Räumliche Authentifikation an großflächigen multi-touch Oberflächen

Klaus Drerup

Institut für Geoinformatik

Westfälische Wilhelms-Universität Münster

Zur Erlangung des Grades

Bachelor of Science

25. August 2009

Erstgutachter	Zweitgutachter
Prof. Dr. Antonio Krüger	Prof. Dr. Edzer Pebesma

Erklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Inhalte sind als solche kenntlich gemacht.

Münster, der 25. August 2009

Kurzfassung

Große multi-touch Benutzerschnittstellen ermöglichen die simultane Interaktion mehrerer Benutzer. Hierfür muss bei vielen multi-touch Anwendungen eine Eingabe einem bestimmten Benutzer zugewiesen werden. In dieser Arbeit wird eine Methode vorgestellt, mit der eine Person mit Hilfe eines üblichen Mobiltelefons für eine Subregion der multi-touch Oberfläche authentifiziert werden kann. Dies erfolgt über Farbsignale, die die Kamera des mobilen Gerätes erfasst. Zuerst wird durch Anzeige von Binärbäumen in Form von zweifarbigen Rechtecken die exakte Position der Kamera bestimmt, dann wird genau an dieser Position ein nur für die Kamera sichtbares Signal, bestehend aus einer Sequenz von unterschiedlichen Farben, übertragen. Wenn dieses korrekt aufgenommen wurde, ist der Benutzer an dieser Stelle authentifiziert.

Abstract

Several users can simultaniously interact at large multi-touch surfaces. Therefore many applications need to know, which touches on the multi-touch surface are related to which user. In this thesis a way to authenticate a person for a subregion of a multi-touch Interface is presented by using a common Smartphone. It is based on a color-signal captured by the mobile device. First you get the exact location of the camera using binarytrees having the shape of two-colored rectangles, next at that location the phone captures a sequence of different colors encoding a signal. If this signal has been received correctly, a user is authenticated at that location.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Glossar

Bluetooth Funkstandard, um über kurze Distanzen verschiedene Geräte zu verbinden und Daten auszutauschen.

DI Diffused Illumination: Bei dieser Umsetzung einer multi-touch Oberfläche wird diese von hinten mit Strahlung, welche meistens Infrarotlicht ist, bestrahlt. Wenn diese Oberfläche von Fingern berührt wird, wird Strahlung reflektiert und von einer dahinter positionierten Kamera erfasst. Bei dieser Technik können neben Fingern auch andere Objekte an der multi-touch Benutzerschnittstelle erfasst werden [24] .

DSA Digital Signature Algorithm, ein Standard für digitale Signaturen.

DSI Diffused Surface Illumination funktioniert ähnlich wie DI, man ist jedoch nicht auf externe Strahlungsquellen angewiesen. In dem Medium, welches als multi-touch Oberfläche dient, sind Partikel enthalten, welche sich wie kleine Spiegel verhalten. Somit wird eine gleichmäßige Ausleuchtung erreicht und eine Kamera kann die Reflektionen von Kontakten mit der multi-touch Benutzerschnittstelle erfassen [24].

EOC Emergency Operation Center. Krisenzentrum zur Bewältigung von Großschadensereignisse. Hier fließen alle Informationen zusammen und von hier aus werden die Einsätze geleitet.

FTIR Frustrated Total Internal Reflection: Elektro-magnetische Strahlen werden in einem Medium ausgestrahlt, welches bei multi-touch Benutzerschnittstellen meistens eine Plexiglasscheibe ist. Wenn diese Strahlung auf den Rand, wo das Medium in ein anderes Medium mit einer geringeren Brechzahl wie z.B. Luft übergeht, in einen geringen Winkel trifft, wird diese total reflektiert. Bei Berührung eines Fingers mit dieser Oberfläche ändert sich dort die Brechzahl und die Strahlung wird nach hinten ausgestrahlt. Für eine multi-touch Oberfläche gibt es hinter dieser eine Kamera, die die reflektierte Strahlung erfasst [24] .

GUI Graphical User Interface. Die grafische Benutzeroberfläche eines Programms, über die Eingaben und Ausgaben des Programmes erfolgen.

IP-Spoofing Beim IP-Spoofing wird in einem Netzwerk die IP-Adresse eines Paketes gefälscht. Somit kann sich ein Computer als ein anderer ausweisen.

Man-in-the-middle-Angriff Wenn bei der Kommunikation zwischen zwei Personen eine dritte Person die Nachrichten der beiden Personen ansieht und ändert, so nennt man das Man-in-the-middle-Angriff. Der jeweils andere Kommunikationspartner denkt dabei, die Nachrichten wären von der richtigen Person zugeschickt worden.

RGB Farbraum, der durch den Farbanteil von drei Komponenten definiert ist: Rot (R), Grün (G), Blau (B).

SoKNOS Service-orientierte Architekturen zur Unterstützung von Netzwerken im

Rahmen öffentlicher Sicherheit. Ein Projekt des Bundesministeriums für Forschung und Bildung, dessen Ziel es ist, Konzepte zu entwickeln und zu erforschen, die staatliche Organe, Unternehmen und andere Organisationen im Bereich öffentlicher Sicherheit wirksam unterstützen [9].

WLAN Wireless Local Area Network. Standard für Netzwerke, deren Geräte per Funk verbunden sind.

1

Einführung

In dieser Arbeit wird eine Umsetzung einer Benutzerauthentifikation an einer multi-touch Benutzerschnittstelle beschrieben. Bisher wurden zu diesem Thema noch keine Lösungen für multi-touch Oberflächen, die auf dem technischen Prinzip von FTIR basieren, erarbeitet. In diesem Kapitel wird auf die Motiviation, Aufbau der Arbeit und einige Grundlagen für das Thema eingegangen.

1.1 Motivation

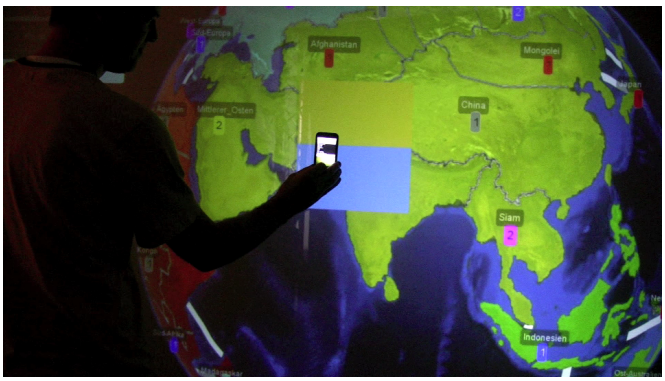

Abbildung 1.1: Authentifizierung mit mobilen Gerät. - Ein Mobiltelefon wird benutzt, um sich an einer multi-touch Benutzerschnittstelle zu authentifizieren [Eigene Abbildung].

In den vergangenen Jahren haben mehrere Entwicklungen zur Verbreitung von multi-touch Benutzerschnittstellen geführt. Zum einen sind diese nun einfacher und günstiger zu realisieren und werden deswegen in mehr Geräte integriert. Ein Beispiel

hierfür ist das iPhone, welches als erstes Mobiltelefon eine multi-touch Bedienoberfläche verwendet und wovon schon Stückzahlen von über zehn Millionen verkauft wurden [18]. Zum anderen ist es einfacher geworden, größere multi-touch Oberflächen zu erstellen. Insbesondere ermöglicht die Verwendung von optischen Verfahren wie FTIR, DI oder DSI die Konstruktion größerer und dennoch günstigen multi-touch Umsetzungen mit mehr als zwei Metern Diagonale [24].

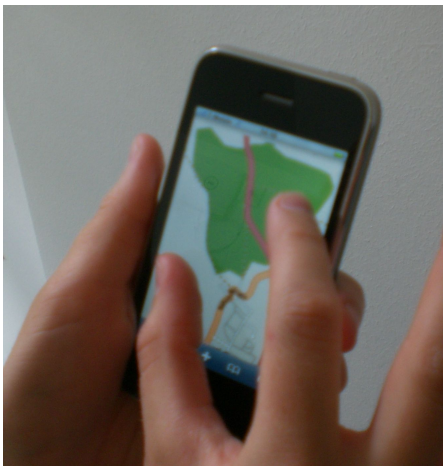

Abbildung 1.2: Ein Benutzer führt eine'pinch'-Geste auf dem iPhone aus. - Diese Geste kann zur Veränderung der Kartengröße genommen werden [Eigene Abbildung].

Die Mensch-Maschine Interaktion mit solchen Benutzerschnittstellen ist gegenüber der klassischen Bedienung von Computern mit Maus und Tastatur eine vollkommen andere. Vor allem hat man gegenüber der Maus durch die Nutzung beider Hände bildlich gesprochen „mehrere Mauszeiger zur Verfügung", wodurch Gesten möglich sind, die zur Eingabe weitaus mächtiger sein können als ein simpler Mauszeiger [1]. Hierdurch kann zum einen die Erledigung einer Aufgabe beschleunigt werden, zum anderen kann für den Benutzer die Interaktion mit dem Computer einfacher werden [17].

Für den Nutzer stellt sich die Eingabemethode als eine neue Erfahrung heraus. Dies ist ein Grund für die weite Verbreitung des iPhone. Die einfache und schnell zu erlernende Bedienung über dessen multi-touch Benutzerschnittstelle macht dieses Mobiltelefon attraktiver im Vergleich zu anderen Mobiltelefonen [27].

Bei multi-touch Oberflächen sind Ein- und Ausgabegerät in einem Gerät vereinigt, wodurch ohne Umwege über externe Eingabegeräte die direkte Manipulation von

Objekten auf Bildschirmen möglich ist und dies eine intuitivere Form der Bedienung darstellt [16].

Multi-touch Benutzerschnittstellen eigenen sich u.a. auch um mit Geodaten zu interagieren. Geodaten sind in der Informationstechnologie weit verbreitet und es werden immer mehr digitale Geo-Daten erfasst. Markant für diese sind geometrische, topologische, thematische und zeitliche Informationen. Zur Verarbeitung und Visualisierung werden meist Geoinfomationssysteme (GIS) verwendet. Von 1975 bis 1990 entstanden die ersten GIS-Implementierungen und seit den 90zigern werden diese weitläufig genutzt [3]. Es kommen dabei die unterschiedlichsten Anwendungsfälle zum Einsatz, z.B. Stadtplanung, Navigation oder Analyse von Umweltdaten.

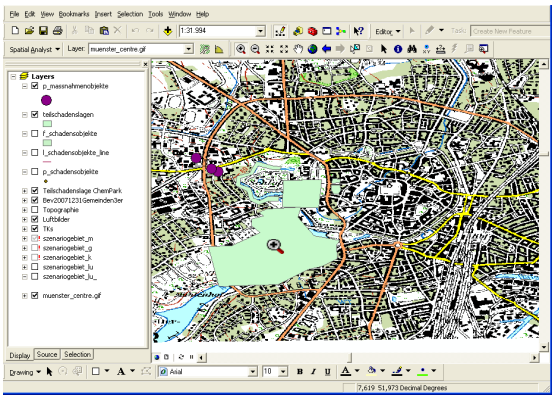

Abbildung 1.3: Ein GIS für die Mausnutzung. - [Screenshot aus ArcGis].

Die Bedienung eines GIS zur Betrachtung dieser Daten ist eine komplexe Aufgabe. Insbesondere wenn dies mit Hilfe von den Standarteingabegeräten Maus und Tastatur erfolgt. Für die Navigation der Szene in einem GIS ist es erforderlich, den Kartenausschnitt zu vergrößern bzw. zu verkleinern, verschieben oder komplett zu drehen. Dabei kann mit der Maus immer nur eine Aktion gleichzeitig ausgeführt werden, für eine andere ist ein Umschalten des Bedienungsmodus erforderlich. Wenn ein GIS über eine multi-touch Benutzerschnittstelle verfügt, so kann die Interaktion einfacher gestaltet werden. Man kann beide Hände zur Interaktion nutzen. Dies war bisher durch den kombinierten Einsatz von Maus und Tastatur auch möglich, allerdings wurde hier eher in einer diskreten und höchstens zwei stetigen Dimension interagiert, bei multi-touch Oberflächen hat man mehrere, stetige Dimensionen. Wenn diese in unterschiedliche Gesten interpretiert werden, können ad hoc unterschiedliche Veränderungen der darge-

stellten Szene erfolgen, ohne dass hierfür die Eingabegeräte anders konfiguriert werden müssen [12]. Die Darstellung eines virtuellen Globus profitiert insbesondere hiervon, da hier die Szene selber schon in 3D dargestellt wird und mit der multi-touch Benutzerschnittstelle genau wie mit einem realen Globus interagiert werden kann.

Durch die Verwendung von größeren multi-touch Oberflächen mit mehr als zwei Metern Diagonale ergeben sich für GIS neue Anwendungsfälle, insbesondere zur Kollaboration. Beim Einsatz von GIS wird dies heutzutage schon oft gefordert, in vielen Situationen sind Gruppen beteiligt statt nur eine Person. Dennoch wird oft ein GIS verwendet, was für den Einsatz am Einzelbenutzerarbeitsplatz gedacht ist [19]. Gegenüber dem Einsatz des klassischen PCs als Plattform können nun mehrere Menschen nebeneinander und gleichzeitig mit dem GIS interagieren. Dies erfordert ein Umdenken bei der Authentifizierung der Benutzer. Der klassische Ansatz, dass sich ein Nutzer mit Kennwort und Passwort an einem PC anmeldet und diesen für sich reserviert, ist unbrauchbar. Hierbei erfolgt die Registrierung auf Betriebssystemebene. Es sollen aber mehrere Benutzer gleichzeitig an der selben Anwendung arbeiten können. Somit ist es notwendig, deren GUI in einzelne Subregionen für die ausschließliche Nutzung zu unterteilen, um eine simultane Kollaboration zu gewährleisten. In dieser Arbeit wird eine Methode entwickelt wie ein Benutzer mit Hilfe eines Mobiltelefons eindeutig an einer multi-touch Oberfläche authentifizieren kann. Um die genauen Anforderungen für diesen Vorgang besser zu illustrieren, werden im folgenden zwei Nutzungs-Szenarien vorgestellt:

1. Krisenzentrum

   In einem Krisenzentrum wird die genaue Lage eines Notfall-Szenarios beobachtet und analysiert, des weiteren erfolgt hier die Koordination von Einsatzkräften. Viele Personen sind hier involviert und benötigen Zugang zu aktuellen Daten, auf Grundlage dessen sie schnell Entscheidungen treffen und ausführen müssen. Dabei übernimmt eine große multi-touch Oberfläche die zentrale Funktion hierfür. Es muss gewährleistet sein, dass die Personen nur die für sie autorisierten Befehle ausführen dürfen.

2. Spiel mit mehreren Personen

   Ein Strategiespiel für mehrere Personen wird auf einem 3D Globus auf einer multi-touch Benutzerschnittstelle ausgeführt. Die Interaktion erfolgt rundenba-

siert. Wichtig ist, dass nur der Spieler Aktionen ausführen kann, der auch gerade am Zuge ist.

Die hier betrachteten Szenarien und aufgestellten Anforderungen benötigen einen neuen Ansatz bei der Authentifizierung. Dabei soll berücksichtigt werden, dass die multi-touch Oberfläche mit Hilfe von FTIR umgesetzt werden soll, um diese kostengünstig realisieren zu können. Allerdings sind bei diesem System nur Informationen über die Lage und die Anzahl der Berührungen bekannt, aber nicht, von wem diese erzeugt wurden. Es ist eine zusätzliche Komponente erforderlich, die eine sichere Authentifizierung gewährleistet und gleichzeitig die Gesamtkosten des Systems nicht unverhältnismäßig in die Höhe treibt.

