsentationen eines Design, die vor dem eigentlichen Produkt entwickelt werden, um wichtige Informationen sowohl für Designentscheidungen als auch für den Design-Prozess zu liefern. Daher ist Prototyping eine wichtige Disziplin bei der Erstellung von interaktiven Systemen. Konkret kann ein Prototyp etwa ein physisches Modell, ein „Stück" Software, eine Slide Show, ein Storyboard, ein Video oder auch ein Karton-Mockup darstellen. [BS00].
Wie man an diesen Beispielen erkennen kann, ist es möglich Prototypen nach der „Fidelity" einzuteilen - also in wie weit der Prototyp dem tatsächlichen User Interface in Form und Funktion entspricht. Typischerweise werden Low-Fidelity Prototypen in frühen Designphasen eingesetzt, da diese geringe Entwicklungskosten aufweisen. High-Fidelity Prototypen werden in späteren Phasen eingesetzt, um sich an das fertige Produkt anzunähern und ein größeres Maß an Interaktivität und Funktionalität gewährleisten zu können. [RSI96].
Ausgehend von dieser Beschreibung wurde in dieser Arbeit ein High-Fidelity Prototyp entwickelt, jedoch wurde auf Frameworks und Tools zurückgegriffen, um ein schnelles Prototyping (= *Rapid Prototyping*) zu gewährleisten.

3.2 Evaluierung - User Test

[Nie93, Seite 165] beschreibt User Tests folgendermaßen: *„User Testing with real users is the most fundamental usability method and is in some sense irreplaceable, since it provides direct information about how people use computers and what their exact problems are with the concrete interface being tested."*
Demnach versteht man unter einem User Test eine grundlegende Methode, bei der Probanden eingesetzt werden, um Software zu evaluieren. Exakter ausgedrückt, stellt bei einem User Test ein Testleiter eine Umgebung (= das Testsetting) zur Verfügung, in der mehrere Testbenutzer typische, zuvor definierte Arbeitsschritte durchführen. Dabei werden sie beobachtet und die Vorgehensweise beziehungsweise mögliche Fehler aufgezeichnet. Aus diesen Aufzeichnungen sollen im nächsten Schritt Schlüsse auf mögliche Mängel in der Software gezogen werden, um entsprechende Verbesserungsmaßnahmen zu definieren.

Das Ziel dieser Evaluierungen ist es, interaktive Systeme, also Systeme in

denen eine Mensch-Computer-Interaktion statt findet, hinsichtlich ihrer Benutzbarkeit und Gebrauchstauglichkeit zu analysieren, um Mängel und Problemgebiete zu erkennen. Die Ziele können von Evaluierung zu Evaluierung variieren und sollten zu Beginn eines jeden Tests definiert werden. In der vorliegenden Arbeit wurden User Tests in zwei Phasen durchgeführt. In der ersten Phase wurde ein informeller User Test eingesetzt, um an einem Prototyp Probleme bei der Benutzung eines Multitouch-Tisches durch mehrere Benutzer zu identifizieren. In der zweiten Phase wurde ein formeller, vergleichender Test eingesetzt, um die nach der Identifikation der Probleme statt gefundenen Verbesserung des Prototyps zu validieren. Die angewendete Methodik dieser beiden Tests kann Kapitel 5.1 (für den Test zur Problemidentifikation) sowie Kapitel 8.1 (für die Evaluierung der Heuristiken) entnommen werden.

3.3 Hardware: Multitouch-Tisch

In dieser Arbeit wurde ein Multitouch-Tisch basierend auf Diffuse Illumination eingesetzt. Dieser wurde vom INSO[16] für diese Arbeit zur Verfügung gestellt und besteht aus einem Bildprojektor, einer Kamera und und Infrarotlichtquellen, die im unteren Teil des Tisches verbaut sind. Der Projektor und die Kamera leuchten über einen Spiegel ebenfalls von unten auf die Tischplatte. Das Setting besteht weiters aus einem Rechner mit Mac OS X 10.6.4, einem Arbeitsspeicher von 4 GB RAM und einem Intel Core 2 Duo Prozessor mit 2,8 GHz. Der Beamer erreicht eine Bildschirmauflösung von 1024x768 Pixel, welche auf ein Display mit 1 m x 0,7 m projiziert wird. Abbildung 3.1 veranschaulicht das Setting.

Abbildung 3.1 – Verwendeter Multitouch-Tisch

[16]INSO steht für Industrial Software und ist eine Forschungsgruppe des Institus für Rechnergestützte Automation der Technischen Universität Wien.

3.4 Software: Eingesetzte Technologien

3.4.1 Überblick über die Architektur

Die nachfolgende Abbildung 3.2 soll einen ersten Überblick über die Architektur der Technologien geben. So werden die Gesten beziehungsweise die Interaktion über eine Kamera aufgezeichnet und sollen von der TUIO-Tracker Applikation erkannt werden. Über das TUIO-Protokoll werden diese Informationen an die TUIO-Client Applikation weitergegeben, um sie dort weiter zu verarbeiten. Über einen Projektor können danach über die Client-Applikation entsprechende Informationen ausgegeben werden.

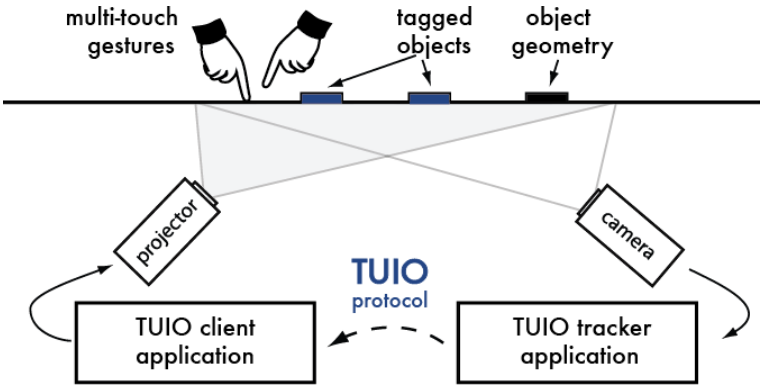


Abbildung 3.2 – Architektur von TUIO[17]

Die einzelnen Technologien werden in den nachfolgenden Kapiteln näher erläutert.

3.4.2 Protokoll: TUIO

Tangible User Interface over OSC[18] (TUIO) ist ein offenes Framework, welches ein Protokoll und eine API[19] für tangible Multitouch Umgebungen definiert. Das Protokoll erlaubt die Übermittlung einer abstrakten Beschreibung des Multitouch-Tisches, beispielsweise Touch-Events und Zustände der tangiblen Objekte. Dazu werden Daten der Tracker-Applikation kodiert und an jegliche Client-Applikationen, die die versendeten Daten dekodieren können, gesendet.

Die Implementierung von TUIO verwendet Open Sound Control (OSC) und folgt deren Syntax. OSC [Wri05] ist ein Protokoll, das für Audio Applikationen entwickelt wurde. Es soll geringe Latenz aufweisen und schnelles kodieren und dekodieren gewährleisten.

[17] nach http://www.tuio.org/images/diagram.png, zuletzt abgerufen am 16.01.2010.
[18] http://www.tuio.org/, zuletzt abgerufen am 16.01.2010.
[19] Application Programming Interface, deutsch: Programmierschnittstelle.

Üblicherweise wird das TUIO-Protokoll über UDP/IP transportiert, um signifikanten Overhead, wie dies etwa bei TCP der Fall wäre, zu vermeiden und die Latenz gering zu halten. TUIO ist in der Lage Touch-Objekte zu repräsentieren, welche vom System erkannt werden, indem die Position und Richtung des Objekts identifiziert wird. Dies geschieht über ein „`Cursor`-Objekt“, welches Attribute wie etwa die Position, die Richtung, die Geschwindigkeit und die Beschleunigung koppelt. Zwei Klassen von Nachrichten werden von TUIO definiert. In einer *Set*-Nachricht werden die Parameter des „`Cursor`-Objekts“ - also der Touch-Punkte - nach einer Zustandsänderung übermittelt. *Alive*-Nachrichten bilden die aktuellen Touch-Punkte ab. Wenn ein Touch-Punkt entfernt wird, wird eine *Alive*-Nachricht übermittelt. Die Client-Applikation kann daher aus *Set*- und *Alive*-Nachrichten folgern, wann ein Touch-Punkt hinzugefügt oder entfernt wird.
Um die Größe der UDP Pakete auszunutzen und eine geringe Latenz zu gewährleisten, werden *Set*-Messages gebündelt. Falls keine Veränderung im Zustand der Touch-Punkte stattfindet, werden in periodischen Abständen *Alive*-Nachrichten gesendet. Da bei UDP im Gegensatz zu TCP Nachrichten verloren gehen können, werden redundante Informationen übermittelt. So wird bei einer *Set*-Nachricht immer zusätzlich eine *Alive*-Nachricht gesendet. Wenn etwa eine *Alive*-Nachricht verloren geht, wird mit dem Hinzufügen oder Entfernen von Touch-Punkten gewartet, bis die nächste *Alive*-Nachricht empfangen wird. Bei Verlust einer *Set*-Nachricht wird genauso verfahren. So ist sichergestellt, dass der Verlust von Paketen nur zu Verzögerungen, jedoch nie zu Mehrdeutigkeiten führen kann. [KBBC06].

Wie bereits erwähnt, kann das Protokoll, da es durch OSC implementiert ist, in allen Plattformen, die dieses Protokoll unterstützen, verwendet werden. Derzeit gibt es beispielsweise Implementierungen für Java, C++, PureData, Max/MSP, Flash, Processing, C#, Quartz Composer und einige mehr[20].

3.4.3 TUIO Tracker

Als Implementierung des TUIO Trackers (siehe Abbildung 3.2)), dies ist die Software, welche für die Erkennung der sogenannten „Blobs“ - also der Berührungspunkte - zuständig ist, wird Community Core Vision (CCV)[21] in der Version 1.2 verwendet. Auf Basis der „Blobs“ werden die Koordinaten der Berührungspunkte an die Applikation mittels TUIO weitergeleitet. CCV ist wie auch TUIO und MT4J quelloffen und gibt auf Basis eines Video-Inputs Tracking-Daten, beispielsweise Koordinaten und Größe der Blobs, aus. Weiters werden Events erzeugt, etwa ob ein Finger bewegt wurde. CCV kann sich mit Applikationen, die TUIO verwenden, verbinden und unterstützt nicht nur das eingesetzte Diffuse Illumination, sondern auch

[20] http://www.tuio.org/?developer, zuletzt abgerufen am 16.01.2010.
[21] vormals tbeta, http://ccv.nuigroup.com/, zuletzt abgerufen am 16.01.2010.

Frustrated Total Internal Reflection.

3.4.4 Programmiersprache und Frameworks

Java und Eclipse. Für die Umsetzung der Client Applikation dient Java, da diese Programmiersprache sowohl von TUIO unterstützt wird als auch Frameworks zur Unterstützung von Rapid Prototyping existieren. Als IDE[22] wird Eclipse[23] eingesetzt.

MT4J. Für die Erstellung des Prototyps (entspricht der Client-Applikation in Abbildung 3.2) wurde als Rahmen das Framework MT4J[24] in der Version 0.81 verwendet. Es handelt sich dabei um eine quelloffene Plattform für Java, das schnelles Entwickeln grafikreicher Applikationen ermöglicht, da es die gebräuchlichsten Gesten, viele graphische Objekte, wie zum Beispiel Ellipsen oder Rechtecke und User Interface Komponenten, etwa Buttons oder Text Labels bereits definiert. MT4J unterstützt verschiedene Geräte als Input, wobei der Fokus auf Multitouch gerichtet ist. Aufgebaut ist es ähnlich wie das bekannte Java Swing Framework und ist daher flexibel, modular und komponentenbasiert strukturiert.

MT4J ist weiters - wie Swing - event-getrieben; dies bedeutet, dass die Applikation auf verschiedene Event-Typen reagiert, die während der Laufzeit, typischerweise vom Benutzer, generiert werden können. Man spricht daher von einer event-getriebenen Architektur, einem Softwarearchitektur-Muster das durch Events gesteuert wird. Unter einem Event wird dabei eine Zuständsveränderung, welche beispielsweise durch das Klicken eines Buttons getriggert wird, verstanden. In MT4J kann das Klicken eines Buttons etwa ein `TapEvent` hervorrufen, welches dann drei statische Zustände kennt: `BUTTON_CLICKED, BUTTON_DOWN, BUTTON_UP`. Weitere Objekte können mit einem `ActionListener` belegt werden, um auf Zustandsveränderungen zu reagieren.

MT4J ermöglicht weiters die Verwendung des eingesetzten TUIO-Protokolls und auch der Tracking-Software. Hinsichtlich der Multitouch-Gesten werden bereits standardmäßig das Tippen auf Objekte, das Ziehen von Objekten, das Drehen von Objekten mit zwei Fingern, das Verändern der Größe von Objekten mit zwei Fingern, das Gruppieren von Objekten mit einer Art „Lasso-Bewegung“ und das Zoomen mit zwei Fingern unterstützt[25]. Diese Gesten können erweitert und überschrieben werden.

[22] Integrated Development Environment, deutsch: Integrierte Entwicklungsumgebung

[23] http://www.eclipse.org/, zuletzt abgerufen am 16.01.2010.

[24] Multitouch for Java, http://www.mt4j.org, zuletzt abgerufen am 16.01.2010.

[25] http://www.mt4j.org/mediawiki/index.php/Multitouch_gestures, zuletzt abgerufen am 16.01.2010.

Kapitel 4

Related Work

Dieses Kapitel stellt existierende Ansätze zur Lösung von Problemen der Mehrbenutzer-Anwendung vor: Methoden, welche zusätzliche Hardware verwenden, sowie Methoden basierend auf speziellen Algorithmen. Für die einzelnen Ansätze werden ebenfalls Schwierigkeiten und Probleme in der Umsetzung diskutiert, die einerseits in den Arbeiten selbst aufgebracht werden und andererseits theoretische Überlegungen beinhalten.

4.1 Ansätze mit zusätzlicher Hardware

Bei der Studie verwandter Arbeiten zu der Thematik fiel auf, dass kaum heuristische oder algorithmische Lösungen gesucht werden, um die Problematik der Multi-User Verwendung eines Multitouch-Tisches zu optimieren. In der Literatur findet man hauptsächlich Verfahren, die zusätzliche Hardware benutzen, um zwischen den Benutzern zu differenzieren.

So benutzt beispielsweise [DL01] Sensoren, welche in der Touch-Oberfläche eingebaut sind, um die Benutzer über deren Stühle, welche eindeutige Signale übermitteln, mit deren Touch-Inputs zu koppeln. Das Signal wird demnach über den Benutzer übertragen, was als verlässlich gilt. Dadurch wird eine simultane Interaktion ermöglicht, bei der mehrere verschiedene Personen mit dem gleichen Gerät interagieren, ohne sich gegenseitig zu stören oder von Objekten einer anderen Person gestört zu werden. **DiamondTouch**, so der Name des Produkts, welches aus dem vorgestellten Paper hervorgegangen ist, wurde von den Mitsubishi Electronics Research Labs (MERL) entwickelt. Dabei wird eine Matrix an Sensoren/Antennen (in Diamantenform, deshalb der Name) benutzt, welche in einer isolierten Touch-Oberfläche eingebaut sind. Daher ist dieses Verfahren nicht mit optischen Multitouch-Lösungen möglich, sondern stellt eine Ausprägung des projezierten kapazitiven Verfahrens dar.
Die Sensoren des Tisches werden üblicherweise über die Stühle der Benutzer

gekoppelt. Falls die Benutzer nicht auf Stühlen sitzen sollten, wäre es möglich adäquate Sensoren in Bodenplatten, auf denen Benutzer stehen, einzubauen. DiamondTouch erfordert elektrische Isolierung zwischen den Stühlen oder Bodenplatten der Benutzer. Die Benutzer dürfen sich daher nicht berühren und nicht nahe aneinander stehen. Diese Anforderung der Verhinderung von interner Kopplung wird jedoch in [DL01, Seite 221] mit *„social norms of personal space“*, also mit „normalen sozialen Normen des persönlichen Raums“ beziehungsweise des Abstands zwischeneinander relativiert.
Betreffend der Anzahl der Antennen am Tisch wird ein 'full matrix' Pattern vorgeschlagen. Dies bedeutet, dass die Antennen in einer hohen Zahl, vergleichbar mit den einzelnen Pixeln eines Bildes, vorzufinden sind. Da die Implementierung dieses Pattern sehr komplex ist, wird angeführt, dass dies für die meisten Applikationen nicht notwendig sein wird, da beispielsweise nur einige große Buttons existieren könnten. Falls dieses „full matrix“ Pattern nicht eingesetzt wird, kann der Touch-Punkt via Signalstärke geschätzt werden.
Eine Anforderung an DiamondTouch war, dass das System von Objekten, die auf dem Tisch liegen gelassen werden, nicht beeinflusst wird. Dies wird durch das Design gewährleistet, jedoch ist es möglich Objekte speziell so zu entwickeln, dass sie vom System erkannt werden, was in tangiblen Umgebungen nützlich sein könnte.

Die kommerzielle Verfügbarkeit von DiamondTouch machte es notwendig, spezielle Frameworks und Toolkits für die Entwicklung von Software zu schaffen, obwohl der in 2001 vorgestellte Tisch bereits über eine API verfügte, die einige Gesten, wie das Drehen und Skalieren von Objekten mit zwei Fingern unterstützte. **DiamondSpin** [SVFR04] stellt ein solches Toolkit dar: Es ermöglicht effizientes Prototyping und das Experimentieren mit Interfaces speziell für Multi-User Anwendungen. Dabei wird eine Engine definiert, um zwischen dem kartesischen Koordinatensystem, das für Zeichentools wie etwa OpenGL benötigt wird, und Polar-Koordinaten, welche vor allem von Applikationen verwendet werden und bei denen die Orientierung der Finger entscheidend ist, zu übersetzen. Diese Orientierung ist deshalb wichtig, da die Benutzer an verschiedenen Seiten des Tisches sitzen/stehen können.

Probleme dieses Ansatzes. Bei der Analyse des Ansatzes, den DiamondTouch verfolgt, fällt auf, dass es sich um die teuerste Methodik in der Herstellung handelt (siehe Kapitel 2.3.3). [SBD$^+$08, Seite 3] stellt zusätzlich die Performance in Frage: *„Their performance is rather worse than many of the other approaches [...].“* Aufgrund der Notwendigkeit einer leitenden Verbindung zwischen Benutzer und Tisch ist der Einsatz von anderen, billigeren Multitouch-Verfahren - beispielsweise optischen Verfahren - nicht möglich.

Zusätzlich zu diesem finanziellen Aspekt, der den Einsatz des Tisches für

Rapid Prototyping nahezu unmöglich macht, wirkt es einschränkend, dass die Benutzer des Tisches auf Stühlen sitzen müssen. Auch der zusätzliche Vorschlag, die zur Identifikation notwendigen Sensoren im Boden zu verbauen, scheint keine optimale Lösung zu sein: Personen, die am Tisch arbeiten, müssten immer auf der gleichen Position stehen beziehungsweise müsste auch der Tisch selbst immer am gleichen Ort stehen. Somit wird die Flexibilität der Benutzer sowie die Flexibilität des Einsatzes des Tisches herabgesetzt. Ein weiteres Argument, das gegen das Verfahren von Diamond-Touch spricht, ist die Notwendigkeit der elektrischen Isolierung zwischen den Stühlen beziehungsweise den Bodenplatten der Benutzer. Etwa könnte es bei kollaborativen Tätigkeiten sehr wohl vorkommen, dass Personen in einem engeren Abstand zueinander stehen. All diese Einschränkungen könnten zu einer geringeren Akzeptanz des Tisches führen.

INTOI [BHH+07], entwickelt von der Fachhochschule Oberösterreich, stellt ein weiteres Verfahren dar, das zusätzliche Hardware benutzt um Benutzer zu identifizieren. Es kombiniert zwei Eingabe-Methoden und dadurch auch zwei Multitouch-Technologien: FTIR für die Finger- und Gesten-Tracking - also für die direkte Eingabe mit der Hand - und die von Anoto[26] entwickelte Technologie um Pen-Tracking zu ermöglichen, bei dem die Erkennung indirekt über einen Stift erfolgt. Die Technologie unterstützt die gleichzeitige Interaktion mit mehrere Stiften und es kann jeder Stift - also auch der Benutzer - eindeutig identifiziert werden. Da die Informationen der Stifte über Bluetooth gesendet werden, können mit einem einzelnen Bluetooth Dongle bis zu acht Benutzer gleichzeitig erfasst werden. Eine Benutzung von mehreren Bluetooth Dongles stelle jedoch kein Problem dar.
Für die Erkennung der Stifte wird eine spezielle Folie mit Rückprojektion benutzt. Weiters werden für die Erkennung auf die Folie in einem Abstand von 0.3mm kleine Punkte gedruckt, die aufgrund der Rückprojektion nicht mehr für den Benutzer sichtbar sind.

FLUX [LPB+09] erweitert dieses Verfahren und erlaubt zusätzlich den Einsatz unter unterschiedlichen Konfigurationen, da es sowohl als Entwurfstisch (in einem bestimmten Winkel gekippt), als Diskussionstisch (horizontal gekippt) und als Tafel (vertikal gekippt) eingesetzt werden kann.

Probleme dieses Ansatzes. Betrachtet man diese Methode Benutzer am Multitouch-Tisch zu identifizieren, so fällt auf, dass die Eindeutigkeit der User nur bei der Benutzung der Stifte gegeben ist. Es werden zwar mehrere Benutzer unterstützt, jedoch kann jeweils nur ein Stift pro Benutzer bedient werden. Dies führt dazu, dass die Interaktion auf Singletouch

[26] Anoto-Stifte sind Kugelschreiber mit einer eingebauten Infrarot-Kamera, die die Bewegung des Stiftes aufzeichnet. http://www.anoto.com/, zuletzt abgerufen am 19.01.2010.

pro User zurückgesetzt wird. Dadurch sind weder Gesten noch komplexere Interaktionen, wie beispielsweise das gleichzeitige Auswählen von mehreren Objekten bei Identifikation der Benutzer gewährleistet. Dies wird auch durch die Definition von Gesten durch [Saf09, Seite 2] untermauert, welcher eine Geste als physische Bewegung bezeichnet, die von einem digitalen System ohne Zuhilfenahme traditioneller Zeigegeräte - wie einer Computermaus oder sensorischen Stiften - interpretiert werden kann. Die Interaktion basiert daher nicht mehr auf Gesten, da die Verwendung eines Stiftes der Definition von Gesten widerspricht.
Weiters leidet die Flexibilität des Einsatzes, da zusätzliches Equipment zur Interaktion benötigt wird. Die Natürlichkeit der Interaktion ist zwar durch die Methaper des Stiftes, der auch sonst zum Schreiben eingesetzt wird, gegeben, jedoch geht die direkte Manipulation von Objekten verloren. Sogar [LPB+09, Seite 3214] beschreibt selbst: *„On the other side, in our observations, users immediately tried to touch the interactive surface with bare hands."* Dieses Ergebnis der Untersuchung geht konform mit [DL01, Seite 219]: *„Also, relying on a separate physical device keeps us from utilizing the natural human tendencies of reaching, touching and grasping."* Ebenso widerspricht dieses zusätzliche Gerät den von [DL01, Seite 220] definierten Anforderungen an eine Multi-User Umgebung eines Multitouch-Tisches (siehe Kapitel 2.1.4), da keine weiteren Geräte zur Interaktion benötigt werden sollten (zB Stifte, Kontakte die am Körper getragen werden).

Die in **[DDSP08]** vorgestellte Methodik zur Verbesserung der Multi-User-Interaktion an einem Multitouch-Tisch benötigt kein zusätzliches Gerät zur Interaktion, stellt jedoch ebenso zusätzliche Anforderungen an die Hardware. Verwendet wird ein Multitouch-Tisch basierend auf FTIR. Dabei werden zwei Probleme dieser Tische behoben: Zum einen ist jeder Berührungspunkt unabhängig. Algorithmen, die auf den Abstand zwischen mehreren Berührungspunkten basieren, können in diesem Fall verwendet werden um diese unabhängigen Events zusammenzufügen. Bei steigender Anzahl von Benutzern und damit einhergehender Steigerung der Komplexität der Interaktion, erhöht sich die Wahrscheinlichkeit von inkorrekten Gruppierungen. Es ist daher wichtig die verschiedenen Benutzer zu identifizieren. Das zweite Problem bezieht sich auf die Beleuchtung, - beispielsweise Tageslicht oder auch Fotografie mit Blitz - durch welche die Infrarot-Strahlen des optischen Multitouch-Tisches gestört werden.
Um die zwei Probleme von FTIR zu lösen, wird eine Kamera, die über dem Tisch befestigt ist, vorgeschlagen. Mit dieser Kamera können die Hände auf und über dem Tisch durch Unterschiede in den Hautfarben erkannt und somit die verschiedenen Benutzer separiert werden. Dadurch muss nicht mehr über Entfernungen der Touch-Punkte zwischen Benutzern differenziert werden.

Für die Identifikation der Hände werden dabei zwei Möglichkeiten vorgeschlagen. Die erste Methode reduziert die Dimensionen der verfügbaren Hautfarben und normalisiert diese. Bei der zweiten Methode werden nur die aufgezeichneten RGB-Farben der Kamera verwendet ohne etwaige Filter einzusetzen. Dadurch erscheinen, aufgrund der Helligkeit des Displays, die Hände als schwarze Regionen in dem Bild. Jede Methode der Identifikation der Hände hat ihre Vor- und Nachteile. Die erste Methode ermöglicht eine bessere Auflösung. Die korrekte Identifikation kann jedoch nur dann gewährleistet werden, wenn die Benutzer Kleidung mit kurzen Ärmeln tragen. Die zweite Methode verbessert dies, jedoch birgt sie Probleme, falls die Applikationen die dargestellten Farben gegenüber den Initialwerten verändert. Da die im Prototyp der Studie vorgestellte Methode zur Unterscheidung zwischen Benutzern nicht den Hautton als solchen verwendet - also die zweite Methode eingesetzt wird -, ist eine geringe Fehlerrate bei der Identifikation nur dann sichergestellt, wenn jeder Benutzer an einer anderen Seite des Tisches steht.
Bei diesem Prototyp werden Events durch die Kombination der Touch-Punkte und der Daten der zusätzlichen Kamera generiert und an die Applikation weiter geleitet. Mehrere zusammengehörende Touch-Punkte werden somit als Teile eines einzigen Events übermittelt. So erhöht das System die Robustheit des Trackings, da zwischen Benutzern differenziert werden kann. Zuvor vorherrschende Probleme - Touches von mehreren Benutzern, die vom System fehlerhaft interpretiert werden und damit ungewollte Reaktionen hervorrufen - können dadurch vermieden werden. Dadurch können Benutzer in gleichen Regionen der Oberfläche oder sogar mit gleichen Objekten interagieren. Dies führt laut [DDSP08, Seite 301] zu einer Steigerung der Effizienz.
Das Problem des Tageslichts wird durch die zusätzliche Kamera über dem Tisch verbessert. Damit existiert eine zusätzliche Quelle, die Eingaben erkennen kann. Ein Nebeneffekt davon ist, dass Gesten nun auch über dem Tisch möglich sind. Das bedeutet, dass der Tisch nicht mehr explizit berührt werden muss. Damit kann ein „hover-Effekt“ erzeugt werden, wie er bei der Interaktion mit der Maus, wenn sich der Cursor lediglich über einem Objekt befindet, bekannt ist.

Eine ähnliche Idee zu [DDSP08] verfolgt **[SG09]**. Es wird dabei ebenfalls eine zusätzliche Kamera über dem Tisch montiert. Diese dient dazu die Silhouetten der Hände zu erkennen, deren Konturen zu analysieren und damit die Hände der Benutzer zu identifizieren. Gleichzeitig werden, unabhängig davon, Touch-Eingaben durch Diffused Illumination erkannt. Diese beiden unabhängigen Daten werden danach in einem gemeinsamen Koordinaten-System analysiert, um zu entscheiden, welche Touch-Eingaben zu welcher "darüber liegenden"Hand gehören.

Probleme dieser Ansätze. Es ist zusätzliche, vom Tisch unabhängige Hardware notwendig und es werden zusätzliche Anforderungen an das Setting gestellt. Es kann mit dieser Methode der Tisch nicht mehr ohne Probleme in jedem beliebigen Raum eingesetzt werden, da eine Montage der Kamera über dem Tisch notwendig ist.
Des weiteren führt der Ansatz, wie bereits in der Beschreibung der Technologie erläutert, dazu, dass jeweils nur ein User pro Tischkante stehen kann [DDSP08, Seite 300], um eine Identifikation zu gewährleisten. Dies bedeutet durch die vier Kanten eines Tisches, dass die Anzahl der Teilnehmer auf vier Personen beschränkt ist und dass sich die Benutzer nicht mehr frei um den Tisch bewegen können.

Abgeleitete Anforderungen. Aus der vorgestellten Literatur zum Thema, kann geschlossen werden, dass die sechs Anforderung an Multitouch-Tische aus Kapitel 2.1.4 von [DL01, Seite 220] erweitert werden sollten. Die sechs Anforderungen:

- **Multitouch**
  Es müssen mehrere Finger gleichzeitig erkannt werden, das bedeutet es muss sich um einen Multitouch-Tisch handeln.
- **Identifizierung**
  Es muss erkannt werden, welcher Finger und welche Geste zu welchem Benutzer gehört.
- **Tolerant gegenüber weiteren Einflüssen**
  Objekte, die auf dem Tisch liegen gelassen werden, sollten die Umgebung nicht beeinflussen.
- **Dauerhaftigkeit**
  Wiederkehrende Kalibrierungen oder Reparaturen sollten nicht notwendig sein.
- **Keine zusätzlichen Belastungen**
  Es sollten keine weiteren Geräte zur Interaktion benötigt (zB Stifte, Kontakte die am Körper getragen werden) werden.
  [LPB+09] und [BHH+07] widersprechen dieser Anforderung.
- **Billige Herstellung**
  Die Herstellung des Geräts sollte preiswert sein.
  [DL01] widerspricht dieser Anforderung.

Der Argumentation aus den vorigen Absätzen folgend, können diese Anforderungen um nachstehenden Punkt erweitert werden:

- **Flexibilität der Benutzung**
  Die Benutzer sollten sich beliebig um den Tisch sammeln können. Dies involviert, dass die User stehen können, sitzen können, dass der Tisch in beliebigen Räumen eingesetzt werden kann und mehrere Personen an einer Seite des Tisches stehen/sitzen können.
  [DL01], [DDSP08] und [SG09] widersprechen dieser Anforderung.

4.2 Algorithmische Ansätze

Ein Ansatz, der keine zusätzliche Hardware verwendet, sondern es ermöglicht mithilfe eines Algorithmus zwischen Benutzern beziehungsweise zwischen Händen zu unterscheiden, wird in [DSA09] vorgestellt. Dieser Ansatz wird in diesem Kapitel beschrieben.
[DSA09] beschreibt, dass Multitouch-Umgebungen nicht nur Informationen über die Position von Touch-Punkten bieten, sondern auch Informationen über die Orientierung beziehungsweise der Richtung der Finger. Dabei wird versucht diese Touch-Punkte auf die zugehörige Hand abzubilden. Die Gestenerkennung und die User Interaktion soll verbessert werden, indem Einschränkungen, die bei der Kombination der Position der Touch-Punkte durch der Orientierung der Finger auftreten, verwendet werden.
Der Hauptfokus des Papers ist jedoch nicht die Behebung der Probleme der Multiuser-Interaktion, sondern die Unterscheidung zwischen der Interaktion mit nur einer Hand und der Interaktion mit mehreren Händen. [MH08] stellt beispielsweise fest, dass der Input mit einer Hand passend für das Bewegen, für das Verändern der Größe und für das Drehen von Objekten ist. Hingegen sind Gesten mit zwei Händen eher für Aufgaben, die sich mit der separaten Kontrolle von Punkten beschäftigen, wie das Selektieren einer Region, geeignet. Sie werden verwendet, um eine präzise Eingabe zu gewährleisten. Zukünftig sollen durch die Unterscheidung der Hände Gesten, die prinzipiell gleich sind, anders interpretiert werden. So soll zwischen der Ausführung einer Geste mit einer beziehungsweise mit zwei Händen differenziert werden. Beispielsweise sollen bei der Eingabe mit zwei Händen Möglichkeiten geboten werden, welche die Präzision der Eingabe erleichtern oder die Geschwindigkeit der Bewegung von Objekten anpassen.
Somit kann das Repertoire an Gesten um natürliche und verfeinerte Gesten erweitert werden. Oftmals bringt das Erweitern eines Repertoires an Gesten jedoch eine geringere Robustheit mit sich, weil sich neue Gesten von existierenden nur durch kleinere Abweichungen unterscheiden. In diesem Fall ist durch das Miteinbeziehen von Gesten mit zwei Händen keine signifikante Verschlechterung der Robustheit zu befürchten, da die Möglichkeit zwischen Gesten mit einer Hand und Gesten mit zwei Händen unterscheiden zu können, eine Klassifizierung der Eingabe zulässt.
Als Nebeneffekt dieses Ansatzes wird erwähnt, dass durch die Identifikation

der Hände Probleme bei der Interaktion von mehreren Benutzern mit einem Multitouch-Tisch behoben werden könnten.
[DSA09] behandelt nur die optischen Multitouch-Technologien FTIR und DI und geht von einer gegebenen Orientierung der Blobs durch die Multitouch-Umgebung aus.

Eigenschaften eines Touch-Punktes. Der vorgestellte Algorithmus bildet Touch-Punkte durch die Analyse der Orientierung der Touch-Punkte auf die Hände der Benutzer ab. Es wird der Ansatz verfolgt, die Hände über verschiedene Charakteristika unterschiedlicher wahrgenommener Finger, zu identifizieren. Diese Charakteristika ergeben sich durch die natürlichen Einschränkungen der menschlichen Hand. Diffuse Illumination würde weitere Eigenschaften bieten, da bei dieser Methode auch zusätzliche Konturen, wie die Ansätze des Fingers und der Hand, aufgezeichnet werden können. Für ein besseres Verständnis sind in Abbildung 4.1 die von der Multitouch-Umgebung aufgezeichneten Eigenschaften angeführt.

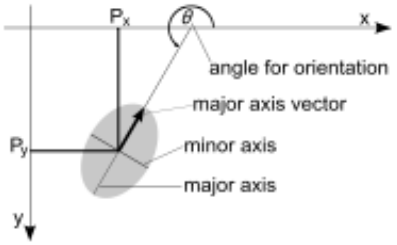


Abbildung 4.1 – Eigenschaften eines Touch-Punktes, [DSA09, Seite 103].

Touch-Punkte werden durch eine Form, die an eine Ellipse erinnert, mit einer Position P_x und P_y - dem Zentrum des Blobs - im Koordinatensystem dargestellt. Der Hauptachsen-Vektor (*major axis vector*) erlaubt es den Winkel der x-Achse und daher die Orientierung des Blobs zu berechnen. Die Bestimmung von verlässlichen Informationen über die Orientierung der Finger ist wesentlich, um Touch-Punkte auf die Hände abbilden zu können.

Eigenschaften mehrerer Touch-Punkte. Wenn man jeweils zwei Blobs betrachtet, ist es möglich, jeweils die Distanz zwischen den Zentren der Blobs und dem inneren Winkel der Orientierung zwischen den Blobs (*interior angle*) zu berechnen. Wie in Abbildung 4.2 ersichtlich, variiert die Orientierung bei der Betrachtung der Blobs einer ganzen Hand nur wenig aufgrund der menschlichen Anatomie. Zusätzlich kann der Abstand zwischen den Fingern nur einen maximalen Wert betragen.
Eine Ausnahme bildet der Daumen, da die Kombination eines Fingers mit dem Daumen sowohl einen höheren inneren Winkel als auch eine höhere Distanz ermöglicht.

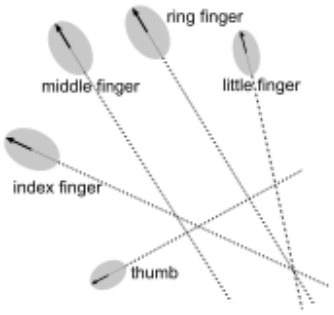


Abbildung 4.2 – Eigenschaften eines Sets an Touch-Punkten, [DSA09, Seite 104].

Parameter des Algorithmus. Abbildung 4.3 fasst die einzelnen Parameter zusammen.