1.2 Weitere Gliederung

Diese Bachelorarbeit gliedert sich des Weiteren wie folgt: Als nächstes werden in diesem Kapitel die wichtigsten Grundlagen erläutert. Danach werden in Kapitel 2 verwandte Arbeiten beschrieben. In Kapitel 3 werden zwei Methoden vorgestellt, mit denen eine räumliche Authentifizierung durchgeführt werden könnte. Dann wird in Kapitel 4 die konkrete Implementierung des Systems vorgestellt. Zum Schluss erfolgt in Kapitel 5 eine abschließende Diskussion über die erzielten Resultate und ein Ausblick auf noch nicht umgesetzte Bereiche.

1.3 Grundlagen

Im Folgenden werden einige, für diese Arbeit grundlegende Themen erklärt.

1.3.1 Multi-touch Interaktion

Multi-touch ist eine besondere Art der Mensch-Maschine Interaktion gegenüber der üblichen Bedienung mit Maus und Tastatur. Ein- und Ausgabe sind in einem Gerät vereinigt, welches registriert, wo die Anzeige berührt wird. Daher ergibt sich ein besonderes „Look and Feel“, man weiß direkt, mit welchen Elementen der virtuellen Welt man interagiert. Dadurch ist eine direkte Manipulation von virtuellen Objekten möglich. So könnte man die Größe, Lage oder Position verändern.

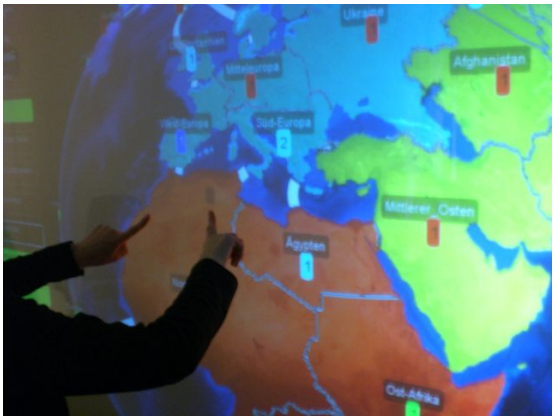

Abbildung 1.4: Multi-touch Interaktion mit Globus. - Risikospiel an multi-touch Wand [Eigene Abbildung].

Multi-touch Interaktion kennzeichnet sich dadurch aus, dass mehrere Berührungen gleichzeitig auf dem selben Eingabegerät erfasst werden. Theoretisch könnte es sich hierbei sogar um eine Tastatur handeln, allerdings hat man durch die Kombination von zwei Tasten zwei Dimensionen zur Verfügung. Gemeint ist bei multi-touch, dass man pro Berührung mit der Benutzerschnittstelle zwei Freiheitsgrade hat, d.h. zwei neue kontinuierliche Dimensionen, was genau den Möglichkeiten eines einzelnen Mauszeigers entsprechen würde und man dann bei mehreren Berührungen quasi mehrere Mauszeiger zur Verfügung hätte.

Zu Berücksichtigen hierbei ist die physikalische Realisierung der multi-touch Benutzerschnittstelle bevor man sich Gedanken macht über die eigentliche Interaktion. So kann dies ein Mobiltelefon sein, und man versteht unter multi-touch die Bewegung von nicht mehr als zwei Fingern auf einem relativen kleinen berührungs - empfindlichen Bildschirm mit nicht mehr als zehn cm Diagonale. Andererseits könnte es sich auch um eine vier Meter breite multi-touch Wand handeln, vor der dann mehrere Personen interagieren und man nicht unterscheiden kann, welche Berührung von welcher Person ausgeht. Es gibt also sehr unterschiedliche Anwendungsfälle.

Auf jeden Fall werden aus den Berührungen Gesten gebildet, die den Vorteil von multi-touch gegenüber klassischer Maus/Tastatur Bedienung mit z.B. einer Maus ausmachen und diese Interaktion so mächtig machen. Eine einfache Geste wäre hierbei z.B. das „Einklemmen“, wo durch die Veränderung des Abstandes von zwei Berührungen die Größe von virtuellen Objekten verändert wird. Es sind noch viele weitere komplexere

Gesten möglich, ein wichtiger Faktor hierbei ist die oben schon erwähnte Größe der multi-touch Oberfläche, die mitbestimmt, ob nur zwei Finger gleichzeitig interagieren können oder auch mehrere Hände.

Eine multi-touch Benutzerschnittstelle zur Eingabe ist nicht in allen Situationen von Vorteil. So kann z.B. eine Tastatur simuliert werden, in dem eine Tastatur auf der Anzeige dargestellt wird, mit der man interagieren kann. Aber das Tippen hiermit ist nicht so effizient als würde man eine reale Tastatur benutzen. Daher ist es nicht immer von Vorteil, nur eine multi-touch Benutzerschnittstelle zu benutzen. [4].

1.3.2 Fingerabdruckerkennung

Fingerabdrücke sind eine Form von biometrischen Merkmalen von Menschen. Genauer gesagt ist ein Fingerabdruck die Darstellung der Epidermis eines Fingers. Dieser besteht aus einem bestimmten Muster und ist für jeden Menschen einzigartig, bis jetzt hatten noch nie zwei Menschen den gleichen Fingerabdruck, selbst bei eineiigen Zwillingen trifft dieses nicht zu. Ein Fingerabdruck lässt sich heutzutage problemlos digital verarbeiten, hierzu werden Automatic Fingerprint Identification Systems (AFIS) genutzt. Der hierfür erforderliche Prozess wird in drei Schritte unterteilt: (1) Die Aufnahme mit Sensoren, (2) die Extraktion der Merkmale eines Fingerabdruckes und (3) der Vergleich mit anderen Aufnahmen auf Gleichheit [14].

1.3.3 Verschlüsselung durch asymmetrische Kryptographie

Die Kryptographie beschäftigt sich mit der Verschlüsselung von Nachrichten, so dass zwei Personen A und B geheime Informationen austauschen können, ohne dass Dritte deren Inhalt lesen können. Die Nachrichten werden dabei mit Hilfe von Schlüsseln von Klartext in Geheimtext umgewandelt, der für Dritte unlesbar ist.

Die asymmetrische Kryptographie kennzeichnet sich gegenüber der symmetrischen dadurch aus, dass zwei Schlüssel verwendet werden und einer davon, der private, nicht ausgetauscht werden muss. Bei der symmetrischen Kryptographie wird ein gemeinsamer Schlüssel zur Ver- und Entschlüsselung verwendet, dieser darf nur den beiden Kommunikationspartnern bekannt sein und muss geheim gehalten werden.

Es gibt bei der asymmetrischen Kryptographie einen privaten und einen öffentlichen Schlüssel. Wenn eine Person A sich verschlüsselte Nachrichten zu kommen lassen möchte,

kann jeder den öffentlichen Schlüssel von A verwenden, um eine Nachricht zu verschlüsseln. Diese kann dann jedoch nur mit den privaten Schlüssel von A wieder in Klartext entschlüsselt werden.

Dieses Verfahren kann umgekehrt verwendet werden, um Nachrichten digital zu signieren, d.h. zu unterschreiben. Damit eine Person B verifizieren kann, dass eine Nachricht von A stammt, wird mit dem privaten Schlüssel von A zu einer Klartextnachricht M eine Signatur erstellt. Diese wird zusammen mit M an B übertragen und B kann mit den öffentlichen Schlüssel diese Signatur wieder entschlüsseln. Wenn die Nachricht wirklich von A stammt, müsste aus der Signatur die gleiche Klartextnachricht M erzeugt werden, die A mitgeschickt hat [2].

2

Verwandte Arbeiten

Die Einführung von FTIR für die Umsetzung von multi-touch Oberflächen stellt eine kostengünstige Realisierung dieses Systems dar [11]. Hierfür werden lediglich bereits ausgereifte und schon lange am Markt eingeführte Komponenten benötigt. Durch die einfache Skalierbarkeit dieser Technik können auch größere multi-touch Oberflächen erstellt werden, auf denen mehrere Personen gleichzeitig interagieren können.

Eine mögliche Anwendungsdomäne ist der kollaborative Einsatz von GIS, z.B. als Hilfsmittel zur Entscheidungsfindung oder zur gemeinsamen Planung. Mit dem GIS Wallboard [7] wird ein Konzept zur multi-modalen Steuerung eines GIS an großen Bildschirmen vorgestellt. Die Interaktion erfolgt hierbei mit Hilfe von einer oder beiden Händen, Körpergesten, Sprache oder zusätzlichen Eingabegeräten. Zudem wird bei der Interaktion in der Distanz zum GIS Wallboard unterschieden, dadurch ergeben sich für den Benutzer unterschiedliche Interaktionsräume.

Maceachren et al stellen ein Framework zur Kollaboration mit räumlichen Daten vor [19], insbesondere werden hier verschiedene Parameter und ihre Beziehungen zueinander der sogenannten „geocollaborativen“ Umgebungen systematisch aufgezeigt. Hierzu wurde u.a. ein Prototyp zur kollaborativen Visualisierung zur Interaktion gleiche Zeit/ gleicher Ort erstellt und auf die definierten Parameter untersucht. Die Gruppe teilt dabei eine gemeinsame Sicht und es kann eine Person aktiv mit der virtuellen Umgebung interagieren. Die Prototypen werden in verschiedenen, grundlegenden Anwendungsszenarien untersucht, so z.B. in der Wissensexploration, allgemein ist bei allen nur eine gemeinsame Sicht möglich. Ein konkreter Anwendungsfall sind EOCs. Fuhrmann et al. [10] stellen hierzu ein durch Sprache und Gesten gesteuertes GIS vor, welches als

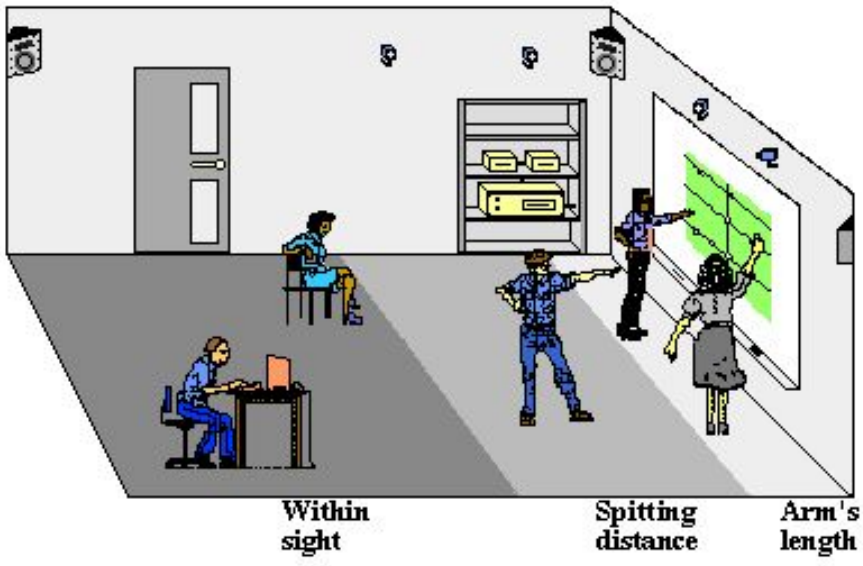

Abbildung 2.1: Interaktionsräume des GIS Wallboard. - Gefunden unter http://www.spatial.maine.edu.

Benutzerstudie für ein EOC eingesetzt wird. Benutzer, die nicht im Umgang mit GIS geschult sind, fällt die Bedienung durch das multi-modale Konzept leichter, welches für die grundlegenden Interaktionen intuitiver und daher schneller zu verstehen ist.

Eine weitere Anwendung von großen multi-touch Oberflächen sind Spiele für mehrere Personen. Zu diesem Bereich gibt es bisher wenige Arbeiten. Ein grundlegendes Framework auf Basis von Flash wird in [6] vorgestellt, für das einfache Spiele implementiert wurden.

Bei fast allen bisherigen Anwendungen für multi-touch Oberflächen haben allerdings alle Benutzer dieselben Zugriffsrechte. Gerade bei EOCs ist es entscheidend, wer welche Eingaben gemacht hat. So sollte bei der Nutzung von größeren, berührungsempfindlichen Oberflächen auch eine Authentifizierung der Benutzer oder zumindest sollten kritische Interaktionen erst nach erfolgreicher Authentifizierung möglich sein. Bei Diamond Touch [5] ist es möglich, auf größeren multi-touch Benutzerschnittstellen mehrere Eingaben direkt verschiedenen Benutzern zuzuordnen. Dies wird durch einen kapazitiven Bildschirm und der Kopplung eines jeden Benutzers hiermit realisiert. Diese Touch-Oberfläche ist als Tisch konzipiert, an dem jeder Benutzer sitzt und es dadurch zu einem Ladungsfluß kommt. Für den Einsatz in EOCs ist diese Konstellation jedoch ungünstig, da sich die Benutzer zwischendurch auch an anderen Orten aufhalten und

bei Bedarf schnellen Zugriff zur multi-touch Oberfläche benötigen und somit eine dauerhafte kapazitive Kopplung hiermit schlecht realisierbar ist.

Vogel und Balakrishnan [29] stellen ein Interaktionsframework für große, öffentliche Anzeigen vor. Dabei gibt es verschiedene Interaktionsräume, in denen dem Benutzer unterschiedliche Eingabemöglichkeiten zur Verfügung stehen. In Abhängigkeit von der Entfernung werden dem Benutzer verschiedene Inhalte angezeigt, je kürzer die Entfernung, desto persönlicher werden die dargestellten Informationen. Außerdem ist es möglich, dass mehrere Personen gleichzeitig interagieren.

Schöning et al. [26] stellen ein Konzept vor, dass die Nutzung von Mobiltelefonen für die Authentifizierung vorsieht. Wenn ein Nutzer mit seinem Mobiltelefon in Berührung mit der multi-touch Oberfläche kommt, wird ein Prozess zur Authentifizierung ausgelöst, der den Benutzer und seine Position identifiziert. Dies kann über das Blitzlicht geschehen, dadurch kann eine Kamera, die bei FTIR schon vorhanden wäre, die genaue Position vor der multi-touch Oberfläche bestimmen. Anschließend kann eine Subregion für die Interaktionen dieser Person freigeschaltet werden.

Für die Positionsbestimmung steht mit dem Blitzlicht ein schmaler Kommunikationskanal zur Verfügung, über dem sich wenige Informationen übertragen lassen. Weiterhin muss sichergestellt werden, dass kein anderer Benutzer mit seinem Blitzlicht das System täuschen kann. Eine andere Möglichkeit ist die Übertragung von Informationen über Farben. Der Vorteil ist, dass lediglich eine Kamera benötigt wird, die von dem sowieso schon vorhandenen Bildschirm des Mobiltelefons Informationen erfasst. Bei C-blink [21] können große öffentliche Anzeigen, an denen eine Kamera angebracht ist, Informationen von Mobiltelefonen empfangen. Auf dem Bildschirm dieser werden die Informationen durch verschiedene Farbtöne dargestellt, die dann abgefilmt werden. Die Zeichen der zu übertragenden Nachrichten werden dabei nicht durch feste Farbwerte kodiert, viel mehr wird das nächste Zeichen durch den Unterschied im Farbton bestimmt. Es können mit n verschiedenen Farbtönen $n\text{-}1$ verschiedene Zeichen kodiert werden. Dies gewährleistet eine bessere Signalerkennung der Kamera, es ist einfacher, zweimal das gleiche Signal hintereinander zu senden.

BlueTable [31] ist ein weiteres System, bei dem Mobiltelefone mit einer größeren Anzeige interagieren. Es gibt eine Projektionsfläche, auf der das Mobiltelefon gelegt wird und dann durch Kombination von Bilderkennung und Bluetooth authentifiziert wird. Die Kamera erkennt das Gerät selber und hat somit seine genaue Position, durch

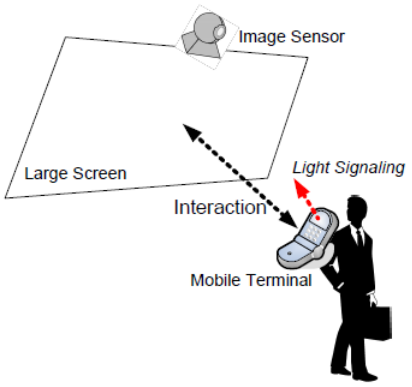

Abbildung 2.2: C-blink Konzept zur Übertragung von Informationen. - [21].

die es eindeutig identifiziert werden kann. Nun kann mit diesem interagiert werden, z.B. können aufgenommene Fotos direkt daneben auf der Projektionsfläche angezeigt werden.

Räumliche Beziehungen verschiedener Geräte zueinander können auch zur Übertragung geheimer Informationen genutzt werden. In [20] wird ein Protokoll, basierend auf ein Ultraschall System, genutzt, damit zwei Geräte sicher miteinander kommunizieren können. Durch die Ultraschall-Technologie kann die Position des jeweils anderen Gerätes verifiziert werden, somit ist eine physikalische Verbindung beider Geräte gegeben.

Weitere Ansätze zur Authentifizierung an multi-touch Oberflächen werden in [25] aufgeführt, so könnte dieser Vorgang aufgrund von biometrischen Merkmalen erfolgen. Dies wäre z.B. der Fingerabdruck des Benutzers. Hierfür wird hinter der multi-touch Oberfläche eine zusätzliche Kamera positioniert, die dann von jeder Berührung mit der Oberfläche einen Fingerabdruck aufnimmt und versucht, diesen zu verifizieren. Für diesen Vorgang müsste die Anzeige durchsichtig geschaltet werden, um die feinen Konturen eines Fingerabdruckes detailliert erfassen zu können, was bei FTIR ad hoc nicht möglich ist. Eine Lösung wäre Torchlight [30]. Das Bild wird von schräg unten auf eine stehende multi-touch Oberfläche projiziert, diese hat eine besondere Beschichtung, damit nur Bilder aus diesen Winkel angezeigt werden und somit für den davor stehenden Benutzer sichtbar wird. Für Lichtstrahlen, die diese Anzeige in einen anderen Winkel passieren, bleibt diese durchsichtig, so wie z.B. für eine Kamera, die Fingerabdrücke erfasst. Bei SecondLight [13] kann der multi-touch Oberfläche sogar abwechselnd durchsichtig und diffus geschaltet werden, so dass fast gleichzeitig hinter der Projektionsscheibe ein Bild

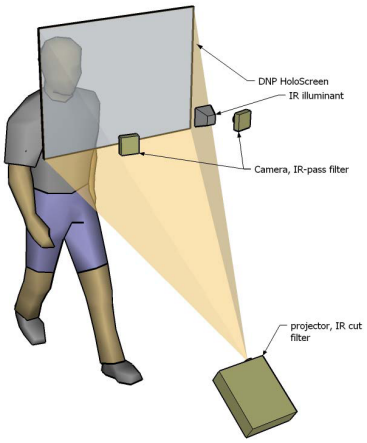

Abbildung 2.3: Touchlight zur Realisierung einer multi-touch Benutzerschnittstelle. - [30].

dargestellt wird und man Aufnahmen von Fingerabdrücken machen kann.

Der in dieser Arbeit vorgestellte Ansatz unterscheidet sich im Wesentlichen von den in [7], [19], [10] und [29] vorgestellten Ansätzen dadurch, dass mehrere Personen gleichzeitig an einer multi-touch Oberfläche interagieren können. Hierbei ist es wichtig, dass zwischen den verschiedenen Benutzern unterschieden werden kann, damit eine Anwendung jederzeit Eingaben einem Benutzer zuordnen kann und hierauf basierend Zugriffsrechte gewährt. Hierzu wird erstmalig eine Authentifizierung direkt an einer multi-touch Oberfläche durchgeführt, ohne aufwändige, technische Verbindung mit dieser oder einen PC mit der klassischen Eingabemethode verwenden zu müssen. Des Weiteren kann die multi-touch Benutzerschnittstelle durch FTIR realisiert werden.

3

Konzept einer räumlichen Authentifizierung

In diesem Kapitel wird die konzeptionelle Umsetzung einer Benutzerauthentifikation an einer multi-touch Benutzerschnittstelle näher erläutert.

3.1 Grundidee

Große multi-touch Oberflächen, die heutzutage in einer Größe von über zwei Meter Breite zu erschwinglichen Preisen realisierbar sind, ermöglichen kollaboratives Arbeiten. Die Akteure können dabei in unterschiedlichen Rollen mit unterschiedlichen Zugriffsrechten auftreten. Bisher wurden diese unterschiedlichen Rechte an einem Computer dadurch gewährleistet, dass man sich mit Kennwort und Passwort dort angemeldet hat und den ganzen Computer für sich reserviert hat. Wenn nun mehrere Personen zusammenarbeiten müssen, und dafür einige Personen Zugang zu einem Computer benötigen, aber auch eine gemeinsame Sicht auf bestimmte Daten, die für die Gruppenarbeit wichtig sind, so sind hierfür mehrere Computer erforderlich, was aber wiederum ein Hindernis für die Kollaboration und einer gemeinsamen Sicht ist. Des Weiteren ist eine direkte Kommunikation nicht ohne weiteres möglich.

An einer großen multi-touch Oberfläche haben alle Beteiligten eine gemeinsame Perspektive, es ist eine natürliche Kommunikation möglich, und durch die multi-touch Interaktionen können auch mehrere Personen gleichzeitig Eingaben machen. Allerdings gibt es bei letzteren keine Unterscheidung, welche Eingaben von wem kommen. Da

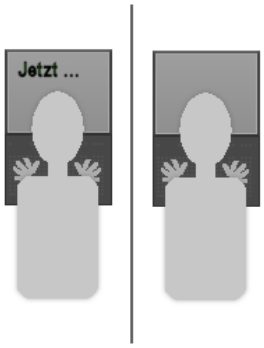

Abbildung 3.1: Bisherige Kollaboration an mehreren Einzelarbeitsplätzen, wenn mehrere Personen gleichzeitig interagieren müssen. - [Eigene Abbildung].

die multi-touch Anzeige von einem üblichen Computer betrieben wird, der über ein zusätzliches Eingabegerät, welches die multi-touch Interaktionen registriert, verfügt, gibt es bisher auch nur den Ansatz, dass sich immer nur ein Benutzer anmelden kann.

Deswegen ist ein neuer Ansatz erforderlich. Die bisherige Anmeldung auf Betriebssystemebene kann nicht ohne weiteres so verändert werden, dass sich mehrere Benutzer an einem PC gleichzeitig anmelden können. Die gleichzeitige Anmeldung ist für viele multi-touch Anwendungen auch gar nicht erforderlich, wie später in diesem Kapitel gezeigt wird. Solche Anwendungen erfordern keine Betriebssystem-kritischen Operationen wie das Installieren neuer Programme bzw. die Administration des PCs. Es reicht, wenn sich die Anwendung selber um eine Benutzerunterscheidung kümmert. Die Authentifizierung erfolgt somit auf Anwendungsebene. Bei der Anmeldung kann man flexibel bleiben, es ist nicht erforderlich, beim Starten der Anwendung festzulegen, wer hiermit arbeiten wird. Ein Benutzer kann bei laufender Anwendung einfach zur multi-touch Anzeige hingehen und sich spontan anmelden. Bei machen Anwendungen ist zusätzlich eine vorherige Registrierung des Benutzers erforderlich.

Um ein reibungsloses Arbeiten mit mehreren Benutzern zu ermöglichen, wird die Anwendungsfläche in mehrere Subregionen unterteilt. Dies geschieht bei Bedarf, ein Benutzer meldet sich an und um ihm herum wird in Interaktionsreichweite eine Fläche für ihn reserviert. Diese entspricht der Reichweite seiner Arme, womit er die multi-touch Eingaben macht. Konzeptionell entspricht dies der Projektion von mehreren Desktops auf einer Anzeige, wo für jeden Desktop gesonderte Zugriffsrechte gelten. Diese Desktops sind nicht festgelegt, sondern werden bei Bedarf geändert. Nun kann der Benutzer mit

dem ihm gewährten Rechten in dieser Subregion arbeiten. Wenn er versucht, außerhalb dieser zu interagieren, werden nur die möglichen Aktionen eines nicht angemeldeten Benutzers gewährt bzw. diese Eingabe wird als ungültig gewertet, je nachdem, welche Interaktionen ein nicht angemeldeter Benutzer machen darf.

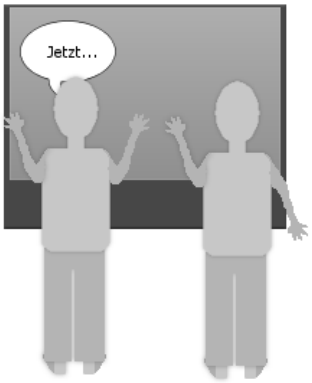

Abbildung 3.2: Simultane Kollaboration an einer multi-touch Benutzerschnittstelle. - [Eigene Abbildung].

Die Authentifizierung eines Benutzers soll nur einmal bei Anmeldung an der multi-touch Benutzerschnittstelle erfolgen, eine ständige Authentifizierung wäre ein großes Hindernis für die eigentliche Interaktion. Dies hängt wiederum auch von der Methode ab, auf jeden Fall würde dies zusätzlichen Aufwand für den Benutzer bedeuten, was auch gerechtfertigt ist angesichts der Freigabe kritischer Aktionen. Beim Desktop Computer ist man bisher ebenso verfahren, nach einmaliger Anmeldung ist der Computer freigeschaltet und die davor sitzende Person hat alle Zugriffsrechte. Es wird nicht laufend überprüft, ob die Person, die gerade Eingaben macht, auch diejenige ist, die sich angemeldet hatte. Man geht davon aus, dass der Anmelder dafür sorgt, dass kein anderer während seiner Sitzung Eingaben macht.

Wenn an der multi-touch Oberfläche sich jemand angemeldet hat, ist eine für diesen Benutzer freigeschaltete Subregion nicht auf unbestimmten Zeitraum gültig. Dies ist notwendig, um bei mehreren Nutzern zu gewährleisten, dass die Interaktionen in einer Subregion auch nur von dem angemeldeten Benutzer ausgelöst wurden. Hierfür hat der Benutzer zum einen die Möglichkeit, sich selber abzumelden. Zum einen geschieht dies mit der gleichen Methode wie auch für die Authentifizierung erforderlich ist oder durch eine besondere multi-touch Geste. Zum anderen geschieht dies automatisch,

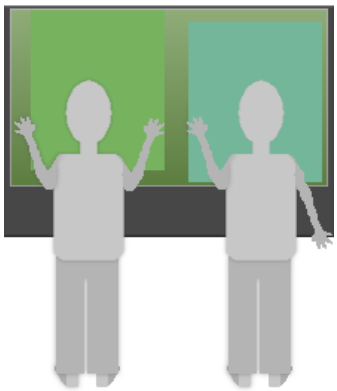

Abbildung 3.3: Unterteilung der multi-touch Oberfläche in Subregionen. - [Eigene Abbildung].

nach einem bestimmten Zeitraum nach der letzten Interaktion mit dieser Subregion werden die hier zusätzlich gewährten Zugriffsrechte wieder aufgehoben. Diese Funktion soll gewährleisten, dass wenn ein Benutzer die multi-touch Benutzerschnittstelle verlässt und vergisst sich abzumelden, seine Sitzung beendet wird und damit auch alle zusätzlichen Benutzerrechte nicht mehr gültig sind.

Des Weiteren könnte in der Anwendung ein besonderer Modus dafür sorgen, dass immer nur eine Person interagieren darf, also eine Unterteilung in Subregionen nicht unbedingt erforderlich ist. Wenn er nun seine Sitzung beendet hat, wird automatisch der Zugang für den nächsten Benutzer freigegeben, der sich authentifizieren muss. Die Kollaboration erfolgt hier nach dem Schema, dass eine Person interagiert und die anderen Personen dies über eine gemeinsame Sicht verfolgen.

Insgesamt soll durch diesen zuvor beschriebenen Authentifizierungsmechanismus eine reibungslose Zusammenarbeit mehrerer Personen an einer multi-touch Oberfläche realisiert werden. Im folgenden werden zwei Techniken vorgestellt, mit denen ein Benutzer authentifiziert werden kann.

3.2 Techniken

In diesem Abschnitt werden methodisch die beiden Techniken vorgestellt, mit denen die Authentifizierung an einer multi-touch Benutzerschnittstelle durchgeführt werden kann.

3.2.1 Fingerabdruckerkennung

Fingerabdrücke stellen ein weitverbreitetes und bewährtes biometrisches Merkmal dar, mit dem Menschen eindeutig identifiziert werden können. Die Authentifizierung durch diese ist ein etabliertes Verfahren in sicherheitskritischen Umgebungen und wird auch bei PCs angewandt, hier insbesondere bei Notebooks. Es bietet sich aber auch bei multi-touch Oberflächen an. Durch den Umstand, dass die Interaktion mit den Fingern erfolgt, ist der Benutzer ständig mit der Oberfläche der Benutzerschnittstelle in Kontakt. Somit kann die Erfassung von Fingerabdrücken direkt hiervon erfolgen, für den Benutzer ist kein zusätzlicher Aufwand erforderlich. Außerdem hat man damit direkt die genaue Position des Benutzers.

Wenn multi-touch Oberflächen durch optische Ansätze wie FTIR oder DI realisiert werden, ist eine Kamera zur Erfassung der Berührungen des Benutzers mit der Oberfläche erforderlich, die hinter dieser angeordnet ist. Für die Aufnahme von Fingerabdrücken wird eine zweite Kamera daneben positioniert. Diese verfügt über ein anderes Objektiv mit starker Vergrößerung, welches nicht die ganze Anzeige erfasst, sondern nur einen kleinen Ausschnitt, gerade groß genug, um ein Bild von einem Finger machen zu können. Wie schon bei der Kamera zur Erfassung der Berührungen mit der multi-touch Oberfläche reicht es auch bei dieser, wenn die vom Finger bei Berührung mit der Benutzerschnittstelle reflektierte Infrarotstrahlung erfasst wird.

Die für die Erfassung der Benutzereingaben an der multi-touch Oberfläche zuständige Kamera gibt der Fingerabdruck-Kamera die Position eines Fingers, diese muss anschließend hierhin bewegt werden. Entweder geschieht dies durch Schwenken und anschließendes Scharfstellen der Kamera oder man verwendet ein bewegliches Modul, mit dem die Kamera parallel zur Projektionsfläche zu jeder Position bewegt wird. Anschließend wird ein Bild vom Finger auf der multi-touch Oberfläche gemacht.