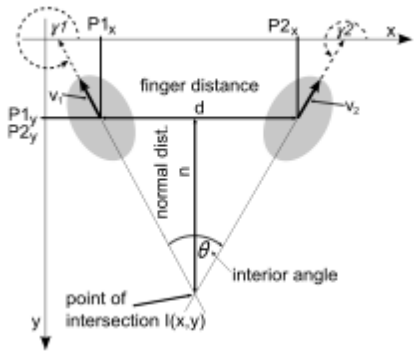


Abbildung 4.3 – Parameter der Heuristik, nach [DSA09, Seite 104].

Die beiden Zentrumspunkte $P1(x, y)$ und $P2(x, y)$ sowie die Orientierung als Winkel der x-Achse γ_1, γ_2 sind dabei üblicherweise gegeben. Dies ermöglicht die Berechnung der Distanz d zwischen den Zentrumspunkten, den Winkel θ der Orientierungslinien (*point of intersection*) des Kreuzungspunkts der Orientierungslinien $I(x, y)$, der Distanz n zwischen Kreuzungspunkt und Distanzkante d und die Berechnung der Gradienten-Vektoren v_1, v_2.

Bedingungen und Einschränkungen. Der Algorithmus definiert drei Bedingungen, die zur Identifikation der Hände beitragen.

- **Distanz und Nähe**
  Es wird eine maximale Distanz $D_{max} = 10,55"(= 26,797cm)$ definiert, die der weitest möglichen Entfernung zwischen dem Daumen und einem anderen Finger entspricht. Wenn d größer ist als D_{max}, dann werden zwei Touch-Punkte als von verschiedenen Händen stammend

erachtet. Zusätzlich wird ein Maximalwert $D_{adj} = 3,5" (= 8,89cm)$ für angrenzende Finger definiert.

- **Kreuzungspunkt und Angrenzung**
  Der Kreuzungspunkt der Orientierungslinien muss „hinter" den beiden Blobs sein. Dies kann mithilfe der Punkte $P1(x,y), P2(x,y)$ und $I(x,y)$ überprüft werden. Ein spezieller Aspekt dieser Bedingung sind parallele Kontaktpunkte, die keinen Kreuzungspunkt aufweisen. In diesen Fällen ist der innere Winkel sehr klein. Daher wird der Winkel $\gamma_{adj} = 45°$ definiert und die Distanz D_{adj} verwendet. Zwei Touchpunkte gehören zu angrenzenden Fingern, wenn γ kleiner als γ_{adj} und d kleiner als D_{adj} ist.

  Theoretisch können zwei Finger einer Hand auch einen Kreuzungspunkt „vor" den beiden Blobs haben, insbesondere wenn bestimmte Gesten ausgeführt werden. Beispielsweise bei der Ausführung einer Skalier- oder einer Grab-Geste[27] kann dies vorkommen, falls diese mit zwei Fingern einer Hand ausgeführt werden. Eine korrekte Klassifizierung kann in diesen Fällen nur durch die Analyse von zeitlichen Sequenzen sicher gestellt werden.

- **Relationen der Entfernungen**
  Die ersten beiden Bedingungen sind noch nicht ausreichend, da ein d für das gilt $Dadj < d < Dmax$ auch durch Finger von verschiedenen Händen gefunden werden kann. Abbildung 4.4 zeigt folgendes: Je weiter entfernt zwei zu analysierende Finger sind, desto größer ist γ und desto kleiner ist n. Daher kann zusätzlich ein maximaler Wert für n - $N_{max} = 14,06" (= 35,7124cm)$ - definiert werden.

In weiterer Folge wird durch Formel 4.1 (nach [DSA09, Seite 105]) die Entfernung n so skaliert, dass sie der Proportion von D_{max} mit N_{max} entspricht.

$$score = (D_{max} - d) - \frac{n \times D_{max}}{N_{max}} \tag{4.1}$$

Da d in inversem Verhältnis zu n steht, wird dieses Ergebnis von der Differenz zwischen D_{max} und d subtrahiert, um den Wert $score$ zu erhalten. Dieser Wert ist nur dann positiv, wenn zwei Touch-Punkte zur selben Hand gehören.

[27] Eine Geste, bei der zwei Finger zusammengezogen werden, wie dies beim Aufheben etwa eine Puzzle-Teils geschieht.

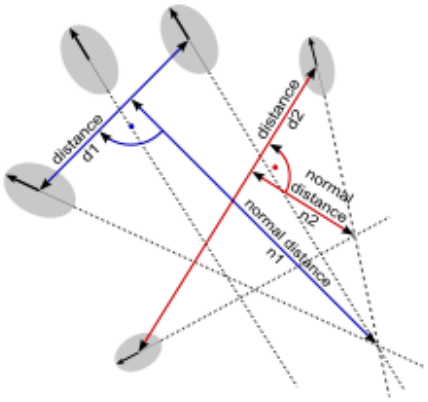


Abbildung 4.4 – Relation von d und n, [DSA09, Seite 105].

Die praktische Verwendung dieses Ansatzes zur Behebung der Probleme der Multi-User Interaktion wird in diesem Paper nicht weiter behandelt. Die theoretischen Überlegungen von [DSA09] sollen als Grundlage einer Heuristik (siehe Kapitel 6) in der vorliegenden Arbeit dienen.

Kapitel 5

Probleme bei Mehrbenutzer-Anwendung

Gesten an einem Multitouch-Tisch bestehen aus mehreren Touch-Punkten. Dies bedeutet, dass jeder Touch-Punkt ein eigenes, unabhängiges Ereignis darstellt und diese Punkte interpretiert werden müssen um Gesten zu erkennen. Betrachtet man die Benutzung eines Multitouch-Tisches durch mehrere Personen, so lässt sich feststellen, dass hierbei die Anzahl der Touch-Punkte steigt, mehrere Gesten und Interaktionen synchron erfasst und die Benutzer identifiziert werden müssen. Nur dadurch lassen sich Gesten korrekt und fehlerfrei interpretieren. Zusammengefasst führen Multi-User Anwendungen demnach zu einer höheren Komplexität. Sofern die Multi-User Eingabe nicht durch zusätzliche Hardware unterstützt wird, beispielsweise zur Erkennung der Benutzer mittels zusätzlichen Kameras, so ist die Folge der erhöhten Komplexität eine gesteigerte Fehlerrate beziehungsweise Fehleranfälligkeit. Welche konkreten Probleme zu dieser erhöhten Fehlerrate beitragen, wird in diesem Kapitel untersucht und erörtert.

5.1 Pilotstudie

Wenngleich in der verwandten Literatur zum Thema von Schwierigkeiten bei der Multi-User Interaktion berichtet wird - beispielsweise in [DDSP08] oder in [DL01] -, so werden keine genauen Probleme erwähnt. Es wird lediglich die Wichtigkeit der Identifikation der einzelnen Benutzer hervorgehoben. So sollen in dieser Arbeit, um eine möglichst fehler- und problemfreie Interaktion von mehreren Benutzern an einem Multitouch-Tisch gewährleisten zu können, konkrete Probleme zuerst identifiziert und definiert werden.

Ziel der Studie. Wie bereits in Kapitel 3 erwähnt, diente zur Identifikation dieser Probleme eine Pilotstudie ausgewählter, bereits existierender Applikationen für einen Multitouch-Tisch. Durch den Test sollten kritische

Designaspekte für Multi-User Applikationen identifiziert werden. Unter kritischen Designaspekten werden in der vorliegenden Arbeit solche Entscheidungen verstanden, die sowohl für das Interaktionsverhalten als auch für die Implementierung einer Multi-User Applikation als kritisch angesehen werden können. Diese zeigen daher Bereiche auf, in denen Probleme bei Multi-User Anwendung vermehrt auftreten. Daher wurden diese im erstellten Prototyp mangelhaft berücksichtigt beziehungsweise gegen diese Aspekte bewusst verstoßen.

Testsetting. Die Studie wurde mit einem Probanden in einem informellen Umfeld durchgeführt. Dabei wurde der bereits in Kapitel 3.3 vorgestellte Multitouch-Tisch verwendet. Der Proband hatte bereits Erfahrung mit Multitouch-Tischen beziehungsweise mit der Technologie Multitouch im Allgemeinen. Dies ermöglicht eine bessere Reproduktion der aufgetauchten Probleme, da diese exakter beschrieben werden können. Die Testperson wurde bei der Interaktion beobachtet, wobei eine aktive Beobachterrolle eingenommen wurde. Es wurde also aktiv am Geschehen teilgenommen, um einerseits eine Multi-User Situation zu simulieren und andererseits Probleme auf diese Weise von innen kennen zu lernen und bei Unklarheiten sofort eingreifen zu können. Weiters wurde dem Probanden Nahe gelegt Probleme und Schwierigkeiten sofort zu melden und laut auszusprechen, damit diese reproduziert und nachverfolgt werden können.

Hinsichtlich des Ablaufs des Tests wurden verschiedene Multitouch Applikationen nacheinander gestartet. Die Testperson wurde gebeten mit diesen zu interagieren beziehungsweise diese „auszuprobieren“ und damit zu „spielen”.

Evaluierte Anwendungen. Als Quelle für existierende Multitouch Anwendungen dienten die Beispiele der Implementierung des TUIO Trackers - Community Core Vision - der NUI-Group[28]. Zusätzlich wurde eine Demo-Applikation des benutzten Frameworks[29] - MT4J - verwendet. Nachstehend werden diese Applikationen vorgestellt:

- *Photo Demo:* Eine Foto- und Videodemonstration, um diese zu betrachten, skalieren und zu drehen.
- *Tank:* Ein Spiel für maximal vier Spieler bei dem Panzer gesteuert werden, die mit einer Schusswaffe gegeneinander antreten.

[28] Natural User Interface Group, Anwendung verfügbar unter http://nuigroup.com/log/comments/tuio_flash_clients_demos/, zuletzt abgerufen am 17.02.2010.

[29] verfügbar unter http://www.mt4j.org/mediawiki/index.php/Examples, zuletzt abgerufen am 17.02.2010.

- *Pegs:* Ein Casual Game[30] bei dem Bubbles mit Bällen zerstört werden sollen.
- *Multikey:* Ein Musikprogramm für zwei Benutzer, bei dem mit Klaviertasten Töne erzeugt werden können.
- *MusicalSquares:* Quadrate können erstellt und so beschleunigt werden, dass sie auf Punkte treffen die verschiedene Töne erzeugen (siehe Abbildung 5.1).
- *Tangram:* eine Art Puzzle-Applikation bei der der Umriss einer geometrischen Form mit einzelnen, kleineren Formen gefüllt werden muss. Um dies zu erreichen können die kleineren Formen mittels Gesten gedreht und vergrößert/verkleinert werden.
- *Modest Maps:* Eine Kartenapplikation des MT4J-Frameworks, bei der eine Weltkarte mit ortsbezogenen Fotos, welche von flickr geladen werden, erkundet werden kann.

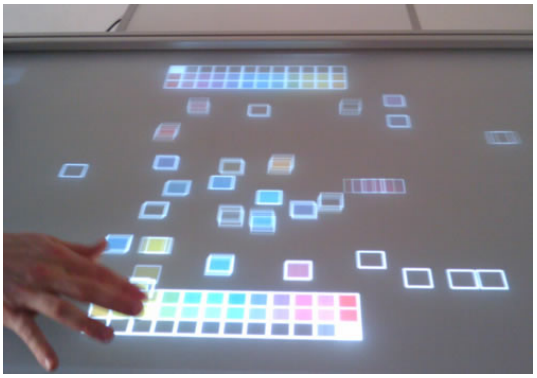


Abbildung 5.1 – Pilotstudie: Evaluierung von Musical Squares

5.2 Probleme und kritische Designaspekte

Als Ergebnis der Pilotstudie dienen die aufgetretenen Probleme und die daraus abgeleiteten kritischen Designaspekte.

[30] Casual Games sind Computerspiele, die kurzweilige Unterhaltung bieten sollen und daher über schnelle Zugänglichkeit, schnelle Erreichung von Erfolge und intuitive Interaktion verfügen.

5.2.1 Aufgetretene Probleme

Ein Problem, das immer wieder in diversen Applikationen auftrat, war das Skalieren des Hintergrunds, also das Verändern des Zoom-Faktors der Applikation. Dieses Interaktionsverhalten ist bei sämtlichen evaluierten Applikationen als Geste implementiert. Dabei berührt der Benutzer mit zwei Fingern die Oberfläche und zieht diese zusammen, um den Zoom-Faktor zu verkleinern, oder auseinander, um den Zoom-Faktor zu vergrößern. Diese Geste kann bei allen Applikationen sowohl mit zwei Fingern einer einzelnen Hand als auch mit zwei verschiedenen Händen ausgeführt werden.
Das Problem bestand darin, dass Berührungspunkte unbeabsichtigt erzeugt wurden, indem sich beispielsweise die Personen mit der zweiten Hand am Tisch abstützten. Treten nun zwei Berührungspunkte gleichzeitig am Hintergrund auf und bewegen sich diese, führt dies bei einigen evaluierten Anwendungen (*Musical Squares, Photo Demo, Multikey, Modest Maps*) sofort zur Interpretation der beiden Berührungspunkte als Vergrößerungs- beziehungsweise Verkleinerungs-Geste, welche die Auflösung des Hintergrundes veränderte. Bereits eine minimale Bewegung einer dieser zwei Berührungspunkte konnte dieses Verhalten hervorrufen. Dies führte dazu, dass ein weiteres Arbeiten erst wieder nach einem erneuten, dieses Mal nach einem gewollten, Skalieren möglich war. Das Problem wurde jedoch nicht nur von einem Probanden alleine hervorgerufen, sondern auch von mehreren Personen gemeinsam. Wenn gleichzeitig zwei Berührungspunkte beispielsweise an zwei gegenüberliegenden Kanten auftraten, wurde die Auflösung des Hintergrunds verändert. Dieses Problem trat auf, obwohl die beiden Berührungspunkte in einer großen Entfernung - nahezu dem gesamten Screen - zueinander lagen und von zwei verschiedenen Benutzern erzeugt wurden.

Ein weiteres Problem, welches mit dem Problem des Skalierens des Hintergrundes verwandt ist, ruft das „Schwenken der Kamera“, also das „Scrollen auf dem Hintergrund“, hervor. Das Schwenken der Kamera ist in den evaluierten Anwendungen einheitlich als Geste mit einem Finger, der den Hintergrund berührt und sich dort bewegt, implementiert.
Die Problematik, die sich dabei ergab, ist, dass eine Person den Hintergrund unbeabsichtigt berührte und bei einer kleineren Bewegung bereits die gesamte Sicht auf den aktuellen Screen verändert wurde. Dadurch wurden die anderen Teilnehmer in ihrer Interaktion unterbrochen. Der Ausgangszustand musste vor dem weiteren Arbeiten erst wieder hergestellt werden. Dieses Problem wurde oftmals unbeabsichtigt hervorgerufen, als sich etwa der Proband mit einer Hand am Tisch abstützte oder Objekte nicht präzise genug selektiert wurden. Beispielsweise fiel bei der Beobachtung des Probanden auf, dass dieser Objekte selektieren und verschieben wollte. Durch eine zu geringe Präzisierung dieser Selektion wurden Berührungspunkte am Hintergrund erzeugt, die keinem Objekt direkt zugeordnet werden konnten.

Dadurch nahmen die betroffenen Applikationen an, dass es sich um eine Pan-Geste handelt. Aufgetreten ist diese Problematik wiederum in den Anwendungen *Musical Squares, Photo Demo, Multikey, Modest Maps.*

Zusätzlich ist in der Anwendung *Photo Demo* das Schwenken der Kamera nicht nur mit der Entfernung der Bewegung des Fingers, sondern auch mit der Geschwindigkeit der Bewegung des Fingers behaftet. Ein schnellere Bewegung führte daher zu einem größeren Schwenk als eine langsamere Bewegung. Diese Geste wird etwa in [Saf09, Seite 60] beschrieben: *„[...] Based on the speed of the fling, the movement of the object or scrolling of the screen will continue after the gesture is clomplete, slowing to a gentle stop.“* Es wird im Gegensatz zu der Verwendung in dieser Applikation jedoch dazu geraten, diese Geste hauptsächlich einzusetzen, um lange Listen zu durchblättern. Durch die, in Bezug auf das vorgeschlagene Einsatzgebiet, inadäquate Verwendung dieser Geste, fiel die Veränderung des Hintergrunds bei unbeabsichtigten Berührungspunkten so signifikant aus, dass die Wiederherstellung des Ausgangszustandes einige Zeit in Anspruch nahm.

Betrachtet man die beiden Probleme der vorigen Absätze, so kann man eine Gemeinsamkeit erkennen: In beiden Problemen wird unbeabsichtigt die aktuelle Sicht auf die Applikation verändert. Bei Multi-User Anwendungen würde jedoch eine Pan- oder eine Zoom-Geste eine Kooperation sämtlicher beteiligter Benutzer benötigen: Wird die aktuelle Sicht auf die Applikation verändert, betrifft dies alle Benutzer. [RMRS$^+$04] beschreibt, dass sich in kollaborativen Tätigkeiten auch Phasen der individuellen Arbeit zeigen (siehe dazu auch Kapitel 2.2): *„Group work frequently involves transitions between periods of active collaboration and periods of individual activity.“* [RMRS$^+$04, Seite 1441]. Ist ein Benutzer gerade mit einer individuellen Tätigkeit beschäftigt und führt ein anderer Benutzer etwa eine Pan-Geste aus, so wird der andere Benutzer gestört und ein weiteres Arbeiten ist nicht möglich. Eine Verwendung einer solchen Geste macht daher nur in kollaborativen Phasen Sinn und benötigt, um keine Konflikte hervorzurufen, die Kooperation der einzelnen Benutzern.

Bei der Ausführung von Gesten fiel auf, dass diese nicht immer exakt so ausführt wurden, wie diese implementiert waren: Der Proband wurde zuvor nicht aufgeklärt wie Gesten auszuführen sind. Betrachtet man beispielsweise die Skaliergeste bei Objekten, welche in der Anwendung *Photo Demo* implementiert ist, so wurde ein zusätzlicher Finger hinzu genommen. Die Geste ist jedoch grundsätzlich so implementiert, dass sich zwei Finger entgegengesetzt auseinander bewegen. Dies führte zu einem unkontrollierten Verhalten der Applikation, welches nicht näher ergründet werden konnte. So verschwand das skalierte Objekt von der aktuellen Stelle und tauchte nach Entfernung der Berührungspunkte an einer anderen Stelle wieder auf. In einer robus-

ten Applikation, sollte eine „freiere“ Ausführung einer Geste nicht zu einem unkontrolliertem Interaktionsverhalten führen. Da auch bei einer exakten Ausführung der angesprochenen Geste ein zusätzlicher Berührungspunkt durch einen anderen Benutzer erzeugt werden könnte, führt dies zu einem weiteren Problem bei der Multi-User-Anwendung.

Während der Ausführung von Gesten zeigten sich weitere Probleme, welche sich mit eben einer solchen Störung der Ausführung einer Geste befassen. Diese Probleme werden ausgelöst, wenn während der Interaktion eines Benutzers, der gerade aktiv eine Geste ausführt, ein zweiter Benutzer hinzukommt, der ebenfalls mit dem gleichen Objekt interagieren möchte. Dieser erzeugt einen zusätzlichen Berührungspunkt auf dem bereits aktiven Objekt und ruft dadurch eine „Störung“ der Geste hervor. Zurückführen kann man diese Problematik auf fehlende Koordination zwischen den einzelnen Benutzern. Dieses Problem wurde ersichtlich, als eine Interaktion in einer geringen räumlichen Entfernung zueinander statt fand.

Abhängig von der Art der Geste nahm dieses Probleme unterschiedliche Ausprägungen an. So führte das gleichzeitige „Ziehen“ eines Objekts (siehe Abbildung 2.1-1b) durch zwei Benutzern, die sich am Tisch gegenüber standen, dazu, dass sich das Objekt vergrößerte. Es wurde also das Ziehen, um ein Objekt zu sich zu bewegen, als Skalier-Geste interpretiert.

Dies deckt sich mit den Beobachtungen von [PKS+08]. In dieser Studie wurde ein großes Multitouch-Display in einer Fußgängerzone in Helsinki, Finnland installiert, um die Interaktion der Benutzer zu beobachten. Durch den Einsatz im öffentlichen Raum offenbarten sich einige Phänomene, die sich auch bei der in dieser Arbeit betrachteten Multi-User-Umgebung ergeben: Parallele Interaktion, Teamarbeit und Situationen der Über- und Weitergabe von Objekten. Der Fokus der Beobachtung lag dabei auf der sozialen Interaktion mit der Multitouch-Umgebung und auf der Untersuchung der Methoden, wie die Benutzer zusammenarbeiten. 72% der Benutzung des Multitouch-Displays fand in Multi-User-Situationen statt. [PKS+08] beschreibt dabei auch Konflikte, die während dieser Situationen der Interaktion von mehreren Benutzern auftraten: Fotos wurden irrtümlicherweise vergrößert, wenn zwei verschiedene Benutzer gleichzeitig versuchten das Foto zu sich zu ziehen. Als Grund wurden Probleme genannt, die Aktionen eines Benutzers mit jenen der anderen Benutzer zu koordinieren. *„It's mine, don't touch“* wurde in einer ähnlichen Situation von einem Teilnehmer dieser Studie geäußert. [PKS+08, Seite 1290].

Abbildung 5.2 illustriert dieses Problem, bei welcher zwei verschiedene Personen versuchen ein Foto durch eine „Zieh-Geste“ zu sich zu bewegen und dabei die Koordination der beiden Benutzer mangelhaft ist.

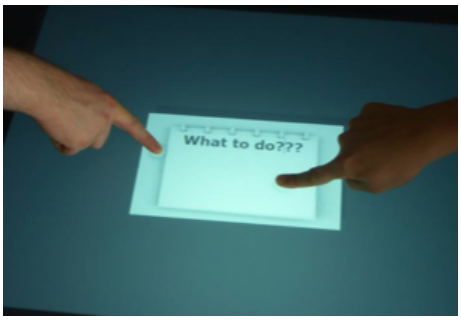


Abbildung 5.2 – Zwei verschiedene Benutzer mit fehlender Koordination, [DSA09, Seite 102]

Eine weitere Ausprägung dieser Problematik konnte bei der Evaluierung der Anwendungen *Tankgram* und *Photo Demo* beobachtet werden. So führte ein zusätzlicher Berührungspunkt eines anderen Benutzers beim Vergrößern/ Verkleinern eines Bildes (siehe Abbildung 2.1-1-d) beziehungsweise einer geometrischen Form dazu, dass die Geste zur Skalierung abgebrochen und statt dessen die Position des aktiven Objekts am Screen scheinbar zufällig verändert wurde. Ein vergleichbarer Konflikt trat ebenso beim Drehen eines Objekts auf. Es handelt sich daher um eine mangelnde Robustheit der Eingabeverarbeitung des Programms und nicht etwa, wie bei anderen aufgezeigten Problemen, um eine Fehlinterpretation der Absichten der Benutzer. In den beiden Applikationen ist das Drehen so implementiert, dass ein Benutzer mit zwei Fingern ein Objekt berührt und einen oder beide Finger dreht (siehe Abbildung 2.1-1-c). Ein zusätzlicher Berührungspunkt auf dem bereits aktiven Objekt führte dazu, dass sich das Objekt ungewollter Weise vergrößerte und somit die eigentliche Geste nicht mehr korrekt interpretiert wurde.

In der Anwendung *Photo Demo* wurde ein weiteres Problem beobachtet: Ein Benutzer vergrößerte mehrere Male unbeabsichtigt Fotos so, dass diese die anderen Benutzer an der weiteren Interaktion hinderten, da das Foto sämtliche Arbeitsbereiche überdeckte. Das Objekt musste, bevor weiter gearbeitet werden konnte, wieder verkleinert werden.
Dies beobachtet auch [PKS+08] und führt als Problem das Fehlen von Grenzen für jeden Benutzer beziehungsweise das Fehlen von Grenzen hinsichtlich der Vergrößerung von Objekten an. Diese Problematik wurde bei den interviewten Personen von [PKS+08] als am störendsten empfunden.

Ein weiteres Problem, welches sich in der Studie zeigte, stellt das Erweitern des Repertoires an Gesten dar. Je mehr Gesten in einer Applikation imple-

mentiert waren, desto weniger eindeutig waren deren Interpretationen. Betrachten wir beispielsweise eine Applikation, die Skalier-Gesten, wie diese in Kapitel 2.1.3 vorgestellt wurde, erlaubt: Ein Objekt kann vergrößert werden indem zwei Finger voneinander weg bewegt werden und dabei das Objekt berühren. Würde man nun etwa zusätzlich eine Rotier-Geste implementieren (ebenfalls in Kapitel 2.1.3 angeführt, zwei ein Objekt berührende Finger drehen sich), so können diese beiden Gesten nicht mehr exakt unabhängig voneinander ausgeführt werden. Dreht man beispielsweise zwei Finger nach rechts und berührt dabei ein Objekt beziehungsweise die Touch-Oberfläche, so bewegen sich unbewusst die beiden Finger zueinander oder auch voneinander weg: Ein Drehen in Form des Zeichnens eines exakten Kreises ist mit den bloßen Fingern nicht möglich. Dadurch werden beide Gesten gleichzeitig wahrgenommen und auch ausgeführt. Ist dieses Verhalten nicht erwünscht, leidet die Robustheit der Applikation, da die beiden Gesten nicht mehr eindeutig interpretiert werden können.

Bei der Evaluierung der Anwendung *Pegs*, ein Casual Game, bei dem die Benutzer mit dem Finger Bälle zerstören und Punkte dafür bekommen, gab es des Weiteren Anmerkungen des Probanden: Es wäre interessanter wenn jeder Benutzer separat Punkte bekommen würde und daher eine Art Wettkampf entstünde. Um dies zu ermöglichen müsste jedoch die Möglichkeit bestehen, Benutzer zu identifizieren.

5.2.2 Zusammenfassung der Probleme

Zusammengefasst konnten die folgenden Probleme identifiziert werden. Die Probleme wurden dabei nach der erforschten Ursache gruppiert:

(1) **Unbeabsichtigt, gemeinsam getriggerte Gesten**

   (a) Durch unbeabsichtigte Berührungspunkte mehrerer Benutzer verändert sich der Zoom-Faktor der gesamten Applikation

   (b) Gleichzeitiges Ziehen eines Objekts führt zu einer Skalier-Geste

(2) **Unbeabsichtigt, alleine getriggerte Gesten**

   (a) Zu stark vergrößerte Objekte stören die anderen Benutzer

   (b) Unbeabsichtigt ausgeführtes Schwenken der Kamera stört die anderen Benutzer (Verbindung der Geste mit der Geschwindigkeit der Bewegung des Fingers verschlechtert dies zusätzlich)

(3) **Störung der Gesten durch andere Benutzer - Räumlich enges Arbeiten der Benutzer - Fehlen von Grenzen der Arbeitsbereiche**

(a) Zusätzlicher Berührungspunkt bei Rotier-Geste führt zu unkontrolliertem Verhalten

(b) Zusätzlicher Berührungspunkt bei Skalier-Geste führt zu unkontrolliertem Verhalten

(c) Zusätzlicher Berührungspunkt beim Ziehen eines Objekts führt zu einer Skalier-Geste

(d) Zu stark vergrößerte Objekte stören die anderen Benutzer

(4) **Fehlende Identifikation der Benutzer**

(a) Einschränkungen bei Casual Games; bestimmte Spiele sind nur durch vorherige Identifikation der Benutzer möglich

(5) **Erweitertes Gesten-Repertoire**

(a) Zusätzliche Gesten verringern die Robustheit der Applikation

5.2.3 Kritische Designaspekte

Aus den identifizierten Problemen der Multi-User Anwendung werden in diesem Kapitel kritische Designaspekte abgeleitet. Entscheidet man sich für den Einsatz der nachfolgend angeführten Möglichkeiten in einer Applikation, so kann dies zu den im vorigen Kapitel erläuterten Problemen bei der Interaktion von mehreren Benutzern führen. Entwirft man eine Multi-User Applikation, sollte man sich daher dieser bewusst sein.

- **Schwenken der Kamera, Scrollen am Hintergrund**
  Wird dies in einer Applikation erlaubt und unterscheidet sich diese Geste nicht signifikant von anderen, können dadurch Probleme bei der Multi-User Anwendung hervorgerufen werden.

- **Zoom-Faktor der Applikation verändern**
  Wird dies mit einer Geste, wie diese im vorigen Kapitel beschrieben ist, in einer Applikation eingesetzt und werden keine Einschränkungen hinsichtlich der Entfernung zwischen den beiden Berührungspunkte der Geste getroffen, können dadurch Probleme bei der Multi-User Anwendung hervorgerufen werden.

- **Vergrößern/Verkleinern von Objekten**
  Wird eine Geste implementiert, um Objekte vergrößern/verkleinern zu können, und weist diese eine geringe Robustheit sowie keine Schranken hinsichtlich der Möglichkeit der Vergrößerung auf, kann dies zu Problemen in der Arbeit auf einem gemeinsamen Medium führen.

- **Kritische Gesten für Störungen**
  Jegliche Gesten können bei der Störung durch einen zusätzlichen Berührungspunkt eines anderen Benutzers zu Problemen führen. Beispiele, die in der Pilotstudie Probleme hervor riefen, sind die Skalier-Geste, die Rotier-Geste und das Ziehen um Objekte zu bewegen (mehr zu diesen Gesten in Kapitel 2.1.3).

- **Zusätzliche Gesten**
  Jede zusätzliche Geste in einer Applikation, kann zu einer geringeren Robustheit der Applikation führen: Zusätzliche Gesten unterscheiden sich oftmals nur geringfügig von existierenden.

Kapitel 6

Heuristiken und Algorithmen

Heuristiken werden in schlecht strukturierten Problembereichen eingesetzt und werden in dieser Arbeit als „Daumenregeln“ betrachtet, deren Anwendung im Allgemeinen zu einer guten Lösung eines Problems führt. Ich orientiere mich dabei an der Definition der Interaction Design Enzyklopädie[31], welche Heuristiken als *„rules of thumb for reasoning, a simplification, or educated guess that reduces or limits the search for solutions in domains that are difficult and poorly understood“* definiert.

Die Heuristiken dieser Arbeit sollen daher Prinzipien darstellen: Folgt man diesen Prinzipien, soll die Qualität von Multi-User Applikationen in den Ausprägungen Korrektheit, Performance, Stabilität und Robustheit gesteigert werden. Die durchgeführte Evaluierung der Heuristiken soll dabei zeigen, ob eine Steigerung dieser Qualitäts-Kriterien erreicht werden kann.

Ähnlich verhält es sich mit dem Begriff „Design Pattern“, der ebenfalls für adäquate Lösungsvorschläge für häufige auftretende Probleme einer spezifischen Domäne steht: *„In essence, patterns are structural and behavioral features that improve the 'habitability' of something - a user interface, a web site, an object-oriented program, or even a building. They make things easier to understand or more beautiful; they make tools more useful and usable. As such, patterns can be a description of best practices within a given design domain. They capture common solutions to design tensions [...]“*. [Tid06, Seite xiii]. Wie die Definition von Tidwell zeigt, überschneiden sich die beiden Begriffe - in dem Wortsinn, wie sie in dieser Arbeit gebraucht werden - weitgehend. Eine eindeutige Abgrenzung und Unterscheidung ist deshalb nicht sinnvoll. In der vorliegenden Arbeit wird zur besseren Stringenz und Lesbarkeit stets auf den Begriff der Heurisitik zurückgegriffen.

[31] Enzyklopädie Interaction Design: http://www.interaction-design.org/encyclopedia/heuristics_and_heuristic_evaluation.html, zuletzt abgerufen am 23.03.2010.

6.1 Bestehende Ansätze für Heuristiken

Dieses Kapitel stellt bestehende Ansätze, welche im nachfolgenden Kapitel 6.2 in den Heuristiken eingesetzt werden, vor.

6.1.1 Größe von Touch-Objekten

Iceberg Tips. „Iceberg Tips“[32] [Saf09, Seite 42-44] adressieren die Problematik zu kleiner Touch-Objekte: Objekte können bei Multitouch-Interfaces auf Grund der Größe des Fingers nicht so präzise ausgewählt werden wie bei Interfaces basierend auf einer Maus. Dies bedeutet, dass bei Multitouch User Interfaces, Objekte groß genug für die menschlichen Finger designed werden müssen. Um eine exakte Selektion auch bei kleineren Elementen zu ermöglichen, schlägt [Saf09, Seite 43] daher so genannte „Iceberg Tips“ vor, bei welchen Objekte ein tatsächlich größeres empfindliches Touch-Ziel aufweisen als für den Benutzer sichtbar ist. Wie Abbildung 6.1 veranschaulicht, ist das Ziel für die Selektion beispielsweise eines Buttons größer als das eigentlich visuell dargestellte Objekt selbst.

Abbildung 6.1 – Iceberg Tips in Form zweier Buttons. Die strichlierte Linie stellt die unsichtbare Grenze des Touch-Objekts dar, adaptiert nach [Saf09, Seite 43].

Wie reale Eisberge, die sich größtenteils unter der Wasseroberfläche befinden, ist das Objekt ebenfalls größer als sein sichtbare Teil. Diese Vergrößerung der Objekte, erfordert daher größere Abstände zwischen benachbarten Elementen.

Diese Methodik ist jedoch nicht nur bei Single-User Anwendung, wie dies in [Saf09] nahe gelegt wird, von Bedeutung. Betrachtet man nochmals die Konflikte bei Multi-User Anwendung aus Kapitel 5.2, so kann diese Heuristik eingesetzt werden, um die Probleme der zu wenig präzisen Selektion zu beheben: Die Probanden führten unbeabsichtigt Gesten am Hintergrund aus, obwohl sie ein Objekt selektieren wollten und störten dadurch die anderen Benutzer. Implementiert man Iceberg Tips und definiert man damit größere Flächen für die Selektion von Objekten, treten unbeabsichtigte Berührungspunkte nicht mehr so oft auf. Durch das Fehlen dieser

[32] deutsch: Eisberg Tips.

Berührungspunkte können diese nicht mehr als Gesten, die den Hintergrund und damit die gesamte Arbeitsfläche verändern, fehl interpretiert werden.

Anpassungsfähige Touch-Objekte. Bei adpativen Touch-Objekten werden Algorithmen verwendet, um die Wahrscheinlichkeiten der als nächstes ausgeführten Aktionen, etwa der Selektion eines Buttons, zu berechnen. Je größer die Wahrscheinlichkeit der Interaktion mit einem Objekt, desto größer wird der nicht sichtbare Bereich eines Iceberg Tips ausgelegt. Typische Algorithmen dieser Form existieren beispielsweise für eingeblendete Tastaturen. Diese können über eine Art Wörterbuch entscheiden, welche Wörter voraussichtlich eingegeben werden, beziehungsweise welche Buchstaben auf andere folgen. [Saf09, Seite 44].
Abbildung 6.2 illustriert dies und zeigt, dass die Bestätigung einer Aktion wahrscheinlicher sein kann, als deren Abbruch.