Das Problem für diese Methode der Authentifizierung ist, dass eine Projektionsfolie, die zur besseren Darstellung der Ausgabe eines Projektors dient, an der multi-touch Oberfläche angebracht wird. Hierdurch kann eine Kamera auf der anderen Seite nur die Finger selber erkennen, jedoch keine feineren Details. Wie auch schon in Kapitel 2 erwähnt, könnte man dieses Problem mit zwei Alternativen zum herkömmlichen Aufbau lösen:

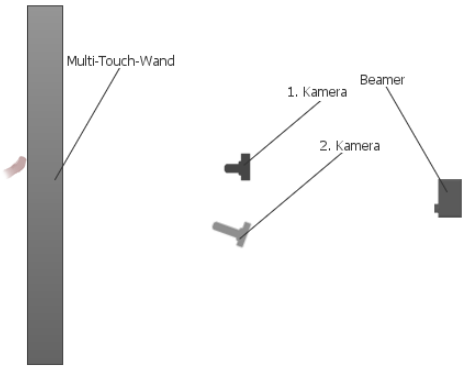

Abbildung 3.4: Mögliche Anordnung zur Fingerabdruckerfassung an einer multi-touch Wand. - Zusätzlich zur FTIR Konfiguration wird eine zweite Kamera für die Aufnahme des Fingerabdrucks hinter der Wand angeordnet und ist hierfür schwenkbar [Eigene Skizze].

1. Es wird die Torchlight Technologie [30] verwendet. Licht, insbesondere das vom Projektor, wird dann nur aus einem bestimmten Winkel an der multi-touch Oberfläche gestreut, also auf der Projektionsscheibe für den davorstehenden Benutzer sichtbar gemacht, ansonsten ist die Wand lichtdurchlässig. Dadurch können auch fein aufgelöste Aufnahmen von Fingerabdrücken gemacht werden.

2. Man nutzt die Secondlight Technologie [13]. Diese hat den Vorteil, dass die Anordnung des Projektors nicht geändert werden muss. Die multi-touch Oberfläche kann abwechselnd lichtdurchlässig oder lichtundurchlässig geschaltet werden. Dadurch kann zum einen ein Bild projiziert werden, zum anderen kann eine klare Aufnahme von einen Fingerabdruck gemacht werden.

Wenn die Kamera ein Bild vom Finger aufgenommen hat, wird dieses digital nachbearbeitet, so dass anschließend die spezifischen Merkmale besser extrahiert werden können. Diese werden dann mit einer Datenbank aller bisher angemeldeten Benutzer abgeglichen und es wird versucht, hiermit einen Benutzer zu verifizieren.

3.2.2 Mobiles Gerät

Viele Menschen besitzen heutzutage ein Mobiltelefon, dessen Funktionsumfang weit über das Telefonieren herausragt. Es bietet sich an, dieses für die Authentifizierung an einer multi-touch Oberfläche zu nutzen, um zusätzliche Anschaffungen zu vermeiden.

Dabei muss das Mobiltelefon über eine bestimmte Mindestausstattung an Hardware verfügen:

- Es ist eine Kamera erforderlich, um geheime Informationen von der multi-touch Oberfläche zu empfangen. Dabei spielt die Auflösung der Kamera keine Rolle.
- Des weiteren ist für die bidirektionale Kommunikation mit einem PC, der die multi-touch Oberfläche ansteuert und für die Authentifizierung aller Benutzer zuständig ist, eine Verbindung mit höherer Bandbreite erforderlich. Hierfür nutzt man entweder Bluetooth oder WLAN. Im Folgenden gehen wir von WLAN aus, im Prinzip würde Bluetooth auch denselben Zweck erfüllen.
- Es wird ein Lage- und Beschleunigungssensor verwendet, damit der Authentifizierungsvorgang automatisch initiiert werden kann.

Bevor eine Person ihr mobiles Gerät zur Authentifizierung einsetzen kann, muss dieses erst in der Anwendung registriert werden und eine ID haben, damit die Anwendung dieses einem Benutzer eindeutig zuordnen kann und für diesen bestimmte Interaktionen freigeben kann.

Wenn das mobile Gerät registriert wurde, kann es für eine Authentifizierung genutzt werden. Diese läuft folgendermaßen ab:

1. Starten des Authentifizierungsprozesses

   Bei laufender Authentifizierungsanwendung werden im Hintergrund alle Daten des Lage- und Beschleunigungssensors erfasst und auswertet. Der Vorgang selber wird dadurch gestartet, dass das Mobiltelefon mit der Kamera zur multi-touch Oberfläche gerichtet an dieser angelegt wird. Der Beschleunigungssensor registriert eine größere negative Beschleunigung durch den Stop an der Oberfläche. Zwar könnte diese auch in anderen Situationen auftreten, aber es wird außerdem noch überprüft, ob die Lage des Mobiltelefons genau der Lage der multi-touch Oberfläche entspricht, was bei multi-touch Wänden kein Problem darstellt, da ein mobiles Gerät im normalen Gebrauch nicht senkrecht stehend an einer Wand gestoppt wird.

   Wenn die multi-touch Oberfläche anders angeordnet wird, z.B. waagerecht, wird die Authentifizierung durch den Benutzer selber in der Anwendung gestartet. Dies

ist erforderlich, da z.B. das Ablegen des mobilen Gerätes auf einen Tisch genau der gleichen Bewegung mit gleicher Endlage entsprechen würde und die Sensoren dies nicht unterscheiden können.

Durch diese Startmethode sind vom Benutzer so wenig manuelle Eingaben wie möglich erforderlich. Er muss selber keine Eingaben in das mobile Gerät machen, sondern lediglich dieses an die multi-touch Wand halten. Wenn Authentifizierungen komplexer werden, neigen die Nutzer dazu, diese so wenig wie möglich durchzuführen und sich dann z.B. nicht abzumelden, um eine erneute Authentifizierung zu umgehen.

2. Ermittlung der ungefähren Kameraposition

   Als nächstes schickt das mobile Gerät über WLAN eine Anfrage an den Computer, der die multi-touch Benutzerschnittstelle nutzt. Für die Bestimmung der Kameraposition gibt es zwei Ansätze, wobei jeder Vor- und Nachteile hat:

   (a) Für einen kurzen Zeitraum wird ein Farbverlauf angezeigt, der von der Kamera im Mobiltelefon aufgenommen wird. Dieses schickt dann die wahrgenommene Farbe an den Computer zurück, anhand der dann die ungefähre Position der Kamera ermittelt werden kann. Es ist hier erforderlich, dass man die von der Kamera wahrgenommenen Farben kalibriert hat, so dass man weiß, was die Minimal- oder Maximalwerte sind.

   (b) Der Benutzer hält sein Mobiltelefon so an die Oberfläche, dass er zusätzlich mit einem Finger die Wand berührt und damit in unmittelbarer Umgebung der Kamera eine für den Computer wahrnehmbare Berührung erzeugt. Da mehrere Personen gleichzeitig an der Wand interagieren könnten, und dies somit nicht die einzige Berührung sein könnte, wird bei mehreren überprüft, ob eine Berührung außerhalb aller bisherigen, schon Personen zugewiesenen Subregionen liegt. Dies wird dann als Position für die Kamera geschätzt.

3. Ermittlung der genauen Kameraposition

   Da später die visuell übertragenen Informationen nur für die Kamera sichtbar sein sollen, muss die genaue Position der Kamera am mobilen Gerät bestimmt werden, damit die Informationen nicht großflächig gezeigt werden müssen. Weiterhin ist

zu berücksichtigen, dass an jedem mobilen Gerät die Kameraposition anders sein kann, insbesondere wenn diese durch eine Berührung des Benutzers mit der multi-touch Oberfläche geschätzt werden soll.

Es wird an der durch 2 erfassten Position ein Quadbaum gezeichnet, ein Rechteck, welches selber in vier gleichgroße Vierecke unterteilt ist, wobei jedes in einer unterschiedlichen Farbe angezeigt wird. Die Kamera erfasst nun eine dieser Farben und schickt den Farbwert über WLAN zum Computer zurück, der anhand diesem weiß, vor welchen Sub-Rechteck des Quadbaumes die Kamera liegt. Nun wird in diesem wieder ein neuer Quadbaum zur genauen Lokalisierung angezeigt und vom mobilen Gerät der Farbwert an den Computer weitergegeben. Dieses wiederholt sich so lange, bis die Größe des Quadbaumes so klein ist, dass dieser vom mobilen Gerät verdeckt wird, und keine Person, die vor der multi-touch Oberfläche steht, eine Farbe erkennen kann.

Abbildung 3.5: Darstellung eines Quadbaumes auf einer multi-touch Oberfläche - [Eigene Abbildung].

4. Erfassung des „geheimen“ Signals zur Authentifikation

   Um sich zu vergewissern, dass die gerade ermittelten Informationen auch vom richtigen mobilen Gerät und damit vom richtigen Benutzer kommen, wird eine Nachricht als eine Sequenz von verschiedenen Farben an die gerade ermittelte Kameraposition als Rechteck angezeigt. Um die Erkennung des Signals durch die Kamera zu erleichtern, werden die einzelnen Zeichen nicht durch feste Farbwerte repräsentiert, was insbesondere bei zwei aufeinanderfolgenden Farben problematisch wäre, sondern durch den Unterschied zur vorherigen Farbe. Die Reihenfolge

der Farben ist festgelegt, somit stellt die Anzahl der Farben, die zwischen der zuletzt angezeigten und der jetzigen liegen, ein Zeichen dar. Insgesamt wird eine vorher festgelegte Anzahl von Farben angezeigt, und daraus entsteht dann eine Nachricht.

5. Übertragung der „geheimen“ Signatur des Signals

   Nun muss diese Nachricht zum Computer, der die multi-touch Oberfläche steuert, übertragen werden. Damit diese Nachricht nicht von jemand anderem empfangen wird und dieser sich hiermit authentifizieren kann, wird zusätzlich die ID vom mobilen Gerät angehängt. Dieses hat zusätzlich einen öffentlichen und einen privaten Schlüssel, letzterer ist nur ihm bekannt. Mit diesem privaten Schlüssel wird die Signatur der Nachricht erstellt und dann zusammen mit dem öffentlichen Schlüssel zum Computer über WLAN übertragen.

   Dieser verifiziert diese Nachricht und wenn die Signatur zur zuvor übertragenen Nachricht passt, ist gewährleistet, dass sich das mobile Geröt wirklich an der Position, wohin das Signal gesendet wurde, befindet. Jetzt wird an dieser Stelle eine Subregion der multi-touch Oberfläche mit den Rechten des Benutzers freigegeben.

Mit dieser Methode kann eine Person sich durch ihr Mobiltelefon anmelden. Dabei ist die Authentifizierung allerdings eher an das Mobiltelefon gebunden anstelle der Person. Diese muss dafür sorgen, dass niemand anderes ihr mobiles Gerät missbraucht. Entweder ist der Zugang zu diesem selber durch Methoden wie die PIN Eingabe oder bei neuen mit berührungsempfindlichen Anzeigen ausgestatteten Geräten durch Gesten gesichert oder die Anwendung zur Authentifizierung wird selber durch ein Passwort geschützt, welches beim Starten eingegeben werden muss.

3.3 Szenarien

Im folgenden eine Beschreibung von 2 möglichen Anwendungsfällen für eine Benutzerauthentifikation.

3.3.1 SoKNOS

SoKNOS ist ein Forschungsprojekt des Bundesministeriums für Forschung und Entwicklung. Es soll die Zusammenarbeit aller beteiligten Parteien im Bereich öffentlicher

Sicherheit verbessern. Insbesondere sollen hier Konzepte zur Bewältigung von Großschadensereignissen erarbeitet werden. Es sollen Lösungsansätze entwickelt werden, die ein weitsichtiges, schnelles, sicheres und effektives Handeln ermöglichen und somit die Entscheidungsprozesse in Einsatzleitungen und Krisenstäben optimieren.

Abbildung 3.6: Interaktion mit der multi-touch Wand des ifgi im Rahmen des SoKNOS-Projektes - [Foto von Alexander C. Walkowski].

Wichtig ist hierbei, dass alle Informationen zentral an einem Ort zusammenfließen und hier analysiert werden, um so schnell wie möglich die Lage im Großschadensfall einschätzen zu können. Dies geschieht in einem Krisenzentrum (EOC), wo die verschiedenen Akteure schnell und koordiniert zusammenarbeiten, um die Katastrophe bewältigen zu können. Dabei ist eine gemeinsame Sicht auf die aktuelle Schadenslage notwendig. Eine großflächige multi-touch Oberfläche kommt hierfür zum Einsatz, an welcher mehrere Benutzer gleichzeitig interagieren können. Die Mensch-Maschine Interaktion erfolgt hier mittels Gesten, die schnell erlernbar sind. Somit ist im Vergleich zur Arbeit am herkömmlichen Desktop-Computer ein geringer Schulungsaufwand der Mitarbeiter im Krisenzentrum erforderlich und es kann schneller gehandelt werden [9].

Für eine optimale Kollaboration müssen in unterschiedlichen Szenarien mehrere Personen gleichzeitig interagieren können. Jede Person hat dabei unterschiedliche Verantwortungsbereiche. Bei der Feuerwehr gibt es einen zentralen Einsatzleiter, der die ganze Verantwortung trägt. Um auch Großschadensereignisse erfolgreich bewältigen zu können, werden die Aufgaben in verschiedene Sachgebiete (S1 - S6) ausgelagert und mit Führungsassistenten besetzt. Für die Interaktion an einer multi-touch Oberfläche sind insbesondere die folgenden Sachgebiete interessant:

- S2, Lage: Dieses Sachgebiet umfasst die Feststellung und Darstellung der aktuellen Schadenslage. Informationen aus verschiedenen Quellen werden gesammelt und dann in einer Lagekarte visualisiert, die dann anderen Einsatzleitungen und Krisenstäben zur Verfügung steht.

- S3, Einsatz: Hier wird die aktuelle Lage beurteilt und entsprechende Einsätze durchgeführt. Dieser Sachbereich ist auch für Lagebesprechungen verantwortlich.

- S5, Presse- und Medienarbeit: Bei einem Großschadensereignis ist es wichtig, die Medien mit passenden Informationen zu versorgen. Diese dürfen weder die aktuelle Lage verharmlosen noch zu schwerwiegend darstellen, was schnell zu panischen Reaktionen in der Bevölkerung führen könnte.

Die für S2 und S3 zuständigen Personen könnten zusammen an einer Schadenskarte arbeiten, wobei erstere die Schadenslage direkt auf der Karte aktualisiert, die für S3 zuständige Person sich dann die jeweiligen genauen Informationen beschafft. Jeder hat seine eigenen Zugriffsrechte, so darf nur der S2 Informationen ändern, der S3 hingegen als einziger Befehle an andere Einsatzstellen und/ oder Einsatzkräfte vor Ort erteilen.

Der S5 erarbeitet mit anderen Personen eine Einschätzung der aktuellen Lage, die für die Öffentlichkeit bestimmt ist. Dies geschieht mit allen beteiligten Personen direkt vor der multi-touch Benutzerschnittstelle, so dass alle einen direkten Überblick auf die aktuelle Lage haben. Die erstellte Lagekarte wird anschließend der Presse vorgeführt, diese darf ausschließlich Zugriff auf diese Informationen haben und andere sensible Informationen müssen vor unberechtigten Zugriff geschützt werden [8].