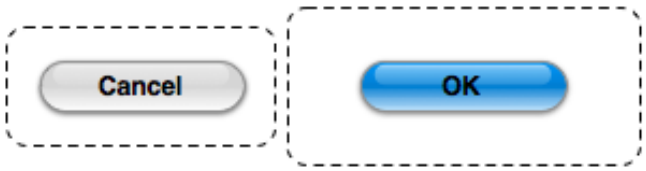


Abbildung 6.2 – Iceberg Tips mit entsprechend der Wahrscheinlichkeit einer Selektion angepasster Größe

Ein ähnlicher Ansatz wird in [BGBL04] durch das Konzept *„Semandic Pointing“* beschrieben: Die Erfassung der klickbaren Ziele in traditionellen graphischen Benutzerschnittstellen soll verbessert werden. Erreicht wird dies durch das Anpassen der Größe der Ziele auf zwei Dimensionen. Es wird sowohl die Größe eines Objekts aufgrund dessen Wichtigkeit für die Interaktion als auch aufgrund der Wichtigkeit dessen Inhalts verändert. Demnach wird folgende Hypothese aufgestellt: *„the difficulty of a pointing task is not directly linked to the on-screen representation of the task, but to the actual difficulty of the movement performed in the physical world to accomplish it.“* [BGBL04, Seite 521].
Wie in den vorigen Absätzen vorgeschlagen, impliziert [BGBL04, Seite 524] ebenfalls, dass die Wichtigkeit erhöht werden soll, falls die Wahrscheinlichkeit der Selektion höher ist als eine andere: *„the importance can be proportional to the probability of being selected.“*

Tatsächliche Größe der Touch-Objekte. Diese Heuristik kann, wie auch die zuvor erwähnten Iceberg Tips, eingesetzt werden um unpräzise Se-

lektionen bei Multitouch zu verhindern. [Saf09, Seite 42] beschäftigt sich dabei mit der tatsächlich notwendigen Größe von Touch-Objekten, damit diese selektiert beziehungsweise mit diesen interagiert werden kann. In [Saf09, Seite 42] wird eine Heuristik vorgeschlagen, welche die Größe der Touch-Objekte abhängig von der durchschnittlichen Größe eines Fingerballens betrachtet: Ein Touch-Objekt sollte demnach ein mindestens $0,4 \times 0,4\ Zoll$ ($1 \times 1\ cm$) großes Quadrat darstellen. Zusätzlich wird angeführt, dass die Auflösung in Pixel des Screens mit einbezogen werden muss, um die tatsächliche Größe zu erhalten. Die Berechnung der tatsächlichen Größe eines Touch-Objekts zeigt Formel 6.1 (nach [Saf09, Seite 42]).

$$target = \frac{(Breite\ des\ Objekts\ in\ Zoll) \times (Breite\ des\ Screens\ in\ Pixel)}{(Breite\ des\ Screens\ in\ Zoll)} \tag{6.1}$$

Das Ergebnis *target* entspricht dabei der tatsächlichen Größe eines Objekts in Pixel, bezieht man die Auflösung des Screens und die Abmessung des Geräts mit ein. Zur Berechnung der Länge müssen die Breitenangaben in obiger Formel adäquat ersetzt werden. Betrachtet man beispielsweise den in dieser Arbeit verwendeten Multitouch-Tisch mit einer Auflösung von $1024 \times 768\ Pixel$ und einer physischen Größe von $100 \times 70\ cm$, so ergibt sich eine für die Mindestgröße von $1 \times 1\ cm$ rund $10,24 \times 10,97$ notwendige Pixel.

Wie bei den Iceberg Tips, können anpassungsfähige Touch-Objekte und eine adäquate Größe dieser wiederum eingesetzt werden, um unpräzise Selektionen zu minimieren. So sollen etwa ebenfalls unbeabsichtigte Pan-Gesten verhindert werden.

6.1.2 Benutzeridentifikation via Orientierung der Finger

Kapitel 4.2 stellte bereits den Algorithmus von [DSA09] vor, welcher verwendet wird, um zu erkennen welche Fingereingaben zu welcher Hand gehören. Natürliche Einschränkungen der menschlichen Hand werden dazu benutzt. So wird beispielsweise festgelegt, dass sich der Schnittpunkt der verlängerten Achse zweier Berührungspunkte einer einzigen Hand im Regelfall hinter den Touch-Punkten befindet. Bei zwei Touch-Punkten einer Hand ist, mit Ausnahme des Daumens, zusätzlich nur eine maximale Entfernung von $8,89\ cm$ möglich. Mit Hilfe dieser beiden und weiteren, in Kapitel 4.2 angeführten, Constraints ist es möglich, Fingereingaben auf die zugehörigen Hände zu mappen. Mit diesem Algorithmus versucht [DSA09] hauptsächlich zwischen semantisch gleichen Gesten zu unterscheiden, ob die Geste mit Fingern einer Hand, oder mit Fingern zweier Hände ausgeführt wurden. Beispielsweise können Zoom Gesten sowohl etwa mit zwei Zeigefingern als auch mit Zeige- und Mittelfinger einer Hand ausgeführt werden. Dadurch soll die gleiche

Semantik dieser Gesten aufgebrochen und zwischen diesen differenziert werden. So wird etwa erwähnt, dass Benutzer die Ausführung einer Geste mit zwei Händen bevorzugen, wenn diese eine höhere Präzision erfordert. Durch die differenzierte Interpretation, könnte man nun beispielsweise zusätzliche Hilfestellung bieten, die eine höhere Genauigkeit der Eingabe ermöglichen.

Als Nebeneffekt führt [DSA09, Seite 102] an, dass diese Unterscheidung der Hände eingesetzt werden kann, um Probleme der Multiuser-Anwendung zu beheben.
So können zumindest Gesten mit einer Hand eindeutig einem Benutzer zugeordnet werden. Dadurch werden einige der erwähnten Probleme ausgeräumt: Werden Gesten von anderen Benutzern gestört, können die Eingaben dieser einem anderen Benutzer zugeordnet werden und damit im nächsten Schritt ignoriert werden. Erinnern wir uns zurück an das in Abbildung 5.2 illustrierte Problem: Zwei verschiedene Benutzer ziehen ein Objekt jeweils in ihre Richtung. Es fehlt die Koordination der beiden Personen. Setzt man nun den vorgestellten Algorithmus ein, könnte die Applikation die Eingabe der zweiten Hand ignorieren und beispielsweise nur die Geste interpretieren, die zeitlich gesehen als erstes ausgeführt wurde. Ein Manko dieses Ansatzes ist jedoch, dass die bei gleichzeitigem Ziehen zweier Hände interpretierte Skalier-Geste auch von einem einzelnen Benutzer mit zwei Händen ausgeführt werden könnte. Dadurch wären Skalier-Gesten nur mehr mit den Fingern einer Hand möglich. Die Benutzer hätten daher nicht mehr so viel Freiraum in der Ausführung der Gesten als zuvor.

6.2 Die Heuristiken

In diesem Kapitel werden die entwickelten Heuristiken vorgestellt. Versucht man Probleme der Multiuser-Anwendung bei Multitouch algorithmisch - in Form von Heuristiken - zu lösen, ist dies durch Segmentierung auf vier verschiedenen Ebenen möglich:

- **Gestenspezifisch**
  Auf dieser Ebene findet eine spezielle Unterscheidung der einzelnen Gesten statt. Gesten, die besondere Handhabung benötigen werden hierbei identifiziert, um sie etwa differenziert von anderen behandeln zu können.

- **Räumlich**
  Heuristiken der räumlichen Segmentierung versuchen durch Einschränkungen im Raum Verbesserungen zu erzielen. So werden beispielsweise Entfernungen benutzt, die sich durch Einschränkungen des menschlichen Körpers ergeben, um Probleme zu lösen. Ebenfalls wird versucht, die Gegebenheiten des Eingabemediums, also der menschlichen Hand

- und daher etwa die Größe der Finger - mit einzubeziehen. Es werden dabei die räumlichen Größen von Objekten optimiert, um das Auftreten von Problemen zu minimieren.

- **Objektspezifisch**
  Bei der objektspezifischen Segmentierung werden die Unterschiede in den Eigenschaften einzelner Objekte ausgenutzt. Ein triviales Beispiel ist die Unterscheidung zwischen dem Hintergrund und einem Objekt im Vordergrund.

- **Benutzerspezifisch**
  Diese Ebene der vorgestellten Heuristiken behandelt die Abbildung der Berührungspunkte auf die Benutzer. Es wird versucht durch Algorithmen zu identifizieren, welcher Berührungspunkt zu welchem Benutzer gehört.

Innerhalb dieser vier Ebenen nehmen die einzelnen Heuristiken verschiedene Ausprägungen an und versuchen durch verschiedene Ansätze Probleme zu lösen. Die Ebenen sind nicht als gänzlich unabhängig voneinander zu betrachten. Die angeführten Heuristiken der nachfolgenden Kapitel werden deshalb der Ebene zugeordnet, in welcher eine Heuristik die größte Ausprägung besitzt.
Dabei wird ebenfalls analysiert, ob der Einsatz einzelner Heuristiken von Beginn an eingeschränkt werden muss; dies bedeutet, ob ein globaler oder lediglich ein auf einzelne Applikationen beschränkter Einsatz gerechtfertigt ist.

Die in den Heuristiken erwähnten Gesten beziehen sich auf jene, welche sich als de-fakto Standard erwiesen haben. Diese wurden bereits in Kapitel 2.1.3 näher vorgestellt. Welche Gesten im erstellten Prototyp implementiert wurden, ist in Kapitel 7.2.1 zu finden.

Hinsichtlich der vorgestellten algorithmischen Umsetzungen sei erwähnt, dass diese in Java-orientiertem Pseudo-Code so allgemein wie möglich gehalten wurden und sich an gängigen event-getriebenen Architekturen und der Objektorientierung ausrichten. Grund dafür ist, dass dieses Softwarearchitektur-Muster in vielen GUI-Programmierschnittstellen, wie beispielsweise Java Swing und im eingesetzten Framework MT4J, Anwendung findet.

6.2.1 Heuristiken der gestenspezifischen Segmentierung

Heuristik 1. *Eine Pan-Geste soll sich in Multi-User Umgebungen signifikant von anderen Gesten unterscheiden.*

- **Warum?**
  In der Pilotstudie zur Evaluierung der Probleme trat bei der Evalu-

ierung jeder (Pan-Gesten unterstützenden) Applikation ein Konflikt zwischen der Pan-Geste und der Geste um ein Objekt zu bewegen auf. Dies war der Fall, da beide Gesten in den Applikationen grundsätzlich gleich implementiert wurden: Ein Finger berührt die Oberfläche und wird in eine Richtung bewegt. Der einzige Unterschied besteht darin, dass bei der Pan-Geste der Hintergrund der Applikation berührt wird, während bei einer „Zieh-Geste“ ein Objekt ausgewählt wird. Wird nun beispielsweise ein Objekt nicht präzise genug selektiert, kommt es vor, dass dies als Pan-Geste interpretiert wird. Ist ein Benutzer im Rahmen einer kollaborativen Tätigkeit individuell beschäftigt, wird dieser in der Ausführung gestört. [Saf09, Seite 40] beschreibt ebenfalls diese Problematik: *„Because of the inaccuracy of our fingers and hands, it is best not to make similar gestures for different actions in the same system for fear of users accidentally triggering one instead of the other.“* [Saf09] schlägt daher eine ähnliche Herangehensweise wie in dieser Heuristik vor: Um eine Missinterpretation zu vermeiden, sollen sich Pan-Gesten in Multi-User Umgebungen signifikant von anderen Gesten unterscheiden.

- **Wann?**
  Wird eine Pan-Geste so implementiert, dass sich diese signifikant von der „Zieh-Geste“ unterscheidet, kann die Natürlichkeit der Ausführung der Geste beeinträchtigt werden. Es wird daher vorgeschlagen, applikations-spezifisch zu differenzieren, ob eine Veränderung einer Pan-Geste adäquat ist.

- **Wie?**
  Ist eine Pan-Geste durch Ziehen mit einem Finger auf dem Hintergrund implementiert und unterscheidet sie sich daher nur gering von anderen Gesten (beispielsweise Zieh-Geste oder Klick-Geste), so wäre eine mögliche Umsetzung die Hinzunahme eines zweiten Fingers bei der Ausführung einer Pan-Geste. [Saf09, Seite 66] führt dies als *„two fingers to scroll“*-Geste an, bei welcher sich zwei Finger (typischerweise Zeige- und Mittelfinger) am Screen in die gewünschte Scroll-Richtung bewegen. Angeführt wird jedoch, dass das Haupteinsatzgebiet Geräte mit indirektem Touch-Input, wie etwa Trackpads, bilden.

  Eine weitere Möglichkeit wäre etwa die Verwendung der ganzen Hand, um eine Pan-Geste auszuführen. Dabei werden die Finger und die Handfläche einer Hand über die Oberfläche in die gewünschte Schwenk-Richtung geführt. Eine Geste dieser Form beischreibt beispielsweise [VWW10, Seite 6].

- **Adressiertes Problem.**
  Ein unbeabsichtigt ausgeführtes Schwenken der Kamera stört die ande-

ren Benutzer. Die Heuristik adressiert das Problem der unbeabsichtigt alleine getriggerten Gesten. (2b).

Heuristik 2. *Jede zusätzliche Geste soll sich signifikant von existierenden unterscheiden.*

- **Warum?**
  Neben der Beobachtung des Problems der Erweiterung des Gesten-Repertoires in der Pilotstudie, führt dieses Problem auch [DSA09, Seite 101] an: *„Often, a repertoire of gestures can only be extended at the expense of lower robustness, since new gestures distinguish from existing ones only by subtle variations."* Es fällt daher auf, dass diese Variationen einer „neuen" Geste so signifikant wie möglich ausfallen sollen, um die Robustheit einer Applikation zu konservieren.

- **Wann?**
  Ein Einsatz dieser Heuristik ist für jene Applikationen sinnvoll, welche zusätzliche, über das Standard-Repertoire hinausgehende Gesten definieren.

- **Wie?**
  Zusätzliche Gesten sollen sich durch signifikante Variationen von existierenden Gesten unterscheiden.

- **Adressiertes Problem.**
  Zusätzliche Gesten verschlechtern die Robustheit einer Applikation. Die Heuristik adressiert das Problem des erweiterten Gesten-Repertoires. (5a).

6.2.2 Heuristiken der räumlichen Segmentierung

Heuristik 3. *Verwendet eine Applikation eine Zoom-Geste, sollte die Entfernung der zwei, bei der Zoom-Geste auftretenden Berührungspunkte, berücksichtigt werden.*

- **Warum?**
  Die Pilotstudie zur Identifikation der Probleme offenbarte, dass zwei unbeabsichtigte Berührungspunkte auf dem Hintergrund der Applikation, welche beispielsweise durch ein Abstützen der Benutzer am Tisch erzeugt werden, zu einer (Fehl-) Interpretation als Zoom-Geste führen können. Befindet sich ein Benutzer in einer individuellen Arbeitsphase, wird dieser in seiner Tätigkeit unterbrochen und gestört. Während der Evaluierung wurde die Geste sogar dann fehlinterpretiert, als die beiden Berührungspunkte auf den gegenüberliegenden Kanten des Tisches erzeugt wurden - also in einem großen räumlichen Abstand zueinander standen. In dieser Heuristik wird daher eine Berücksichtigung

der Entfernung zwischen den zwei Berührungspunkte einer Zoom-Geste vorgeschlagen.

Bei Analyse einer Zoom-Geste fällt auf, dass diese sowohl mit zwei Fingern einer Hand, als auch mit zwei Fingern zweier Hände ausgeführt werden kann. Die Berührungspunkte dieser Geste können daher einerseits in einem kleinen räumlichen Abstand zueinander, bei Ausführung mit einer Hand, statt finden. Andererseits - führt man die Geste mit zwei Händen aus - kann dieser Abstand theoretisch auch beinahe die gesamte Spannweite der menschlichen Arme betragen.

Zusätzlich kann bei der Analyse ein Unterschied zwischen der Art der Zoom-Geste ausgemacht werden: Wird der Zoom-Faktor erhöht (= $ZoomIn$) oder verringert (= $ZoomOut$)? Da sich bei einem *Erhöhen des Zoom-Faktors* die beiden involvierten Finger voneinander weg bewegen, können die Berührungspunkte am Beginn der Geste nur in einem geringeren Abstand zueinander stehen als die Spannweite der menschlichen Arme. Die Definition einer maximalen Entfernung der beiden involvierten Berührungspunkte erscheint in diesem Fall daher adäquat: Wird diese Entfernung überschritten, werden die beiden Berührungspunkte nicht als Zoom-Geste interpretiert. *Verringert* man hingegen den Zoom-Faktor, bewegen sich die beiden Finger zueinander. Es scheint daher naheliegend, dass die Finger zu Beginn der Geste weiter voneinander entfernt sein können.

Weiters gilt es die Größe des Interfaces zu analysieren: Bei sehr großen Multitouch-Interfaces, etwa großen Wänden, können - wie bereits angesprochen - die beiden Berührungspunkte keine größere Entfernung als die Spannweite der menschlichen Arme aufweisen. Die Ausführung einer Zoom-Geste, bei welcher die beiden Berührungspunkte in dieser Entfernung zueinander stehen, erscheint außerdem wenig natürlich. Daraus folgt, dass man in diesem speziellen Fall wiederum eine maximale Entfernung definieren kann, welche weniger als die durchschnittliche Spannweite eines Menschen beträgt. Aufgrund der erheblichen Unterschiede zwischen den Größen der Interfaces sowie derer Orientierungen (horizontal als Tisch, oder vertikal als Tafel), muss der konkrete Wert einer adäquaten maximalen Entfernung jedoch von Fall zu Fall analysiert werden. So kann in dieser Arbeit auch kein allgemein gültiger Wert angegeben werden. Dabei gilt es zusätzlich zu beachten, dass das Ziel dieser Heuristik nicht die Verhinderung einer Fehlinterpretation um jeden Preis sein kann. Es gilt mögliche Nebeneffekte zu beachten: Eine fälschlicherweise verhinderte Interpretation kann zu einem Bruch der Intuition der Ausführung der Geste führen. Es erscheint daher sinnvoll, nur jene Fälle zu behandeln, bei welchen man mit sehr hoher Wahrscheinlichkeit auf eine Fehlinterpretation schließen kann.

Zusammenfassend lässt sich daher feststellen, dass bei sehr großen Interfaces die Definition einer adäquaten maximalen Entfernung der beiden Berührungspunkte passend ist. Bei einer Erhöhung des Zoom-Faktors - als einer *Zoom-In-Geste* kann dieser Wert zusätzlich verringert werden. Bei kleineren Interfaces sollte ebenfalls bei der Identifikation einer Zoom-In-Geste ein adäquater maximaler Wert für die Entfernung gefunden werden. Betrachtet man eine *Zoom-Out-Geste*, könnte man zusätzlich eine Interpretation verhindern, wenn die beiden Berührungspunkte exakt an den gegenüberliegenden Kanten erkannt werden. Dadurch würde man dem ursprünglich identifizierten Problem der Fehlinterpretation der beiden unbeabsichtigten Berührungspunkte auf den gegenüberliegenden Seiten des Tisches entgegen wirken.

An dieser Stelle sei erwähnt, dass bei Adaptierung dieser Heuristiken auf noch kleinere Geräte mit Multitouch-Input - etwa Smart-Phones - die Berücksichtigung der beiden bei der Zoom-Geste auftretenden Berührungspunkte wiederum differenziert betrachtet werden muss. Die Definition etwa einer maximalen Entfernung, würde in diesem Fall aufgrund des kleinen Interfaces und der Größe der menschlichen Finger Probleme bereiten.

- **Wann?**
  Ein Einsatz dieser Heuristik wird in sämtlichen Applikationen, die Zoom-Gesten verwenden, mit Ausnahme von Geräten mit kleinen Multitouch-Interfaces, als sinnvoll erachtet.

- **Wie?**
  Einen Ansatz der Definition einer maximalen Distanz zwischen beiden Fingern der Geste im Falle eines `ZoomInEvent` wird in nachfolgendem Pseudocode anhand einer ereignis-getriebenen Architektur vorgeschlagen:

```
// define some variables
float zoomDetectRadius = Application.width/2;
Cursor finger1;
Cursor finger2;

//compute distance between fingers
float fingerDistance = Vector3D.distance(
    new Vector3D(finger1.getPosX(), finger1.getPosY()),
    new Vector3D(finger2.getPosX(), finger2.getPosY())
);

//check if fingers are in distance
if (fingerDistance < zoomDetectRadius) {
```

```
        fireGestureEvent(new ZoomInEvent(this, finger1,
            finger2));
    }
```

- **Adressiertes Problem.**
  Ein unbeabsichtigtes, applikationsweites Zoomen stört die anderen Benutzer. Die Heuristik adressiert das Problem der unbeabsichtigt gemeinsam getriggerten Gesten. (1a).

Heuristik 4. *Zur Vermeidung unbeabsichtigter Berührungspunkte aufgrund von unpräzisen Gesten, sollen „Iceberg Tips“ für sämtliche selektierbare Objekte verwendet werden.*

- **Warum?**
  Unpräzise Berührungspunkte, beispielsweise um Objekte zu verschieben, führten in der Pilotstudie zur Evaluierung der Probleme dazu, dass die Eingabe als Pan-Geste interpretiert wurde. Diese Fehlinterpretation hat Auswirkungen auf alle Benutzer und stört diese, da die gesamte Sicht auf die Applikation verändert wird.

  Um diesem Problem der unpräzisen Berührungspunkte entgegen zu wirken, sollen daher die in Kapitel 6.1.1 vorgestellten und in Abbildung 6.1 illustrierten Iceberg-Tips implementiert werden. Das für Gesten berührungsempfindliche Ziel eines Objekts ist dabei größer als sein im Interface sichtbarer Teil. Berührt man beispielsweise einen Button knapp neben dem eigentlich sichtbaren Teil, so wird dies trotzdem als zum Button zugehörig und zum Beispiel als Selektion des Buttons interpretiert. Durch diese Heuristik werden ebenfalls Ungenauigkeiten in der Erkennung der Berührungspunkte ausgeglichen.

  Wie bereits in den bestehenden Ansätzen für Heuristiken angeführt (Kapitel 6.1.1), dienen „Iceberg-Tips“ generell zur Verbesserung der Präzision bei der Touch-Eingabe. So wird dadurch beispielsweise eine präzisere Selektion von Objekten ermöglicht, sowie ungenaues Tracking ausgeglichen.

- **Wann?**
  Die Heuristik soll global eingesetzt werden.

- **Wie?**
  Die Heuristik könnte beispielsweise umgesetzt werden, indem ein Multitouch-Button mit einem unsichtbaren erweiterten Interaktionsbereich definiert wird.

- **Adressiertes Problem.**
  Ein unbeabsichtigt ausgeführtes Schwenken der Kamera stört die ande-

ren Benutzer. Die Heuristik adressiert das Problem der unbeabsichtigt, alleine getriggerten Gesten. (2b).

Heuristik 5. *Zur Vermeidung unbeabsichtigter Berührungspunkte aufgrund von unpräzisen Gesten, sollen selektierbare Objekte mindestens eine Größe von* 1×1 *cm* $(0,4 \times 0,4$ *Zoll*$)$ *aufweisen.*

- **Warum?**
  Wie bereits bei Heuristik 4, ist die Motivation dieser Heuristik die Fehlinterpretation unpräziser Berührungspunkte - beispielsweise um Objekte zu verschieben - als Pan-Geste beziehungsweise die dadurch hervorgerufene Störung der anderen Benutzer. Um diese Fehlinterpretation zu verhindern, sollte die in [Saf09, Seite 42] hauptsächlich für Single-User-Umgebungen angedachte Heuristik (siehe dazu auch Kapitel 6.1.1) berücksichtigt werden: Touch-Objekte sollen eine Mindestgröße von 1×1 *cm* aufweisen. Diese Minimalgröße ist wie bereits in Kapitel 6.1.1 erwähnt, ein allgemeiner Ansatz, um der geringeren Präzision der Selektion - als etwa bei der Eingabe mit einer Maus - entgegen zu wirken.

  Kann in einer Applikation der Zoom-Faktor verändert werden, ist es wesentlich, dass das Einhalten dieser Heuristik immer noch sichergestellt ist. Verringert man beispielsweise den Zoom-Faktor, könnten ansonsten selektierbare Elemente ein Ausmaß kleiner als die Mindestgröße von 1×1 *cm* aufweisen.

- **Wann?**
  Die Heuristik soll global eingesetzt werden.

- **Wie?**
  Die Heuristik kann durch die Definition einer globalen Mindestgröße, welche auch bei Abänderung des Zoom-Faktors sicher gestellt ist, umgesetzt werden.

- **Adressiertes Problem.**
  Ein unbeabsichtigt ausgeführtes Schwenken der Kamera stört die anderen Benutzer. Die Heuristik adressiert das Problem der unbeabsichtigt, alleine getriggerten Gesten. (2b).

Heuristik 6. *Zur Vermeidung unbeabsichtigter Berührungspunkte aufgrund von unpräzisen Gesten, sollen skalierbare Objekte nicht unter eine Größe von* 3×3 *cm* $(1,2 \times 1,2$ *Zoll*$)$ *verkleinert werden können.*

- **Warum?**
  Diese Heuristik erweitert Heuristik 5 und soll wiederum der Fehlinter-

pretation unpräziser Berührungspunkte als Pan-Geste - beziehungsweise der dadurch hervorgerufenen Störung der anderen Benutzer - entgegen wirken. So sollen Objekte nicht unter eine Größe von maximal 3×3 cm verkleinert werden dürfen, damit Objekte etwa wieder präzise genug vergrößert werden können. Die dreifache Größe aus Heuristik 5 ($= 3 \times 3$ cm) wird deshalb verwendet, da zwei Finger in einem gewissen Abstand zueinander benötigt werden, um ein Objekt mit einer Skalier-Geste wieder zu vergrößern. Die Mindestgröße von 1×1 cm ist für die Eingabe mit einem Finger, beispielsweise für die Selektion eines Buttons, optimiert. Bei einer Skaliergeste berühren jedoch zwei Finger in einem gewissen Abstand zueinander ein Objekt. Dadurch wird sichergestellt, dass verkleinerte Objekte nicht nur präzise genug selektiert, sondern diese auch wieder etwa auf die Ausgangsgröße skaliert werden können.

Wie schon bei Heuristik 5, gilt es ebenfalls die Möglichkeit der Veränderung des Zoom-Faktors der Applikation mit einzubeziehen. Kann in einer Applikation dieser variiert werden, ist es wesentlich, dass das Einhalten dieser Heuristik immer noch sichergestellt ist. Verringert man beispielsweise den Zoom-Faktor, könnten ansonsten skalierbare Elemente ein Ausmaß kleiner als die Mindestgröße von 3×3 cm aufweisen. Im obigen Pseudocode wird dies durch Abfrage des Zoom-Faktors mit der Methode `App.getZoomFactor()` sicher gestellt.

- **Wann?**
  Die Heuristik soll global eingesetzt werden.

- **Wie?**
  Der nachfolgende Pseudocode illustriert exemplarisch, wie durch Einsatz von Formel 6.1 eine minimale Größe eines Ziel-Objekts (`target`) gewährleistet werden kann. Es wird dabei ebenfalls der Zoom-Faktor der Applikation berücksichtigt. Dieser entspricht der Variablen, welche von der Methode `App.getZoomFactor()` zurückgeliefert wird. Diese hat einen Ausgangswert von 1; bei einem Wert kleiner 1 wurde der Zoom-Faktor verkleinert; vice versa für einen Wert größer 1:

```
//define some variables
final int minWidth = (3*App.widthPx/App.widthCm)
                       /App.ZoomFactor();
final int minHeight = (3*App.heightPx/App.heightCm)
                      /App.getZoomFactor();
TouchTarget target;

//target will be smaller than minWidth OR minHeight
//AND target would be scaled down
```

```
if ((target.getWidth <= minWidth ||
        target.getHeight <= minHeight)
   && (scaleEvent.getScaleFactorX()<=1
        || scaleEvent.getScaleFactorY()<=1)) {
    //do nothing
} else {
    //only scale in this case
    target.scale(scaleEvent.getScaleFactorX(),
    scaleEvent.getScaleFactorY());
}
```

- **Adressiertes Problem.**
  Ein unbeabsichtigt ausgeführtes Schwenken der Kamera stört die anderen Benutzer. Die Heuristik adressiert das Problem der unbeabsichtigt, alleine getriggerten Gesten. (2b).

Heuristik 7. *Sind Grenzen zwischen Benutzern erforderlich, sollen private und öffentliche Arbeitsbereiche explizit definiert werden.*

- **Warum?**
  Beobachtet man Kollaborationen an traditionellen Tischen, so fällt auf, dass für jede Person eigene, voneinander unabhängige Arbeitsflächen gebildet werden. Auf diesen Flächen wird individuellen Aufgaben, die sich im Rahmen kollaborativer Tätigkeiten ergeben, nachgegangen. Wie bereits in Kapitel 2.2 erwähnt, untersucht [Sco05] solche Kollaborationen auf Tischen im Rahmen einer Studie basierend auf Papierzetteln. Es wird dabei gezeigt, dass diese Zetteln für individuelle Tätigkeiten in einen eigenen Teil des Tisches - in eine private Arbeitsfläche - bewegt werden und dadurch für sich reserviert werden. Nach Fertigstellung der individuellen Arbeit werden diese wieder in die Mitte des Tisches gelegt, um damit zu zeigen, dass sie für die Gruppe frei verfügbar sind. Zusätzlich wird erläutert, dass die Mitte des Tisches so geteilt wird, dass jene Person für die in der Mitte befindlichen Artefakte verantwortlich ist, die räumlich gesehen am nächsten zu ihnen ist.

  Wie die Studie von [SCH05] anhand einer Fotoapplikation zeigt, unterstützen Multitouch-Tische die Fähigkeit zur Bildung von Arbeitsbereichen grundsätzlich auch: Die Teilnehmer bildeten virtuelle „Container“ für Fotos im Zentrum des Tisches, als diese die Fotos gemeinsam betrachteten. Für individuelle Arbeiten wurden die Container zu sich selbst, in die individuelle Arbeitsfläche, bewegt. Dabei wurde auch Rücksicht auf die anderen Mitarbeiter genommen: Es wurde versucht, diese in deren Arbeiten so wenig wie möglich zu stören.

Die Pilotstudie zeigte jedoch, dass bei räumlichen Teilungen in Arbeitsflächen Mängel auftreten können: Es existieren keine expliziten Grenzen für die Arbeitsbereiche. So wurden beispielsweise Fotos unbeabsichtigt derartig vergrößert, dass die anderen Benutzer in ihren Tätigkeiten gestört wurden.

- **Wann?**
  Falls im Rahmen einer kollaborativen Tätigkeit getrennt in privaten und öffentlichen Bereichen gearbeitet wird, soll diese Heuristik angewendet werden, um diese Bereiche explizit zu unterstützen. Daher wird ein applikations-spezifischer Einsatz vorgeschlagen.

- **Wie?**
  Falls dies die kollaborative Arbeit nahe legt, sollen Arbeitsbereiche explizit definiert und adäquat im User Interface abgebildet werden. Die privaten Arbeitsflächen sollen dabei abgeschlossene Einheiten bilden. Vergrößert man nun beispielsweise ein Objekt in einer solchen Arbeitsfläche, bilden die Grenzen der Arbeitsfläche die Grenze für die Vergrößerung. Durch Zieh- und Schiebe-Gesten können Objekten aus der öffentlichen Fläche in die private gezogen werden beziehungsweise vice versa von einer privaten Arbeitsfläche wieder in die öffentliche geschoben werden. Bei der Überführung von Objekten in eine private Arbeitsfläche ergeben sich dabei Herausforderungen hinsichtlich der Orientierung der Objekte: Wird etwa ein Objekt von einer privaten Arbeitsfläche in eine andere geschoben, so könnte beispielsweise die Orientierung des Objekts entsprechend der Sicht eines anderen Benutzers auf das Objekts verändert werden.

- **Adressiertes Problem.**
  Diese Heuristik adressiert das Problem des Fehlens von Grenzen der Arbeitsbereiche. (3d).

Heuristik 8. *Wird Heuristik 7 nicht angewandt, soll eine Obergrenze für die Vergrößerung von Objekten eingeführt werden.*

- **Warum?**
  Die Motivation für diese Heuristik stellt die Problematik, welche bereits in Heuristik 7 beschrieben und behandelt wurde, dar. Entscheidet man sich gegen den Einsatz von Heuristik 7, so sollen Obergrenzen für die Vergrößerung von Objekten definiert werden. Dadurch soll sicher gestellt werden, dass Objekte, die unbeabsichtigt zu groß skaliert wurden, andere Benutzer nicht stören. Als Obergrenze könnte die Hälfte der tatsächlichen Größe des Screens verwendet werden. So würden bei einer unbeabsichtigten Vergrößerung eines Objekts andere

Benutzer nicht in ihren individuellen Arbeitsphasen gestört werden, da deren implizit definierte Arbeitsflächen nicht gänzlich mit jenem Objekt überdeckt wären. Vor der Definition eines fixen Wertes, erscheint jedoch eine applikations-spezifische Analyse sinnvoll, da die maximale Größe etwa von Art und Zweck der Software sowie von der Anzahl der Benutzer abhängig ist.

Wie schon bei Heuristik 5 und 6, gilt es wiederum die Möglichkeit der Veränderung des Zoom-Faktors der Applikation mit einzubeziehen. Kann in einer Applikation dieser variiert werden, ist es wesentlich, dass das Einhalten dieser Heuristik immer noch sichergestellt ist. Erhöht man beispielsweise den Zoom-Faktor, könnten ansonsten vergrößerbare Elemente ein Ausmaß größer als die Maximalgröße aufweisen. Im obigen Pseudocode wird dies durch Abfrage des Zoom-Faktors mit der Methode `App.getZoomFactor()` sicher gestellt.

- **Wann?**
  Entscheidet man sich gegen den Einsatz von Heuristik 7 und lässt sich beobachten, dass im Rahmen einer kollaborativen Tätigkeit getrennt in privaten und öffentlichen Bereichen gearbeitet wird, so soll diese Heuristik implementiert werden.

- **Wie?**
  Der nachfolgende Pseudocode illustriert exemplarisch die Definition einer Obergrenze (die Hälfte des Flächeninhalts der Arbeitsfläche, unter Berücksichtigung des Zoom-Faktors der Applikation) für das Skalieren von Objekten:

```
//define some variables
final int maxArea = App.width*App.height/2
                      /App.getZoomFactor();
TouchTarget target;

//target will be bigger than maxWidth OR maxHeight
//AND target would be scaled up
if ((target.getWidth()*target.getHeight()
    >= maxArea)
   && (scaleEvent.getScaleFactorX()>=1
        || scaleEvent.getScaleFactorY()>=1)) {
    //do nothing
} else {
    //only scale in this case
    target.scale(scaleEvent.getScaleFactorX(),
    scaleEvent.getScaleFactorY());
}
```

- **Adressiertes Problem.**
  Diese Heuristik adressiert das Problem unbeabsichtigt alleine getriggerter Gesten sowie das Problem des Fehlens von Grenzen der Arbeitsbereiche. (2a und 3d).

6.2.3 Heuristiken der objektspezifischen Segmentierung

Heuristik 9. *Zur Vermeidung unbeabsichtigter Berührungspunkte aufgrund von unpräzisen Gesten, sollen zusätzlich zu Iceberg-Tips anpassungsfähige Touch-Objekte realisiert werden.*

- **Warum?**
  Während Heuristik 4, Heuristik 5 und Heuristik 6 durch räumliche Segmentierung versuchen das Manko unbeabsichtigter Berührungspunkte zu lösen, fokussiert diese Heuristik neben der räumlichen Ebene auch die objektspezifische Segmentierung. [Saf09, Seite 44] stellt bereits einen Ansatz für anpassungsfähige Touch-Ziele vor. Adaptive Touch-Ziele stellen - wie bereits in Kapitel 6.1.1 ausführlich erläutert - einen allgemeinen Ansatz dar, um das Manko der unpräzisen Selektion bei Touch-Interfaces zu verbessern.

  Dieser Ansatz soll in dieser Heuristik benutzt werden, um unbeabsichtigte Berührungspunkte zu minimieren.

- **Wann?**
  Diese Heuristik kann nur eingesetzt werden, wenn auch Heuristik 4 implementiert wird. Weiters soll applikations-spezifisch über den Einsatz entschieden werden, da nicht in allen Applikationen sinnvolle Algorithmen zur Abschätzung von Wahrscheinlichkeiten der nächsten Interaktion abgeleitet werden können. Ein Einsatz wird daher nur in Applikationen vorgeschlagen, bei welchen immer wiederkehrende, sequentielle Abläufe im Interaktionsverhalten vorzufinden sind.

- **Wie?**
  Wie bereits in Kapitel 6.1.1 erläutert, verwenden anpassungsfähige Touch-Objekte Algorithmen, um die Wahrscheinlichkeiten der als nächstes statt findenden Interaktion, etwa der Selektion eines Buttons, zu berechnen. Je größer diese Wahrscheinlichkeit, desto größer wird der nicht sichtbare Bereich eines Iceberg Tips ausgelegt. Ein oftmals zitiertes Beispiel stellt hierbei die Tastatur dar, bei welcher sich die Wahrscheinlichkeiten der als nächstes selektierten Buchstaben aus den vorangegangen Buchstaben ergeben. Ist es etwa möglich aus den vorangegangen Buchstaben und einem zusätzlichen Buchstaben ein Wort beziehungsweise den Teil eines Wortes zu bilden, so ist dessen Wahrscheinlichkeit der Selektion höher - und somit sollte auch die Größe des

Iceberg Tips erhöht werden. Heuristik 4 bildet demnach eine Grundlage für die Anwendung dieser Heuristik.

- **Adressiertes Problem.**
  Die Heuristik adressiert das Problem der unbeabsichtigt, alleine getriggerten Gesten. (2b).

Heuristik 10. *Wird ein Objekt aktiv durch eine Geste manipuliert, sollen keine Gesten am Hintergrund (zum Beispiel Zoom oder Pan) möglich sein.*

- **Warum?**
  Die Pilotstudie zeigt, dass die Ausführung einer Geste auf dem Hintergrund der Applikation - eine Zoom- oder Pan-Geste - die anderen Benutzer stören kann. Insbesondere wenn sich Personen im Rahmen einer kollaborativen Tätigkeit in einer individuellen Arbeitsphase befinden, wirkt sich eine Veränderung der gesamten Sicht auf die Applikation durch eine andere Person negativ auf den Arbeitsfluss aus, da in einer individuellen Phase ebenfalls Interaktion aktiv statt findet. So ist es möglich das Problem einzudämmen, indem bei aktiver Interaktion globale Applikations-Gesten gesperrt werden. Führt ein Benutzer daher gerade eine Geste aus beziehungsweise interagiert dieser mit der Applikation, sollen Zoom- und Pan-Gesten deaktiviert werden. Diese sollen in einem solchen Systemzustand nicht mehr interpretiert werden.

- **Wann?**
  Ein Einsatz dieser Heuristik wird in sämtlichen Applikationen, die Pan- oder Zoom-Gesten verwenden, als sinnvoll erachtet.

- **Wie?**
  Im nachfolgenden Quellcode wird exemplarisch dargestellt, wie im `DragProcessor`, welcher für die Erkennung der Zieh-Gesten zuständig ist, Zoom und Pan beim Start der Geste deaktiviert werden. Nach Beendigung der Geste werden sie schließlich wieder aktiviert.

```
public void cursorStarted(){
   [...]
   //deactivate Zoom and Pan Gesture
   canvas.setGestureAllowance(ZoomProcessor.class, false);
   canvas.setGestureAllowance(PanProcessor.class, false);
   //start the Drag Gesture event
   this.fireGestureEvent(new DragEvent(), Event.START);
   [...]
}
```

```
public void cursorEnded(){
   [...]
   //reactivate Zoom and Pan Gesture and end Dragging
   canvas.setGestureAllowance(ZoomProcessor.class, true);
   canvas.setGestureAllowance(PanProcessor.class, true);
   this.fireGestureEvent(new DragEvent(), Event.END);
   [...]
}
```

- **Adressiertes Problem.**
  Ein unbeabsichtigt ausgeführtes Schwenken der Kamera und Verändern des Zoom-Faktors der Applikation stören die anderen Benutzer. Die Heuristik adressiert das Problem der unbeabsichtigt (alleine sowie gemeinsam) getriggerten Gesten. (1a und 2b).

Heuristik 11. *Um zu vermeiden, dass andere Benutzer Gesten korrumpieren, sollen Objekte gesperrt werden, sobald eine Geste eindeutig identifiziert wurde.*

- **Warum?**
  Diese Heuristik behandelt das identifizierte Problem der Störung von Gesten durch andere Benutzer. So konnten in der Pilotstudie Gesten nicht mehr korrekt interpretiert werden, wenn in räumlicher Nähe zusätzliche Berührungspunkte von anderen Benutzern aufgezeichnet wurden. Es wurde beobachtet, dass dieses Problem in einer zeitlichen Abfolge statt findet:

  – *Zustand 1:* Die Berührungspunkte von Benutzer A werden korrekt als Geste interpretiert.
  – *Zustand 2:* Zusätzliche von Benutzer B stammende Berührungspunkte werden in räumlicher Nähe aufgezeichnet.
  – *Zustand 3:* Geste von Benutzer A wird abgebrochen und nicht mehr korrekt interpretiert.

  Diese drei Zustände und deren Übergänge werden in Abbildung 6.3 illustriert. Dabei berührt Benutzer „B“ ein Objekt, auf welches bereits eine Geste durch Benutzer „A“ erkannt wurde.

  Die vorgestellte Heuristik nimmt diese zeitliche Sequenz als Grundlage: Sobald Zustand 1 erreicht ist, also eine korrekte Interpretation statt findet, soll das Objekt für jegliche andere Gesten gesperrt werden. Erst wenn die Geste als beendet angesehen wird, also die zugehörigen Berührungspunkte nicht mehr erkannt werden, wird diese Sperre entfernt.

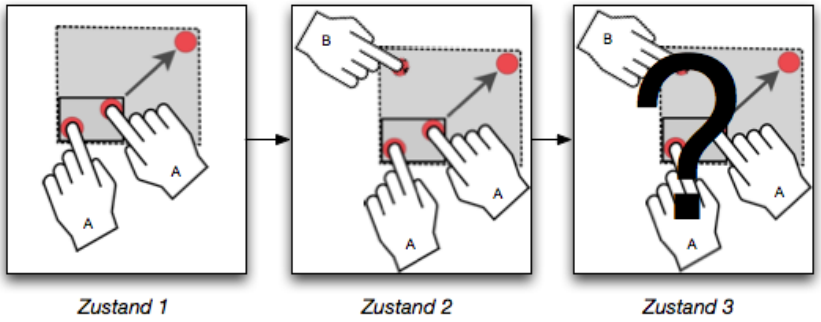


Abbildung 6.3 – Zusätzliche Berührungspunkte bei Ausführung einer Geste

- **Wann?**
  Ein globaler Einsatz dieser Heuristik wird als sinnvoll erachtet.

- **Wie?**
  Wenn eine Skalier-Geste erkannt wurde, kann das Objekt für jegliche andere Benutzereingaben gesperrt beziehungsweise diese ignoriert werden. Dies wird in nachfolgendem Pseudo-Code dargestellt, indem - bevor eine Skalier-Geste gestartet wird - überprüft wird, ob die Geste bereits genug `Cursor`-Objekte (also Berührungspunkte) besitzt. Die Sperrung der restlichen Gesten kann ähnlich wie in Heuristik 10 implementiert werden.

```
//define some variables
List<Cursor> lockedCursors = this.getCursorsOfGesture;
List<Cursor> unUsedCursors;

public void cursorStarted(Cursor c1) {

 //can gesture be started?
 if (lockedCursors.size() >= 2){
   //scale with 2 fingers already in progress
   unUsedCursors.add(m);
   logger.debug("gesture has already enough cursors");

 } else {
   //not enough cursors -> scaling could be started
   if (unUsedCursors.size() == 1) {
     //we can try to start gesture with unUsedCursor
     Cursor c2 = unUsedCursors.get(0);
     [...]
     this.fireEvent(new ScaleEvent(this, Gesture.START,
```

```
            c1, c2));
        [...]

      } else {
        //no second cursor was available
        unUsedCursors.add(c1);
      }
    }
```

- **Adressierte Probleme.**
  Die Interpretation von Gesten kann durch Berührungspunkte anderer Benutzer verfälscht werden. Die Heuristik adressiert das Problem der Störung von Gesten durch andere Benutzer. (1b, 3a, 3b und 3c).

Heuristik 12. *Um zu vermeiden, dass andere Benutzer Gesten korrumpieren, sollte objektspezifisch segmentiert werden: Ein Berührungspunkt auf ein Objekt x interagiert nicht mit einem zusätzlichen Berührungspunkt auf ein Objekt y.*

- **Warum?**
  Wie bereits Heuristik 11 behandelt diese Heuristik den in der Pilotstudie aufgetretenen Mangel der Störung von Gesten durch andere Benutzer. Wurden bei Ausführung einer Geste in einem gewissen räumlichen Umkreis (der zu dieser Geste gehörenden Berührungspunkte) weitere Berührungspunkte erkannt, so führte dies in einigen evaluierten Applikationen zu einem unerwarteten Systemzustand. Es wurde weder die ursprüngliche Geste korrekt ausgeführt, noch die durch den zusätzlichen Berührungspunkt initiierte Geste interpretiert.

  In dieser Heuristik wird daher vorgeschlagen, objektspezifisch zu unterscheiden: Werden ein *Objekt x* und ein *Objekt y* gleichzeitig (von verschiedenen Benutzern) selektiert (beziehungsweise eine andere Geste auf ein Objekt ausgeführt), so sollen diese beiden Berührungspunkte nicht miteinander interagieren. Sie sollen vom System unabhängig voneinander betrachtet werden. Dies soll nicht nur für die Selektion, sondern auch für komplexere Gesten zutreffen.

- **Wann?**
  Ein globaler Einsatz dieser Heuristik wird als sinnvoll erachtet.

- **Wie?**
  In einer ereignis-getriebenen Architektur ist diese Heuristik bereits implizit gegeben und kann trivial umgesetzt werden: Ereignisse können unabhängig voneinander für Objekte gefeuert werden.

- **Adressierte Probleme.**
  Die Interpretation von Gesten kann durch Berührungspunkte anderer Benutzer verfälscht werden. Die Heuristik adressiert das Problem der Störung von Gesten durch andere Benutzer. (3a, 3b und 3c).

6.2.4 Heuristiken der benutzerspezifischen Segmentierung

Heuristik 13. *Ist eine Identifikation der Benutzer zwingend erforderlich, so können die Finger-Orientierung und die natürlichen Einschränkungen der menschlichen Hand verwendet werden, um zwischen Benutzern zu differenzieren.*

- **Warum?**
  Die Pilotstudie zeigte, dass eine Identifikation der Benutzer zusätzliche Möglichkeiten für das Game-Design von Multi-Player-Games mit sich bringt: Durch die Identifikation können die Spieler getrennt voneinander betrachtet werden und es können Wettbewerbe entstehen.

  Zudem führen [DL01, Seite 219] und [SG09, Seite 1] an, dass eine Identifikation zusätzliche, die kollaborative Tätigkeit unterstützende, Interaktionsformen ermöglicht, auf welche im nachfolgenden Kapitel 6.3.2 genau eingegangen wird. Beispielsweise werden dadurch komplexe, kooperative Gesten, wie sie in [RMHPW06] vorgestellt werden, denkbar. Weiters werden Ansätze für Zugriffsberechtigungen von einzelnen Dokumenten bei gemeinsamen Tätigkeiten bei einer Multitouch-Umgebung möglich. Eine solche Methodik wird etwa in [RMRS$^+$04] präsentiert. Dabei werden die Zugriffsberechtigungen durch Gesten geregelt. Dies befähigt beispielsweise öffentliche Zugriffsrechte auf ein Dokument während aktiven, kollaborativen Phasen beziehungsweise private Schreibrechte des Autors während privaten Phasen im Rahmen einer Zusammenarbeit an einem Multitouch-Tisch.

  Diese Heuristik versucht daher das Defizit des Fehlens einer Benutzeridentifikation von klassischen Multitouch-Umgebungen zu beheben.

  Betreffend der Gruppierung der Heuristik, wurde diese der benutzerspezifischen Segmentierung zugeteilt, weil sie versucht Hände beziehungsweise Benutzer zu identifizieren. Betrachtet man den verwendeten Algorithmus, fällt auf, dass diese Heuristik ebenfalls starke Ausprägungen auf der Ebene der räumlichen Segmentierung offenbart: Räumliche Entfernungen und Winkel der Finger werden für die Differenzierung verwendet.

- **Wann?**
  Aufgrund der Komplexität des Algorithmus und der sich daraus ergebenden Schwierigkeit die Fehleranfälligkeit und Robustheit der Methodik zu evaluieren, wird in dieser Arbeit eine applikations-spezifische

Verwendung dieser Heuristik vorgeschlagen. Die Heuristik soll daher verwendet werden, wenn eine Identifikation der Benutzer zwingend erforderlich ist.

- **Wie?**
  Die in [DSA09] präsentierte Methode der Verwendung der Einschränkungen der menschlichen Hände soll verwendet werden, um zwischen Benutzern zu differenzieren. Dieser Algorithmus wurde bereits in Kapitel 4.2 eingehend erläutert.

- **Adressiertes Problem.**
  Es ist nicht möglich zwischen Benutzern zu differenzieren. Die Heuristik adressiert das Problem der fehlenden Identifikation der Benutzer. (4).

Heuristik 14. *Ist eine Identifikation der Benutzer zwingend erforderlich (beispielsweise in Spielen), so kann ein simpler Algorithmus - der Form Eingabe mit einem Finger = Benutzer* $1, Eingabe\ mit\ n\ Fingern =$ *Benutzer n - verwendet werden, um zwischen Benutzern zu differenzieren.*

- **Warum?**
  In der Pilotstudie wurde deutlich, dass durch eine Identifikation der Benutzer die Möglichkeit gegeben ist, zusätzliche Interaktionsmöglichkeiten zu definieren. Eine davon wäre die Erweiterung von Casual Games, wie etwa das evaluierte Spiel *Pegs*. So könnten im Spiel mehrere Benutzer unterstützt und zwischen den Benutzern unterschieden werden, damit eine Art Wettkampf entstünde.

  Wie bereits erwähnt, ist jedoch eine Identifikation ohne zusätzliche Hardware grundsätzlich nicht möglich. In dieser Heuristik wird daher ein simpler Algorithmus vorgestellt, um eine Identifikation zu ermöglichen.

- **Wann?**
  Ein Einsatz in Casual Games wird als sinnvoll angesehen. Aufgrund der nachfolgend angeführten Nachteile muss ein Einsatz in anderen Applikationen von Fall zu Fall analysiert werden.

- **Wie?**
  Formel 6.2 zeigt einen simplen Algorithmus, wie jeder Benutzer trotzdem eindeutig identifiziert werden könnte: Benutzer 1 nimmt dabei für die Interaktion immer einen Finger einer Hand, Benutzer 2 nimmt zwei Finger, und so weiter.

$$Benutzer_i = i\ Finger\ pro\ Touch, \quad i = 1 \ldots 5 \tag{6.2}$$

So kann zumindest in Casual Games eine Unterscheidung der Benutzer getroffen werden. Ein Einsatz in anderen Bereichen muss jedoch auf Grund der sich ergebenden Nachteile applikations-spezifisch analysiert werden:

  - Gesten sind nicht mehr mit einer Hand alleine möglich
  - Gesten sind auf zwei Berührungsflächen reduziert (pro Hand ein Berührungspunkt)
  - Benutzeridentifikation ist mit fünf Benutzern beschränkt
  - Natürlichkeit der Interaktion wird beeinträchtigt
  - Präzision der Eingaben wird beeinträchtigt

- **Adressiertes Problem.**
  Es ist nicht möglich zwischen Benutzern zu differenzieren. Die Heuristik adressiert das Problem der fehlenden Identifikation der Benutzer. (4a).

6.3 Zusammenfassung der vorgestellten Heuristiken

Analysiert man die vorgestellten Methoden, lässt sich feststellen, dass für jedes identifizierte Problem mindestens eine Verbesserungsmöglichkeit in Form einer Heuristik vorgestellt wurde. Es existiert daher für jedes Problem mindestens ein Lösungsansatz. Dies zeigt auch Tabelle 6.1, welche die Heuristiken und deren Intentionen gegenüber stellt. Aus dieser Tabelle wird weiters ersichtlich, dass die meisten Heuristiken für das Problem der unbeabsichtigt, alleine getriggerten Gesten gefunden wurden. Vor allem hinsichtlich der erwähnten unpräzisen Berührungspunkte, die eine Ursache für das Problem darstellen, wurden mehrere Heuristiken - wie beisielsweise adaptive Touch-Ziele basierend auf Iceberg-Tips - vorgeschlagen.

In wie weit sich die Heuristiken eignen die identifizierten Mängel zu verbessern beziehungsweise diese zu beheben, wird in Kapitel 8 - basierend auf einer Evaluierung in Form eines User Tests - vorgestellt.

6.3.1 Kombination und Abhängigkeiten der Heuristiken

Bei Betrachtung der vorgestellten Methodiken zur Verbesserung der Multi-User Anwendung fällt auf, dass diese einerseits Abhängigkeiten zwischen einander aufweisen, sowie andererseits gleiche (Teil-)Probleme adressieren und daher die Möglichkeiten der Kombination sowie des gegenseitigen Ausschließens analysiert werden müssen.

	Heuristik 1	Heuristik 2	Heuristik 3	Heuristik 4	Heuristik 5	Heuristik 6	Heuristik 7	Heuristik 8	Heuristik 9	Heuristik 10	Heuristik 11	Heuristik 12	Heuristik 13	Heuristik 14
(1) Unbeabsichtigt, gemeinsam getriggerte Gesten			x							x				
(2) Unbeabsichtigt, alleine getriggerte Gesten	x			x	x	x		x	x	x				
(3) Fehlen von Grenzen der Arbeitsbereiche							x	x			x	x		
(4) Fehlende Identifikation der Benutzer													x	x
(5) Erweitertes Gesten-Repertoire		x												

Tabelle 6.1 – Heuristiken versus Probleme

Hinsichtlich der Abhängigkeiten lässt sich feststellen, dass Heuristik 8 unbedingt angewendet werden soll, wenn Heuristik 7 nicht implementiert beziehungsweise nicht als notwendig erachtet wird. So sollen in diesem Fall Obergrenzen für die Vergrößerung von Objekten eingeführt werden: Ein unbeabsichtigtes Vergrößern - und damit einhergehend das Verdecken der räumlichen Arbeitsfläche eines anderen Benutzers - wirkt für diese störend. Die beobachteten Personen in [PKS+08] empfanden diesen Konflikt bei Multi-User-Anwendung als am unangenehmsten. Diese beiden Heuristiken sind jedoch nicht als sich gegenseitig ausschließend zu sehen, sondern können ebenso in Kombination miteinander verwendet werden. Heuristik 7 kann auch verwendet werden, wenn Heuristik 8 eingesetzt wird.

Eine ähnliche Relation lässt sich bei Heuristik 1 mit Heuristik 4, Heuristik 5, Heuristik 6 und Heuristik 9 feststellen: Wird Heuristik 1 nicht verwendet, soll zumindest ein Teil der vier weiteren angeführten Heuristiken implementiert werden. Diese Abhängigkeit ergibt sich, da alle fünf Heuristiken ein ähnliches Problem adressieren. Unbeabsichtigte Berührungspunkte führten in der Pilotstudie dazu, dass eine Pan-Geste ausgeführt wurde und so die Sicht auf die gesamte Applikation geändert wurde. Heuristik 1 setzt dabei

an einer anderen Stelle als die anderen vier an: Die Pan-Geste an sich soll so verändert werden, dass sie sich signifikant von den existierenden Gesten unterscheidet. Die anderen vier Heuristiken versuchen hingegen das Auftreten von unbeabsichtigten Berührungspunkte zu minimieren, indem beispielsweise Objekte groß genug für die menschliche Hand designed werden. Unterscheidet sich daher die Pan-Geste nach Einsatz von Heuristik 1 bereits von den anderen Gesten, so haben unbeabsichtigte Berührungspunkte nicht mehr die gleiche problematische Auswirkung wie zuvor. Eine Kombination der fünf Heuristiken ist dabei jedoch dennoch möglich, da sich die Heuristiken keinesfalls widersprechen. Wird jedoch beispielsweise aus Gründen der Verringerung der Natürlichkeit der Interaktion Heuristik 1 nicht eingesetzt, so können unbeabsichtigte Berührungspunkte zu Konflikten führen und es sollte in diesem Fall die Einsetzbarkeit von Heuristik 4, Heuristik 5, Heuristik 6 und Heuristik 9 analysiert werden. Dies impliziert ebenfalls, dass eine Kombination dieser vier Ansätze als sinnvoll erachtet wird.

Weiters zeigen Heuristik 4 und Heuristik 9 eine Abhängigkeit: Heuristik 9 kann nur verwendet werden, wenn auch Heuristik 4 eingesetzt wird, da Iceberg-Tipps für adaptive Touch-Objekte eine Grundlage darstellen.

Die nachstehende Abbildung 6.4 illustriert die vorgestellten Abhängigkeiten und Relationen.

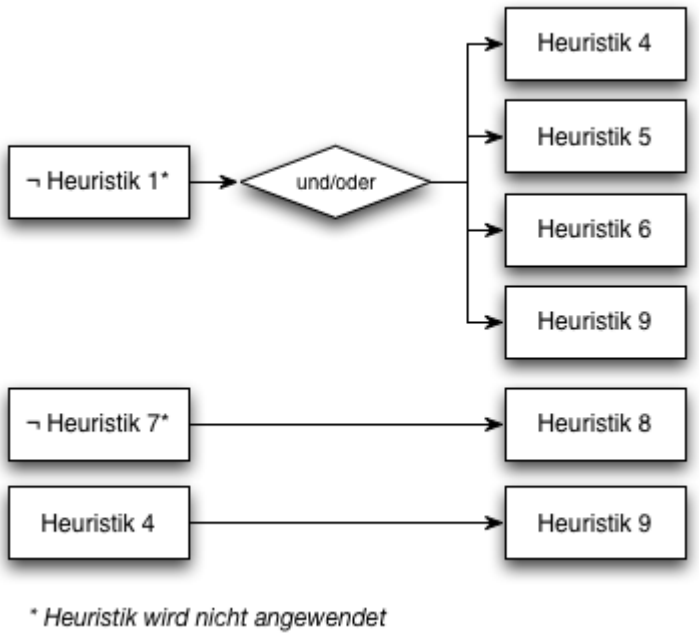


Abbildung 6.4 – Abhängigkeiten und Relationen der Heuristiken

6.3.2 Grenzen der vorgestellten Heuristiken

Ohne jegliche weitere Evaluierung fällt bei Betrachtung der vorgestellten Heuristiken auf, dass diese vor allem hinsichtlich dem Manko der Identifikation der Benutzer an ihre Grenzen stoßen: Heuristik 14 eignet sich etwa nur für den Einsatz in Casual Games. Eine Benutzeridentifikation würde jedoch zusätzliche Interaktionsmöglichkeiten - wie beispielsweise „kooperative“ Gesten -, Möglichkeiten der Regelung von Zugriffsberechtigung und User Interfaces, die mit besonderen Eigenschaften für Multi-User aufwarten können, bieten:

[DL01, Seite 219] führt etwa aus, dass durch eine unabhängige Interaktion, also durch eine Identifikation der Personen, kollaborative Tätigkeiten verbessert werden können und führt diese Eigenschaft in seinen Anforderungen an Multitouch-Umgebungen (siehe Kapitel 2.1.4) an. So definiert [RMHPW06] durch die Erkennung der Benutzer „kooperative Gesten“ und ermöglicht damit neue Funktionalitäten und Interaktionsmethoden im Kontext von Multitouch-Umgebungen mit Multi-User Anwendung. Als kooperative Gesten werden hierbei Interaktionen verstanden, welche die Fähigkeit besitzen, Gesten von mehreren Benutzern als einzelne, gemeinsame Geste mit einer adaptierten Bedeutung zu interpretieren. Diese wären beispielsweise bei Interaktionen, die wichtige, „kritische“ Aktionen - wie etwa das Löschen einer Vielzahl von Daten oder das Beenden der Applikation - hervorrufen, sinnvoll. So könnten alle Beteiligten simultan mit dem Tisch interagieren, um eine Lösch-Geste auszuführen. Als Vorteile dieser Form der Kooperation nennt [RMHPW06, Seite 1201 - 1202] folgendes:

- Kollaboration wird erhöht - eine engere Zusammenarbeit wird geführt, was zu Gruppendynamiken und verbessertem Teamwork führt.
- Wichtige Gesten werden bewusst ausgeführt - beispielsweise für destruktive Aktionen der Applikation wie etwa das Schließen der Applikation oder das Löschen von Daten.
- Reichweite auf großen Displays wird verbessert - ein Benutzer kann das Ziel auswählen, ein anderer die Aktion spezifizieren.
- Implizite Zugriffsberechtigungen sind möglich - nur ein spezifischer Benutzer kann ein Objekt bearbeiten, die anderen können nur lesend darauf zugreifen.
- Unterhaltung - kooperative Gesten machen Spaß.

Analysiert man diese Ausführungen über „kooperative Gesten“ und die Probleme, die bei der Multi-User Anwendung auftraten, so fällt auf, dass ein

Einsatz dieser auch für „globale“ Gesten, wie Zoom oder Pan, sinnvoll erscheint. Zoom- und Pan-Gesten hatten in der Evaluierung der Multi-User Anwendungen diverse Probleme verursacht (Problem1a und 2b): Es wurde beispielsweise durch mehrere unbeabsichtigte Berührungspunkte der Zoom-Faktor der gesamten Applikation verändert und so Personen in individuellen Phasen während einer Kollaboration gestört. Diese beiden Gesten können daher in Multi-User Umgebungen ebenfalls - zu den von [RMHPW06] angeführten - als „kritische Aktionen“ angesehen werden. Ein Einsatz dieser kooperativen Gesten für „globale“ Operationen würde daher einige Probleme der Multi-User Anwendung lösen.
Jedoch muss im Gegensatz zu der in [RMHPW06] erforderlichen Verwendung eines DiamondTouch Tisches, eine Identifikation der Benutzer nicht zwingend notwendig sein. Es würde bereits ausreichen, eine Interaktionsmöglichkeit zu definieren, durch welche bei gemeinsamer Interaktion „kritische“ Gesten getriggert werden können. Ein simples Beispiel wäre die Definition eines Buttons in den einzelnen Arbeitsbereichen der Benutzer. Erst wenn diese Buttons gleichzeitig betätigt werden, wäre das Ausführen einer Zoom-Geste möglich. Dadurch werden diese Gesten, beispielsweise sämtliche Teilnehmer beeinflussende Gesten, bewusst ausgeführt.

[RMRS+04] verwendet einen ähnlichen Ansatz, um Zugriffsberechtigungen auf elektronische Dokumente zu regeln. Es wird dabei angeführt, dass bei kollaborativen Tätigkeiten an einem Tisch persönliche, private Arbeitsflächen gebildet werden. Ebenso würden sich bei Gruppenarbeiten sowohl Phasen der aktiven Kollaboration, als auch Phasen individueller Aktivitäten abzeichnen. Der Übergang dieser Phasen wird in [RMRS+04] untersucht und soll vereinfacht werden: Die Autoren stellen Gesten vor, welche die Zugriffsrechte elektronischer Dokumente während dieser Phasen regeln. So können Dokumente durch spezielle Interaktionsmuster während kollaborativer Phasen öffentlich zugreifbar gemacht werden. In privaten Phasen besitzt etwa nur der Autor Schreibrechte.

In [REE+05] wird des Weiteren eine Methode vorgestellt, um die Benutzeridentifikation für ein adaptives User Interface zu benutzen. Es werden dabei Komponenten einer Anwendung definiert, die Informationen über die Identität des Benutzers als Parameter verwenden. Dadurch ist es möglich diese Komponenten zu individualisieren und dynamisch an Personen oder Gruppen anzupassen. [REE+05, Seite 1127]. Beispielsweise könnte in einer Geschäftsbesprechung das Erscheinungsbild, der Inhalt und die Interaktionsmöglichkeiten einer Komponente angepasst werden, je nachdem ob ein Manager oder etwa ein Spezialist damit interagiert.

Zusammengefasst ergeben sich daher durch eine Benutzeridentifikation zusätzliche Möglichkeiten: komplexe, kooperative Gesten und neue Interakti-

onsmethoden (wie beispielsweise in [RMHPW06]), Regelung von Zugriffsberechtigungen (wie beispielsweise in [RMRS+04]) sowie für Multi-User Anwendung optimierte Interfaces (wie beispielsweise in [REE+05]).

[RMHPW06], [RMRS+04] sowie [REE+05] verwenden einen DiamondTouch-Tisch (siehe Kapitel 4.1), um zwischen Benutzern zu differenzieren. Eine Identifikation ist in den meisten existierenden Multitouch-Umgebungen ohne den Einsatz zusätzlicher Hardware jedoch nicht möglich. Auch durch die beiden vorgestellten Heuristiken im Bereich der benutzerspezifischen Segmentierung ist eine Identifikation und damit das zur Verfügung stellen der erweiterten Möglichkeiten nur bedingt möglich: Das Einsatzgebiet von Heuristik 14 wurde auf Casual Games beschränkt. Durch Heuristik 13 wird es möglich Hände zu identifizieren, jedoch nicht die Benutzer selbst. Befinden sich die Benutzer auf separaten Kanten des Tisches, könnten die beiden Hände eines Benutzers wiederum durch Algorithmen, welche die Winkel der Finger ausnutzen, zusammengefügt werden. Es ergibt sich jedoch auf alle Fälle der gleiche Nachteil, der bereits in Kapitel 4.1 bei [DDSP08] beobachtet wurde. Die Benutzer können sich nicht mehr frei um den Tisch bewegen und die Anzahl der Teilnehmer ist auf vier Personen - entsprechend der vier Kanten des Tisches - beschränkt.

Kapitel 7

Prototyp

Im Zuge der Diplomarbeit wurde zur Evaluierung der vorgestellten Heuristiken ein Prototyp entwickelt. Als Ausgangspunkt wurde dafür eine Applikation implementiert: Diese dient im weiteren Verlauf als Basis des zu evaluierenden Prototyps. Für die Evaluierung wird diese Applikation so angepasst, dass für jedes Szenario der Studie[33] ein eigener, maßgeschneiderter Prototyp gestartet werden kann.

7.1 Ausgangspunkt des Prototyps

Als Ausgangspunkt des für die Evaluierung implementierten Prototyps wurde eine Foto- und Nachrichtenapplikation entwickelt. Diese ist der Applikation von Jefferson Han[34], der damit Multitouch wieder ins Gedächtnis rief, nachempfunden. Sie ermöglicht es nicht nur Fotos zu betrachten, sondern auch Nachrichten, die über die Nachrichtenplattform twitter[35] gesendet werden, darzustellen. So ist es möglich, dass über eine Multitouch Tastatur Suchbegriffe eingegeben werden und zu diesen Daten - Fotos von flickr[36] und Nachrichten von twitter - geladen werden. Mithilfe von Multitouch-Gesten können die Fotos und Nachrichten verändert und manipuliert werden. Es können dabei mehrere Tastaturen eingeblendet werden. Daher besteht die Möglichkeit simultan mehrere verschiedene Suchabfragen durchzuführen und mit den geladenen Daten gemeinsam zu interagieren. Der Prototyp eignet sich demnach zur Benutzung von mehreren Usern. Als zentrales Element im Prototyp dient das Menü-Objekt, über welches Daten geladen werden, Informationen zum Programm angezeigt werden können und die Applikation beendet werden kann.

[33] Im Rahmen der Evaluierung der Heuristiken wird ein User Test, in welchem die Benutzer mehrere Szenarien bearbeiten müssen, durchgeführt; mehr dazu in Kapitel 8.1.

[34] http://www.youtube.com/watch?v=QKh1Rv0PlOQ, zuletzt abgerufen am 19.02.2010.

[35] http://twitter.com/, zuletzt abgerufen am 19.02.2010.

[36] http://www.flickr.com/, zuletzt abgerufen am 19.02.2010.

Um eine bessere Vorstellung des Prototyps zu erhalten, zeigt Abbildung 7.1 das Interface der Applikation, wie sich dieses nach dem Laden von einigen Daten präsentiert. Augenscheinlich ist hierbei, dass die einzelnen geladenen Objekte zufällig gedreht werden, damit die Benutzer aufgefordert werden mit diesen zu interagieren und diese zum Beispiel zu drehen, um sie zu betrachten.

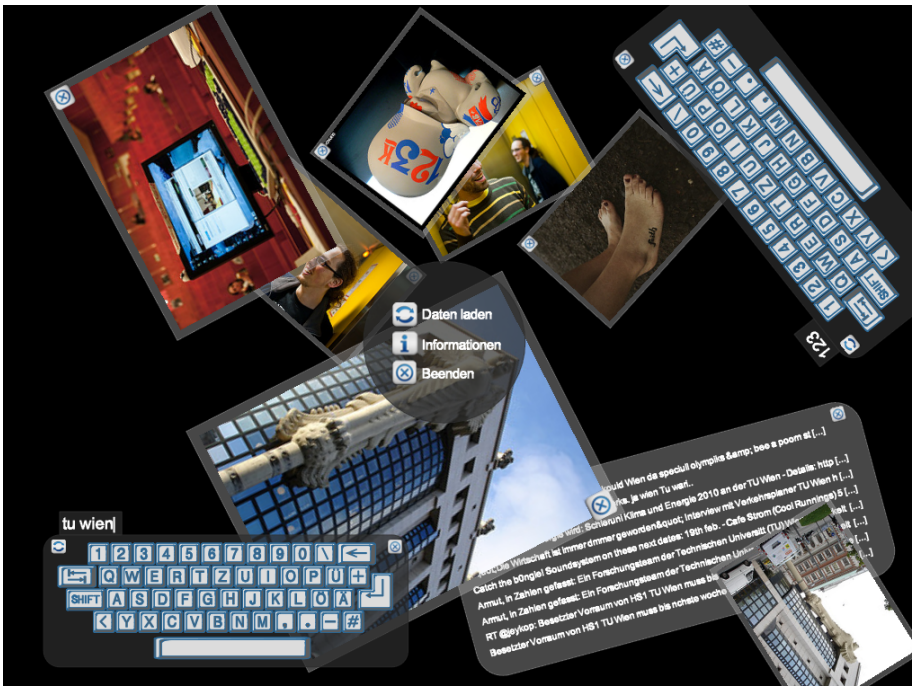


Abbildung 7.1 – Interface der Ausgangsapplikation des Prototyps

Ein typischer Programmablauf wäre beispielsweise wie folgt: Ein Benutzer klickt auf den Button „Daten laden“ und erhält eine Tastatur, die in der gleichen z-Rotation wie das Menü dargestellt wird. Danach werden Daten mittels Tastatur eingetippt. Mittels Drücken der „Enter“-Taste oder des Buttons zur Bestätigung des Ladens der Daten werden darauf hin relevante Fotos und Nachrichten zufällig am Screen angeordnet. Der Benutzer führt auf diesen Objekten Gesten aus und betrachtet die Fotos.

7.2 Prototyp der Evaluierung

Für die Evaluierung wurde die vorgestellte Applikation adaptiert. So ist es möglich, für jedes bei der Studie durchzuführende Szenario einen maßgeschneiderten Prototyp zu starten. Fotos werden nun nicht mehr via flickr geladen, sondern lokal von der Festplatte. Die Fotos werden sofort bei Start eines Tasks angezeigt und müssen - bei den meisten Tasks - nicht mehr via

Tastatur geladen werden. Eine zufällige Anordnung sowohl in der Reihenfolge als auch in der Rotation der z-Achse in der Mitte des Tisches soll dabei gewährleisten, dass Interaktion von verschiedenen Benutzern in geringem räumlichen Abstand zueinander statt findet. Für die diversen Szenarien werden zusätzlich beispielsweise unterschiedliche Fotos und Nachrichten angezeigt. In Tasks bei denen Fotos nicht ausgeblendet werden dürfen, wird kein Button zum Schließen der Fotos angeboten. Wie die Applikation nach Start eines bestimmten Tasks genau aussieht, kann den Vorbedingungen der Tasks, welche im Anhang zu finden sind, entnommen werden. Abbildung 7.2 illustriert exemplarisch wie sich die Applikation nach Start von Task 3[37] und einiger Interaktion zeigt. Hierbei werden beispielsweise keine Buttons zum Schließen der Fotos angezeigt und es ist keine Zoom- und Pan-Geste erlaubt.

Abbildung 7.2 – Interface des Prototyps nach Start von Task 3

7.2.1 Interaktion - Implementierte Gesten

Zur Interaktion mit dem Prototyp sind in diesem verschiedene Gesten zur Manipulation von Objekten sowie zur Änderung des Hintergrunds implementiert. Als Ausgangspunkt dienen dabei die in [Saf09, Seiten 45-69] definierten Modelle für Touchscreens und interaktive Umgebungen. Nachfolgend werden diese Gesten vorgestellt und näher erläutert:

- **Tippen, um etwas zu öffnen beziehungsweise zu aktivieren**
  - *Abbildung 2.1-1a*
  [Saf09, Seite 46] beschreibt, dass diese Geste immer eingesetzt werden soll, wenn Objekte aktiviert beziehungsweise Buttons gedrückt werden

[37] Die Aufgabe von Task 3 ist ein konkurrierendes Sortieren und Skalieren mehrerer Fotos: Jede Testperson bekommt ein Fotomotiv zugeteilt. Vier dieser Fotos müssen zu sich bewegt und entsprechend den Markierungen skaliert und sortiert werden.

sollen, da diese die einfachste manuelle Geste ist und den natürlichen Ersatz des Mausklicks darstellt. Im Prototyp wird diese Geste eingesetzt um etwa die Tastatur zu bedienen, Aktionen im Menü auszuwählen oder um Fotos zu schließen.

- **Ziehen, um ein Objekt zu bewegen**
  - *Abbildung 2.1-1b*
  Diese Geste soll immer eingesetzt werden, wenn Slider bedient oder Objekte auf eine andere Stelle am Screen bewegt werden und wird in der Applikation daher zur Bewegung sämtlicher Objekte, beispielsweise des Menüs oder der Fotos, verwendet. [Saf09, Seite 50].