3.3.2 Multiplayer Game: Risiko

Risiko ist ein Strategiespiel für mehrere Personen. Es findet auf einer Weltkarte statt, die in Kontinente und des Weiteren in Länder unterteilt ist. Vor Spielbeginn werden alle Länder an die Spieler verteilt.

Ziel des Spiels ist es je nach Spielart, dass ein Spieler entweder die ganze Welt erobert, alle Hauptstädte eingenommen hat oder eine von verschiedenen Missionen erfüllt. Dies kann das Auslöschen einen Mitspielern sein, der Besitz von bestimmten Kontinenten oder der Besitz einer bestimmten Anzahl von Ländern.

Das Spiel selber ist rundenbasiert, die Spieler machen nacheinander ihre Züge. Während ein Spieler an der Reihe ist, muss er versuchen, durch Erobern anderer Länder seine Mission zu erfüllen. Hierfür bekommt er zu Beginn eine bestimmte Anzahl von Armeen.

In der klassischen Form wird dieses Spiel als Brettspiel mit Figuren, Karten und Würfeln gespielt. Es existieren bereits verschiedene digitale Umsetzungen dieses Spieles. Im Studienprojekt „Multi-touch Risk“ (zusammen mit Wadim Hamm) wurde ein bestehendes Risiko-Spiel, welches für einen 3D-Globus entwickelt wurde, für eine großflächige multi-touch Wand am ifgi weiterentwickelt. Durch die große Darstellung können mehrere Spieler problemlos das Spielgeschehen mit verfolgen, was bei der bisherigen Spielweise an einem Desktop-PC nicht leicht möglich war. Das Spiel selber wird komplett durch verschiedene Gesten gesteuert.

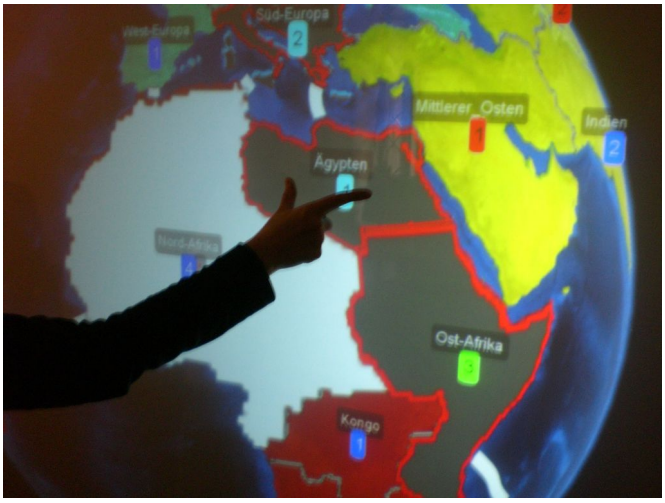

Abbildung 3.7: Risiko als multi-touch Anwendung. - [Eigene Abbildung].

Da das Spiel rundenbasiert ist, darf zu jeder Zeit nur ein Spieler interagieren, und dieser kann dann auch nur seine Einheiten bewegen und kämpfen lassen. Deswegen ist es wichtig, zu diesem Zeitpunkt Interaktionen der Gegenspieler zu verhindern, zudem einmal ausgeführte Befehle nicht mehr rückgängig gemacht werden können. Des Weiteren gibt es bestimmte Spielinformationen wie z.B. die Mission zum Gewinn des Spiels, die nur ein Spieler selber einsehen darf und vor anderen geheim gehalten werden müssen.

3.4 Umsetzung Technik auf Szenarien

Aus den im vorherigen Abschnitt vorgestellten Nutzungs-Szenarien können mehrere Anforderungen an die Authentifizierung abgeleitet werden:

Im ersten Szenario ist eine Interaktion von mehreren Benutzern simultan notwendig. Jedem Nutzer wird eine eigene Subregion zugeordnet, in der er interagieren kann. Dabei muss insbesondere der Sicherheitsaspekt berücksichtigt werden, so dass für kritische Befehle eine besondere Autorisierung erforderlich ist. Des Weiteren ist in beiden Anwendungsfällen eine schnelle und unkomplizierte Authentifizierung notwendig. Diese sollte direkt an der multi-touch Oberfläche machbar sein. Eine klassische Anmeldung per Kennwort und Passwort kommt nicht in Frage.

3.4.1 Authentifizierung mit mobilen Gerät

In diesem Abschnitt soll die konzeptionelle Umsetzung der in 3.2.2 erläuterten Authentifizierung mit Hilfe eines mobilen Gerätes auf die zuvor erwähnten Szenarien dargestellt werden.

3.4.1.1 SoKNOS

SoKNOS ist eine sicherheitskritische Anwendung, die auch eine multi-touch Oberfläche verwendet. In einem Krisenzentrum werden Ereignisse gesteuert, die Einfluss auf die Realwelt haben, wie z.B. das Aussenden von Einsatzkräften. Dies soll durch den Einsatz von Mobiltelefonen zur Authentifizierung sicher gemacht werden. Jede Person, die sicherheitskritische Aktionen ausführen darf, hat hierfür ein Mobiltelefon, auf dem eine wie schon in 3.2.2 beschriebene Applikation läuft. Bevor das Gerät zur Authentifizierung eingesetzt werden kann, muss erst erstmal in SoKNOS angemeldet werden. Dabei muss einer Person ein mobiles Gerät zugeordnet werden.

Ein hierfür verantwortlicher Systemadministrator vergibt eindeutige IDs. Diese werden einer Person per Email zugeschickt. Anschließend muss der Benutzer diesen Code in das mobile Gerät eingeben, dieses kann dann fortlaufend über diese ID von SoKNOS zugeordnet werden. Die ID an sich muss geheim gehalten werden, damit sich kein anderes Mobiltelefon hiermit ausweisen kann.

Beim Authentifizierungsvorgang wird zuerst eine Anfrage vom mobilen Gerät an den Computer, der die multi-touch Benutzerschnittstelle nutzt, geschickt. Dann wird

wie in 3.2.2 schon beschrieben das mobile Gerät und damit den Benutzer lokalisiert, damit dieser mit allen ihm zustehenden Zugriffsrechten interagieren kann. Des Weiteren sollte ein automatischer Abmeldemechanismus gesetzt werden, da die Handlungen in einem Krisenstab schnell erfolgen müssen und es vorkommen könnte, dass ein Benutzer vergisst, sich abzumelden.

Um das Personal im Krisenzentrum nicht mit unnötigen Authentifizierungsvorgängen zu belasten, muss sich ein Benutzer nur dann ausweisen, wenn sicherheitskritische Aktionen ausgeführt werden. Ansonsten kann jeder allgemeine Interaktionen an der multi-touch Benutzerschnittstelle ausführen.

3.4.1.2 Risiko

Bei diesem Strategiespiel muss gewährleistet werden, dass nur derjenige Interaktionen an einer multi-touch Benutzerschnittstelle ausführen kann, der auch am Zuge ist und Missbrauch hiervon durch konkurrierende Mitspieler verhindert wird. Deswegen ist es erforderlich, dass sich jeder Spieler vor Beginn seiner Runde authentifiziert.

Zuerst muss jedoch jedem Spieler ein mobiles Gerät eindeutig zugewiesen werden. Da die Mitspieler und Konfiguration von Spiel zu Spiel variieren, reicht es aus, dass die Zuweisung der Geräte für ein Spiel gültig ist. Zu Beginn des Spiels ist es immer erforderlich, dass einige Einstellungen vorgenommen werden, wie Anzahl Spieler, ob ein Spieler von Mensch oder Computer gesteuert wird oder die Gewinnbedingung. Weiterhin gibt es dann einen zusätzlichen Optionenschirm, an dem schon die Namen und Farben aller Spieler gesetzt wurden. Nun kann hier zusätzlich für jeden Spieler eine Subregion definiert und angezeigt werden, vor der die Kamera seines Mobiltelefons gehalten wird. Dabei hat die Subregion die Farbe des Spielers, die vorher eingestellt wurde, und diesen Spieler auch eindeutig identifiziert.

Anschließend wird wie schon allgemein in 3.2.2 beschrieben, die genaue Position der Kamera bestimmt. Nun wird eine Sequenz von Farben angezeigt, durch die die ID des mobilen Gerätes dargestellt wird. Anschließend wird von dieser eine Signatur mit Hilfe des privaten Schlüssels des mobilen Gerätes erstellt und an der Computer mit der multi-touch Benutzerschnittstelle übertragen. Anschließend wird zusätzlich der öffentliche Schlüssel übertragen und der Computer kann verifizieren, ob die ID richtig übertragen wurde.

Wenn dieser Vorgang erfolgreich abgeschlossen wurde, ist der Spieler mit seinem mobilen Gerät mit einer bestimmten Spielerfarbe im Spiel registriert und kann sich hiermit zu Beginn seiner Spielrunde durch die in 3.2.2 beschriebene Vorgehensweise authentifizieren. Es ist zu berücksichtigen, dass der Spieler die ganze multi-touch Oberfläche für sich beansprucht, jegliche Interaktion wird als seine angesehen. Somit kann der für die Schätzung der Kameraposition erforderliche Quadbaum an die Position einer Berührung mit der multi-touch Oberfläche angezeigt werden. Bei mehreren und zudem weit auseinander liegenden Berührungen sollte das nicht geschehen, da davon ausgegangen werden kann, dass nur der eine Benutzer mit der Wand interagiert und es in diesem Fall mehrere Personen sein könnten, man also von einen Täuschungsversuch ausgehen kann.

Bei Erfolg ist der Spieler angemeldet und kann seine Spielaktionen ausführen. Des weiteren werden auf dem Mobiltelefon geheime Spielinformationen wie seine Mission oder auch seine Karten angezeigt, die somit kein Gegenspieler einsehen kann. Nachdem der Spieler seine Spielzüge gemacht hat und seine Runde beendet hat, muss sich der nächste Spieler authentifizieren und die multi-touch Oberfläche ist bis dahin für weitere Spiel-Interaktionen gesperrt. Allerdings kann eine Person schon einige, aber nur allgemeine Aktionen ausführen, die keine Bedeutung für das eigentliche Spielgeschehen haben wie z.B. die Navigation des Globus, um sich einen Überblick über die aktuelle Lage verschaffen zu können.

3.4.2 Authentifizierung mit Fingerabdrücken

In diesem Abschnitt soll die konzeptionelle Umsetzung der in 3.2.1 erläuterten Authentifizierung mit Hilfe einer Fingerabdruckerkennung auf die beiden zuvor erwähnten Anwendungszenarien dargestellt werden.

3.4.2.1 SoKNOS

Der Fingerabdruck ist ein gutes biometrisches Merkmal zur Identifikation von Personen für sicherheitskritische Anwendungen. Eine davon wäre die Steuerung von SoKNOS mit einer multi-touch Oberfläche. Diese wird in einem Krisenzentrum eingesetzt, von wo für die Realwelt kritische Aktionen ausgeführt werden wie z.B. das Aussenden von Einsatzkräften.

Bevor ein Benutzer SoKNOS als multi-touch Anwendung mit seinen Zugriffsrechten nutzen kann, muss sein Fingerabdruck registriert werden. Dies geschieht durch einen Administrator, der als einziger Nutzerrechte in Kombination mit einem Fingerabdruck vergeben kann. Dieser nimmt den Fingerabdruck eines neuen Benutzers in eine Datenbank auf und ordnet diesen der Person zu und vergibt Freigaben für bestimmte Interaktionen.

Anschließend kann der Benutzer SoKNOS mit seinen Zugangsrechten nutzen. Allgemeine Aktionen wie das Betrachten einer Lagekarte kann dieser ohne Authentifizierung durchführen, sicherheitskritische Aktionen erfordern eine Authentifizierung. Hierfür muss der Benutzer lediglich den Finger auf die Oberfläche der Anzeige drücken, den er auch zur Registrierung genommen hat, und die Anwendung erfasst seinen Fingerabdruck und versucht diesen zu identifizieren. Bei Erfolg kann der Benutzer seine Aktion ausführen und sich anschließend wieder abmelden. Des Weiteren gibt es einen automatischen Abmeldemechanismus. Wenn ein Benutzer sich für bestimmte Interaktionen authentifiziert hat, läuft diese Freigabe nach einem kurzen Zeitraum ab, für weitere sicherheitskritische Handlungen muss sich diese Person wieder authentifizieren. Damit ist gewährleistet, das wenn ein Benutzer vergisst, sich abzumelden und die multi-touch Oberfläche verlässt, kein anderer unbegrenzten Zugang mit bestimmten Sicherheitsfreigaben hat.

3.4.2.2 Risiko

Bei dieser Umsetzung des Strategiespiels für eine multi-touch Oberfläche ist es wichtig, dass nur die Person interagieren darf, die auch am Zuge ist. Deswegen ist eine Authentifizierung vor Beginn seiner Runde erforderlich.

Vor Beginn des Spiels ist eine Zuordnung eines Fingerabdrucks zu einem Spieler erforderlich. In dem Startmenü des Spiels gibt es hierzu neben den Elementen, wo von einem Spieler Name und Farbe gesetzt werden, auch eine Fläche, auf die ein Spieler seinen Finger legt und einem virtuellen Spieler zugeordnet wird. Das System nimmt anschließend von dieser Fläche einen Fingerabdruck auf und kann diesen zur Verifikation nutzen. Wenn dies alle Spieler gemacht haben, kann das Spiel beginnen.

Zu Beginn einer jeden Runde eines Spielers muss sich dieser erst authentifizieren, indem er seinen Finger, den er auch vor Beginn des Spiel für die Fingerabdruckzuordnung genommen hat, auf die Oberfläche der Anzeige legt und hiervon ein Bild erfasst

wird. Anschließend wird versucht, diesen Fingerabdruck zu verifizieren, bei Erfolg kann der Spieler seine Züge machen. Wenn er fertig ist, beendet er seine Runde, und der nächste Spieler muss sich per Fingerabdruck anmelden. Wenn ein anderer Spieler versucht, Interaktionen auszuführen, so ist dies nur bei allgemeinen, nicht spielrelevanten möglich, wie z.B. die Navigation des Globus, um sich die aktuelle Situation anschauen zu können.

4

Implementierung

Dieses Kapitel beschreibt die konkrete Umsetzung der beiden Ansätze zur Authentifizierung.

4.1 Multi-touch Wand am ifgi

Im Folgenden wird die multi-touch Wand (MTW) des Institut für Geoinformatik in Münster kurz beschrieben, für welche im Rahmen dieser Bachelorarbeit verschiedene Ansätze zur Authentifizierung entwickelt wurden.

Die MTW verwendet die FTIR Technologie. Die Wand selber ist 2,2 x 1,4 m groß und besteht aus einer stehenden Plexiglas-Scheibe, die in einem Rahmen eingebaut ist 3.6. In den Seiten der Plexiglas-Scheibe sind Infrarot-Dioden integriert, welche Infrarot-Licht in die Scheibe emittieren. Auf der Rückseite ist eine Projektionsfolie angebracht, hier wird vom drei Meter dahinter liegenden Projektor das Bild in einer Auflösung von 1920 x 1080 Pixeln projiziert. Zwischen Wand und Projektor ist eine Kamera, die zwei Meter hinter der Wand steht, aber nicht im Projektionsfeld. Diese Kamera vom Typ PointGrey Dragonfly 2 hat einen Infrarot-Filter vor ihren Objektiv, womit die in ihre Richtung gestreute Infrarotstrahlung erfasst wird. Das Objektiv selber ist ein Weitwinkel-Objektiv vom Typ M13VM308, damit hinter der Wand nicht so viel Raum beansprucht wird. Die Kamera verfügt über eine Auflösung von 1024 x 768 Pixeln und schafft dabei 30 Bilder pro Sekunde. Die erfasste Infrarotstrahlung wird dann auf Softwareebene in Fingerberührungen mit der multi-touch Oberfläche umgerechnet.