- **Zwei Finger drehen, um ein Objekt zu drehen**
  - *Abbildung 2.1-1c*
  In den vorgeschlagenen Patterns wird diese Geste nicht angeführt, stellt jedoch einen de-fakto Standard dar um Objekte zu drehen und wird in dieser Art ebenfalls in den Microsoft Windows Touch Gestures[38] angeführt. Die Geste kann sowohl mit zwei Fingern einer Hand als auch mit zwei Fingern zweier Hände ausgeführt werden.

- **Zusammenziehen/Auseinanderziehen, um ein Objekt / den gesamten Screen zu verkleinern / vergrößern**
  - *Abbildung 2.1-1d (Verändern eines Objekts), Abbildung 2.1-2a (Verändern des gesamten Screens)*
  Dieses Interaktionsverhalten soll verwendet werden um die Größe eines Objekts oder des gesamten Hintergrunds zu verändern und findet in der Implementierung auf beinahe allen Objekten Anwendung. [Saf09, Seite 64].

- **Einen Finger am Hintergrund bewegen, um zu schwenken**
  - *Abbildung 2.1-2b*
  Diese Geste wird wiederum in den Microsoft Windows Touch Gestures angeführt und ermöglicht es den Hintergrund zu schwenken beziehungsweise am Hintergrund zu „scrollen“. Weiters führt [Saf09, Seite 52] diese Geste an, um den sichtbaren Arbeitsbereich zu verändern.

7.3 Implementierung

Wie bereits angeführt, wurde der Prototyp mithilfe des Frameworks MT4J implementiert. MT4J nimmt dem Programmierer dabei viel Arbeit ab, indem es die gebräuchlichsten Gesten, viele graphische Objekte, wie zum Bei-

[38] http://msdn.microsoft.com/en-us/library/cc872774.aspx#gestures, zuletzt abgerufen am 14.08.2010.

spiel Ellipsen oder Rechtecke und auch User Interface Komponenten, etwa Buttons oder Text Labels, bereits definiert und so Rapid Prototyping zulässt. Diese vordefinierten Objekte wurden zum Teil erweitert, so dass das Erscheinungsbild dieser individuell angepasst werden konnte. Beispielsweise wurde es so möglich eigene Buttons für das Schließen der Fotos und anderer Objekte zu definieren oder Farben anzupassen. Des weiteren wurden die bereits definierten Objekte verwendet, um darauf aufbauend komplexere Objekte - wie etwa die Statusleiste zur Ausgabe von Informationen über den Systemzustand oder das Objekt, welches die Twitter-Nachrichten beinhaltet - zu erstellen.

Die Inhalte der Applikation werden im Ausgangsprototyp, wie erwähnt, über die Plattformen twitter und flickr geladen. Dabei ist zu erwähnen, dass beide Plattformen offene APIs, welche auf REST basieren, anbieten. In Bezug auf twitter wird dabei JTwitter[39] eine Java Library der API[40], verwendet. Als Java Library für die flickr API[41] wird flickrj[42] eingesetzt.

Zusätzlich fiel im Zuge der Implementierung eine Schwachstelle der TUIO Tracker Applikation (Community Core Vision 1.2 - siehe Kapitel 3.4.3) auf: Blobs die außerhalb des kalibrierten Koordinatensystems der Applikation erkannt werden, wird der Nullpunkt als Koordinate zugewiesen. Dies bedeutet, wenn ein solcher Blob registriert wird, wird dieser Wert an die Applikation weitergeleitet. Dies hat zur Folge, dass etwa während der Ausführung einer Zieh-Geste das Objekt auf diesen Punkt referenziert. Beobachtet werden kann dies mit einem plötzlichen Verschwinden und Wiedererscheinen des Objekts am Nullpunkt. Zur Steigerung der Robustheit der gesamten Applikation wurde dieser Punkt im Prototyp ignoriert. Ein genereller Lösungsvorschlag wäre jedoch, dass Community Core Vision keine Punkte außerhalb des Systems aufzeichnet und somit in der Applikation die gleiche Routine eingeleitet wird, zu welcher ein entfernter Blob führen würde.

7.4 Berücksichtigung der identifizierten, kritischen Designaspekte

Bei der Implementierung des Prototyps wurde darauf geachtet, dass die in Kapitel 5.2 definierten kritischen Designentscheidungen in den Prototyp mit einflossen. Eine mangelnde Berücksichtigung dieser Aspekte kann zu Problemen in Multi-User Anwendungen führen. Gegen diese wurde deshalb bewusst verstoßen, um einerseits möglichst viele Probleme ausfindig machen

[39] JTwitter: http://www.winterwell.com/software/jtwitter.php, zuletzt abgerufen am 19.02.2010.

[40] twitter API: http://apiwiki.twitter.com/, zuletzt abgerufen am 19.02.2010.

[41] flickr API: http://www.flickr.com/services/api/, zuletzt abgerufen am 19.02.2010.

[42] flickrj: http://flickrj.sourceforge.net/, zuletzt abgerufen am 19.02.2010.

zu können und andererseits möglichst viele dieser Probleme durch Heuristiken verbessern zu können.
So wurde sowohl das Schwenken der Kamera mit einem Finger als auch das Verändern des Zoom-Faktors der Applikation implementiert. Bei der Zoom-Geste ist die Entfernung der beiden zu interpretierenden Berührungspunkte im Prototyp egal. Hinsichtlich Gesten, die auf Objekte angewendet werden, wurden sämtliche identifizierte, kritische Gesten - wie das Skalieren, das Rotieren und das Ziehen - implementiert. Beim Vergrößern der Objekte wurde darauf geachtet, dass auf keine obere Schranke oder ähnliches Rücksicht genommen wird. Diese Gesten finden in sehr vielen Applikationen Anwendung, stellen de-fakto Standards dar und werden etwa auch in [Saf09] als bewährte Lösung vorgeschlagen.

7.5 Implementierte Heuristiken

Um die in Kapitel 6.2 vorgestellten Heuristiken zu validieren, deren Eignung zur Beseitigung der Probleme der Multi-User-Anwendung (siehe Kapitel 5) zu überprüfen, sowie die beschriebenen Einsatzgebiete und die vorgeschlagenen Werte für Konstanten (etwa die vorgeschlagenen Werte für die Mindestgröße von skalierbaren Objekten) zu verifizieren, wurde der Prototyp erweitert beziehungsweise eine Version mit Heuristiken implementiert. Zur Gewährleistung einer vergleichenden Evaluierung des Prototyps, wurde diese Erweiterung so implementiert, dass der Prototyp sowohl mit als auch ohne den umgesetzten Heuristiken gestartet werden kann.

Die folgenden Heuristiken wurden im Prototyp implementiert:

- Heuristik 1: *Eine Pan-Geste soll sich in Multi-User Umgebungen signifikant von anderen Gesten unterscheiden.*

  Dies wurde so implementiert, dass eine Pan-Geste durch Ziehen zweier anstelle von einem Finger auf dem Hintergrund ausgeführt werden kann. Bei der Implementierung mit nur einem Finger wäre kein signifikanter Unterschied zu einer Zieh-Geste beziehungsweise einer Klick-Geste gegeben.

- Heuristik 3: *Verwendet eine Applikation eine Zoom-Geste, sollte eine adäquate maximale Entfernung der zwei, bei der Zoom-Geste auftretenden Berührungspunkte definiert werden.*

  Bei der Implementierung wurden die Ausführungen aus Kapitel 6.2 beachtet. So wird für die Verringerung des Zoom-Faktors (= $ZoomOut$) eine wesentlich größere maximale Entfernung als bei einer Erhöhung dieses Faktors (= $ZoomIn$) gesetzt. Als Parameter für die maximale Entfernung bei $ZoomIn$ wurde die Hälfte der Länge des Screens (also $\frac{1}{2}$ Meter) gewählt. Bei $ZoomOut$ sollen nur zwei Berührungspunkte

an den gegenüberliegenden Kanten nicht interpretiert werden. Es wurde daher in diesem Fall die maximale Entfernung mit $0,9$ Meter initialisiert. Die Evaluierung der Heuristik soll zeigen, ob diese Werte zweckmäßig sind, oder ob etwa die Geste dadurch als weniger intuitiv empfunden wird.

Anzumerken ist hierbei, dass ein Ansatz der Umsetzung dieser Heuristik bereits im Framework existiert. In diesem wurde als maximale Entfernung immer die Hälfte der Länge des Screens gewählt. Für die Prototyp-Version ohne Heuristiken wurde daher der entsprechende Code, welcher die Entfernung der beiden Berührungspunkte überprüft, entfernt.

- Heuristik 4: *Zur Vermeidung unbeabsichtigter Berührungspunkte aufgrund von unpräzisen Gesten, sollen „Iceberg Tips" für sämtliche selektierbaren Objekte verwendet werden.*

  „Iceberg Tips" wurden sowohl für sämtliche Buttons der Applikation, als auch für die Objekte der Applikation - etwa für Fotos und twitter-Nachrichten - umgesetzt. Die Größe des unsichtbaren Bereichs beträgt in der Applikation rund einen Zentimeter (bei einer Auflösung von 1024x768 und einer physischen Größe des Display von 100x70 cm; entsprechend der Beschreibung des Settings in Kapitel 3.3).

- Heuristik 5: *Zur Vermeidung unbeabsichtigter Berührungspunkte aufgrund von unpräzisen Gesten, sollen selektierbare Objekte mindestens eine Größe von* 1×1 *cm) aufweisen.*

  Sämtliche selektierbaren Objekte weisen im Prototyp diese entsprechende Größe auf. Dies wird auch bei einer Verringerung des Zoom-Faktors sichergestellt.

- Heuristik 6: *Zur Vermeidung unbeabsichtigter Berührungspunkte aufgrund von unpräzisen Gesten, sollen skalierbare Objekte nicht unter eine Größe von* 3×3 *cm verkleinert werden können.*

  Dies wird im Prototyp wiederum auch bei Verringerung des Zoom-Faktors sichergestellt. Da im Prototyp die Fotos bei einigen Szenarien der Evaluierung über Buttons verfügen, wird ebenso sichergestellt, dass für diese Buttons Heuristik 5 eingehalten wird. Dies bedeutet, dass skalierbare Fotos nicht bis zu der Grenze von 3×3 *cm* verkleinert werden können.

- Heuristik 8: *Wird Heuristik 7 nicht angewandt, soll eine Obergrenze für die Vergrößerung von Objekten eingeführt werden.*

  Als Obergrenze wurde die Hälfte der Screengröße gewählt. Diese Obergrenze wird auch bei Erhöhung des Zoom-Faktors gewährleistet.

- Heuristik 10: *Wird ein Objekt aktiv durch eine Geste manipuliert, sollen keine Gesten am Hintergrund (zum Beispiel Zoom oder Pan) möglich sein.*

  Dies wurde implementiert wie in der Definition beschrieben.

- Heuristik 11: *Um zu vermeiden, dass andere Benutzer Gesten korrumpieren, sollen Objekte gesperrt werden, sobald eine Geste eindeutig identifiziert wurde.*

  Diese Heuristik wurde prinzipiell entsprechend der Definition implementiert. Dies bedeutet, dass alle aktivierten Gesten eines Objekts deaktiviert werden, sobald eine bestimmte Geste eindeutig identifiziert wurde.

  Eine Ausnahme bildet hierbei jedoch die Zieh-Geste, weil eine Relation zwischen dieser und der Skalier- und Rotier-Geste besteht: Verschiebt man beispielsweise ein Objekt mit einem Finger, das heißt es findet eine Zieh-Geste statt, könnte immer noch eine Skalier- beziehungsweise Rotier-Geste folgen. Ungenau ausgedrückt könnte man feststellen, dass Skalier- und Rotier-Gesten in gewissem Maße eine Zieh-Geste erweitern: Die Zieh-Geste wird dabei mit zwei Fingern, die sich in eine andere Richtung bewegen ausgeführt.

  Abhilfe schafft die Analyse der zeitlichen Sequenz: Wird in einem gewissen zeitlichen Abstand zu einer eingangs identifizierten Zieh-Geste kein weiterer Berührungspunkt auf dieses Objekt aufgezeichnet, so gilt die Geste als eindeutig identifiziert. In der weiteren Folge können Skalier- und Rotier-Geste des Objekts für den Benutzer gesperrt werden. Dadurch ist die Möglichkeit der Ausführung einer Skalier- oder Rotier-Geste immer noch gegeben. Würde man diese beiden Gesten sofort deaktivieren, wäre dies nicht gewährleistet.

  In der Implementierung wurde diese zeitliche Sequenz-Analyse berücksichtigt, indem die Anzahl an gefeuerten Änderungen der Zieh-Geste gezählt werden. Erst nach einer bestimmten Anzahl an Änderungen, das heißt erst wenn die Zieh-Geste eine bestimmte Zeit aktiv ausgeführt wird, werden die entsprechenden Gesten gesperrt. Als Anzahl an Veränderungen (= `updateCount`) wurde aufgrund von ersten informellen Tests der Wert 24 gewählt.

```
public void cursorUpdated(InputCursor m,
                AbstractCursorInputEvt posEvent) {

   IMTComponent3D comp = posEvent.getTargetComponent();
   dragContext.updateDragPosition();
   updateCount++;
```

```
   if (updateCount == 24 && MT4jSettings.heuristics) {
      logger.debug("drag in progress: " + updateCount);
      logger.debug("disallowing scale, rotate and tap");
      for (AbstractComponentProcessor ip
             : object.getInputProcessors()){
         if (ip instanceof ScaleProcessorHeuristic)
             object.unregisterInputProcessor(ip);
         if (ip instanceof RotateProcessorHeuristic)
             object.unregisterInputProcessor(ip);
    }
}

  this.fireGestureEvent(new DragEvent(
    MTGestureEvent.GESTURE_UPDATED, comp, m,
       dragContext.LastPos(), dragContext.NewPos()));

}
```

Analysiert man diese Implementierung im Detail, so fällt auf, dass die eindeutige Identifikation nicht direkt von der Zeit abhängig ist, sondern von der Aktivität der Ausführung der Geste durch den Benutzer. So ist es etwa immer noch möglich, eine Skalier- oder Rotier-Geste auszuführen, wenn der eine Berührungspunkt der Zieh-Geste beispielsweise mehrere Minuten unverändert bleibt, da dies zu keinem Aufruf der Methode `cursorUpdated()` führt. Somit wird der Benutzer in der Ausführung der Skalier- und Rotier-Geste nicht unter „Zeitdruck“ gesetzt.

Zusätzlich gilt es bei der Implementierung der Heuristik zu beachten, dass auch „Eltern-“ beziehungsweise „Kind-Elemente“ eines Objekts gesperrt werden müssen. Besitzt beispielsweise ein Objekt einen Button, so gilt es nicht nur die Gesten des Objekts zu sperren, sondern ebenfalls die des Buttons. Dies gilt auch vice versa: Betätigt man den Button, sollten die Operationen des in Relation stehenden Objekts gesperrt werden.

Durch die Implementierung dieser Heuristik wird unter anderem das in Abbildung 5.2 aus Kapitel 5.2.1 erörterte Problem gelöst: Dieses beschäftigt sich mit der fehlenden Koordination zweier Benutzer und mit der aufgrund mangelnder Robustheit der Eingabeverarbeitung hervorgerufenen Fehlinterpretationen der Absichten der Benutzer. Dieses Problem gilt als wesentliches Problem der Mehrbenutzer-Anwendung und findet in zahlreichen Publikationen zum Thema Er-

wähnung[43].

Zusätzlich sei erwähnt, dass ein Ansatz dieser Heuristik bereits in MT4J implementiert ist. Dieser ignoriert jedoch lediglich zusätzliche Berührungspunkte, die bei einer Skalier- beziehungsweise Rotier-Operation auftreten. Für die Evaluierung des Prototyps ohne Heuristiken wurde daher der entsprechende Code entfernt.

- Heuristik 12: *Um zu vermeiden, dass andere Benutzer Gesten korrumpieren, sollte objektspezifisch segmentiert werden: Ein Berührungspunkt auf ein Objekt x interagiert nicht mit einem zusätzlichen Berührungspunkt auf ein Objekt y.*

  Diese Heuristik ist bereits implizit durch die ereignis-getriebene Architektur des verwendeten Frameworks gegeben. Daher ist diese Heuristik in beiden Versionen des Prototyps - der Version mit beziehungsweise der ohne Heuristiken - enthalten. Eine Evaluierung dieser Heuristik durch einen vergleichenden User Test ist dadurch nicht möglich.

Im Prototyp nicht unterstützte Heuristiken. Bei der Implementierung ausgespart wurden demnach Heuristik 2, Heuristik 7, Heuristik 9, Heuristik 13 und Heuristik 14:

- Heuristik 2: *Jede zusätzliche Geste soll sich signifikant von existierenden unterscheiden.*

  Bei dem von dieser Heuristik adressierten Problem der Erweiterung des Gesten-Repertoires handelt es sich um ein bekanntes Problem, welches ebenfalls in der Literatur angeführt wird. Beispielsweise erwähnt [DSA09, Seite 101], dass die Gesamtheit an Gesten nur unter dem Abstrich der Verringerung der Robustheit der Applikation erweitert werden kann. Die Lösung dieses Problems liegt auf der Hand - Gesten sollen sich voneinander signifikant unterscheiden - und es bedarf daher keiner weiteren Evaluierung dieser Heuristik.

- Heuristik 7: *Sind Grenzen zwischen Benutzern erforderlich, sollen private und öffentliche Arbeitsbereiche explizit definiert werden.*

  Diese Heuristik wurde nicht implementiert, da zuvor keine detaillierte Analyse über die Existenz von individuellen und kollaborativen Arbeitsphasen bei der Interaktion mit dem Prototyp stattgefunden hat. Zur Behebung des adressierten Problems, wurde daher nur Heuristik 8 implementiert.

- Heuristik 9: *Zur Vermeidung unbeabsichtigter Berührungspunkte aufgrund von unpräzisen Gesten, sollen zusätzlich zu Iceberg-Tips anpassungsfähige Touch-Objekte realisiert werden.*

[43] vgl. etwa [PKS+08] und [DSA09].

Adaptive Touch-Objekte wurden aufgrund des Fehlens von sequentiellen Mustern beim Interagieren mit dem Prototyp bei der Implementierung ausgespart. Wegen dieses Fehlens können keine sinnvollen Wahrscheinlichkeiten geschätzt werden. Einzig bei der Tastatur könnten solche Anpassungen stattfinden: Jedoch handelt es sich hierbei um sehr komplexe Algorithmen, deren Implementierung den Rahmen dieser Arbeit überschreiten würde. Algorithmen dieser Form werden beispielsweise bei der Texteingabe am Apple iPhone verwendet [Saf09, Seite 44]. [HBBS08] stellt beispielsweise ebenfalls einen Algorithmus dieser Form vor.

- Heuristik 13: *Ist eine Identifikation der Benutzer zwingend erforderlich, so können die Finger-Orientierung und die natürlichen Einschränkungen der menschlichen Hand verwendet werden, um zwischen Benutzern zu differenzieren.*

  Ebenso wie bei der vorigen Heuristik würde die Implementierung dieses Algorithmus über den Rahmen dieser Arbeit hinaus gehen, da dieser sehr komplex ist. [DSA09] beschreibt zusätzlich zwar Konfliktsituationen, jedoch werden keine Fehlerraten bei der Identifikation in einer Software angeführt. Zusätzlich sind im implementierten Prototyp keine zusätzlichen Interaktionsformen - wie beispielsweise kooperative Gesten - erforderlich.

- Heuristik 14: *Ist eine Identifikation der Benutzer zwingend erforderlich (beispielsweise in Spielen), so kann ein simpler Algorithmus - der Form Eingabe mit einem Finger = Benutzer* $1, Eingabe\ mit\ n\ Fingern = Benutzer\ n$ *- verwendet werden, um zwischen Benutzern zu differenzieren.*

  Wie in der Beschreibung der Heuristik angeführt, ist ein Einsatz dieser nur in Casual Games sinnvoll und sie wurde deshalb in der Implementierung nicht berücksichtigt.

7.5.1 Abdeckung der Probleme

Durch die implementierten Heuristiken werden die in Tabelle 7.1 angeführten Probleme adressiert. Nicht implementierte Heuristiken werden dabei nicht angeführt. Es werden demnach alle Probleme mit mindestens einer Heuristik, bis auf das Problem des erweiterten Gesten-Reperoires und der fehlenden Identifikation der Benutzer, abgedeckt. Die mögliche Verbesserung dieser Problemgruppen beziehungsweise der Eignung der entsprechenden Heuristiken zur Problemlösung kann daher evaluiert werden.

	Heuristik 1	Heuristik 3	Heuristik 4	Heuristik 5	Heuristik 6	Heuristik 8	Heuristik 10	Heuristik 11	Heuristik 12
(1) Unbeabsichtigt, gemeinsam getriggerte Gesten		x					x		
(2) Unbeabsichtigt, alleine getriggerte Gesten	x		x	x	x		x		
(3) Fehlen von Grenzen der Arbeitsbereiche						x		x	x
(4) Fehlende Identifikation der Benutzer									
(5) Erweitertes Gesten-Repertoire									

Tabelle 7.1 – Implementierte Heuristiken und deren adressierte Probleme

7.6 Einbindung der Heuristiken in die Architektur

Bei Analyse der vorgestellten Heuristiken erscheint eine Berücksichtigung dieser auf Ebene eines Frameworks sinnvoll. Beispielsweise sollte es die Möglichkeit geben, für Touch-Objekte „Iceberg Tips“ (sowie deren Größe) mithilfe des Frameworks zu setzen. So wurde bereits im Zuge der Implementierung, da es sich um eine quelloffene Software handelt, MT4J um einige Möglichkeiten hinsichtlich der Heuristiken erweitert: Der Benutzer des Frameworks kann nun etwa entscheiden welche Art der Pan-Geste er einsetzt oder ob er „Iceberg Tips“ verwenden will. Zusätzlich wurden Heuristiken, wie beispielsweise die Maximalgröße bei der Vergrößerung oder das Sperren des Objekts bei der eindeutigen Identifikation einer Geste, fix im Framework verankert.

Ruft man sich die Architektur von TUIO wieder ins Gedächtnis, so fällt auf, dass die Implementierung daher im Rahmen der *„TUIO client application“*, jedoch auf Ebene eines Frameworks statt fand. Dies illustriert Abbildung 7.3.

Die Einbindung auf Ebene der Client-Applikation kann argumentiert werden, weil es sich bei den Heuristiken um applikations-spezifische Lösungen handelt. Entwirft man eine Applikation, so muss man zuerst applikationsspezifisch analysieren, etwa welche Heuristiken man einsetzen will oder auch welche Maximal- oder auch Minimalwerte man setzt. Da es sich um keine „globalen“, immer geltende Lösungen handelt, ist eine Einbindung auf Ebene der „TUIO tracker application“ nicht sinnvoll.

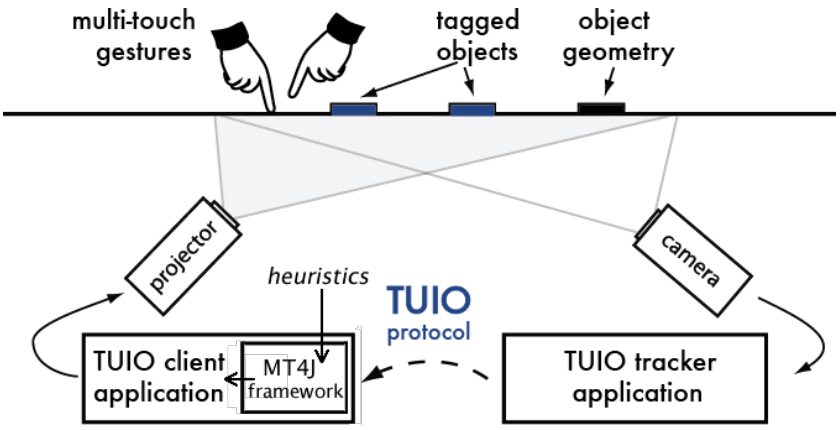


Abbildung 7.3 – Architektur von TUIO mit implementierten Heuristiken[44]

[44] Adaptiert nach http://www.tuio.org/images/diagram.png, zuletzt abgerufen am 28.06.2010.

Kapitel 8

Evaluierung der Heuristiken

In diesem Kapitel wird die durchgeführte Evaluierung der Heuristiken näher erläutert sowie deren Ergebnisse angeführt. Dabei wird sowohl die Methodik des User Tests in Form eines Testplans erläutert, als auch die Ergebnisse des Tests angeführt und Rückschlüsse auf die entwickelten Heuristiken gezogen.

8.1 Testplan

Für die Evaluierung des Prototyps wurde am Beginn der in diesem Kapitel vorgestellte Testplan erstellt, der die Methodik und Vorgehensweise während des Tests beinhaltet. Als Grundlage für diesen Testplan dienen die vorgestellten Vorgehensweisen und Methodiken von [RC08].

8.1.1 Ziel des Tests

Ziel der Evaluierung ist es, zum einen die in der Pilotstudie lokalisierten Probleme noch einmal zu identifizieren. Es sollen Fragen beantwortet werden wie etwa: *Treten die identifizierten Probleme wieder auf? Welche zusätzlichen Probleme können identifiziert werden?*

Zum anderen soll in der Studie die Zweckmäßigkeit der Verbesserungsmaßnahmen zur Beseitigung der Probleme beziehungsweise zur Verringerung ihrer Auswirkungen überprüft werden: *Treten die Probleme nach Implementierung der Heuristiken wieder auf? Welche Probleme treten nicht mehr (noch immer) auf? Können die Szenarien durch Implementierung der Heuristiken durch die Probanden schneller abgearbeitet werden, weil etwa weniger Probleme auftreten? Sind die definierten Konstanten in den Heuristiken adäquat? Sind die Probanden bei Benutzung des Prototyps mit den Heuristiken subjektiv „zufriedener“ als bei Benutzung des Prototyps ohne Optimierung?*

Ziel ist daher die Beobachtung der Probleme der Mehrbenutzer-Anwendung durch qualitative Verfahren. Außerdem sollen die Zeiten erfasst werden, welche die Testpersonen zur Erfüllung der Szenarien benötigen, um diese

zwischen den beiden Versionen vergleichen zu können. Bei Analyse obiger Fragen, ergibt sich zusätzlich das Ziel der Messung der subjektiven Zufriedenheit der Probanden.

8.1.2 Testdesign

Art des Tests. Nach der Einteilung von Jeffrey Rubin bildet den Rahmen für die Evaluierung ein „Exploratory Test“. Diese Art eines User Tests findet früh im Design-Prozess statt, in der es gilt, wichtige grundlegende Fragen das Design betreffend zu klären. Ziel ist es dabei qualitative Aussagen zur Mensch-Computer Interaktion zu machen. [RC08, Seiten 29 - 34].
Der „Exploratory Test“ wird in Verbindung mit einem „Comparison Test“ ausgeführt. Dieser kann laut [RC08, Seite 37] in jeder Phase des Design-Prozesses eingesetzt werden und dient allgemein zum Vergleich verschiedener Design-Konzepte beziehungsweise zum Vergleich mit Konkurrenzprodukten. Im Rahmen der Evaluierung wird diese Methodik eingesetzt, um die Version des Prototyps mit Heuristiken mit der ohne Heuristiken zu vergleichen. Dadurch sollen etwa mögliche Unterschiede im Auftreten von Problemen der Multi-User-Anwendung offenbart werden.
Dabei werden die beiden Versionen des Prototyps in zwei verschiedenen Test-Sessions getestet. Der Testplan sowie die Probanden bleiben dabei gleich. Dadurch wird die Möglichkeit eines Vergleichs gewährleistet, da Metriken und Beobachtungen miteinander verglichen werden können. Um Lerneffekte von der einen Session zur nächsten gering zu halten, werden die beiden Tests in einem Abstand von zehn Tagen statt finden.
Zusätzlich wird, zur Verminderung dieser Lerneffekte, in den einzelnen Test-Sessions die Reihenfolge der verwendeten Prototypen balanciert: In der zeitlich gesehen ersten Session wird die Version des Prototyps ohne implementierten Heuristiken abwechselnd mit der Version mit Heuristiken evaluiert.

Ausführung der Szenarien. Szenarien entsprechen den Aufgaben, welche die Probanden in der Durchführung des Tests erledigen müssen. Hinsichtlich der Abarbeitung dieser Szenarien wird die Methodik des „Within Subject Design“ gewählt. Dies bedeutet, dass im Gegensatz zu „Between Subject Design“ jede Testperson in jeder Bedingung getestet wird[45]. Jede Testperson führt also jeden Szenario aus. Dadurch werden weniger Testpersonen benötigt. Dem Nachteil der Lerneffekte zwischen den Szenarien wird durch das Balancieren der Szenarien mittels Latin Square Design entgegen gewirkt. Dadurch findet jede Testperson eine andere Reihenfolge der Abarbeitung der Szenarien vor und jede Reihenfolge der Abarbeitung findet nur genau einmal statt. Das verwendete Latin-Square befindet sich im Anhang.

[45] Siehe hierzu etwa [Nie93, Seite 178].

Lernphase. Weiters gibt es vor jeder Test-Session eine Lernphase, welche mit dem Ausgangsprototyp durchgeführt wurde (siehe Kapitel 7.1). Diese dient dazu, Lerneffekten, welche sich sowohl durch das Anwendung findende „Within Subject Design“ als auch durch den vergleichenden Test ergeben, zu kompensieren. In dieser Lernphase wird es den Probanden ermöglicht, die Applikation durch ein „freies“, ungezwungenes Benutzen kennen zu lernen. Ebenso wird die Art der Implementierung der Gesten den Testpersonen mithilfe eines Einführungsblattes vorgestellt. Zur Verhinderung von gruppendynamischen Effekten vor der eigentlichen Ausführung des Tests, wird diese Phase von jeder Testperson einer Gruppe getrennt absolviert.

Thinking Aloud. Die Testpersonen werden aufgefordert während des Tests „laut“ zu denken - also ihre Gedanken auszusprechen. Diese Methodik, welche als „Thinking Aloud“ bezeichnet wird, wird eingesetzt, weil Kommentare der Probanden wichtige Informationen über deren Probleme liefern können. „Thinking Aloud“ besagt, dass die Testpersonen bei der Durchführung des Tests „laut“ mitdenken. Dadurch erfährt man sofort warum Probleme auftreten und kann sie genau lokalisieren. Ein Nachteil dieser Methode ist, dass es für die Testpersonen gewöhnungsbedürftig sein kann alle Gedanken laut auszusprechen. Außerdem wird dadurch die Arbeitsgeschwindigkeit verringert und die zeitlichen Benchmarks sind nicht mehr mit dem realen „Betrieb“ des Systems vergleichbar. [RC08, Seiten 204-206].
Im Rahmen des „Exploratory Tests“ sind diese zeitlichen Benchmarks hinsichtlich des Realbetriebs jedoch nicht repräsentativ.

Rolle des Testleiters/Moderators. Während der Durchführung des Tests wird der Moderator eine passive Beobachterrolle einnehmen. Die Situation wird also möglichst zurückgezogen beobachtet, in dem Bemühen, das Geschehen möglichst nicht zu beeinflussen. In einem „Exploratory Test“ kommt es gezwungenermaßen durch die Art des Tests zu mehr Interaktion zwischen Testperson und Testleiter. Diese Interaktion soll nicht verhindert werden - Fragen der Probanden sollen beantwortet werden.

Pilottest. Um das vorbereitete Testszenario selbst vorab zu testen, wird ein Pilottest durchgeführt. Dabei werden alle Szenarien anhand der Beschreibungen abgearbeitet und es wird so versucht einerseits ein Gefühl für die benötigte Zeit zu bekommen. Zum anderen sollen dadurch Probleme identifiziert werden, welche für Schwierigkeit in der Durchführung sorgen können.

Post-Test Fragebogen. Nach der Durchführung des Tests wird, zur Messung der subjektiven Einschätzung der Benutzbarkeit, von den Testpersonen ein Post-Test Fragebogen ausgefüllt. Als Post-Test Fragebogen wird der *IBM*

Computer Usability Satisfaction Questionnaire (CSUQ)[Lew95] ins Deutsche übersetzt verwendet. Bei diesem Fragebogen handelt es sich um einen standardisierten Katalog an Aussagen, die anhand einer Likert-Skala mit sieben Punkten bewertet werden. Die Bewertung findet dabei in einem Bereich von „stimme gar nicht zu“ bis „stimme voll und ganz zu“ statt. Von den 19 Aussagen des Computer Usability Satisfaction Questionnaire (CSUQ) wurden die Aussagen 1, 2, 4, 5, 6, 7, 17 und 19 als anwendbar für diese Studie ausgewählt. Diese Auswahl hat sich bereits etwa in der Studie von [SBG09] bewährt, in welcher ebenfalls eine vergleichende Evaluierung[46] durchgeführt wurde. Zusätzlich zu der in [SBG09, Seite 586] verwendeten Aussagen wurde Aussage 1 verwendet, da diese einen zusammenfassenden Charakter die Benutzbarkeit betreffend aufweist und daher als sinnvoll erachtet wird. Auf Aussage 9 wurde im Gegensatz zu [SBG09, Seite 586] verzichtet, da im Prototyp keine Fehlermeldungen - welche man in dieser Aussage bewerten sollte - existieren. Die erhobenen Daten werden in den beiden Versionen des Prototyps verglichen und mithilfe eines Wilcoxon Vorzeichen Rang Tests[47] analysiert.

Teilstrukturiertes, Problemzentriertes Interview. Nach dem eigentlichen Test wird ein teilstrukturiertes, problemzentriertes Interview mit jeder Person der Testgruppe einzeln durchgeführt.
Bei der Methode des teilstrukturierten Interviews wird anhand eines Gesprächsleitfadens gefragt. Der Gesprächsleitfaden dient dazu eine Struktur in den Verlauf des Gesprächs zu bringen. Des Weiteren sind keine Antwortmöglichkeiten vorgegeben; dadurch ist ein offener Ausgang des Gespräches zu erwarten. Im Rahmen des teilstrukturierten Interviews wird das „Problemzentrierte Interview“ angewendet. Diese Art des Interviews wird relativ offen durchgeführt. Dies gewährleistet nach dem Leitfaden grob vorgehen zu können und somit durch die Fragestellungen die Probanden dazu zu bewegen, über das Auftreten von Problemen bei der Multi-User-Anwendung frei zu erzählen. [BP09, Seiten 15-16].
Der Interview-Leitfaden ist im Anhang zu finden.

8.1.3 Testumgebung und Equipment

Für die Evaluierung wird der bereits in Kapitel 3.3 vorgestellte Multitouch-Tisch verwendet. Der Test findet im Labor der Technischen Universität Wien

[46] Inhalt der Studie war der Vergleich von direkte und indirekter Multitouch-Eingabe auf großen Oberflächen.

[47] Unter einem Wilcoxon Vorzeichen Rang Test versteht man einen nicht-parametrischen Test, welcher aufgrund gepaarter Stichproben (in unserem Fall der verschiedenen Versionen) die zentralen Tendenzen der zugrunde liegenden Grundgesamtheiten analysiert; siehe hierzu etwa [Zöf08, Seite 139-142].

des Forschungsbereichs DECO[48] in ruhiger Atmosphäre statt.

Während des Tests wird das folgende Equipment eingesetzt:

- **Bildschirmaufzeichnung** - QuickTime Player Version 10.0.

  Damit sämtliche Interaktion zwischen den Probanden und dem Tisch aufgezeichnet werden kann, wird der Bildschirm aufgezeichnet. Dadurch ist eine bessere Auswertung des Tests möglich. Die Bildschirmaufzeichnung mithilfe von QuickTime Player ermöglicht auch die Aufzeichnung der Zeit. Diese braucht daher nicht mehr separat aufgezeichnet werden.

- **Audioaufzeichnung** - integriertes Mikrofon (MacBook Pro, siehe 3.3) und QuickTime Player Version 10.0.

  Zur besseren Auswertung des Tests - und wegen des Einsatzes von „Thinking Aloud“, werden die Kommentare der Probanden aufgezeichnet.

- **Videoaufzeichnung** - Sony HD bloggie MHS-PM5.

  Um zusätzlich zur Interaktion mit der Applikation auch Gestik und Mimik aufzeichnen zu können, werden die Probanden während des Tests gefilmt.

- **Pre-Test Fragebogen**

  Im Pre-Test Fragebogen, welcher sich im Anhang der Arbeit befindet, werden Daten der Probanden erhoben, wie beispielsweise deren Technikaffinität oder deren Erfahrungen mit Multitouch, um bei der Auswertung etwaige Rückschlüsse ziehen zu können.

- **Einführungsblatt**

  Vor der Durchführung eines Tests wird den Testpersonen die Art der Gestenimplementierung anhand eines Einführungsblattes (siehe Anhang) vorgestellt. Zusätzlich werden die Probanden aufgefordert, mit der Applikation zu „spielen“. Es wird dabei die Applikation in einem Zustand gezeigt, in welchem bereits einige Bilder und Nachrichten vorhanden sind. Zusätzlich können auch weitere Daten geladen werden.

- **Post-Test Befragung** - IBM Computer Usability Satisfaction Questionnaire, teilstrukturiertes problemzentriertes Interview.

  Die Post-Test Befragung findet, wie bereits erwähnt, mithilfe des CSUQ und eines teilstrukturierten, problemzentrierten Interview mit jeder

[48] DECO steht für Designing Comfort und ist ein Forschungsbereich des INSO (steht für Industrial Software und ist eine Forschungsgruppe des Institus für Rechnergestützte Automation der Technischen Universität Wien) mit dem Schwerpunkt der Mensch-Computer Interaktion.

Testperson einer Test-Session einzeln statt. Da der CSUQ hauptsächlich auf die Zufriedenheit mit der Benutzbarkeit abzielt, soll dieses Interview zusätzliche Informationen über die Qualitätsmerkmale der Robustheit und Korrektheit der Eingabeverarbeitung liefern.

8.1.4 Zielgruppe und Teilnehmer

Da der Prototyp keinem fertigen Produkt mit einem klar definierten, praxisorientierten Einsatzgebiet entspricht, ist es schwierig eine adäquate Zielgruppe zu definieren. Generell kann man jedoch nach der Technikaffinität und dem Alter einteilen. So wird die Zielgruppe von Multitouch-Tischen im Allgemeinen in naher Zukunft eher technikaffinen Personen, die weitreichende Erfahrungen mit Computern haben und jüngeren Generationen zuzuordnen sind, entsprechen. Aufgrund der Natürlichkeit der Interaktion bei Multitouch, welche eine intuitive Benutzung ohne jegliche Erfahrungen mit der Technologie ermöglicht, kann jedoch auch dieser Personenkreis einen Teilbereich der Zielgruppe bilden.

Die Studie wurde mit 8 Personen (5 männliche, 3 weibliche), aufgeteilt auf 4 Zweier-Gruppen, durchgeführt. Bei der Auswahl der Probanden wurde darauf geachtet, dass diese noch keine Erfahrungen mit dem Multitouch-Tisch des INSO haben. Bei den Probanden handelt es sich größtenteils um Personen aus dem studentischen Umfeld. Sie weisen ein Durchschnittsalter von 24,38 Jahren (maximales Alter: 26 Jahre, minimales Alter: 22 Jahre) auf und sind durchschnittlich eher als technikaffine Personen (Durchschnitt von 5,88 auf einer Skala von 1 für sehr gering bis 7 für sehr hoch) zu sehen. Alle Testpersonen bis auf eine haben bereits Erfahrung mit der Touch-Technologie - zumindest durch Fahrkartenterminals oder ähnlichem. 1 von 8 Personen hat bereits einen Multitouch-Tisch (MS Surface) benutzt und 3 weitere Personen kannten Multitouch-Tische bereits zuvor. Eine detaillierte Übersicht über die ausgewählten Probanden sowie deren Charakteristika kann im Anhang gefunden werden.

Wie bereits angeführt, wird der vergleichende Test in zwei Sessions durchgeführt. In jeder Session wird die jeweils andere Version der Applikation getestet. Für Test-Sessions werden zur Simulation der Multi-User-Anwendung Gruppen von 4x2 Personen gebildet. Die Gruppenbildungen werden in den beiden Sessions verändert. Dies soll möglichen gruppendynamischen Effekten in der zweiten Test-Session vorbeugen. Zweier-Gruppen bilden die Grundlage beziehungsweise den Basisfall der Mehrbenutzer-Anwendungen an einem Multitouch-Tisch, wie ihn der des INSO darstellt. Auf diese Gruppenbildung soll daher der Fokus der Untersuchung gerichtet werden.

8.1.5 Szenarien

Ingesamt werden fünf Szenarien - mit dem Ziel das Auftreten der bereits identifizierten Probleme hervorzurufen - definiert. Um dies zu erreichen, werden einerseits kollaborative Szenarien, bei welchen die Probanden eng zusammen arbeiten, festgelegt. Andererseits werden konkurrierende Aufgaben bestimmt. In diesen sollen zeitkritische Situationen hervorgerufen werden, die in Folge einer Stresssituation zu den identifizierten Problemen führen sollen. Die enge Zusammenarbeit in den kollaborativen Szenarien und das schnelle „gegeneinander“ arbeiten in den konkurrierenden Szenarien, soll möglichst intensive, gleichzeitige Interaktion aller Testpersonen forcieren, um somit möglichst viele Probleme der Mehrbenutzer-Anwendung zu erhalten. Dadurch soll das Auftreten der Probleme in den beiden Versionen des Prototyps vergleichbar werden. Einen Überblick über die Szenarien gibt Tabelle 8.1.

Szenario Nummer	**Aufgabe**
Szenario 1	Sortierung und Löschen von Fotos: Fotos mit Blumen sollen auf die linke Seite des Tisches, Fotos mit Tieren auf die rechte Seite des Tisches bewegt werden. Unpassende Fotos sollen entfernt werden.
Szenario 2	Konkurrierendes Suchen und Finden eines Fotos.
Szenario 3	Konkurrierendes Sortieren und Skalieren mehrerer Fotos: Jede Testperson bekommt ein Fotomotiv zugeteilt. Vier dieser Fotos müssen zu sich bewegt und entsprechend den Markierungen skaliert und sortiert werden.
Szenario 4	Gemeinsames Sortieren, Anordnen und Entfernen nach selbstständig entwickelten Maßstäben von Fotos.
Szenario 5	Laden von Fotos via Tastatur. Suchen der zum Suchbegriff gehörenden Fotos.

Tabelle 8.1 – Übersicht der Szenarien der Evaluierung

So soll beispielsweise Szenario 1 - bei dem die Benutzer gemeinsam Fotos von der Mitte des Tisches sortieren - zu einer engen Zusammenarbeit anregen. Die Fotos befinden sich alle in der Mitte des Tisches, so wird gewährleistet, dass Berührungspunkte von verschiedenen Benutzern in knappen räumlichen Abständen zueinander auftreten. In einer konkurrierenden Situation befinden sich die Probanden beispielsweise in Szenario 2 und Szenario 3, bei welchen es gilt Fotos zu finden. Dadurch soll eine intensive und auch räumlich enge Interaktion der Testpersonen statt finden. Bei Szenario 3 kommt hinzu, dass die Fotos zusätzlich in eine vorgegebene Markierung abgelegt werden.

Dazu müssen die Fotos sowohl skaliert, als auch gedreht werden. Dies soll jene Probleme forcieren, die bei diesen beiden Gesten aufgetreten sind.

Damit zusätzlich die Zeiten der Abarbeitung der Szenarien zwischen den Versionen verglichen werden können, werden konkrete Endzustände der Szenarien vorgegeben. Eine detaillierte Liste an Szenarien - mit etwa deren Endzuständen und Formulierungen für die Testpersonen - ist ebenso wie die verwendete Reihenfolge (Latin Square) im Anhang zu finden.

Im Prototyp werden, je nachdem welches Szenario gestartet wird, programmatisch die entsprechenden Vorbedingungen hergestellt. So werden etwa in der ersten Aufgabe 25 zum Szenario passende Fotos eingelesen und in der Mitte des Screens zufällig angeordnet. Das Menü wird in dieser Aufgabe nicht benötigt und wird daher ausgeblendet.

8.1.6 Metriken und Messungen

Wie bereits bei der Zielformulierung der Evaluierung angeführt, wird primär eine qualitative Beobachtung durchgeführt. Bei dieser gilt es, das Auftreten von Problemen zu analysieren. Zusätzlich dient die Zeit als Effizienzfaktor: Wird ein Szenario schneller erledigt, kann dies Auskunft über ein verringertes Auftreten von Problemen geben.

Weiters finden subjektive Messungen durch die Post-Test Befragung statt. In diesem werden beispielsweise Informationen über die subjektive Zufriedenheit - etwa des Antwortverhaltens der Applikation - mithilfe von Likert-Skalen erfragt und durch Bildung des Mittelwertes und der Standardabweichung Analysen vorgenommen.

Hinsichtlich der in der Arbeit definierten Kriterien der Korrektheit, Performance, Stabilität und Robustheit beziehungsweise der durch die Heuristiken möglichen Verbesserung auf diesen Ebenen, gilt es daher diese Werkzeuge einzusetzen: Durch qualitative Beobachtung soll das Auftreten von Problemen erkannt werden, der Faktor Zeit soll für Vergleiche und Analysen eingesetzt werden und subjektive Messungen der Post-Test Befragung sollen ausgewertet werden.

8.1.7 Evaluierung mit vier Testpersonen

Zusätzlich wird zur Evaluierung eines Grenzfalles eine Vierer-Gruppen observiert. Eine Vierer-Gruppe bildet am betrachteten Multitouch-Tisch deshalb einen Grenzfall, weil eine Anwendung mit mehr als vier Personen aufgrund des beschränkten Platzes am Tisch nicht möglich ist. Diese Evaluierung findet nach dem eigentlichen, formellen Test statt und soll in getrennter Auswertung Auskunft darüber geben, wie die Applikation beziehungsweise die Heuristiken bei der Interaktion von mehr als zwei Benutzern skalieren.

Dies geschieht in einem informellen Setting: Die Tests werden von Probanden absolviert, welche bereits in einer Zweier-Gruppe beobachtet wurden. Lerneffekte werden in diesem Grenzfall akzeptiert. Die Metriken, welche durch diese Lerneffekte nicht mehr aussagekräftig wären, werden nicht aufgezeichnet. Ebenso verhält es sich mit den Post-Test Fragebögen: Diese wären aufgrund der Lerneffekte zwischen den beiden Versionen und aufgrund der Durchführung von nur einem einzelnen Test -bedeutungslos. Der Test wird demnach auf eine rein qualitative Beobachtung reduziert.

8.1.8 Zusammenfassung und Ablauf des Tests

Zu Zwecken der Nachvollziehbarkeit wird hier der Ablauf der Evaluierung zusammengefasst:

(1) Ankunft im Labor und Begrüßung der Probanden.

(2) Pre-Test Fragebogen wird von den Testpersonen der Test-Session einzeln ausgefüllt.

(3) Testpersonen einer Test-Session werden mittels Einführungsblatt in die Materie eingeführt und auf den Ablauf der Evaluierung hingewiesen.

(4) Multitouch-Tisch wird den Testpersonen einzeln vorgestellt und diese werden aufgefordert mit der Applikation zu interagieren.

(5) Szenarien werden einzeln ausgegeben, wobei die Reihenfolge durch die Latin-Square-Methode bestimmt wird. Erst wenn ein Szenario abgeschlossen ist, wird die nächste Angabe eines Szenarios ausgegeben.

(6) Der Test wird durchgeführt und aufgezeichnet; beobachtete Probleme werden dokumentiert.

(7) Post-Test Fragebogen wird von den Testpersonen der Test-Session einzeln ausgefüllt.

(8) Testpersonen der Test-Session werden einzeln befragt.

(9) Debriefing und Verabschiedung der Testpersonen.

8.2 Ergebnisse der Evaluierung

8.2.1 Ergebnisse des Pilottests

Durch die Durchführung eines Pilottests wurde das Testsetting selbst vorab einem Test unterzogen. Es zeigte sich dabei, dass einige Dinge noch adaptiert werden mussten:

Im Pilottest wurde als Post-Test Fragebogen der *System Usability Scale*[49] eingesetzt - doch waren einige der anhand einer Likert-Skala zu bewertenden Aussagen nicht für den Einsatz passend. So waren Aussagen wie beispielsweise „Ich denke, ich würde die Applikation regelmäßig verwenden“ aufgrund des Prototypstatus der Applikation schlicht ungeeignet. Die Probanden des Pilottest waren daher bei diesen Aussagen unschlüssig und wussten nicht recht wie sie diese bewerten sollten. Deshalb wurde der System Usability Scale durch eine auf passende Fragen reduzierte Version des CSUQ ersetzt. Des Weiteren wurde offensichtlich, dass für die zwei Test-Sessions das Einführungsblatt angepasst werden musste: Die Pan-Geste wurde in den dort vorgestellten Heuristiken nur mit einem Finger erläutert, was zu Verwirrungen in der zweiten Test-Session (bei welcher die Pan-Geste mit zwei Fingern implementiert ist) führte.
Nach dem Pilottest wurde außerdem Szenario 5 der Szenarien überarbeitet: Die Fotos und Textnachrichten wurden zuvor über das Internet von den Plattformen flickr und twitter geladen. Dadurch ist man einerseits von einer Internetverbindung sowie von den jeweiligen Plattformen abhängig und andererseits sind die Ergebnisse der geladenen Daten nicht oder nur schwer kontrollierbar. Daher wurde die Implementierung dieses Szenarios so abgeändert, dass die Daten lokal von der Festplatte gelesen werden. Beobachtbar war weiters, dass im Pilottest Zoom-Gesten nur selten eingesetzt wurden. Daher wurden in diesem Szenario die Schriftgrößen der geladenen Nachrichten so angepasst, dass das Ausführunen von Zoom- und Skalier-Geste erforderlich ist.

Zusätzlich zu den erwähnten Anpassungen, wurden zeitliche Benchmarks für die einzelnen Szenarien erhoben. Diese sind bei der Beschreibung der Szenarien im Anhang zu finden. Sie sollen einen groben Richtwert bieten, um einerseits den Test zeitlich besser planbar zu machen und andererseits einen Vergleichswert für mögliche Testauswertungen zu erhalten.

8.2.2 Beobachtungen und Interaktionsverhalten

Bei der Beobachtung der Probanden während des Tests wurde darauf geachtet, welche Probleme auftraten beziehungsweise wie die Testteilnehmer gemeinsam am Multitouch-Tisch mit der Applikation interagierten.

Version ohne Heuristiken. In jener Version, in welcher Heuristiken nicht implementiert wurden[50], fiel auf, dass sämtliche erwarteten und bereits in der Pilotstudie identifizierten Probleme wieder auftraten:

[49] System Usability Scale (SUS) - vorgestellt in [Bro96].
[50] In der weiteren Analyse als Version 1 bezeichnet.

- *Unbeabsichtigt, gemeinsam getriggerte Gesten: Durch unbeabsichtigte Berührungspunkte mehrerer Benutzer verändert sich der Zoom-Faktor der gesamten Applikation. (1a).*

  Dieses Problem konnte in 3 von 4 Sessions beobachtet werden.

  Durch die Anwendung von Thinking Aloud wurden exemplarisch folgende Zitate beim Auftreten dieses Problems geäußert: *„Huch, achso wir dürfen nicht zu zweit oder wie?!“*; *„Achso, wenn wir beide Ziehen wird alles größer.“*; *„Hoppala, was ist jetzt?!“*.

  Aus den ersten beiden Zitaten kann man folgern, dass es den Testteilnehmern teilweise bewusst war, warum ein Problem auftrat, sie zum Teil jedoch auch völlig überrascht waren und mit einer solchen Reaktion der Applikation nicht rechneten. Das letzte Zitat zeigt, dass die Testpersonen durch die Ausführung der Zoom-Geste gestört wurden.

- *Unbeabsichtigt, gemeinsam getriggerte Gesten: Gleichzeitiges Ziehen eines Objekts führt zu einer Skalier-Geste (1b)* sowie *Störung der Gesten durch andere Benutzer - Räumlich enges Arbeiten der Benutzer - Fehlen von Grenzen der Arbeitsbereiche: Zusätzlicher Berührungspunkt beim Ziehen eines Objekts führt zu einer Skalier-Geste. (3c).*

  Dieses Problem konnte in 4 von 4 Sessions beobachtet werden.

  Folgendes Zitat wurde während des Tests geäußert: *„Wenn wir beide mit einem Finger klicken, dann sind es wieder zwei.“*. Zusätzlich führte ein Proband bereits während der Test-Session an, dass er sich durch das Problem gestört fühlt und es beim nächsten Test verhindert werden sollte: *„Man könnte vielleicht für den nächsten Test das Zoomen (Anmerkung: das Zoomen von Bildern - das Skalieren) deaktivieren, damit die Bilder nicht vergrößerbar sind - dass sie auf der gleichen Größe bleiben, wenn wir beide darauf greifen.“*

- *Unbeabsichtigt, alleine getriggerte Gesten: Zu stark vergrößerte Objekte stören die anderen Benutzer (2a)* sowie *Störung der Gesten durch andere Benutzer - Räumlich enges Arbeiten der Benutzer - Fehlen von Grenzen der Arbeitsbereiche: Zu stark vergrößerte Objekte stören die anderen Benutzer. (3d).*

  Dieses Problem konnte in 2 von 4 Sessions beobachtet werden.

  Das Problem wurde vor allem in konkurrierenden Szenarien identifiziert und wurde dort als störend empfunden, da für die zweite Testperson die Möglichkeit der Erfüllung der Aufgabe verzögert wurde. Typische Zitate waren daher ziemlich fordernd und ungeduldig: *„Mach' das Foto weg da!“*; *„Hey, mach' das nicht so groß!“*.

- *Unbeabsichtigt, alleine getriggerte Gesten: Unbeabsichtigt ausgeführtes Schwenken der Kamera stört die anderen Benutzer. (2b).*

Dieses Problem trat in mehreren Ausprägungen auf:

  - Während des User-Tests zeigte sich, dass Pan-Gesten sehr oft etwa durch unpräzise Selektion unbeabsichtigt ausgeführt wurden. Dabei wurde die Ausgangssituation vor Auftreten des Problems jeweils von der anderen Person wieder hergestellt.

    Das Problem offenbart sich in den folgenden Zitaten: *„Ich ziehe immer alles statt einem.“*; *„Jetzt habe ich schon wieder alles weg gezogen.“*; *„Das ruckelt irgendwie so heute“* (Zweite Session); *„Das verzeiht jetzt nicht mehr sehr viele Fehler.“* (Zweite Session). Auch in folgender Konversation zwischen zwei Teilnehmern zeigt sich das Problem: *„Wieso ruckelt das manchmal so?“* (Proband A) - *Ich glaube da kommt jemand am Hintergrund an.“* (Proband B).

    Es lässt sich folgern, dass die Ursache des Problems einigen Testpersonen bewusst ist. Die Wortwahl in den Zitaten indiziert weiters, dass die Testteilnehmer sichtlich gestört und genervt durch das wiederholte Auftreten des Problems sind.

    Diese Ausprägung des Problems konnte in 4 von 4 Sessions beobachtet werden.

  - Eine weitere Ausprägung dieses Problem zeigte sich darin, dass Fotos so stark verkleinert wurden, dass sich diese nicht mehr vergrößern ließen. Beim Versuch die Fotos wieder auf eine ansprechende Größe zu skalieren, wurde wiederum unbeabsichtigt eine Pan-Geste ausgeführt.

    Diese Ausprägung des Problems konnte in 2 von 4 Sessions beobachtet werden.

- *Störung der Gesten durch andere Benutzer - Räumlich enges Arbeiten der Benutzer - Fehlen von Grenzen der Arbeitsbereiche: Zusätzlicher Berührungspunkt bei einer Rotier-Geste führt zu unkontrolliertem Verhalten (3a)* sowie *zusätzlicher Berührungspunkt bei einer Skalier-Geste führt zu unkontrolliertem Verhalten (3b).*

  Dieses Problem konnte in 1 von 4 Sessions beobachtet werden.

  Folgendes Zitat wurde beim Auftreten des Problems geäußert: *„Irgendwie passieren da jedes Mal, wenn wir gemeinsam mit dem Bild etwas machen, irgendwelche 'Random'-Sachen.“*

Zusätzlich zum Auftreten der bereits identifizierten Probleme, ließ sich während 2 von 4 Test-Sessions beobachten, dass sich die Art der Interaktion veränderte: Die Benutzer interagierten, um keine weiteren Probleme mehr hervorzurufen, nur mehr abwechselnd: Sie arbeiteten nur mehr nacheinander und nicht mehr miteinander. Dies war in der Version mit Heuristiken nicht

der Fall. Dies könnte mitunter zu einer Verminderung der Geschwindigkeit in der Abarbeitung der Szenarien führen.

Die restlichen Probleme (4a und 5a) konnten, wie zu erwarten war, in dieser Evaluierung nicht beobachtet werden: Problem 4a wurde im Rahmen der Pilotstudie zur Evaluierung der Probleme nur im Zusammenhang mit Casual Games beobachtet. Hinsichtlich Problem 5a wurden in beiden Versionen des Prototyps keine Gesten definiert, welche über das Standard-Repertoire hinausgehen.

Version mit Heuristiken und Vergleich der Versionen. In der Version mit implementierten Heuristiken[51] konnte die Tendenz beobachtet werden, dass Probleme weniger häufig oder gar nicht mehr auftraten:

- Ein Auftreten der Probleme 2a, 2b, 3a, 3b sowie 3d konnte in Version 2 nicht mehr beobachtet werden. Die Heuristiken, welche diese Probleme adressieren, können somit zur Verbesserung dieser Probleme als bestätigt angesehen werden (siehe hierfür die zusammenfassende Tabelle 6.1 der Heuristiken und deren adressierte Probleme in Kapitel 6.3).

- Problem 1a trat in Version 2 weniger häufig auf. Ein Auftreten konnte nur mehr in 1 von 4 Sessions und damit in zwei Sessions weniger als in Version 1 beobachtet werden. Wie bereits in der Beschreibung der Heuristik angeführt, ist es durch die Definition eines `ZoomDetectRadius`, wie dies in Heuristik 3 geschieht, nicht erwünscht sämtliche Probleme zu verhindern und diesen Erkennungs-Radius entsprechend klein zu wählen, da es gilt mögliche Nebeneffekte zu beachten: Eine fälschlicherweise verhinderte Interpretation kann zu einem Bruch der Intuition der Ausführung der Geste führen. Diese Heuristik verbessert somit das Problem, löst es jedoch nicht gänzlich. Eine Lösung kann hierbei nur die Identifikation der Benutzer etwa durch zusätzliche Hardware bringen.

- Ebenso traten die Probleme 1b, 3c in zwei Sessions weniger als in Version 1 auf; damit nur mehr in 2 von 4 Sessions. Es kann somit gefolgert werden, dass für die Wahl des Parameters für Heuristik 11, welcher den Zeitpunkt der Sperre der Skalier-Geste nach Identifikation einer Zieh-Geste regelt, Optimierungsbedarf besteht. Diese Zeitverzögerung in der Sperre der Skalier-Geste wurde eingeführt, da aufgrund der Abhängigkeiten in der Erkennung einer Zieh-Geste, diese nicht mehr ausgeführt werden könnten. Ein gesonderter User Test kann Auskunft über einen optimalen Wert für diesen Parameter geben.

[51] In der weiteren Analyse als Version 2 bezeichnet.

8.2.3 Qualitative Interviews

Im Rahmen des User Tests wurden teilstrukturierte, problemzentrierte Interviews durchgeführt, welche auf negative und positive Erfahrungen hinsichtlich der Interaktion mit dem Multitouch-Tisch in einer Gruppe fokussierten.

Version ohne Heuristiken. Die Interviews zu Version 1 zeigten, dass die in der Beobachtung identifizierten Probleme auch von den Testpersonen beschrieben wurden. So schildert beispielsweise eine Testperson: *„Mir ist negativ aufgefallen, dass wir uns gegenseitig behindert haben, wenn wir zum Beispiel an etwas gemeinsam gearbeitet haben, aber jeder ein anderes Ziel hatte.“* Hinsichtlich dieses Problems wurden in den Interviews unterschiedliche Ausprägungen, etwa die Zoom-Geste und die Skalier-Geste betreffend, beschrieben: *„Bei der ersten Aufgabe habe ich bemerkt, dass das Zoomen des Bildschirms unbeabsichtigt ausgeführt wurde und andere Sachen auch noch.“* beziehungsweise *„Es war [...] oft so, dass das System angenommen hat, wir wären nur eine Person, die zum Beispiel etwas auseinander zieht.“*

Zusätzlich wurden in den Interviews nicht nur Probleme unbeabsichtigt, gemeinsam getriggerter Gesten (Probleme der Gruppe 1), von welchen in den angeführten Zitaten berichtet wird, erläutert, sondern auch unbeabsichtigt, alleine getriggerte Gesten (Probleme der Gruppe 2): *„Beim Desktop verschieben (Anmerkung: Bei Pan-Gesten) ist mir aufgefallen, wenn man nicht genau am Bild landet, dann verschiebt sich der Desktop und der andere wird dadurch gestört.“*

Von Problemen der Gruppe 3 wurde ebenfalls in den Interviews zu Version 1 berichtet: *„Blöd war, dass ein paar Mal Fotos so groß wurden, dass man gar nichts mehr anderes gesehen hat und nicht mehr weiterarbeiten konnte.“*; *„Wenn man gemeinsam nicht genau aufpasst, sind irgendwelche 'Random'-Sachen mit dem Bild passiert - gedreht, verzerrt je nachdem wie man es gerade zufällig erwischt.“*; *„Es war problematisch, dass es keine Unterscheidungsmöglichkeit gab, welcher Finger jetzt zu wem gehört - wer welche Geste ausführt. Das ist meiner Meinung nach ein Bug kein Feature. Aber das ist halt so, wenn zwei Personen am Tisch arbeiten.“* Hinsichtlich räumlich engem Arbeiten der Benutzer, einem Hauptaugenmerk von Problemgruppe 3 zeigen die Interviews, dass die Applikation nur dann adäquat auf die Eingaben von mehreren Benutzern reagierte, wenn jeder auf einem eigenen Teil des Tisches arbeitete: *„In der Mitte kommt es schon zu Konflikten, zum Beispiel, dass beide auf das selbe Bild drücken und dann passiert halt irgendetwas. Da muss man sich dann mehr absprechen. Ich will nicht sagen, dass es deshalb schlecht funktioniert, aber man muss halt auf den anderen aufpassen.“*; *„Wenn jeder in seiner Hälfte arbeitet, dann funktioniert das ganz gut - dann drückt der eine dem anderen nicht drein.“*; *„Gut funktioniert hat, wenn man lokal getrennt gearbeitet hat - das war sicher besser. Ansonsten*

ist manchmal etwas Komisches passiert. Das Blöde ist wirklich, wenn einer etwas macht und der zweite pfuscht dann auch mit.“

Im Interview wurde ebenfalls erfragt, in wie weit sich die Testteilnehmer von Problemen gestört fühlten. Dabei fiel auf, dass unbeabsichtigt, alleine getriggerte Gesten - wie etwa das Ausführen einer Pan-Geste - die Probanden irritierte. Dieses Problem wird als „Ruckeln“ der Applikation wahrgenommen und als fehlende Fehlertoleranz: *„Die Fehlertoleranz war dadurch (Anmerkung: durch unbeabsichtigt getriggerte Pan-Geste) gleich Null, wenn man zum Beispiel daneben greift, dann zieht man nicht das Foto, sondern alles. Und dann waren gleich einmal alle Fotos weg oder 'verzoomt' (Anmerkung: viel zu groß/klein).“* beziehungsweise *„Das Programm hat manchmal ein bisschen geruckelt.“* Weiters führten mehrere Testpersonen an, dass die Arbeit dadurch erschwert wurde und dies in der Ausführung der Aufgaben störte: *„Das (Anmerkung: unbeabsichtigt, alleine getriggerte Gesten) hat mich bei der Benutzung in meiner Tätigkeit gestört - und ich denke auch meinen Partner -, weil oft eine andere Geste ausgeführt wurde als die, die ich machen wollte.“*

Bezüglich der Störung der Probanden durch unbeabsichtigt, gemeinsam getriggerte Gesten, zeigt die Analyse der Interviews, dass dieses Problem ebenso größtenteils als negativ empfunden wird. Mehrere Testteilnehmer führten an, dass dies bei der Arbeit irritierte: *„Wir haben uns gegenseitig behindert. Heute (Anmerkung: zweite Session) war es dadurch schwerer zu arbeiten auf jeden Fall.“* Eine Testperson führte jedoch zusätzlich an, dass sie dieses Problem nicht als störend empfand, da diese Schwachstelle auch für die gemeinsame Interaktion genutzt werden und diese gezielt eingesetzt werden kann: *„Vielleicht ist ja das (Anmerkung: gemeinsam getriggerte Zoom-Geste) gezielt, da man es zu zweit kleiner machen kann - das ist ja ganz normal.“*

Generell lässt sich jedoch der Trend erkennen, dass die Probleme der Mehrbenutzer-Anwendung die Probanden der Evaluierung irritierten. Die Probleme stellten jedoch kein Hindernis dar, die gestellten Aufgaben zu erfüllen. Nach einiger Zeit konnte die Situation vor Auftreten der Probleme wieder hergestellt und weiter gearbeitet werden. Dies zeigen auch die Interviews: *„Wir haben die Konflikte dann schon lösen können. Man merkt, dass etwas nicht stimmt. Bei der Erfüllung der Aufgaben habe ich deshalb keine Probleme gehabt“* Es wird daher der Arbeitsfluss gebrochen, jedoch nicht die Möglichkeit der Erfüllung der Aufgabe eingeschränkt.

Wie auch schon bei der Beobachtung der Testpersonen, zeigt die Analyse der Interviews zu Version 1 eine Veränderung in der Art der Interaktion: Die Probanden berichteten, während des Tests die Art der Interaktion abgewandelt zu haben. Nach dem Auftreten der Probleme wurde nicht mehr gleichzeitig, miteinander gearbeitet, sondern man wechselte sich ab beziehungsweise sprach sich ab: *„Das (Anmerkung: Probleme der Mehrbenutzer-Anwendung) hat ein Problem bei der Benutzung dargestellt, weil wir [...]*

dadurch viel mehr aufeinander Rücksicht nehmen mussten. Wir haben dann eher nacheinander getippt und nicht mehr gleichzeitig.“ Im Zuge dessen war ebenfalls eine Testperson der Meinung, dass die Arbeit dadurch verlangsamt wird und sie sogar alleine schneller wäre: *„Wenn man alleine daran arbeitet, dann braucht man sich nichts vereinbaren - dann geht es in diesem Sinne schneller. Wir haben das nach den Problemen so gemacht, dass wir einzeln den Tisch berührt haben.“*

Warum Konflikte auftreten war den Testpersonen zum Teil klar: *„Heute war es auf jeden Fall so, dass das Grid (Anmerkung : der Hintergrund) sehr nervös war, dadurch dass man es mit einem Finger bewegen konnte.“*; *„Es hat nicht gut funktioniert, dass wir teilweise den Bildschirm verschoben haben (Anmerkung: eine Pan-Geste ausgeführt haben), dadurch dass wir gleichzeitig gearbeitet haben [...]. Man muss halt ganz genau auf das Bild klicken.“* Teilweise waren die Probleme jedoch auch unklar und schienen für die Testteilnehmer zufällig aufzutreten: *„Das Bild hat manchmal so geruckelt, aber ich weiß nicht warum.“*

Version mit Heuristiken und Vergleich der Versionen. Die Auswertung der Interviews zu Version 2 zeigt ähnliche Tendenzen wie die Beobachtungen des vorigen Paragraphen. Einige der Probleme treten weniger oft auf - einige gar nicht mehr. Die Probanden erwähnten lediglich das Auftreten der folgenden Probleme in den Interviews:

- Von unbeabsichtigt hervorgerufenen Zoom-Gesten wird in den Interviews berichtet (Problem 1a): *„Das Zoomen vom Bildschirm hat nicht gut funktioniert: Es hat einmal gezoomt, obwohl wir ziemlich weit auseinander etwas gemacht haben. Das hat den Arbeitsprozess unterbrochen.“*; *„Wenn ich in der Nähe von ihr (Anmerkung: der anderen Testperson) etwas gemacht habe, hat es gezoomt, aber nicht immer. Das verzögert das Ganze.“* An der Wortwahl der Probanden erkennt man jedoch einen deutlichen Unterschied zu Version 1: Das Problem ist nicht so oft vorgekommen (*„Es hat einmal gezoomt [...]“* und *„[...] aber nicht immer“*). Ebenso erwähnen Testpersonen bei der zweiten Session auch, dass Zoom-Gesten nicht mehr unbeabsichtigt aufgetreten sind: *„Das wir den Hintergrund unabsichtlich vergrößert haben ist dieses Mal gar nicht passiert.“*

- Von unbeabsichtigt hervorgerufenen Skalier-Gesten wird in den Interviews berichtet (Probleme 1b, 3c): *„Manchmal kommt es auch vor, wenn man den selben Gedankengang hat, dass beide ein Foto bewegen wollen und es sich dadurch vergrößert.“*; *„Negativ aufgefallen ist mir, dass wir uns manchmal gegenseitig behindert haben. Wenn zwei Leute in das gleiche Bild greifen, ist es fehleranfällig.“*; *„ Es hat Momente gegeben in denen wir Fotos skaliert haben, obwohl wir nicht wollten.“*;

„Es war wieder (Anmerkung: zweite Session) so, dass es in der Mitte des Tisches Konflikte gegeben hat, wenn beide das gleiche Bild zu sich ziehen wollten.“

Bei der Analyse der Interviews fiel auf, dass Benutzer den Eindruck hatten, dass diese beiden noch immer existierenden Probleme nur bei konkurrierenden Aufgaben ins Gewicht fielen: *„Es hat Momente gegeben in denen wir Fotos skaliert haben, obwohl wir nicht wollten. Man kann sich damit arrangieren, aber bei konkurrierenden Aufgaben gerät man da aneinander.“* Ein anderer Testteilnehmer beschreibt sogar, dass diese Probleme nur in konkurrierenden Situationen aufgetreten sind: *„[...] bei Aufgaben, die man miteinander macht, haben wir uns nicht gestört; die haben problemlos funktioniert.“*

Betrachtet man die Interviews zur verbesserten Version noch einmal genauer, sticht heraus, dass unbeabsichtigt, alleine getriggerte Gesten (Problemgruppe 2) kein Problem mehr darstellten: Auf der einen Seite wurden Pan-Gesten nicht mehr unbeabsichtigt ausgeführt: *„Mit dem Hintergrund (Anmerkung: hinsichtlich Pan-Gesten) hatten wir keine Probleme, weil wir das mit zwei Fingern machen mussten - dadurch ist es wahrscheinlich stabiler.“* Andererseits stellten nun auch unbeabsichtigt zu groß skalierte Objekte kein Problem mehr dar: *„Das (Anmerkung: zu groß skalierte Bilder) war besser als beim letzten Mal (Anmerkung: Zweite Session), wo das Foto so groß geworden ist, dass man nicht mehr arbeiten konnte. Das war dieses Mal nicht so, da ist das Foto gleich groß geblieben.“*

Betrachtet man diese Analyse der qualitativen Interviews, so lässt sich eine generelle Tendenz feststellen, dass die Version mit Heuristiken besser für die Mehrbenutzer-Anwendung geeignet ist: Probleme werden von den Probanden in Version 2 weniger häufig geschildert oder finden gar keine Erwähnung mehr. Dies zeigen auch die Antworten auf Fragen nach positiven Aspekten bei der gemeinsamen Arbeit: *„Gut funktioniert hat das, was einzeln auch gut funktioniert hätte.“* Einige Testpersonen finden zusätzlich dann positive Worte, wenn sie die Applikation in der zweiten Session getestet haben: *„Das einzige was ich nicht gut finde, ist wenn man sich in die Quere kommt, aber das war heute nicht der Fall. Prinzipiell ist es für mich heute viel besser gegangen.“; „Es hat besser funktioniert als das letzte Mal, aber ich weiß nicht wieso.“*

8.2.4 Zufriedenheit der Testpersonen

Zusätzlich zur Erhebung der zeitlichen Daten, wurden die Probanden gebeten einen Post-Test Fragebogen zur Erhebung der Zufriedenheit auszufüllen. Es wurde dabei in Anlehnung an [Lew95] eine gekürzte Version des CSUQ verwendet, bei welchem verschiedene Aussagen die Zufriedenheit betreffend

anhand einer 7-stufigen Likert-Skala bewertet werden. Ausgewählt wurden die Aussagen 1, 2, 4, 5, 6, 7, 17 und 19 (siehe hierzu auch Kapitel 8.1.2).