Diese werden dann über das TUIO-Protokoll an eine multi-touch Anwendung geschickt [15].

Die MTW steht in einen abgedunkelten Teil eines Raumes, damit die Kamera nur Infrarotlicht von der multi-touch Oberfläche empfängt und um andere Lichtquellen hierfür auszuschließen.

4.2 Authentifizierung durch Fingerabdruckerkennung

In diesem Kapitel wird der Versuchsaufbau beschrieben, womit von der multi-touch Oberfläche Fingerabdrücke erfasst wurden. Wie aus dem letzten Kapitel 5.1.1 dieser Arbeit deutlich wird, wurde dieser Ansatz nicht weiter umgesetzt.

4.2.1 Hardwarekomponenten

- Kamera

  Um einen Fingerabdruck aufnehmen zu können, wurden Bilder mit Hilfe einer Dragonfly Kamera gemacht, die gleiche wird auch dazu verwendet, um die Berührungen an einer multi-touch Oberfläche zu erfassen. Als Auflösung wurde die maximal mögliche von 1024 x 768 verwendet. Das Objektiv der Kamera wurde gegen ein spezielles Objektiv, ein 1.23FM50SP ebenfalls von der Firma Tamrom, ausgetauscht, um ein Maximum an Details des Fingerabdrucks zu erfassen.

- Multi-touch Oberfläche

  Die MTW am Institut für Geoinformatik in Münster konnte nicht ohne weiteres für die Aufnahme von Fingerabdrücken verwendet werden, da für die Anzeige des Bildes eine streuende Folie an der Plexiglasscheibe angebracht ist, durch welche feine Details wie die Konturen der Finger nicht erkannt werden können. Deswegen wurde ein Glastisch verwendet, welcher zum Testen von unterschiedlichen multi-touch Oberflächen für FTIR verwendet wird. In dem Tisch ist eine Plexiglasscheibe mit der Größe 20 x 40 cm integriert und in einem Holzrahmen eingebaut, in dessen Seiten Infrarotdioden eingelassen sind, die Infrarotlicht in die Scheibe emittieren. Ein Teil der Scheibe war mit Silicon bedeckt und der Rest war frei von Beschichtungen, Folien oder ähnliches, was die Sicht auf den Finger verhindert hätte.

4.2.2 Aufnahme der Fingerabdrücke

Die Kamera wurde auf den Boden gelegt, um vom darüber liegenden Glastisch Aufnahmen zu machen. Der Abstand zwischen beiden betrug 55 cm, dies ist optimal um genau den Ausschnitt von einem Finger zu erfassen. Der Glastisch befand sich im gleichen Teil des Raumes, wo sich auch die MTW befindet. Somit waren die Lichtbedingungen dieselben wie bei der MTW, die Kamera hat nur dort Infrarotlicht empfangen, wo es von einem auf der Plexiglasscheibe liegenden Finger gestreut wurde.

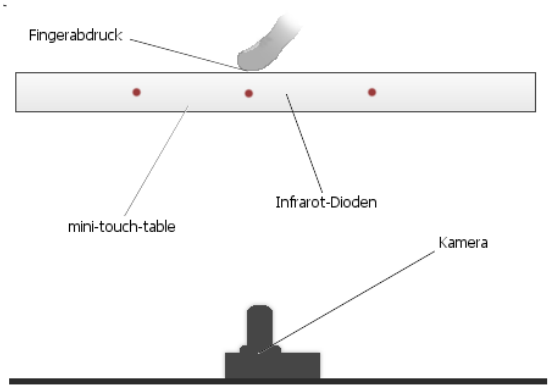

Abbildung 4.1: Versuchsaufbau Fingerabdruckentnahme FTIR. - Der Finger wird auf eine Plexiglasscheibe gelegt, es werden dann IR-Strahlen Richtung Kamera reflektiert [Eigene Abbildung].

Für die Erfassung eines Fingerabdruckes wurde ein Finger auf einen von der Kamera fokussierten Teil der Scheibe gedrückt. Die Kamera ist über Firewire direkt mit einen Computer verbunden gewesen, über den Treiber wurde direkt ein Bild aufgenommen und im jpg-Format abgespeichert. Dieses wurde bei insgesamt fünf Personen durchgeführt, jede Person hat zehnmal hintereinander denselben Finger, den rechten Zeigefinger, auf die Scheibe gelegt und ein Foto hiervon machen lassen. Dieser ist auch ein Finger, der häufig für multi-touch Gesten eingesetzt wird und deswegen häufig in Kontakt mit der Oberfläche kommt.

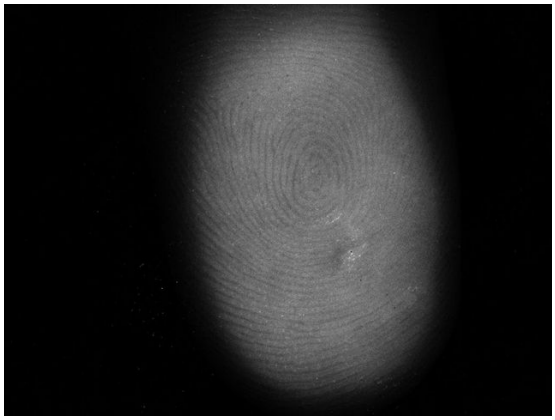

Abbildung 4.2: Fingerabdruck auf FTIR-Versuchsaufbau. - [Eigene Abbildung].

4.2.3 Erkennung und Verifizierung eines Fingerabdrucks

Für die weitere Verarbeitung der Bilder wurde das Programm „Verifinger algorithm demo“ der Firma NEURO Technologie in der Version 6.0.0.0 genutzt [28]. Dieses verwendet einen ausgereiften Algorithmus zur Erkennung und Extraktion der Epidermis und ihre Merkmale, um verschiedene Finger zu unterscheiden. Anschließend wurden diese Merkmale in einer Datenbank gespeichert und konnten zur Verifizierung oder Identifikation einer Person durch deren Fingerabdruck verwendet werden.

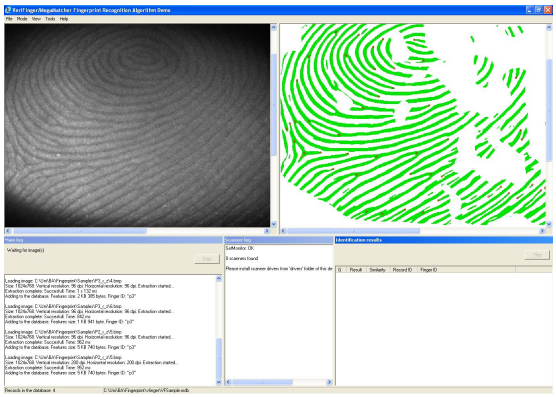

Abbildung 4.3: Aufnahme von Merkmalen durch Verifinger - Links das Eingabebild, rechts der erkannte Fingerabdruck durch das Programm [Screenshot aus VeriFinger].

4.3 Authentifizierung durch ein mobiles Gerät

In diesem Teil werden die einzelnen Soft- und Hardwarekomponenten für die Implementierung des in 3.2.2 erwähnten Ansatzes beschrieben. Dann wird auf die Implementierung der Client- und Serverseite eingegangen. Diese ist prototypisch für das in 3.4.1.2 erwähnte Strategiespiel erstellt worden. Somit nimmt die Kamera des mobilen Gerätes zuerst die Farbe eines Spielers auf, ermittelt dann die genaue Kameraposition des Gerätes vor der multi-touch Oberfläche und überträgt hier eine Farbsequenz, die die ID des Benutzers darstellt. Nun ist der Benutzer in der Server-Anwendung registriert und kann sich jederzeit an der MTW authentifizieren.

4.3.1 Verwendete Komponenten

Die Authentifizierung wird an der in 4.1 beschriebenen multi-touch Oberfläche durchgeführt. Als Server dient hierfür ein handelsüblicher Desktop-PC, an dem die in 4.2.1 erwähnte Kamera und der Projektor angeschlossen sind. Des Weiteren ist dieser PC mit dem Netzwerk der Universität Münster verbunden. Als Betriebssystem ist Windows XP installiert, als Entwicklungsumgebung wird Eclipse 3.2 verwendet. Das JDK ist Java SE Development Kit 6 Update 11.

Das eingesetzte mobile Gerät ist ein „Android Dev Phone“, welches über keinerlei Hardwaresperren verfügt. Dessen Betriebssystem ist Android 1.5 („Cupcake“). Zur Verbindung mit dem Server wurde das integrierte WLAN genutzt, hierzu ist das Gerät im WLAN der Universität Münster eingewählt. Zudem ist in dem Gerät eine 3,2 Megapixel Kamera integriert.

Abbildung 4.4: Android Dev Phone - [Eigene Abbildung].

4.3.2 Aufbau Android-Anwendung

Die Android-Anwendung besteht aus der Hauptklasse `MTWUser`, den Klassen `Preview` und `WorkingThread`. Die Hauptklasse erzeugt die GUI, welche aus zwei Schaltfeldern, einem Textfeld zur Informationsanzeige und bei Bedarf einer Kameravorschau besteht. Die eine Schaltfläche dient zum setzen der Position der MTW, die andere zum manuellen Abmelden eines authentifizierten Benutzers oder für einen Abbruch des jetzigen Authentifizierungsvorgangs. Das Textfeld zeigt an, ob ein Benutzer erfolgreich authentifiziert wurde und kann weiterhin anwendungsspezifische und nur für diesen Benutzer sichtbare Informationen anzeigen. Die Kameravorschau muss angezeigt werden, ansonsten bekommt man von der Kamera keine Vorschaubilder. Weiterhin wird in der Hauptanwendung zur Authentifikation ein neuer Thread zur Kommunikation mit dem Server gestartet. Damit dieser Vorgang gestartet wird, werden laufend die Daten des Lage- und des Beschleunigungssensors ausgewertet. Zwei zentrale Methoden zur Steuerung des Authentifikationsvorgangs sind `startAut()` und `stopAut()`.

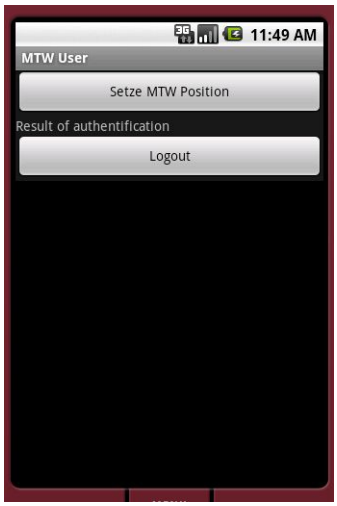

Abbildung 4.5: Android GUI - GUI vor dem Starten eines Authentifizierungsvorgangs [Eigene Abbildung].

Durch erstere wird der Vorgang gestartet. Es wird über einen Socket eine Verbindung mit dem Server aufgebaut und zur Kommunikation ein In- und Outputstream erzeugt, die Nachrichten in Form von Bytes zum Server schicken bzw. von diesem empfangen. Als nächstes wird ein Thread erzeugt, der eingehende Nachrichten vom Server weiterverarbeitet. Weiterhin beginnt die Vorschau der Kamera, die Klasse `WorkingThread`

zur Weiterverarbeitung der Vorschaubilddaten wird initialisiert und es werden keine Daten mehr vom Beschleunigungssensor empfangen.

Dementsprechend stoppt `stopAut()` die Kameravorschau und registriert wieder den Beschleunigungssensor, damit hiervon Daten empfangen werden. Danach wird an den Server eine Nachricht geschickt, damit dieser den Authentifizierungsvorgang stoppt und der Socket zu diesem inklusive des In- und OutputStreams wird geschlossen.

Wenn der Benutzer noch nicht authentifiziert wurde, wird dieser Vorgang durch erreichen der Lage der MTW gestartet. Gestoppt wird der Vorgang, nachdem dieser komplett durchlaufen wurde, das Handy nicht die Lage der MTW hat oder sich der Benutzer vorher manuell abgemeldet hat. Im Folgenden werden die einzelnen Teile der Applikation näher erläutert.

4.3.2.1 Lage- und Beschleunigungssensoren

Die Android-Anwendung sollte so konzipiert sein, dass der Benutzer so wenige manuelle Eingaben wie möglich machen muss, um die Authentifizierung möglichst einfach zu gestalten. Nach dem Start des Programms wird einmalig die genaue Lage der MTW eingespeichert, dies sind absolute Koordinaten in einem kartesischen Koordinatensystem mit x,y und z-Werten. Hierzu werden zuerst zwei `SensorListener` registriert, der eine für den Lagesensor und der andere für den Beschleunigungssensor. Dann kann der Benutzer durch eine Schaltfläche die jetzige Position als die Lage der MTW definieren.

Jetzt wird laufend die aktuelle Position mit dieser abgespeicherten verglichen. Damit das mobile Gerät die Position der MTW erreicht hat, müssen bei diesem Vergleich die Differenzen zwischen beiden Werten kleiner sein als je nach Achse ein Schwellenwert 'epsx', 'epsy' oder 'epsz'. Der erste und letzte Schwellenwert wurden auf zehn gesetzt, 'epsy' auf fünf. Weiterhin wird die Beschleunigung in Z-Richtung überprüft. Die Z-Richtung ist diejenige, die orthogonal zum Bildschirm des mobilen Gerätes verläuft. Dabei muss ein Beschleunigungswert von mindestens drei g innerhalb der letzten fünf Messungen erreicht worden sein, was einem leichten Stoß mit der Plexiglasscheibe entspricht. Die letzten fünf Messungen deshalb, weil die Messwerte der beiden Sensoren nicht synchronisiert ankommen. Insgesamt wird der Start der Anwendung durch folgenden Java-Code realisiert:

Listing 4.1: Bedingungen zum Start der Authentifizierung

```
if (fixedPosition // mtw position?
        && !tryAut // aut. Process gestartet?
        && (maxZ > 3 || not1stPosCheck) // welche z-beschleuning
        && Math.abs(x - mtwX) < epsX  //
        && Math.abs(y - mtwY) < epsY
        && Math.abs(z - mtwZ) < epsZ
){      if (not1stPosCheck){
        if (System.currentTimeMillis() - oldTime > epsT){
            showDialog(DIALOG_MTWPosFound);
            tryAut = true;        } }
```

Wenn die Authentifizierung gestartet wurde, wird laufend durch den Lagesensor überprüft, ob das Handy noch die Lage der MTW hat, wenn nicht, wird der Vorgang abgebrochen. Dies soll verhindern, dass sich das Handy nicht direkt vor der Wand befindet, aber andere Personen das gezeigte Farbsignal sehen können.

4.3.2.2 Kameravorschau

Für die Übertragung eines geheimen Signals von der multi-touch Wand zum Dev Phone und zur Lokalisierung von diesem muss ein durch Farben codiertes Signal von der Kamera aufgenommen werden.