Tabelle 8.2 und Abbildung 8.1 geben einen Überblick über die erhobenen Daten im Post-Test Fragebogen und zeigen berechnete Mittelwerte sowie Standardabweichungen im Vergleich zwischen den beiden Versionen. Tabelle 8.2 zeigt zusätzlich die Mediane der einzelnen Aussagen und Versionen.

	Mittelwert V1 (ohne)	**Mittelwert V2 (mit)**	**Median V1 (ohne)**	**Median V2 (mit)**
Aussage 1	4,50 (1,60)	6,00 (0,76)	4,00	6,00
Aussage 2	5,63 (1,51)	6,25 (0,71)	6,00	6,00
Aussage 4	4,13 (1,81)	5,63 (0,74)	4,00	6,00
Aussage 5	4,25 (1,39)	4,75 (0,71)	4,00	5,00
Aussage 6	4,38 (1,60)	4,50 (1,60)	4,50	5,00
Aussage 7	6,75 (0,46)	6,88 (0,35)	7,00	7,00
Aussage 17	5,13 (1,46)	5,50 (1,60)	5,00	6,00
Aussage 19	5,38 (0,92)	5,88 (0,64)	5,00	6,00

Tabelle 8.2 – Vergleich der Auswertung der Post-Test Fragebögen. Standardabweichungen sind in Klammern angeführt.

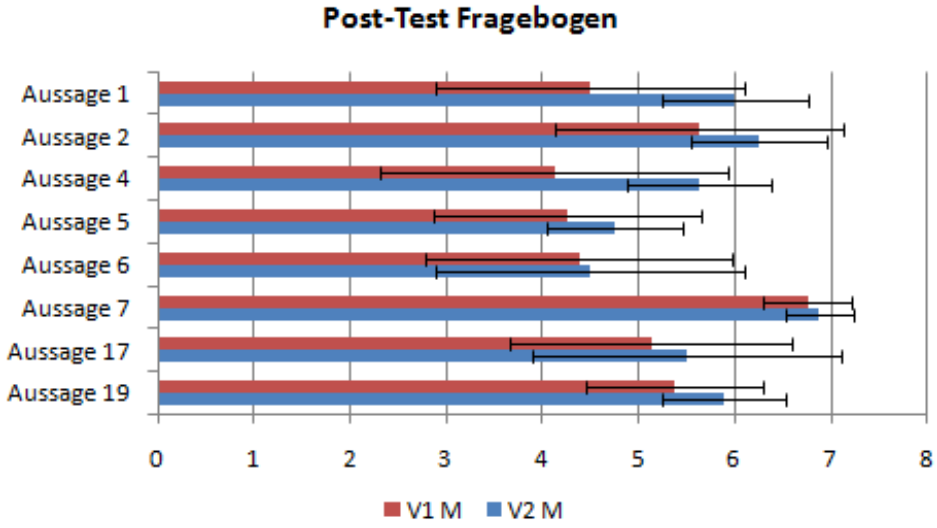


Abbildung 8.1 – Vergleich der Auswertung der Post-Test Fragebögen

Sämtliche von den Testteilnehmern zu bewertenden Aussagen wurden in der Prototyp-Version mit Heuristiken durchschnittlich besser bewertet. Es kann somit die generelle Tendenz festgestellt werden, dass die Testteilnehmer zufriedener mit Version 2 des Prototpys waren und keinerlei Abweichungen

der Zufriedenheit der Testteilnehmer in den einzelnen Aussagen bestanden.

Die größten Bewertungs-Unterschiede in den Bewertungen liegen bei den Aussagen 1 und 4:

- Aussage 1: *Insgesamt bin ich zufrieden wie leicht es ist das System zu benutzen.*
- Aussage 4: *Ich kann meine Aufgaben schnell erledigen, wenn ich das System benutze.*

Aus den Bewertungen der Testpersonen erschließt sich der Gesamteindruck, dass die Version mit Heuristiken deutlich leichter zu benutzen ist als die Version ohne Heuristiken. Zusätzlich hatten die Testpersonen das Gefühl mit der verbesserten Version ihre Aufgaben schneller erledigen zu können.

Der geringste Bewertungsunterschied liegt bei den Aussagen 6 und 7:

- Aussage 6: *Es war leicht zu erlernen wie man das System benutzt.*
- Aussage 7: *Ich mag die Benutzung des Interfaces des Systems.*

Bezüglich Aussage 6 wurde mit Ausnahme der Pan-Geste, welche in Version 2 mit zwei Fingern statt mit einem ausgeführt wird, nichts Offensichtliches - hinsichtlich der Eingabe und der Möglichkeit diese zu erlernen - verändert. So wurde jedoch sogar in Aussage 7 die Version mit Heuristiken - also die Version mit einer Pan-Geste mit zwei Fingern - als leichter zu erlernen bewertet. Die Testteilnehmer hatten insgesamt keinerlei Probleme die Benutzung der Applikation zu erlernen - auch bei Ausführung der Pan-Geste in einer abgewandelten Form.

Hinsichtlich Aussage 7 lässt sich ein Trend erkennen, dass die Testteilnehmer in beiden Versionen die Benutzung des Interfaces mögen. Dies kann etwa auf die Natürlichkeit, Popularität der Technologie Multitouch, sowie auf die Neuheit der Interaktion auf einem Interface in Tischgröße zurückgeführt werden.

Generell am besten bewertet in beiden Versionen wurden die Aussagen 2 und 7:

- Aussage 2: *Die Benutzung des Systems war einfach.*
- Aussage 7: *Es war leicht zu erlernen wie man das System benutzt.*

Die einfache Erlernbarkeit und Benutzung der Applikation lässt sich zurückführen auf den wesentlichen Vorteil bei der Interaktion mithilfe von Multitouch: Die Eingabe ist natürlich, direkt und intuitiv. Dies bestätigt die Studie.

Überprüfung auf Signifikanz. Für die Überprüfung auf Signifikanz wurde, wie bereits im Test-Design erwähnt, auf einen Wilcoxon Vorzeichen Rang Test zurückgegriffen. Basierend auf der Nullhypothese, die Grundgesamtheiten der beiden Stichproben besäßen die gleichen zentralen Tendenzen (Mediane) - also es bestünde kein signifikanter Unterschied zwischen den paarweise verbundenen Stichproben der beiden Versionen, ergab die Durchführung des Tests und dessen Analyse ergab, dass sich die Bewertungen von Aussage 4 (*„Ich kann meine Aufgaben schnell erledigen, wenn ich das System benutze.“*) in den Versionen signifikant unterscheidet. Die Bewertung fällt in der Version mit Heuristiken signifikant mit $p < 0,05$ und $p > 0,04$ besser aus. Die minimale Rangsumme (T) von $1,5$ der Beobachtungen ist bei einer Stichprobenanzahl von 7[52] (n) bei beidseitiger Fragestellung kleiner als der kritische Wert für $\alpha = 0,05$ ($= 2$) und größer als der kritische Wert für $\alpha = 0,04$ ($= 1$).

Für die restlichen Aussagen konnte nur mehr eine zuverlässige Bewertung der Signifikanz durchgeführt werden: Hinsichtlich Aussage 1 (*„Insgesamt bin ich zufrieden wie leicht es ist das System zu benutzen.“*) konnte kein signifikanter Unterschied mit einem Niveau von $\alpha = 0,05$ in den beiden Versionen festgestellt werden: Die minimale Rangsumme ist mit $1,5$ größer als der kritische Wert 0 der Tabelle der kritischen Werte.

Tabelle 8.3 zeigt einen Überblick des Ergebnisses der Tests für die Aussagen 1 und 4:

	Mittelwert	**T**	**z**	**n**	**p**
Aussage 1	$4,50$ (ohne) und $6,00$ (mit)	$1,5$	$-1,89$	6	$p > 0,05$
Aussage 4	$4,13$ (ohne) und $5,63$ (mit)	$1,5$	$-2,11$	7	$p < 0,05$

Tabelle 8.3 – Ergebnisse des Wilcoxon Vorzeichen Rang Tests

Bei den übrigen Aussagen war eine Analyse nicht möglich, da paarweise Beurteilungen, welche in beiden Versionen gleich sind, für den Test gekürzt werden und dadurch weniger als 6 Stichproben übrig blieben. Eine aussagekräftige Analyse der Signifikanz ist bei diesen Aussagen des Post-Test Fragebogens daher nicht möglich: Die Tabelle der kritischen Werte beinhaltet für eine solch geringe Anzahl an Stichproben keine Werte. [Zöf08, Seite 266].

[52] Bei 8 Probanden schied eine paarweise verbundene Stichprobe aus, da diese eine gleiche Bewertung in beiden Versionen aufwies.

8.2.5 Performance Faktor Zeit

Für jede Testgruppe wurde pro Szenario die benötigte Zeit erhoben und getrennt für die Version ohne Heuristiken sowie die mit Heuristiken der Mittelwert und die Standardabweichung berechnet.

Bei der Analyse der zeitlichen Auswertung muss vorweg zwischen den kollaborativen und konkurrierenden Aufgaben unterschieden werden: Aufgrund der unterschiedlich langen Diskussionen der Probanden, beispielsweise über die Art der Sortierung und Anordnung von Bildern, sind zeitliche Aufzeichnungen kollaborativer Szenarien nicht so aussagekräftig wie jene der konkurrierenden.

Abbildung 8.2 und Tabelle 8.4 geben einen Überblick über die von den Probanden für die Abarbeitung der Szenarien benötigten Zeiten. Es fällt dabei auf, dass kollaborative Szenarien (Szenario 1 und 4) in der Version mit Heuristiken (Version 2) deutlich schneller abgearbeitet wurden. Die aufgezeichneten Zeiten von Szenario 4 sind jedoch mit einer sehr hohen Standardabweichung behaftet, was auf unterschiedlich lange Diskussionen und unterschiedliche Arten der Sortierung der Fotos zurückzuführen ist.

Hinsichtlich der konkurrierenden Szenarien (Szenarien 2, 3 und 5) lässt sich folgendes feststellen: Bei einer sehr geringen Standardabweichung konnten die Szenarien 2 und 3 wiederum in der verbesserten Version schneller erledigt werden. Szenario 5 wurde hingegen durchschnittlich gesehen schneller in der Version ohne Heuristiken erfüllt, jedoch mit einer deutlich höheren Standardabweichung als bei Szenario 2 und 3.

Aufgrund der geringen Anzahl an Stichproben (vier für jede Version - bei Zweier-Gruppen und acht Testteilnehmern) sind die zeitlichen Aufzeichnungen und die Schlüsse jedoch lediglich als generelle Tendenzen zu sehen bei denen noch keinerlei aussagekräftige Signifikanz festgestellt werden kann. Auf eine weiterführende Überprüfung auf Signifikanz wird aus diesem Grund verzichtet.

	Mittelwert V1 (ohne)	**Mittelwert V2 (mit)**	**Differenz V1 und V2**	**Benchmark**
Szenario 1	1,55 (0,41)	1,18 (0,35)	0,37	3,50
Szenario 2	1,12 (0,30)	0,99 (0,24)	**0,13**	1,25
Szenario 3	0,96 (0,21)	0,87 (0,10)	**0,09**	1,50
Szenario 4	3,23 (2,24)	2,30 (1,05)	0,93	3,00
Szenario 5	1,20 (0,59)	1,28 (0,56)	**-0,08**	1,50

Tabelle 8.4 – Vergleich der benötigten Zeiten in Minuten. Standardabweichungen sind in Klammern. Der Vergleich der konkurrierenden Szenarien ist fett gedruckt. Benchmark entspricht der im Pilottest erhobenen Zeit.

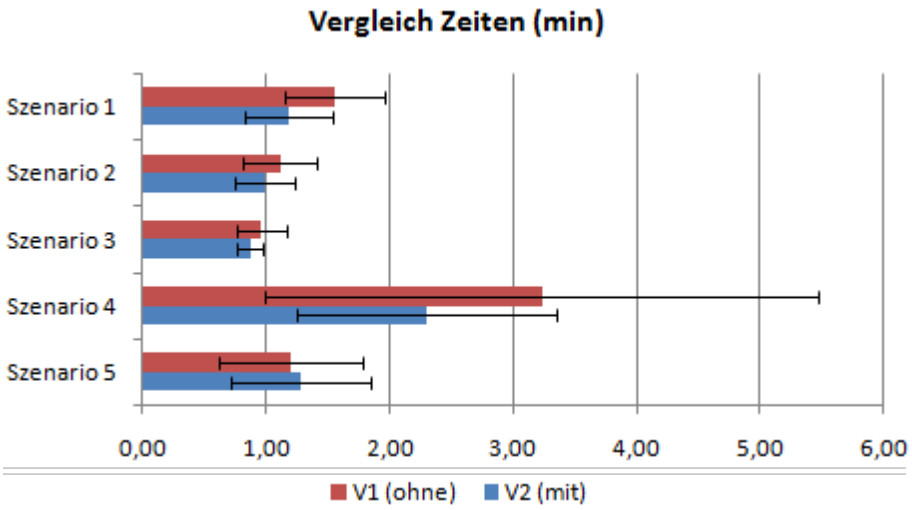


Abbildung 8.2 – Vergleich der benötigten Zeiten in Minuten

8.2.6 Ergebnis der Evaluierung des Tests in Vierer-Gruppe

Bei der Evaluierung mit vier Testpersonen, welche zur Simulation eines Grenzfalles eingesetzt wurden, stach heraus, dass Probleme in Version 1 noch häufiger auftraten. Vor allem die unbeabsichtigte Ausführung der Zoom- und Pan-Geste störten die Testteilnehmer immer wieder bei der Ausführung der Aufgaben und unterbrach den Arbeitsfluss. Im Gegensatz zu den Tests in zweier Gruppen, fiel es den Probanden sichtlich schwerer den Ausgangszustand vor Auftreten eines Problems wieder herzustellen: Zu viel wurde sofort nach Auftreten des Problems wieder von anderen Personen verändert und die Testpersonen brauchten einige Zeit bis sie sich einigten wer den Systemzustand nun wieder herstellt.

Weiters fiel auf, dass in der Version ohne Heuristiken - wie auch bereits in den Tests mit zwei Personen - Fotos oftmals zu groß skaliert wurden. Die Auswirkungen waren hierbei jedoch deutlich störender, da dieses Mal nicht nur der Arbeitsprozess von einer weiteren Person gestört wurde, sondern von drei.

In der Version mit Heuristiken gehen die Ergebnisse konform mit denen der Tests in Zweier-Gruppen: Das Problem der unbeabsichtigt, gemeinsam ausgeführten Zoom-Geste ist in dieser Evaluierung nicht mehr aufgetreten. Ebenso wurde keine Pan-Geste unbeabsichtigt ausgeführt. Das Problem der gemeinsam ausgeführten Skalier-Geste - ein Bild wurde vergrößert als es zwei Personen zu sich ziehen wollten - trat wieder auf, jedoch nur in den konkurrierenden Szenarien.

8.2.7 Zusammenfassung und Diskussion der Ergebnisse der Evaluierung

Die in diesem Kapitel vorgestellten Ergebnisse der Evaluierung geben folgendes Bild:

Identifikation von Problemen. In der Version ohne implementierte Heuristiken traten sämtliche, bereits in einer informellen Studie identifizierten Probleme der Mehrbenutzer-Anwendung wieder auf. So wurden etwa Gesten unbeabsichtigt getriggert: sowohl einzeln als auch gemeinsam. Unbeabsichtigt getriggerte Pan-, Zoom- und Skalier-Gesten wurden als störend für die Benutzer empfunden. Räumlich enges arbeiten stellte ebenfalls ein Problem dar, durch welches unerwartete Systemzustände hervorgerufen wurden. Zusätzliche Probleme, die über jene der Pilotstudie hinausgehen, konnten nicht beobachtet werden.

Auswirkungen der Probleme. In den Ergebnissen der Evaluierung kann generell der Trend beobachtet werden, dass die Probleme der Mehrbenutzer-Anwendung die Probanden der Evaluierung störten und irritierten. Die Probanden konnten trotzdem ihre Aufgaben erfüllen, wurden jedoch in ihrem Arbeitsfluss unterbrochen. Dies zeigte sich sowohl in den Beobachtungen als auch in den qualitativen Interviews: *„Man findet sich mit den Problemen zurecht, weil man weiß dass man einen Fehler gemacht hat, aber es stört.“*

Vergleich der beiden Versionen. Vergleicht man die Version mit implementierten Heuristiken mit der ohne, resultiert aus den Ergebnissen, dass einige Probleme gar nicht mehr beziehungsweise einige weniger häufig auftreten: Die Auswertungen zu den Beobachtungen und zu den Interviews gehen dabei konform und indizieren kein Auftreten der Probleme 2a, 2b, 3a, 3b sowie 3d im Gegensatz zu der Version ohne Heuristiken. Die Heuristiken, welche diese Probleme adressieren, können somit zur Verbesserung dieser als bestätigt angesehen werden:

- Heuristik 8 für Problem 2a.
- Heuristiken 1, 4, 5, 6 und 10 für Problem 2b.
- Heuristik 11 für die Probleme 3a und 3b.
- Heuristik 8 für Problem 3d.

Die Probleme 1a, 1b und 3c sind in der Version mit Heuristiken wieder, wenn weniger häufig, aufgetreten. Dies zeigen sowohl Beobachtungen als auch die Wahrnehmung der Probanden beziehungsweise deren Aussagen in

den Interviews. Wie bereits angeführt, ist es jedoch nicht möglich respektive gewünscht durch Heuristik 3 alle Probleme dieser Form zu lösen, da dies ansonsten zu einem Bruch der Intuition in der Ausführung der Geste führen kann. Hinsichtlich der Probleme 1b, 3c kann festgehalten werden, dass die Wahl des Parameters zur Regelung der Sperre der Skalier-Geste nach Identifikation einer Zieh-Geste noch optimiert werden kann. Ein gesonderter User Test kann Auskunft über einen adäquaten Wert für diese Konstante geben. Aufgrund des weniger häufigen Auftretens der Probleme in der Version mit Heuristiken, lässt sich feststellen, dass diese beiden Heuristiken die negativen Auswirkungen dieser Probleme verminderten, jedoch die Probleme nicht gänzlich lösten.

In einem Vergleich der beiden Versionen lässt sich demnach der generelle Trend erkennen, dass die Version mit Heuristiken besser geeignet für die Mehrbenutzer-Anwendung ist. Dies zeigen einerseits die Interviews, in welchen Probanden etwa äußerten: *„Das einzige was ich nicht gut finde, ist wenn man sich in die Quere kommt, aber das war heute nicht der Fall. Prinzipiell ist es für mich heute viel besser gegangen.“* Andererseits bestärkt diesen Trend auch die Erhebung der subjektiven Zufriedenheit der Testpersonen im Post-Test-Fragebogen, in welchem jede Aussage in der Version mit Heuristiken durchschnittlich besser bewertet wurde als in der Version ohne Heuristiken.

Zeitliche Ergebnisse und Art der Interaktion. Wie in der Analyse des Performance Faktors Zeit festgestellt, lässt sich eine grundsätzliche Tendenz feststellen, dass die Aufgaben in der Version mit Heuristiken schneller abgearbeitet werden können als bei der Version ohne Heuristiken. Aufgrund der geringen Anzahl an Stichproben, sind diese Ergebnisse jedoch nur als grundsätzliche Richtung zu sehen, bei welcher keine zuverlässige Signifikanz festgestellt werden kann. Bestärkt wird diese Tendenz durch die qualitativen Elemente der Studie: Die Testpersonen sprechen in den Interviews dabei etwa davon, dass sie nicht mehr gleichzeitig interagierten, sondern abwechselnd nacheinander. Dies hat zur Folge, dass sich durch das wiederholte Auftreten der Probleme der Mehrbenutzer-Anwendung nicht nur die Szenarien nicht mehr so schnell abgearbeitet werden können, sondern auch, dass sich die generelle Art der Interaktion verändert. Diese Beobachtung konnte auch in der Observation der Testpersonen gemacht werden. In den Interviews äußerte sich dieser Umstand etwa folgendermaßen: *„Das (Anmerkung: unbeabsichtigt, gemeinsam getriggerte Gesten) hat ein Problem bei der Benutzung dargestellt, weil wir [...] dadurch viel mehr aufeinander Rücksicht nehmen mussten. Wir haben dann eher nacheinander getippt und nicht mehr gleichzeitig.“*

Qualitätsmerkmale. Die Heuristiken dieser Arbeit zielen darauf ab, Multitouch-Systeme bei Mehrbenutzer-Anwendung hinsichtlich den Qualitätskriterien der Robustheit, Korrektheit, Performance sowie Stabilität zu verbessern.

Bezüglich der Robustheit der Applikation lässt sich feststellen, dass einige der Probleme als mangelnde Fehlertoleranz wahrgenommen werden: *„Die Fehlertoleranz war dadurch (Anmerkung: durch unbeabsichtigt getriggerte Pan-Geste) gleich Null, wenn man zum Beispiel daneben greift, dann zieht man nicht das Foto, sondern alles. Und dann waren gleich einmal alle Fotos weg [...].“* In dieser Arbeit wird die in diesem Zitat angeführte Fehlertoleranz entsprechend der Definition in Kapitel 2.4 als Robustheit verstanden. In der verbesserten Version sind die beschriebenen Probleme, welche etwa durch unpräzise Selektion oder durch einen zusätzlichen Berührungspunkt auf ein gerade skaliertes/rotiertes Objekt hervorgerufen werden, nicht mehr aufgetreten. Es lässt sich daher von einer gesteigerten Robustheit in der Version mit Heuristiken ausgehen.

Ebenso wurde in der Version mit implementierten Heuristiken die Korrektheit verbessert: Entsprechend der Definition dieses Qualitätskriteriums in Kapitel 2.4, wird die Korrektheit in dieser Arbeit auf Multitouch User Interfaces bezogen interpretiert: Werden alle Eingaben richtig von der Software als Gesten interpretiert? In der Version ohne Heuristiken wurden Gesten falsch interpretiert, wenn zwei Benutzer den Hintergrund schwenken oder ein Objekt zu sich ziehen wollten. Diese beiden Berührungspunkte wurden dann jeweils als Zoom- beziehungsweise Skalier-Geste fehl interpretiert. In der Version mit Heuristiken traten diese beiden Probleme weniger häufig auf. Daher lässt sich auf eine verbesserte Korrektheit schließen.

Hinsichtlich der Performance, worunter in dieser Arbeit die Flüssigkeit im Programmablauf verstanden wird, konnten keine Unterschiede zwischen den beiden Versionen erkannt werden. Vor der Evaluierung wurde angenommen, dass die Analyse zeitlicher Sequenzen, wie dies in Heuristik 11 geschieht, negative Auswirkungen auf die Flüssigkeit der Interaktion im Allgemeinen hat. Dies konnte in der Evaluierung nicht bestätigt werden.

Ähnlich verhält es sich mit dem Qualitätskriterium der Stabilität. Unter diesem Merkmal wird in der Arbeit verstanden, dass Gesten stabil und kontinuierlich über die Zeit hinweg korrekt interpretiert werden sollen. Wird etwa eine zusätzliche Geste in das Repertoire aufgenommen, sollen die vorhandenen Gesten noch immer korrekt interpretiert werden können. In diesem Teilbereich des Qualitätskriteriums waren keine Veränderungen beobachtbar. Jedoch erscheint diese Analyse nicht aussagekräftig, da auf eine Erweiterung des Gestenrepertoires in der Implementierung des Prototyps mit Heuristiken beziehungsweise im Testdesign verzichtet wurde. Zusätzlich wirken die Heuristiken 10 und 11 in diesem Qualitätsbereich: Diese Heuristiken konnten im Zuge der Evaluierung bestätigt werden. Sie sorgen dafür, dass

Gesten korrekt über die Zeit hinweg interpretiert werden und etwa weniger Probleme der inkorrekten Gruppierung von Berührungspunkten auftreten.

Kapitel 9

Diskussion der Ergebnisse

In dieser Arbeit wurde am Beispiel eines Multitouch-Tisches deutlich, dass Multitouch-Systeme eine vielversprechende Technologie darstellen, welche eine natürliche, direkte und intuitive Mensch-Computer-Interaktion ermöglicht. Es werden die Fähigkeiten des Menschen benutzt, um via Gesten das System über eine direkte Kopplung von Eingabe- und Ausgabegerät zu steuern.

Probleme der Mehrbenutzer-Anwendung. Neben der Natürlichkeit der Interaktion, ist ein weiterer Vorteil der Multitouch-Technologie die Möglichkeit der Mehrbenutzer-Anwendung, die gerade durch die in dieser Arbeit betrachteten Multitouch-Tische aufgrund der Größe des Interfaces naheliegend ist. Diese Fähigkeit hebt sich klar von traditionellen Systemen ab, bei welchen die Interaktion immer nur einer Person vorbehalten ist beziehungsweise beispielsweise mehrere indirekte Zeiger aufgrund des Einsatzes mehrerer Computer-Mäuse die Benutzer verwirren würde. Die Mehrbenutzer-Anwendung steht jedoch noch einigen Erschwernissen gegenüber, da jede Geste aus mehreren, unabhängigen Touch-Punkten besteht und diese interpretiert und gruppiert werden müssen: Bei steigender Anzahl von Benutzern und damit einhergehender Steigerung der Komplexität der Interaktion, erhöht sich die Wahrscheinlichkeit von inkorrekten Gruppierungen.

Diese Probleme der Mehrbenutzer-Anwendung wurden in der vorliegenden Arbeit identifiziert und definiert. Probleme, welche in der Literatur angeführt werden, konnten durch eine informelle Pilotstudie bestätigt werden sowie neue identifiziert werden.

So werden etwa Touch-Punkte mehrerer, unterschiedlicher Benutzer nicht korrekt zu Gesten gruppiert. Dies kann dazu führen, dass Gesten unbeabsichtigt, gemeinsam getriggert werden (Problemgruppe 1). Eine Ausprägung dieses Problems wäre, dass durch mehrere unbeabsichtigte Berührungspunkte auf dem Hintergrund eine Zoom-Geste erkannt wird. Diese unbeabsichtigten Berührungspunkte können beispielsweise durch unpräzise Selektion, ei-

nem wesentlichen Nachteil der Multitouch-Technologie, hervorgerufen werden. Weiters kann es vorkommen, dass aufgrund fehlender Koordination der Benutzer das gleichzeitige ziehen eines Objekts zur Interpretation einer Skalier-Geste führt. Die Identifikation dieses Problem geht dabei konform mit [PKS+08].

Zusätzlich führen unbeabsichtigt, alleine getriggerte Gesten (Problemgruppe 2) zu einem Problem. Dabei wird zwar keine Geste durch mehrere Berührungspunkte verschiedener Benutzer fehl interpretiert, jedoch werden aufgrund mangelnder Robustheit des Systems etwa Objekte zu stark vergrößert oder eine Pan-Geste unbeabsichtigt ausgeführt. Dies stört die restlichen beteiligten Benutzer vor allem in individuellen Arbeitsphasen im Rahmen der kollaborativen Tätigkeit.

Weitere identifizierte Probleme treten aufgrund der Störung von Gesten durch andere Benutzer, durch räumlich enges Arbeiten der Benutzer, durch das Fehlen von Grenzen der Arbeitsbereiche (Problemgruppe 3), durch fehlende Identifikation der Benutzer (Problemgruppe 4) sowie durch eine Erweiterung des Gesten-Repertoires (Problemgruppe 5) auf. Vergleicht man dieses Ergebnis mit verwandten Arbeiten zum Thema fällt auf, dass hierbei einige Probleme bestätigt, wie etwa die Erweiterung des Gesten-Repertoires oder die Missinterpretation einer gleichzeitigen Zieh-Geste als Skalier-Geste, sowie einige neu identifiziert werden konnten. In verwandten Arbeiten - wie beispielsweise in [DL01], [DDSP08] oder [DSA09] - findet sich der Standpunkt, dass eine Identifikation der Benutzer diese Problematiken löst. [DL01, Seite 220] führt etwa an, eine Identifikation der Benutzer sei eine Anforderung an Multitouch-Umgebungen. Diesen Annahmen stehen jedoch Probleme entgegen, welche nicht durch die fehlerhafte Gruppierung von Touch-Punkten verschiedener Benutzer rühren: Unbeabsichtigt, alleine getriggerte Gesten unterbrechen den Arbeitsfluss anderer Benutzer, welche etwa in individuellen Arbeitsphasen tätig sind.

Wie aus der Studie zur Evaluierung der Heuristiken hervor ging, fühlten sich die Testteilnehmer durch das Auftreten der Probleme gestört und meinten, dadurch auch andere Teilnehmer zu stören. Ein Proband führte zusätzlich an, dass die Arbeit dadurch erschwert wurde. Die Aufgaben, die im Rahmen der Studie an die Probanden gestellt wurden, konnten dennoch erfüllt werden.

Heuristiken versus zusätzliche Hardware. In dieser Arbeit wurden Heuristiken, welche als Prinzipien zur Verbesserung der Qualität in den Ausprägungen Korrektheit, Performance, Stabilität und Robustheit gesehen werden, entwickelt, um den erwähnten Problemen der Mehrbenutzer-Anwendung entgegen zu treten. Im Gegensatz zu den meisten verwandten Arbeiten zum Thema, handelt es sich dabei um Ansätze, die durch Algorithmen versuchen die Probleme zu beseitigen. In der verwandten Literatur

wird hauptsächlich zusätzliche Hardware eingesetzt, um beispielsweise über Kameras[53], spezielle Stifte[54] für die Eingabe oder spezielle Sensoren[55] zwischen Benutzern zu differenzieren und diese Information zur Behebung der Probleme zu benutzen. Eine Analyse der Methodiken zeigte jedoch Defizite in der Flexibilität der Benutzung: Werden etwa zusätzliche Kameras über dem Tisch benötigt, kann dieser nicht mehr in jedem Raum eingesetzt werden. Werden zusätzliche Stifte eingesetzt, um Benutzer zu identifizieren, widerspricht dies den in [DL01, Seite 220] definierten Anforderungen an Multitouch-Umgebungen. Die Benutzer werden zusätzlich belastet. Außerdem wird die Möglichkeit der Interaktion auf „Singletouch“ pro Hand zurückgesetzt. In dieser Arbeit wurde daher ein softwareseitiger Ansatz vorgestellt. Eine vergleichbare Methodik wählt auch [DSA09], verfolgt diese jedoch nur am Rande: Die Möglichkeit der Identifikation der Hände der Benutzer durch den in [DSA09] vorgestellten Algorithmus - und damit die Aussicht zur Behebung einiger Probleme der Mehrbenutzer-Anwendung - wird jedoch nur als Nebeneffekt erwähnt.

In der Arbeit wurden zahlreiche Möglichkeiten aufgezeigt, wie mit Algorithmen und Design Guidelines die Qualität hinsichtlich der Fähigkeit der Mehrbenutzer-Anwendung gesteigert werden kann: Diese erschließen sich auf den Ebenen der gestenspezifischen, räumlichen, objektspezifischen und benutzerspezifischen Segmentierung. So konnte für jedes identifizierte Problem mindestens eine Verbesserungsmöglichkeit aufgezeigt werden. Dabei wurden auch bestehende Ansätze für Heuristiken aufgegriffen und hinsichtlich der Mehrbenutzer-Anwendung interpretiert. Aufgegriffen wurden so beispielsweise „Iceberg Tips“ [Saf09, Seite 43], welche in den Heuristiken eingesetzt werden, um unbeabsichtigte Berührungspunkte aufgrund unpräziser Gesten zu minimieren. Dadurch wird ein ungewolltes Ausführen einer Pan-Geste und damit eine Störung der anderen beteiligten Benutzer vermieden. Bezüglich der in dieser Arbeit mehrfach erwähnten und in Kapitel 4.1 erweiterten Anforderungen an Multitouch-Umgebungen müssen, setzt man die vorgestellten Heuristiken um, lediglich Abstriche hinsichtlich der Identifikation der Benutzer gemacht werden: Durch die beiden vorgestellten Heuristiken im Bereich der benutzerspezifischen Segmentierung ist eine Identifikation nur bedingt möglich, da das Einsatzgebiet von 14 auf Casual Games beschränkt wurde. Durch Heuristik 13 wird es zwar möglich Hände zu identifizieren, jedoch nicht die Benutzer selbst.

Eine solche Identifikation der Benutzer, wie es teilweise Systeme mit zusätzlicher Hardware gewährleisten, kann nicht nur zur Verbesserung der Probleme der Mehrbenutzer-Anwenung verwendet werden - es ergeben sich dadurch auch zusätzliche Interaktionsmöglichkeiten. So können komplexe,

[53] etwa in [DDSP08] und [SG09].
[54] etwa in [LPB+09] und [BHH+07].
[55] etwa in [DL01].

kooperative Gesten und neue Interaktionsmethoden (wie beispielsweise in [RMHPW06]), Regelung von Zugriffsberechtigungen (wie beispielsweise in [RMRS+04]) sowie für Multi-User Anwendung optimierte Interfaces (wie beispielsweise in [REE+05]) erschlossen werden.

Zusammengefasst lässt sich daher feststellen, dass der flexible und schnelle Einsatz der Heuristiken sowie die angeführten Nachteile der Verfahren mit zusätzlicher Hardware, den zusätzlichen Interaktionsmöglichkeiten dieser Systeme gegenüber stehen.

Eignung der Heuristiken. Zur Evaluierung der vorgestellten Heuristiken wurde in dieser Arbeit ein explorativer, vergleichender User Test durchgeführt. Es wurde dabei eine Version eines Prototyps, bei welcher die Heuristiken implementiert waren, mit einer Version ohne Heuristiken verglichen. Dieser Test zeigte, dass in der Version mit implementierten Heuristiken einige der Probleme nicht mehr, einige weniger häufig auftraten. Somit können die zugehörigen Heuristiken als bestätigt angesehen werden. Bei anderen besteht hingegen noch Verbesserungsbedarf. So ist die Wahl des Parameters bis ein Objekt nach der eindeutigen Identifikation einer Zieh-Geste gesperrt wird noch nicht optimal gewählt: Zwei zeitlich knapp aufeinander folgende Zieh-Gesten verschiedener Benutzer wurden in den Tests, wenn auch weniger häufig, noch immer als Skalier-Geste interpretiert. Weiters fiel im Rahmen der Evaluierung auf, dass zwei Berührungspunkte verschiedener User am Hintergrund noch immer fälschlicherweise als Zoom-Geste interpretiert wurden. Dies ist in der verbesserten Version wiederum weniger häufig aufgetreten. Eine gänzliche Eliminierung dieses Problems mit der vorgeschlagenen Heuristik ist jedoch, wie bereits angeführt, nicht möglich. Zu groß wären die Einschränkungen hinsichtlich der Intuition in der Ausführung einer Zoom-Geste. Es lässt sich daher folgern, dass sich bei steigender Anzahl von Benutzern die Wahrscheinlichkeit von inkorrekten Gruppierungen erhöht. Diese Wahrscheinlichkeit kann durch die Heuristiken verringert werden, jedoch ist es nicht möglich inkorrekte Gruppierungen gänzlich zu verhindern.

In der Evaluierung besonders bewährt hat sich der Einsatz der Heuristiken zur Beseitigung von Störungen aufgrund von unbeabsichtigt, alleine getriggerten Gesten. Probleme mit dieser Ausprägung konnten in der verbesserten Version des Prototyps nicht mehr beobachtet werden. Auch die durch den Einsatz einer Pan-Geste basierend auf zwei Fingern erwarteten Einbußen hinsichtlich der Natürlichkeit der Interaktion konnten nicht bestätigt werden: Die Auswertung des Post-Test-Fragebogens - im Hinblick auf wie leicht es war die Benutzung des Systems zu erlernen - , ergab keine signifikanten Unterschiede zwischen den beiden Versionen. Die Version mit Heuristiken wurde in dieser Aussage sogar durchschnittlich etwas besser beurteilt. Da Probleme hervorgerufen durch unbeabsichtigt, alleine getriggerte Gesten nicht durch die Identifikation von Benutzern - wie dies etwa durch den Ein-

satz zusätzlicher Hardware in den verwandten Arbeiten angestrebt wird - abgedeckt werden, wäre hierbei auch eine Kombination sinnvoll: Verfahren mit zusätzlicher Hardware können mit den Heuristiken kombiniert werden, um so alle Probleme der Mehrbenutzer-Anwendung abzudecken.

Ein Phänomen, das im Rahmen des Tests beobachtet werden konnte und auch in den qualitativen Interviews angeführt wurde, war, dass sich die Art der Interaktion durch das Auftreten der Probleme änderte. Benutzer sprachen etwa davon, dass sie nach den Problemen nur mehr nacheinander und nicht mehr gleichzeitig, miteinander interagierten. Dies könnte ebenfalls Auswirkungen auf die zeitliche Performance haben: Die Testteilnehmer waren bei der Abarbeitung ihrer Aufgaben in der Version ohne Heuristiken durchschnittlich langsamer. Diese zeitlichen Diffenzen sind jedoch aufgrund der geringen Anzahl an Stichproben nur als genereller Trend zu sehen.