Hierzu ist die Klasse `Preview` zuständig. Im Prinzip würde es reichen, wenn man sich in dieser Klasse die Instanz der Kamera holen würde, und dann die Schnittstelle `Camera.PreviewCallback` implementiert und bei der Kamera registriert. Leider startet in der jetzigen Version von Android die Kamera nach einem `mCamera.startPreview()` nicht ohne weiteres die Vorschau, für diesen Vorgang muss über die Methode `mCamera.setPreviewDisplay(holder)` ein `SurfaceView` registriert werden, welches die Vorschau auch wirklich auf dem Bildschirm ausgibt. Deswegen ist diese Klasse vom Typ `SurfaceView` und es ist eine minimale Vorschau des Kamerabildes zu sehen.

Dieses Vorschaubild wird auf dem unteren Teil der Anwendungs-GUI ausgegeben, es gibt die Methoden `startCam` und `stopCam`, die für die Steuerung der Vorschau und der Kamera zuständig sind und bei Bedarf diese Ressource wieder freigeben bzw. beanspruchen.

Die Kamera an sich wurde auf eine Auflösung von 176 x 144 Pixel eingestellt, um so wenig Ressourcen wie möglich zu verbrauchen und eine hohe Wiedergaberate der Vorschaubilder zu gewährleisten. Dies ist die niedrigst-mögliche Auflösung, hierbei werden

ungefähr zehn Vorschaubilder pro Sekunde angezeigt. Für die weitere Verarbeitung ist diese Bildgröße auch vollkommen ausreichend, da nur der Durchschnittsfarbwert vom gesamten Bild berechnet wird. Die Auflösung ist auch schon fast die einzige Einstellung der Kameravorschau, die man bei der jetzigen Version von Android vornehmen kann. Für die Farberkennung ist eine Deaktivierung des automatischen Weißabgleichs sehr wichtig, dies lässt sich jedoch nicht einstellen. Dies führt dann ab und zu dazu, dass eine Farbe bei längerer Anzeige auf der MTW als eine andere erkannt wird, weil der Weißabgleich versucht, durch die Helligkeit der multi-touch Oberfläche die Helligkeit des Vorschaubildes anzupassen. Somit war es nicht möglich, an die rohen Bildwerte des Kamerasensor zu gelangen.

Ein Vorschaubild wird im YUV 4:2:0 Format ausgegeben. In diesem Farbraum gibt eines Kanal (Y) für die Helligkeit und zwei (U und V) Farbkanäle. Dabei liegen für vier Pixel im Helligkeitskanal jeweils ein Pixel für einen Farbkanal vor. Für meine Anwendung ist diese Farbcodierung von Vorteil, da nur die Farbinformationen benötigt werden, und weniger Daten im Vergleich zur Standardfarbcodierung RGB weiterverarbeitet werden müssen. Außerdem sind die Helligkeitsinformationen unbrauchbar, da der automatische Weißabgleich selbst aus Schwarz nach einer gewissen Zeit kein Schwarz, sondern ein dunkles Weiß erkennt. Das Vorschaubild wird in einem Feld von Bytes gespeichert, diese können in Java ganzzahlige Werte von -128 bis 127 annehmen.

4.3.2.3 Weitere Bildverarbeitung der Kameravorschau

Die weitere Verarbeitung des Vorschaubildes und Erkennung und Signalbildung von Farben erfolgt in der Klasse `WorkingThread`. Diese ist als wartender Thread konzipiert, jedes mal wenn ein neues Vorschaubild in `Preview` angezeigt wird, wird dem `WorkingThread` dieses als byte-Feld übergeben und dieser Thread wird dann benachrichtigt, dass er weiterlaufen soll. Es gibt drei Modi, in denen der Thread läuft:

`NO_CAM, REG_COL` und `REC_SIG`.

Diese werden in der Hauptklasse `MTWUser` gesetzt, und dienen zur Steuerung, ob der Thread die Farbe der Vorschau zurückgeben, aus allen Vorschaubildern ein Signal zusammensetzen oder keine Weiterverarbeitung erfolgen soll. Für die Erkennung einer Farbe und/oder eines Signals daraus ist die Bildverarbeitung der Kameravorschau gleich:

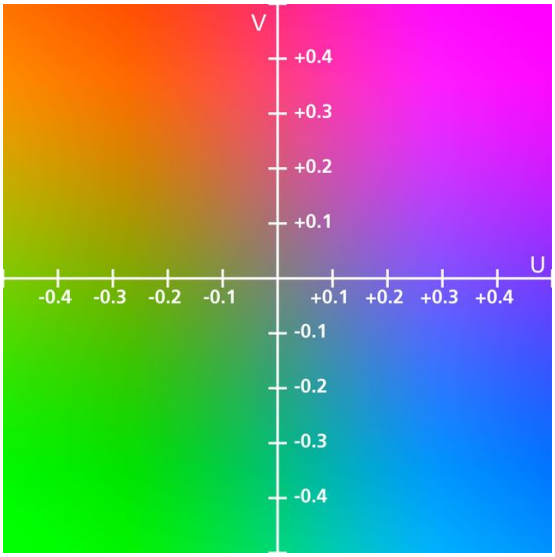

Abbildung 4.6: YUV-Farbraum bei Y = 0.5. - Gefunden unter http://upload.wikimedia.org.

Aus dem übergebenen Byte-Feld werden in der Methode `extractUV()` die Durchschnittswerte der beiden Farbkanäle U und V ermittelt, dies sind für U die Bytes ab 2/3 bis 5/6 der Feldgröße und für V die restlichen Bytes. Zu berücksichtigen ist hier, dass die Hardware des verwendeten Android Dev Phone nicht direkt Gleitkommazahlen verarbeiten kann und man aus Geschwindigkeitsgründen möglichst nur ganzzahlige Operationen durchführen sollte.

Listing 4.2: Ermittelung Durchschnittswerte für U und V

```
int begin = bytes.length * 2 / 3;
int end = bytes.length;
for (int i = begin; i < end ;){
        sumV += bytes[i];
        i++;
        sumU += bytes[i];
        i++;
}
int noUV = (end - begin)/2;
avgU = sumU/noUV;
avgV = sumV/noUV;
```

Diese Farbwerte müssen nun in konkrete Farben übersetzt werden. Damit die Erkennung dieser so zuverlässig wie möglich funktioniert, wurden Farben ausgewählt,

die maximale bzw. minimale Werte für U und V haben, siehe hierzu Abbildung 4.6. Dies sind auf jeden Fall die beiden Farben Orange und Dodger-Blau. Erstere ist das Standardorange aus der Java-Klasse `Color`, zweitere hat die Werte R: 30, G: 144 B: 255 und ist ein helleres Blau gegenüber dem Standard-Blau aus Java. Tests mit dem Android Handy haben gezeigt, dass dieses hinsichtlich U- und V-Werte besser erkennt wird. Diese beiden Farben werden für die Darstellung der Binärbäume genutzt. Für die Kodierung des Farbsignals, bestehend aus zwei verschiedenen Zeichen, ist eine dritte Farbe erforderlich. Hierzu wurde das Standard-Grün von Java genommen, welches nicht so zuverlässig wie die beiden vorherigen erkannt wird, aber durch einen langsameren Wechsel der Farben dennoch ausreichend gut erkannt wird.

Für die Bildung von Quadbäumen wäre eine vierte Farbe erforderlich gewesen, die im maximalen Kontrast zu den bisher beschriebenen Farben steht, was im YUV-Farbraum Pink bis Violett gewesen wäre. Wegen des vorher schon erwähnten nicht abschaltbaren Weißabgleichs und allgemeiner Probleme der Kamera bei der Erkennung wurde auf diese Farbe verzichtet und es wird mit Binärbäumen statt Quadbäumen gearbeitet. Als mögliche Farbe hatte sich Tief-Pink herausgestellt mit den Farbwerten R: 255 , G: 20, B: 147. Jedoch wurde des Öfteren durch die Kamera ein Blau erkannt, welches auch durch die Vorschau auf dem Handy-Displays ersichtlich wurde, jedoch auf der MTW eindeutig ein Pink angezeigt wurde. Oder es wurde ein Orange erkannt. Beide Fehlerkennungen würden die Zuverlässigkeit der Bestimmung der Kameraposition und der Erkennung des übertragenen Farbsignals erheblich senken, deswegen wurde diese Farbe zu Lasten der Gesamtgeschwindigkeit der Applikation nicht verwendet.

Andere Farben kamen als vierte Farbe nicht in Betracht, es wurden hier schon die besten Möglichkeiten des YUV-Farbraumes erwähnt. Auch Schwarz wurde nicht als Schwarz erkannt durch den Weißabgleich, Weiß wurde des Öfteren sogar wegen ähnlicher U- und V-Werte als Blau erkannt.

Um aus den Durchschnittswerten für U und V konkrete Farben zu bestimmten, wurden bestimmte Schwellenwerte gesetzt, siehe hierzu Tabelle 4.1 .

Diese Farbwerte wurden so großzügig wie möglich definiert, weil die Kamera nicht immer konstante Werte für eine Farbe erkannt hat und solange es keinen Konflikt mit einer anderen Farbe gibt, führt dies zu einer zuverlässigeren Erkennung.

Entsprechend der aktuellen Farbwerte wird eine Zeichenkette `regcol` auf „orange“, „blue“, „green“ oder „pink“ gesetzt, letzteres wurde im Programm gelassen, falls

Farbe	Min U	Max U	Min V	Max V
Orange	30			0
Blau		- 20	20	100
Grün	- 20		20	
Pink		- 50	20	

Tabelle 4.1: Die Minimal- und Maximalwerte für den U- und V-Wert einer jeden Farbe, die durch diese definiert und erkannt wird.

es ein Handy mit besserer Kamera bzw. besser einzustellender Kamera geben wird, welches auch vier Farben gut erkennt. Um allgemein die Zuverlässigkeit der Farberkennung zu erhöhen und um zu wissen, ob noch die gleiche oder schon die nächste Farbe erfasst wurde, gibt es eine ganzzahlige Variable `colorCounter`, die mitzählt, wie lange die gleichen Farben hintereinander erkannt wurden. Dies ist notwendig, da beim Wechsel der Farben nicht sofort die neue Farbe erkannt wird bzw. zwischendurch kurz eine andere Farbe erkannt wird. Damit dies funktioniert, wird zusätzlich die zuletzt erkannte Farbe gespeichert. Nachdem eine Farbe erkannt wurde, wird je nach Modus des `WorkingThread` eine bestimmte Nachricht an den Server geschickt.

Wenn die Kameraposition bestimmt werden soll und das Handy hierzu einfach nur die aktuelle Farbe erkennen soll, wird der Modus auf `REG_COL` gesetzt, um die aktuelle Farbe an den Server zu senden. Als Vorbedingung muss mindestens fünfmal hintereinander dieselbe Farbe erkannt worden sein. Als nächstes wird über einem Outputstream, der vorher von `MTWUser` initialisiert wurde, eine Nachricht an den Server geschickt.

Wenn ein Signal aus einer Farbsequenz erkannt werden soll, wird der Modus des Threads auf `Rec_Sig` gesetzt. Damit eine Farbe für die Signalsequenz gültig ist, muss sie mindestens zweimal hintereinander erkannt worden sein und nicht der zuletzt erkannten Farbe entsprechen. Diese Farbe wird anschließend einer Liste hinzugefügt. Die Erkennung des Signals ist abgeschlossen, wenn eine vorher festgelegte Anzahl an Farben erkannt wurde. Für diese Implementation war die Sequenz sechs Farben lang. Der nächste Schritt ist die Umwandlung der Farben in einen Binärcode bei Verwendung von drei Farben. Dies geschieht durch die Methode `decodeSignal(String[] colors)`:

Listing 4.3: Umwandlung Farben in einen Binärcode

```
private final String[] SigColorNames = {"orange","green","blue"};

private int getSigColorPos(String s){
  List<String> colList = Arrays.asList(SigColorNames);
  return colList.indexOf(s);}

String decodeSignal(String[] colors){
  int lastPos = 0;
  String result = "";
  for (String curColor : colors){
        int pos = getSigColorPos(curColor);
        int diff = (pos + SigColorNames.length - lastPos)
        lastPos = pos;
        int bin =  diff >> 1;
        result += bin;
  }
  return result;}
```

Erst wird die absolute Position der jetzigen Farbe ermittelt, welche in dem Feld `SigColorNames` vorgegeben ist. Dann wird der Abstand zur Position der vorherigen Farbe auf einer zyklischen Skala bestimmt, was bei drei Farben entweder eins oder zwei ist und zum Schluss wird durch eine bitweise Verschiebeoperation diese Differenz in Binärcode umgewandelt, hat also den Wert „0“ oder „1“.

Dieses Signal wird zwischengespeichert und die Signatur hiervon zum Server verschickt, nachdem von diesem eine Nachricht empfangen wurde, dass die Farbsequenz zu Ende ist.

4.3.2.4 Kommunikation mit Server

In `MTWUser` wird beim Starten der Kommunikation ein Thread `comThread` gestartet. Dieser dient zur Kommunikation mit dem Server und ist als eigener Thread konzipiert, damit dieser auf Nachrichten vom Server warten kann. Diese bestehen aus einer Zeichenkette, welches durch einen Zeilenumbruch abgeschlossen wird. Die folgenden Zeichenketten sind als gültige Server-Befehle definiert:

- „CamPosFound“

  Wenn genügend Binärbäume zur Bestimmung der Kameraposition gezeichnet

wurden und eine bestimmte Mindestgröße unterschritten wurde, d.h. die angezeigte Farbe wird durch das Handy komplett verdeckt, sendet der Server dieses Signal. Danach erfasst der `WorkingThread` alle Vorschaubilder als Teil einer durch Farben codierten Nachricht.

- „SigEnd“

  Das Anzeigen des geheimen Farbsignals auf der MTW ist vorbei, was durch diese Zeichenkette signalisiert wird. Von der `WorkingThread` aufgezeichneten und in einen Binärcode verwandelten Nachricht wird die Signatur zum Server zurückgeschickt.

- „NeedPubKey“

  Das mobile Gerät ist noch nicht beim Server registriert, dieser braucht zum verifizieren der zugeschickten Signatur noch den öffentlichen Schlüssel des mobilen Gerätes. Dies wird dem Server als Feld von Bytes zugeschickt.

- „AutSuccess“

  Der Server konnte mit der Signatur die Nachricht verifizieren, diese stimmt mit dem von der Kamera des mobilen Gerätes erfassten Signals überein. Wenn der Benutzer noch nicht beim Server registriert war, ist es dieser nun und hat eine eindeutige ID, welche der übertragenen Nachricht entspricht. Oder der Benutzer ist bereits registriert und ist erfolgreich an der MTW authentifiziert und kann dort mit seinen Zugriffsrechten interagieren.

- „AccessDenied“

  Der Server konnte mit der Signatur die Nachricht nicht verifizieren, der Authentifizierungsvorgang ist fehlgeschlagen.

4.3.2.5 Sicherheitsaspekte, Signatur der Nachricht

Bei dieser Anwendung ist es wichtig, dass wirklich nur das mobile Gerät authentifiziert wird, welches von der MTW das Geheimsignal aufgenommen hat.