Betrachtet man die angeführten Qualitätskriterien, welche durch die Heuristiken untersucht werden sollten, ließ sich im Rahmen des User Tests feststellen, dass die Robustheit und Korrektheit gesteigert werden konnten. Hinsichtlich der Performance konnten keine Unterschiede festgestellt werden. Eine aufgrund der Analyse von zeitlichen Sequenzen erwartete Einschränkung in diesem Kriterium kann daher nicht bestätigt werden. Bezüglich der Stabilität wurden ebenfalls keine Unterschiede festgestellt.

Kapitel 10

Conclusio

Diese Arbeit beschäftigt sich damit, Probleme der Mehrbenutzer-Anwendung bei Multitouch zu erkennen und Möglichkeiten aufzuzeigen, um deren Auswirkungen in Form von Heuristiken und Algorithmen zu minimieren. Ausgegangen wurde von der These, es gäbe Algorithmen und Design Guidelines zur Optimierung der Multitouch-Interaktion mit Applikationen, die mehrere Benutzer unterstützen.

Wie bereits die theoretische Abhandlung zeigt, handelt es sich um ein aktuelles Thema: Die Technologie Multitouch erhält große Aufmerksamkeit, verschiedene Produkte welche Mehrbenutzer-Anwendung zulassen werden angedacht beziehungsweise bereits vorgestellt. Wissenschaftlich debattiert werden unterschiedliche Lösungsansätze für Probleme, welche die Mehrbenutzer-Anwendung mit sich bringt. Die meisten Vorschläge versuchen durch den Einsatz zusätzlicher Hardware Benutzer zu identifizieren und so den Problemen entgegen zu wirken. Wie diese Arbeit zeigt, bringen diese Systeme jedoch allesamt Einschränkungen mit sich und können nicht alle der identifizierten Probleme lösen. Hinsichtlich dieses Verbesserungspotentials wird in dieser Arbeit der Ansatz vorgestellt, wie man auf Softwareseite die Qualität einer Mehrbenutzer-Multitouch-Applikation steigern kann.

Für jedes der identifizierten Probleme wird daher mindestens eine Heuristik vorgestellt, welche darauf abzielt, die Mehrbenutzer-Fähigkeit in den Qualitätskriterien Robustheit, Performance, Stabilität und Korrektheit zu steigern. Die Heuristiken verfolgen dabei unterschiedliche Ansätze der Segmentierung: gestenspezifisch, räumlich, objektspezifisch und benutzerspezifisch. Dabei werden sowohl applikationsspezifische als auch globale Verbesserungsmethoden vorgeschlagen. In der Praxis können die Heuristiken dadurch individuell angepasst und für unterschiedliche Interfaces basierend auf der Multitouch-Technologie eingesetzt werden.

Eine explorative, vergleichende Studie mit acht Probanden zwischen einer

Version mit implementierten Heuristiken und einer ohne, zeigte, dass sich durch die Heuristiken die Zufriedenheit bei der Benutzung der Applikation steigern lässt und gleichzeitig weniger Probleme auftreten. Dadurch kann sowohl die Robustheit als auch die Korrektheit der Eingabeverarbeitung verbessert werden. Wurde bei der Version ohne Heuristiken aufgrund der auftretenden Probleme nicht mehr gleichzeitig interagiert, sondern abwechselnd nacheinander, konnte dies bei der Version mit Heuristiken nicht beobachtet werden. In der Folge waren die Testperson in der Version mit Heuristiken schneller in der Abarbeitung der Tasks.

Einige der identifizierten Probleme traten in der verbesserten Version jedoch wieder, wenn auch in geringerer Anzahl, auf. Zum einen auf Grund einer Fehlinterpretation zweier unbeabsichtigter Berührungspunkte am Hintergrund als Zoom-Geste. Die Heuristik, welche dieses Problem adressiert, kann das Auftreten dieses Problem nur reduzieren, nicht beseitigen - ansonsten müssten Abstriche hinsichtlich der Intuition in der Ausführung einer Zoom-Geste gemacht werden. Zum anderen wurden zwei zeitlich knapp aufeinander folgende Zieh-Gesten zweier verschiedener Benutzer in der Version mit Heuristiken fälschlicherweise noch immer als Skalier-Geste interpretiert.

An der Heuristik, welche diese Fehlinterpretation zweier Zieh-Gesten adressiert, könnten zukünftige Arbeiten ansetzen, um einen optimalen Wert für den Parameter der Heuristik zu finden. Weitere Forschungsarbeit wäre zusätzlich notwendig, um die Heuristiken, welche im Prototyp der Studie nicht angewendet wurden, zu evaluieren. Dadurch sollen mögliche Schwächen und Vorteile dieser Heuristiken erkannt werden. Dies betrifft vor allem die applikationsspezifisch vorgeschlagenen Ansätze, welche aufgrund dieser Spezifika in der Studie keine Anwendung finden konnten. Des weiteren sollte noch einmal detaillierter in einer Folgestudie überprüft werden, ob eine Abänderung des de-fakto Standards einer Pan-Geste - wie dies in einer Heuristik vorgeschlagen wird - Auswirkungen auf die Natürlichkeit der Interaktion hat. Im Rahmen der in dieser Arbeit durchgeführten Studie konnten keine Unterschiede festgestellt werden. Die Version mit Heuristiken wurde in der Aussage des Post-Test-Fragebogens hinsichtlich der leichten Erlernbarkeit der Benutzung der Applikation sogar durchschnittlich etwas besser beurteilt als die Version ohne Heuristiken. Bezüglich der statistischen Auswertung der Evaluierung wird weiters - um aussagekräftigere Ergebnisse zu erhalten - eine zusätzliche Studie mit einer größeren Stichprobengröße vorgeschlagen.

Zusammenfassend lässt sich festhalten, dass sich bei steigender Anzahl von Benutzern - und damit einhergehender steigender Komplexität der Interaktion - die Wahrscheinlichkeit von inkorrekten Gruppierungen beziehungsweise inkorrekter Interpretation von Gesten erhöht. Diese Wahrscheinlichkeit kann softwareseitig durch Heuristiken, welche verschiedene Ebenen der

Segmentierung nützen, verringert und in einigen Ausprägungen minimiert werden, jedoch ist es nicht möglich inkorrekte Gruppierungen gänzlich zu verhindern. Trotzdem erscheint die Technologie Multitouch geeignet, um eine deutliche Erleichterung und Verbesserung der Interaktionsmöglichkeiten zu erzielen: Mehrere Benutzer können auf einer Touch-Oberfläche interagieren, gemeinsam Aufgaben lösen und sollen durch die im Rahmen dieser Arbeit präsentierten Maßnahmen unterstützt werden, möglichst störungs- und fehlerfrei arbeiten zu können.

Literaturverzeichnis

[BGBL04] Renaud Blanch, Yves Guiard, and Michel Beaudouin-Lafon. Semantic pointing: Improving target acquisition with control-display ratio adaption. *CHI '04: Proceedings of the SIG- CHI conference on Human factors in computing systems*, pages 519–526, 2004.

[BHH+07] Peter Brandl, Michael Haller, Michael Hurnaus, Verena Lugmayr, Juergen Oberngruber, Claudia Oster, Christian Schafleitner, and Mark Billinghurst. An adaptable rear-projection screen using digital pens and hand gestures. *Conference on Artificial Reality and Telexistence*, pages 49–54, November 2007.

[BHR85] William Buxton, Ralph Hill, and Peter Rowley. Issues and techniques in touch-sensitive tablet input. *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 215–224, 1985.

[BP09] Andrea Birbaumer and Martin-David Preyer. Qualitative methoden der gestaltung von multimediasystemen. Skriptum zur gleichnamigen Vorlesung und Übung, Technische Universität Wien, 2009.

[Bro96] John Brooke. Sus - a quick and dirty usability scale. *P. W. Jordan, B. Thomas, B. A. Weerdmeester, I. L. McClelland: Usability Evaluation in Industry, Taylor and Francis Ltd*, pages 189–194, 1996.

[BS00] Marion Buchenau and Jane Fulton Suri. Experience prototyping. *DIS '00: Proceedings of the 3rd conference on Designing interactive systems*, pages 424–433, 2000.

[DDSP08] K.C. Dohse, Thomas Dohse, Jeremiah D. Still, and Derrick J. Parkhurst. Enhancing multi-user interaction with multi-touch tabletop displays using hand tracking. *Proceedings of the First International Conference on Advances in Computer-Human Interaction*, pages 297–302, Februar 2008.

[DL01] Paul Dietz and Darren Leigh. Diamondtouch: A multi-user touch technology. *ACM symposium on User interface software and technology*, pages 297–302, November 2001.

[DSA09] Chi Tai Dang, Martin Straub, and Elisabeth André. Hand distinction for multi-touch tabletop interaction. *Proceedings of Interactive Tabletops and Surfaces 2009*, pages 101 – 108, 2009.

[FWSB07] Clifton Forlines, Daniel Wigdor, Chia Shen, and Ravin Balakrishnan. Direct-touch vs. mouse input for tabletop displays. *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 647–656, 2007.

[Han05] Jefferson Y. Han. Low-cost multi-touch sensing through frustrated total internal reflection. *Symposium on User Interface Software and Technology*, pages 115–118, Oktober 2005.

[HBBS08] Johannes Hirche, Peter Bomark, Mikael Bauer, and Pawel Solyga. Adaptive interface for text input on large-scale interactive surfaces. *IEEE Tabletops and Interactive Surfaces 2008*, pages 163–166, 2008.

[IU97] Hiroshi Ishii and Brygg Ullmer. Tangible bits: towards seamless interfaces between people, bits and atoms. *CHI '97: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 234–241, 1997.

[JGAK07] Sergi Jordà, Günter Geiger, Marcos Alonso, and Martin Kaltenbrunner. The reactable: exploring the synergy between live music performance and tabletop tangible interfaces. *TEI '07: Proceedings of the 1st international conference on Tangible and embedded interaction*, pages 139–146, 2007.

[KAD09] Kenrick Kin, Maneesh Agrawala, and Tony DeRose. Determining the benefits of direct-touch, bimanual, and multifinger input on a multitouch workstation. *Proceedings of Graphics Interface*, pages 119–124, 2009.

[KBBC06] M. Kaltenbrunner, T. Bovermann, R. Bencina, and E. Costanza. Tuio: A protocol for table-top tangible user interfaces. *Proceedings of the The 6th International Workshop on Gesture in Human-Computer Interaction and Simulation*, 2006.

[Lew95] J.R. Lewis. Ibm computer usability satisfaction questionnaires: Psychometric evaluation and instructions for use. *Int. J. Human-Computer Interaction*, pages 57–78, 1995.

[LPB+09] Jakob Leitner, James Powell, Peter Brandl, Thomas Seifried, Michael Haller, Bernahrd Dorray, and Paul To. Flux: a tilting multi-touch and pen based surface. *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, pages 3211–3216, April 2009.

[MH08] Tomer Moscovich and John F. Hughes. Indirect mappings of multi-touch input using one and two hands. *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1275–1284, 2008.

[Nie93] Jakob Nielsen. *Usability Engineering.* Academic Press, Morgan Kaufmann, San Francisco, California, 1993.

[PKS+08] Peter Peltonen, Esko Kurvinen, Antti Salovaara, Giulio Jacucci, Tommi Ilmonen, John Evans, Antti Oulasvirta, and Petri Saarikko. It's mine, don't touch!: interactions at a large multi-touch display in a city centre. *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1285–1294, 2008.

[RC08] Jeffrey Rubin and Dana Chisnell. *Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests.* Wiley Publishing, Inc.; Indianapolis, Indiana, 2. Auflage 2008.

[REE+05] Kathy Ryall, Alan Esenther, Katherine Everitt, Clifton Forlines, Meredith Ringel-Morris, Chia Shen, Sam Shipman, and Frédéric D. Vernier. idwidgets: Parameterizing widgets by user identity. *Human-Computer Interaction - INTERACT '05*, pages 1124–1128, 2005.

[RL03] Yvonne Rogers and Slian Lindley. Collaborating around vertical and horizontal large interactive displays: which way is best to meet? *http://www.informatics.sussex.ac.uk/research/groups/interact/papers/pdfs/pervasive_environments_and_ubiComp/Shared_interaction_spaces/Rogers.displays2003.pdf, zuletzt abgerufen am 19.01.2010*, 2003.

[RMHPW06] Meredith Ringel-Morris, Angi Huang, Andreas Paepcke, and Terry Winograd. Cooperative gestures: multi-user gestural interactions for co-located groupware. *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 1201–1210, April 2006.

[RMRS+04] Meredith Ringel-Morris, Kathy Ryall, Chia Shen, Clifton Forlines, and Frédéric D. Vernier. Release, relocate, reorient, resize: fluid techniques for document sharing on multi-user in-

teractive tables. *CHI '04: Proceedings*, pages 1441 – 1444, 2004.

[Rot08] Tim Roth. Dsi - diffused surface illumination. *http://iad.projects.zhdk.ch/multitouch/?p=90, zuletzt abgerufen am 14.08.2010*, 2008.

[RSI96] Jim Rudd, Ken Stern, and Scott Insensee. Low vs. high-fidelity prototyping debate. *Interactions*, 3(1):76–85, 1996.

[Saf09] Dan Saffer. *Designing Gestural Interfaces*. O'Reilly, 1. Auflage 2009.

[SBD+08] Johannes Schöning, Peter Brandl, Florian Daiber, Florian Echtler, Otmar Hilliges, Jonathan Hook, Markus Löchtefeld, Nima Motamedi, Laurence Muller, Patrick Olivier, Tim Roth, and Ulrich von Zadow. Multi-touch surfaces: A technical guide. Technical report, TUM-l0833, Technical Reports of the Technical University of Munich, Oktober 2008.

[SBG09] Dominik Schmidt, Florian Block, and Hans Gellersen. A comparison of direct and indirect multi-touch input for large surfaces. *INTERACT 2009*, pages 582–594, 2009.

[SCH05] Stacey D. Scott, M. Sheelagh T. Carpendale, and Stefan Habelski. Storage bins: Mobile storage for collaborative tabletop displays. *IEEE Computer Graphics and Applications*, 25(4):58 – 65, 2005.

[Sco05] Stacey D. Scott. Territoriality in collaborative tabletop workspaces. *University of Calgary*, page 301, 2005.

[SG09] Dominik Schmidt and Hans Gellersen. Show your hands: A vision-based approach to user identification for interactive surfaces. *Interactive Tabletops and Surfaces, Banff, Canad*, 2009.

[SRF+06] Chia Shen, Kathy Ryall, Clifton Forlines, Alan Esenther, Frédéric D. Vernier, Katherine Everitt, Mike Wu, Daniel Wigdor, Meredith Ringel-Morris, Mark Hancock, and Edward Tse. Informing the design of direct-touch tabletops. *IEEE Computer Graphics and Applications*, 26(5):33–46, Oktober 2006.

[SVFR04] Chia Shen, Frédéric D. Vernier, Clifton Forlines, and Meredith Ringel. Diamondspin: An extensible toolkit for around-the-table interaction. *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 167–172, 2004.

[Tid06] Jennifer Tidwell. *Designing Interfaces*. O'Reilly, 1. Auflage 2006.

[TR09] Philip Tuddenham and Peter Robinson. Territorial corrdination and workspace awareness in remote tabletop collaboration. *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, pages 2139 – 2148, 2009.

[TTP+06] Anthony Tang, Melanie Tory, Barry Po, Petra Neumann, and Sheelagh Carpendale. Collaborative coupling over tabletop displays. *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 1181 – 1190, 2006.

[VWW10] Craig Villamor, Dan Willis, and Luke Wroblewski. Touch gesture reference guide. *http://www.lukew.com/touch/TouchGestureGuide.pdf, zuletzt abgerufen am 22.04.2010*, 2010.

[WR09] Feng Wang and Xiangshi Ren. Empirical evaluation for finger input properties in multi-touch interaction. *Conference on Human Factors in Computing Systems*, pages 1063–1072, April 2009.

[Wri05] Matthew Wright. Open sound control: an enabling technology for musical networking. *Organised Sound 10*, pages 193–200, März 2005.

[WRL08] Feng Wang, Xiangshi Ren, and Zhen Liu. A robust blob recognition and tracking method in vision-based multi-touch technique. *International Symposium on Parallel and Distributed Processing with Applications*, pages 971–974, Oktober 2008.

[ZGK04] Wolfgang Zuser, Thomas Grechenig, and Monika Köhle. *Software Engineering mit UML und dem Unified Process*, volume 2., überarbeitete Auflage. Pearson Studium, 2004.

[Zöf08] Peter Zöfel. *Statistik für Psychologen*. Pearson Studium, 2008.

Abbildungsverzeichnis

Tabellenverzeichnis

Anhang

Testmaterial: Einführungsskript

Einführungsblatt

Herzlich Willkommen

Heute testen Sie eine Applikation für einen Multitouch-Tisch. Ein Multitouch-Tisch ist ein Tisch mit einer besonderen berührungs-empfindlichen Oberfläche für die Eingabe von Daten. Es werden dabei mehrere Berührungen, die beispielsweise mit den Fingern ausgeführt werden, erkannt. Aufgrund dieser Eigenschaft, können auch mehrere Personen gleichzeitig an diesem Tisch arbeiten.

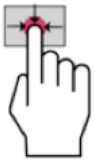
Den heutigen Test werden Sie in einer Gruppe absolvieren. Wir werden während des Tests Bildschirm-, Video- und Audio-Aufzeichnungen machen und möchten Sie darauf hinweisen, dass nicht Sie sondern die Applikation getestet wird.

Bei der zu testenden Applikation handelt es sich keineswegs um ein fertiges Produkt. Es kann daher vorkommen, dass es zu Problemen während der Benutzung kommt. Wir würden Sie daher bitten, während des Tests „laut" zu denken. Sprechen Sie also ihr Gedanken und Probleme laut aus. Dies wird uns helfen den Test auszuwerten.

Für Fragen stehen wir auch während des Tests jederzeit zur Verfügung!
Danke, dass Sie sich an dieser Studie beteiligen!

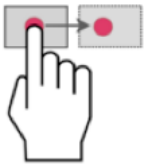
Verfügbare Gesten

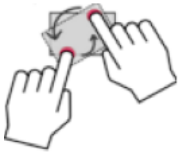
Durch die Erkennung von mehreren Berührungspunkten, können bei Multitouch spezielle Bedienmethoden, sogenannte Gesten, zum Einsatz kommen. Von der Applikation werden die folgenden Gesten unterstützt:



Tippen, um etwas zu öffnen / zu aktivieren
Beispiel für Einsatz: Ausblenden von Fotos

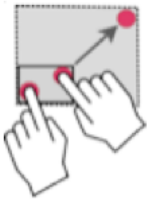
Ziehen, um ein Objekt zu bewegen
Beispiel für Einsatz: Fotos am Bildschirm bewegen

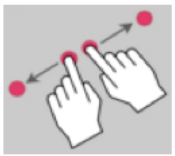




Zwei Finger drehen, um ein Objekt zu drehen
Beispiel für Einsatz: Fotos drehen

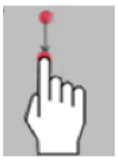
Zwei Finger zusammen/auseinanderziehen, um ein Objekt zu verkleinern/vergrößern
Beispiel für Einsatz: Fotos vergrößern/verkleinern





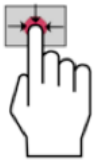
Zwei Finger am Hintergrund zusammen-/auseinanderziehen, um den Screen zu verkleinern/vergrößern
Beispiel für Einsatz: Zoom erhöhen/vermindern

Einen Finger am Hintergrund bewegen, um den Screen zu schwenken
Beispiel für Einsatz: Scrollen

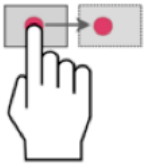


Verfügbare Gesten

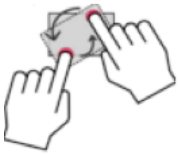
Durch die Erkennung von mehreren Berührungspunkten, können bei Multitouch spezielle Bedienmethoden, sogenannte Gesten, zum Einsatz kommen. Von der Applikation werden die folgenden Gesten unterstützt:



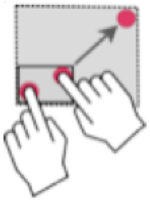
Tippen, um etwas zu öffnen / zu aktivieren
Beispiel für Einsatz: Ausblenden von Fotos



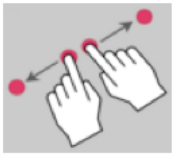
Ziehen, um ein Objekt zu bewegen
Beispiel für Einsatz: Fotos am Bildschirm bewegen



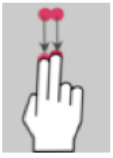
Zwei Finger drehen, um ein Objekt zu drehen
Beispiel für Einsatz: Fotos drehen



Zwei Finger zusammen/auseinanderziehen, um ein Objekt zu verkleinern/vergrößern
Beispiel für Einsatz: Fotos vergrößern/verkleinern



Zwei Finger am Hintergrund zusammen-/auseinanderziehen, um den Screen zu verkleinern/vergrößern
Beispiel für Einsatz: Zoom erhöhen/vermindern



Zwei Finger am Hintergrund bewegen, um den Screen zu schwenken
Beispiel für Einsatz: Scrollen

Testmaterial: Pre-Test Fragebogen

Pre-Test Fragebogen

Bitte füllen Sie folgende Angaben in Blockschrift aus.

Alter: ____

Beruf: _____

Ausbildung: _____
(höchste abgeschlossene)

Bitte kreuzen Sie Zutreffendes an und erweitern Sie Ihre Auswahl gegebenenfalls mit Kommentaren.

Ich stehe neuen Technologien aufgeschlossen gegenüber.

stimme nicht zu — stimme zu

1	2	3	4	5	6	7
☐	☐	☐	☐	☐	☐	☐

Kommentar:

Wie würden Sie Ihre Computerkenntnisse einschätzen?

☐ Grundkenntnisse ☐ Fortgeschritten ☐ Profi

Kommentar:

Haben Sie Erfahrungen mit der Benutzung von Touch-Screens (etwa durch Handys, Fahrkartenterminals, etc.)?

☐ Ja
☐ Nein

Kommentar:

Kennen Sie Multitouch-Tische?

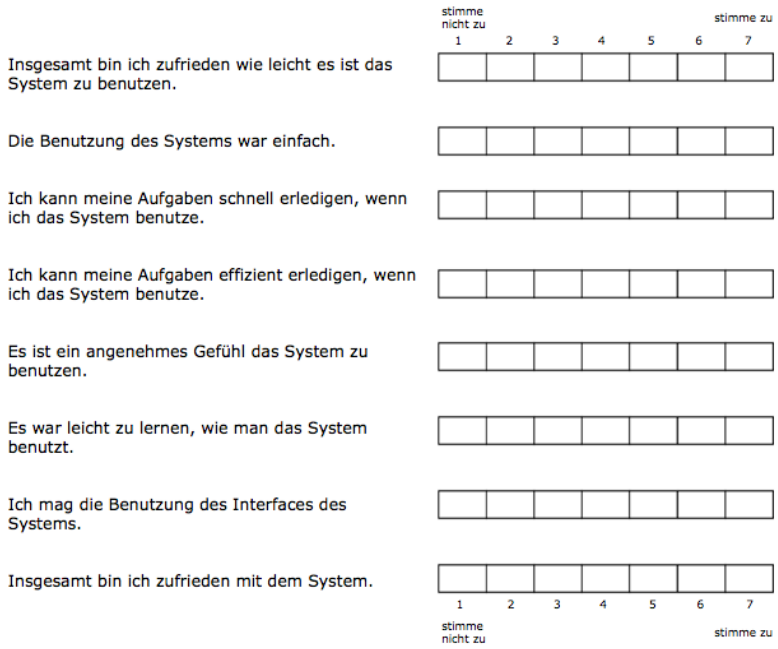
☐ Ja und ich habe einen solchen
  ☐ noch nicht benutzt
  ☐ bereits benutzt.
  Wenn ja, wann und welchen?
  _____
☐ Nein

Kommentar:

Testmaterial: Post-Test Fragebogen

Post-Test Fragebogen

Bitte kreuzen Sie Zutreffendes an.



Testmaterial: Interviewleitfaden

Interviewleitfaden

Was hat bei der gemeinsamen Arbeit gut funktioniert?

Was hat bei der gemeinsamen Arbeit nicht so gut funktioniert?

Welche Probleme hatten Sie bei der Benutzung der Applikation?

Denken Sie, dass einige dieser Probleme durch ander Benutzer verursacht wurden?

Wurden Sie durch die andere Personen während der Benutzung gestört?

Wenn ja, wie?

Finden Sie, dass sich die Applikation bei der Eingabe (zum Beispiel durch Gesten) so verhalten hat, wie Sie es erwartet haben?

Wenn nein, warum nicht?

Testmaterial: Probanden

Pilottest

Person	**Alter**	**Beruf**	**Erfahrung**[56]	**Technik-affinität**[57]
Testperson A	24	Student	Erfahrung mit Touch-Screens (Smart Phone, Fahrkartenterminals), kennt Multitouch-Tische	7
Testperson B	24	Studentin	Erfahrung mit Touch-Screens (Fahrkartenterminal)	4
Durchschnitt	24			5,5

Tests in Vierer-Gruppen

Person	**Alter**	**Beruf**	**Erfahrung**	**Technik-affinität**
Testperson 1	26	Dissertant	Erfahrung mit Touch-Screens (Fahrkartenterminals)	4
Testperson 3	24	Studentin	Erfahrung mit Multitouch (Fahrkartenterminals)	4
Testperson 5	25	Student	keine	6
Testperson 7	22	Studentin	Erfahrung mit Touch-Screens, kennt Multitouch-Tische	7
Durchschnitt	24,25			5,25

[56] Erfahrungen mit Touch-Screens (beispielsweise durch Smart Phones) und/oder mit Multitouch-Tischen.

[57] Technikaffinität auf einer Skala von 1 (sehr gering) bis 7 (sehr hoch).

Erste Iteration

Person	Alter	Beruf	Erfahrung	Technik-affinität
Erste Session - Version ohne Heuristiken				
Testperson 1	26	Dissertant	Erfahrung mit Touch-Screens (Fahrkartenterminals)	4
Testperson 2	26	Student	Erfahrung mit Touch-Screens (Smart Phone), Erfahrung mit Multitouch-Tische (MS Surface)	7
Zweite Session - Version mit Heuristiken				
Testperson 3	24	Studentin	Erfahrung mit Multitouch (Fahrkartenterminals)	4
Testperson 4	24	Konstrukteur, Student	Erfahrung mit Multitouch (Fahrkartenterminals, Laptops)	6
Dritte Session - Version ohne Heuristiken				
Testperson 5	25	Student	keine	6
Testperson 6	24	Dissertantin	Erfahrung mit Touch-Screens (Smart Phone)	6
Vierte Session - Version mit Heuristiken				
Testperson 7	22	Studentin	Erfahrung mit Touch-Screens, kennt Multitouch-Tische	7
Testperson 8	24	Student	Erfahrung mit Touch-Screens, kennt Multitouch-Tische	7
Durchschnitt	24,38			5,88

Zweite Iteration

Person	Alter	Beruf	Erfahrung	Technik-affinität
Erste Session - Version ohne Heuristiken				
Testperson 3	24	Studentin	Erfahrung mit Multitouch (Fahrkartenterminals)	4
Testperson 7	22	Studentin	Erfahrung mit Touch-Screens, kennt Multitouch-Tische	7
Zweite Session - Version mit Heuristiken				
Testperson 1	26	Dissertant	Erfahrung mit Touch-Screens (Fahrkartenterminals)	4
Testperson 6	24	Dissertantin	Erfahrung mit Touch-Screens (Smart Phone)	6
Dritte Session - Version ohne Heuristiken				
Testperson 4	24	Konstrukteur, Student	Erfahrung mit Multitouch (Fahrkartenterminals, Laptops)	6
Testperson 8	24	Student	Erfahrung mit Touch-Screens, kennt Multitouch-Tische	7
Vierte Session - Version mit Heuristiken				
Testperson 2	26	Student	Erfahrung mit Touch-Screens (Smart Phone), Erfahrung mit Multitouch-Tische (MS Surface)	7
Testperson 5	25	Student	keine	6
Durchschnitt	24,38			5,88

Testmaterial: Szenarien und Abarbeitung

Szenarien

Szenario 1	**Alter**
Aufgabe	Sortierung und Löschen von Fotos: Fotos mit Blumen sollen auf die linke Seite des Tisches, Fotos mit Tieren auf die rechte Seite des Tisches bewegt werden. Unpassende Fotos sollen entfernt werden.
Vorbedingung	Pool an Fotos (25 Stück) in der Mitte des Tisches, 10 mit Tieren, 10 mit Blumen, 3 andere Motive, 2 mit Tieren und Blumen.
Endzustand	Alle Fotos befinden sich auf der richtigen Seite des Tisches, drei unpassende Fotos wurden gelöscht, über das verbleibende Foto mit beiden Motiven wurde diskutiert beziehungsweise es wurde umsortiert oder gelöscht.
Benchmark	03:30 Minuten
Formulierung für die Testperson	Sortieren Sie gemeinsam alle Fotos, dass sich die Fotos mit Blumen auf der linken Seite und Fotos mit Tieren auf der rechten Seite des Tisches befinden. Fotos mit anderen Motiven sollen ausgeblendet werden.
Anmerkung	kollaboratives Szenario

[58] Der Wert ist abhängig von der Anzahl der Testpersonen pro Testsession (zwei oder vier).

Szenario 2	Alter
Aufgabe	Konkurrierendes Suchen und Finden eines Fotos.
Vorbedingung	Pool an Fotos (40 Stück) in der Mitte des Tisches, 1 mit Blume, Rest mit anderen Motiven.
Endzustand	Foto mit der Blume als Motiv wurde von einer Person gefunden und zu sich bewegt. Fotos können nicht entfernt werden.
Benchmark	01:15 Minuten
Formulierung für die Testperson	Finden Sie das Foto mit der Blume. Wer als Erster das Foto findet und zu sich bewegt gewinnt.
Anmerkung	konkurrierendes Szenario

Szenario 3	Alter
Aufgabe	Konkurrierendes Sortieren und Skalieren mehrerer Fotos: Jede Testperson bekommt ein Fotomotiv zugeteilt. Vier dieser Fotos müssen zu sich bewegt und entsprechend den Markierungen skaliert und sortiert werden.
Vorbedingung	Pool an Fotos (23/31 Stück[58]) in der Mitte des Tisches, 4 mit Tieren, 4 mit Blumen (, 4 mit Autos, 4 mit Uhren)[58], 15 mit anderen Motiven. Fotos können nicht entfernt werden.
Endzustand	4 passende Fotos wurde von einer Testperson gefunden, zu sich bewegt und entsprechend der Markierungen skaliert und sortiert.
Benchmark	01:30 Minuten
Formulierung für die Testperson	Finden Sie vier Fotos mit Tieren/Blumen (Autos/Uhren)[58]. Bewegen Sie diese Fotos zu sich und sortieren beziehungsweise skalieren Sie diese entsprechend der Markierungen in ihrem Arbeitsbereich. Wer als Erster alle Fotos richtig sortiert hat gewinnt.
Anmerkung	konkurrierendes Szenario

Szenario 4	**Alter**
Aufgabe	Gemeinsames Sortieren, Anordnen und Entfernen nach selbstständig entwickelten Kriterien von Fotos.
Vorbedingung	Pool an Fotos (24 Stück) in der Mitte des Tisches mit unterschiedlichen Motiven.
Endzustand	Fotos wurden von der Gruppe sortiert und Diskussionen über Sortierungen/Gruppierungen sind abgeschlossen.
Benchmark	03:00 Minuten
Formulierung für die Testperson	Sortieren und ordnen Sie die angezeigten Fotos nach Ihren eigenen Kriterien an. Sie können über diese Sortierung gemeinsam entscheiden und diskutieren. Unpassende Fotos können Sie auch ausblenden.
Anmerkung	kollaboratives Szenario

Szenario 5	**Alter**
Aufgabe	Laden von Fotos via Tastatur. Suchen der zum Suchbegriff gehörenden Fotos.
Vorbedingung	2 (4)[58] Tastaturen sind vorhanden. Fotos können nicht entfernt werden.
Endzustand	Eine Person hat die passenden Fotos und Nachrichten geladen und diese zu sich bewegt.
Benchmark	01:30 Minuten
Formulierung für die Testperson	Benutzen Sie Ihre Tastatur und suchen Sie nach folgendem Suchbegriff: „Blume“ („Tiere“/„Haus“/„Uhren“)[58]. Nach dem Laden der Daten suchen Sie die zum Suchbegriff gehörenden Fotos und Nachrichten und bewegen Sie diese zu sich. Wer als Erster alle entsprechenden Fotos und Nachrichten zu sich bewegt hat gewinnt.
Anmerkung	konkurrierendes Szenario

Latin Square - Abarbeitung der Szenarien

Die Szenarien werden in der folgenden Reihenfolge abgearbeitet. Die Zeilen entsprechen der Test-Session, die Spalten der Szenario-Reihenfolge. Die Szenario-Sequenz wurde dabei in Anlehnung an den von Chris Chatham vorgeschlagenen Algorithmus[59] zur Erstellung von Balancierten Latin Squares für experimentelles Design erstellt.

$$Sequenz = \begin{pmatrix} Szenario1 & Szenario2 & Szenario5 & Szenario3 & Szenario4 \\ Szenario2 & Szenario3 & Szenario1 & Szenario4 & Szenario5 \\ Szenario3 & Szenario4 & Szenario2 & Szenario5 & Szenario1 \\ Szenario4 & Szenario5 & Szenario3 & Szenario1 & Szenario2 \\ Szenario5 & Szenario1 & Szenario4 & Szenario2 & Szenario3 \end{pmatrix}$$

Testauswertung

Zeiten

Version 1 (min)	**1**	**2**	**3**	**4**	**M**	**SD**
Szenario 1	1,35	1,73	2,03	1,10	1,55	0,41
Szenario 2	0,83	1,12	0,98	1,53	1,12	0,30
Szenario 3	1,17	1,07	0,90	0,70	0,96	0,21
Szenario 4	2,67	6,55	1,83	1,88	3,23	2,24
Szenario 5	1,17	-	0,63	1,80	1,20	0,59

Version 2 (min)	**1**	**2**	**3**	**4**	**M**	**SD**
Szenario 1	1,67	0,92	0,93	1,20	1,18	0,35
Szenario 2	0,92	0,75	0,95	1,33	0,99	0,24
Szenario 3	0,90	0,93	0,72	0,93	0,87	0,10
Szenario 4	3,75	2,38	1,57	1,50	2,30	1,05
Szenario 5	2,12	0,95	0,95	1,11	1,28	0,56

[59] http://rintintin.colorado.edu/ chathach/balancedlatinsquares.html, zuletzt abgerufen am 14.07.2010.

Post-Test Fragebögen

Version 1	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**Mi**	**SD**	**Me**
Aussage 1	7	3	4	4	6	6	3	3	4,50	1,60	4,00
Aussage 2	7	4	6	5	7	6	7	3	5,63	1,51	6,00
Aussage 4	7	3	4	4	5	6	2	2	4,13	1,81	4,00
Aussage 5	5	2	6	4	4	6	4	3	4,25	1,39	4,00
Aussage 6	2	3	6	5	6	6	3	4	4,38	1,60	4,50
Aussage 7	7	6	7	7	7	7	7	6	6,75	0,46	7,00
Aussage 17	2	5	5	6	5	7	6	5	5,13	1,46	5,00
Aussage 19	7	5	5	5	6	6	5	4	5,38	0,92	5,00

Version 2	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**Mi**	**SD**	**Me**
Aussage 1	6	6	5	7	6	6	7	5	6,00	0,76	6,00
Aussage 2	5	6	6	7	6	6	7	7	6,25	0,71	6,00
Aussage 4	6	6	6	6	6	6	5	4	5,63	0,74	6,00
Aussage 5	5	4	6	5	4	5	4	5	4,75	0,71	5,00
Aussage 6	2	3	5	6	6	6	3	5	4,50	1,60	5,00
Aussage 7	7	6	7	7	7	7	7	7	6,88	0,35	7,00
Aussage 17	2	5	6	7	5	7	6	6	5,50	1,60	6,00
Aussage 19	6	5	7	6	6	6	6	5	5,88	0,64	6,00