Zum einen wird das Signal optisch geheim gehalten, da nur die Kamera dieses Gerätes die angezeigte Farbsequenz sehen kann, andere Benutzer haben hierauf keine Sicht. Zum anderen muss bei der Übertragung gewährleistet sein, dass keine andere Person die Nachricht, die über WLAN an den Server geschickt wird, abfängt, modifiziert

und versucht, sich selber damit authentifizieren zu lassen. Damit der Server weiß, dass ein bereits registriertes Gerät auch wirklich das ist, welches vorher registriert wurde, wird zum einen zusätzlich zum Geheimsignal die ID des Benutzers angehängt. Zum anderen wird von dieser zusammengesetzten Nachricht dann mit den privatem Schlüssel eine Signatur erstellt. Bei der Registrierung wurde der öffentliche Schlüssel des mobilen Gerätes mitübertragen, nur mit diesem kann mit Hilfe der Signatur die Nachricht verifiziert werden.

Da das Erzeugen eines Schlüsselpaares, bestehend aus öffentlichen und privaten Schlüssel, auf einem mobilen Gerät wie dem Dev Phone mindestens 20 Sekunden dauert, wurde das Schlüsselpaar vorher schon am PC erzeugt und auf dem mobilen Gerät in einer Schlüsseldatenbank vom Typ " ´.bks“ auf der SD-Speicherkarte gespeichert. Hierzu wurde das Programm „KeyTool IUI “ [22] verwendet, der Schlüssel hat eine Länge von 1024 Bit und ist vom Typ DSA. Diese Datenbank ist mit einem allgemeinen Passwort geschützt und zusätzlich benötigt man für den Zugriff auf den öffentlichen Schlüssel ein Kennwort und für den dazugehörigen privaten Schlüssel ein weiteres Passwort. Beim Programmstart muss diese Datenbank in ein `Keystore` Objekt geladen werden, was mit einer Dauer von ca. einer Sekunde eine schnellere Lösung darstellt. Aus diesem Objekt kann dann mit dem Kennwort und dem Passwort das erzeugte Schlüsselpaar geladen werden.

4.3.3 Aufbau MTW-Anwendung

Diese Applikation, die auf dem PC läuft, der auch die multi-touch Benutzerschnittstelle nutzt, ist als Server konzipiert. In der Hauptklasse `AuthServerThread` wird ein Thread gestartet, der fortlaufend auf eingehende Anfragen von Clients wartet, die authentifiziert werden sollen. Wie schon in 4.3 beschrieben, wird zur Registrierung eines Benutzers für das Strategiespiel eine Spielerfarbe angezeigt, die dann von dem mobilen Gerät erfasst und zurückgeschickt wird. Als nächstes werden kleiner werdende Binärbäume angezeigt, um die Kameraposition zu bestimmen, hiernach wird dann an der ermittelten Kameraposition ein Farbsignal dargestellt. Anschließend wird dem Server zuerst die signierte Nachricht und dann der öffentliche Schlüssel zugeschickt, womit dann verifiziert wird, ob die Nachricht vom richtigen Gerät stammt. Bei Erfolg wird ein neuer Benutzer mit der Spielerfarbe auf dem Server angelegt und dieser kann sich nun jederzeit für den Zugang zur MTW authentifizieren. Die Anzeige der Farbsignale

erfolgt auf Anwendungsebene. Um den Implementierungsaufwand möglichst gering zu halten, wird hierfür eine Schnittstelle definiert. Im Folgenden werden die einzelnen Teile der Applikation näher erläutert.

4.3.3.1 Speicherung eines Benutzers

Ein registrierter Benutzer wird durch die Klasse `AutUser` instanziiert. In dieser werden Werte für ID, Position und PublicKey gespeichert. Durch die ID kann der Benutzer eindeutig identifiziert werden, diese wurde ihm vorher geheim zugeschickt und ist öffentlich nicht bekannt. Die Position ist diejenige, die während des Authentifizierungsvorgangs ermittelt wurde und ist wie alle anderen Koordinaten auch als relative Bildschirmkoordinate gespeichert. Diese sind durch Gleitkommazahlen definiert und müssen im Bereich von null bis eins liegen. Dadurch ist man nicht an feste Bildschirmgrößen gebunden. Des weiteren wurde bei der Authentifizierung der öffentliche Schlüssel zugeschickt, womit verifiziert werden kann, ob eine Nachricht von diesem Benutzer signiert wurde. Bei erneuter Authentifizierung muss dieser Schlüssel nicht mehr zugeschickt werden und der Vorgang läuft schneller ab. Zusätzlich zu den gerade aufgeführten notwendigen Parametern kann noch der Parameter `info` mit Text belegt werden, um anwendungsspezifische Informationen wie z.B. die Spielerfarbe in dem Strategiespiel zu speichern.

4.3.3.2 Modellierung eines Farbsignals

Es werden im Folgenden verschiedene Klassen verwendet, um die verschiedenen Farbsignale zu modellieren:

- Rechtecke zur Benutzer-Farb-Zuweisung

  Am Anfang des Strategiespiels muss jedem Spieler eine Farbe zugewiesen werden. Informationen werden hierzu in der Klasse `QuadIDColors` gespeichert. Diese enthält Informationen zur Anzahl der verschiedenen Farben, die Namen der Farben als Zeichenketten, wo das zugehörige Rechteck gezeichnet wird und die Größe von einem Rechteck.

- Binärbaum

  Für eine Authentifizierung muss zuerst die Kameraposition bestimmt werden. Dies geschieht durch sequentielle Anzeige von Binärbaumen, die durch ein Rechteck repräsentiert werden, welches bei jeder Iteration abwechselnd horizontal oder

vertikal halbiert wird und somit die mögliche Fläche, wo die Kamera sein könnte, halbiert wird. Der nächste Binärbaum wird nun in dem Teil angezeigt, dessen Farbe vom mobilen Gerät erkannt wurde. Ein Binärbaum wird durch eine Instanz der Klasse `BinTree` implementiert. Bei Erstellung dieser wird folgender Konstruktor aufgerufen:

Listing 4.4: Konstruktor BinTree

```
public BinTree(Point2D.Double position, Point2D.Double size,
Color[] colors)
```

`position` gibt die Anfangsposition an, `size` die Anfangsgröße eines Binärbaums und `colors` gibt die beiden Farben des Binärbaumes an. Dies sind aus den schon in 4.3.2.3 erklärten Gründen die Farben Orange und Dodger-Blau.

Eine Methode `nextIteration(int color)` erzeugt die nächste Iteration des Binärbaums, dieser wird dann in der jeweiligen Hälfte angezeigt, welche als Wert für `color` übergeben wurde. Hierbei entspricht 1 = Orange und 2 = Blau, erstere ist immer links bzw. oben angeordnet.

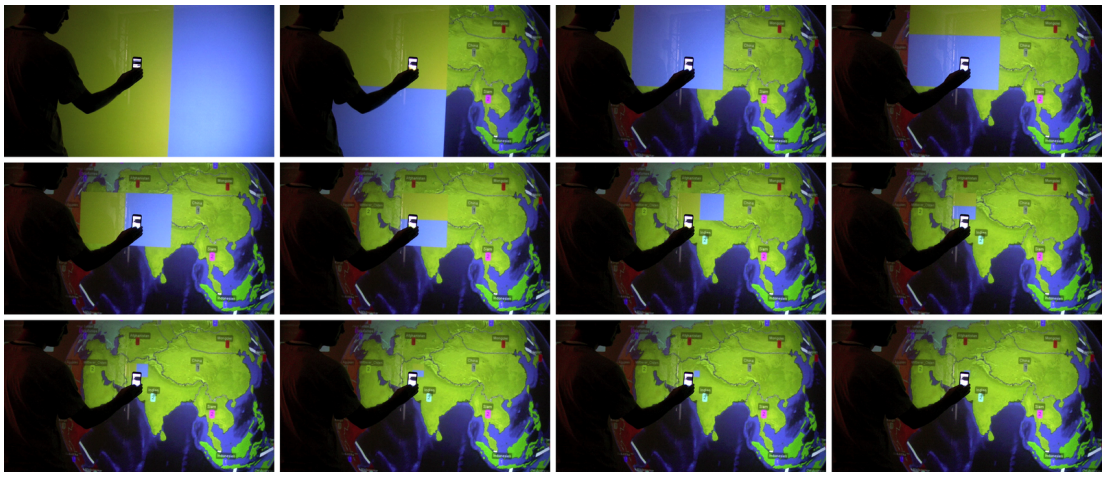

Abbildung 4.7: Ermittelung der Kameraposition durch Binärbäume. - Reihenfolge von oben links nach rechts unten, zuerst wird ein Binärbaum auf der kompletten multi-touch Oberfläche gezeichnet, dann solange in der vom Mobiltelefon erkannten Hälfte, bis eine Mindestgröße unterschritten wird [Eigene Abbildung].

- Quadbaum

  Analog zum zuvor erwähnten Binärbaum modelliert ein Objekt der Klasse `Quadtree`

einen Quadbaum, der auch zur Bestimmung der Kameraposition verwendet werden kann. Im Gegensatz zum Binärbaum ist dieser in vier gleich große Rechtecke unterteilt, die sich in der Mitte berühren, als Farben werden Orange, Tief-Pink, Grün und Dodger-Blau verwendet. Wenn eine dieser Farben erkannt wurde, wird analog zum Binärbaum durch die Methode `nextIteration(int color)` der nächste Quadbaum erzeugt und in der durch eine Farbe erkannten Subregion angezeigt. Bei der Umsetzung der Authentifizierung im Rahmen dieser Arbeit mit dem Android Dev Phone konnten keine Quadbäume verwendet werden, da die Erkennung von vier Farben nicht zuverlässig genug war. Aber dies könnte mit anderen Endgeräten für Android funktionieren, deswegen wurde der Quadbaum mit implementiert.

- Farbsignal

  Um sicherzustellen, dass sich die Kamera des mobilen Gerätes wirklich an einer bestimmten Position befindet, wird an dieser eine Sequenz aus Farben, bestehend aus Orange, Dodger-Blau und Grün, angezeigt. Diese wird durch die Klasse `AutSignal` instanziiert, welche die Position und Größe analog zum Binärbaum speichert und zusätzlich die Farbsequenz abspeichert, die dann angezeigt werden soll.

4.3.3.3 Erzeugung Farbsignal

Die Klasse `MTWColors` dient zur Speicherung der verwendeten Farben, Umwandlung dieser in Zeichenketten, der Erzeugung eines Binärcodes und der Umrechnung von diesem in Farbwerte. Um eine konsistente Verwendung der Farben zu gewährleisten, werden diese in einem Feld vom Typ `Color` gespeichert, weiterhin werden deren Namen in einem Feld als Zeichenkette gespeichert. In dieser Klasse werden sowohl die Farben und deren Reihenfolge für die Binär- und Quadbäume als auch für das Farbsignal gesetzt. Die Methode `createSignal()` wird für die Erzeugung eines Signals in Form einer Farbsequenz verwendet:

Listing 4.5: Erzeugung einer Farbsequenz

```
Color[] SigColors = {Color.orange, Color.green, dodgerBlue};
String[] SigColorNames = {"orange","green","blue"};
```

```
public Color[] createSignal(){
        int num = 0; int c = 0; sigString = "";
        Color[] result = new Color[sigSize];
        for ( int i = 0; i < sigSize; i++){
          int tmp = (int)Math.round(Math.random());
          num = num <<1 | tmp;

          int diff = 1 << tmp;
          c = c + diff;

          result[i] = SigColors[c\% SigColors.length];
          signal[i] = SigColorNames[c\% SigColorNames.length];
          sigString += tmp;
        }
        return result;}
```

Für die Länge der Farbsequenz wird hier eine Schleife durchlaufen, in der zuerst per Zufallsfunktion eine „0“ oder „1“ erzeugt wird. Diese Ziffer wird dem bisherigen Binärcode angehängt, nachdem dieser um eine Stelle nach links verschoben wurde. Dann wird die Farbdifferenz erzeugt, bei einer „0“ beträgt diese „1“, bei einer „1“ ist diese „2“. Zum Schluss werden sowohl der Farbwert als auch der Name abgespeichert.

4.3.3.4 Kommunikation mit mobilen Gerät

Wenn die Hauptklasse über einen Socket eine Verbindung zu einem Client aufgebaut hat, werden für die Kommunikation mit diesem ein In- und Outputstream erzeugt. Wenn schon alle Spielerfarben einem mobilen Gerät zugeordnet wurden, wird über die Methode `firstBinTree()` ein Binärbaum angezeigt. Anschließend wird laufend auf Nachrichten des Clienten gewartet, um entsprechend darauf zu reagieren und schließlich einen Benutzer zu authentifizieren. Beendet wird die Konversation mit dem Clienten erst, nachdem von diesem eine entsprechende Nachricht geschickt wurde. Im Folgenden werden die Nachrichten genauer beschrieben. Jede Nachricht besteht aus mindestens einem Wort, danach können noch weitere Informationen folgen.

- „RecCol“

  Von der Kamera des mobilen Gerätes wurde eine Farbe erkannt, diese ist die zweite Zeichenkette und hat entweder den Wert „orange“ oder „blue“. Je nach Modus wird nun die Farbe eines möglichen Spielers gesetzt oder ein Teil eines

Binärbaumes wurde erkannt und es wird die nächstkleinere Iteration gezeichnet. Wenn der nächste Binärbaum erzeugt wird, wird auch überprüft, ob dieser eine bestimmte Größe unterschreitet, ab der ein gezeichnetes Rechteck nicht mehr vor dem mobilen Gerät sichtbar ist, sondern hinter diesem verschwindet. Dann schickt der Server eine Nachricht an den Clienten, dass die Kameraposition gefunden wurde. Es wird ein Binärcode erzeugt, in eine Farbsequenz umgewandelt und angezeigt. Anschließend wird das mobile Gerät benachrichtigt, dass die Farbsequenz zu Ende ist.

- „RecSigSigned“

  Das mobile Gerät hat ein Farbsignal aufgezeichnet und in einen Binärcode umgewandelt. Danach wird die Signatur des Signals an den Server zurückgeschickt und stellt die zweite Zeichenkette der mit „RecSigSigned“ beginnenden Nachricht dar. Wenn der Benutzer bereits registriert ist, wird mit dem schon bekannten öffentlichen Schlüssel die Nachricht verifiziert. Danach wird eine Nachricht an den Clienten geschickt, ob die Authentifikation erfolgreich war. Ansonsten wird erst der öffentliche Schlüssel angefordert.

- „PublicKey“

  Es wird der öffentliche Schlüssel des mobilen Gerätes übertragen, dieser ist die zweite Zeichenkette der Nachricht. Hiermit wird analog wie schon zuvor die Signatur der Nachricht verifiziert und dem Clienten der Erfolg der Authentifikation mitgeteilt.

- „bye“

  Der Authentifizierungsvorgang ist beendet oder das mobile Gerät hat diesen vorzeitig beendet. Die Socketverbindung und der In- und Outputstream werden geschlossen und der Server wartet wieder auf eingehende Verbindungen.

- „LogOut“

  Der Benutzer hat sich manuell abgemeldet und über die Schnittstelle `SignalDrawer` wird der Benutzer in der Anwendung abgemeldet.

4.3.3.5 Schnittstelle zur Anwendung

Damit diese Authentifizierungsmethode möglichst generisch in Bezug auf eine mögliche MTW-Anwendung bleibt, reicht für die Authentifizierung die Verwendung der Schnittstelle `SignalDrawer` aus. Hierdurch wird gewährleistet, dass die farblichen Signale angezeigt werden und Anwender hinzugefügt oder entfernt werden können.

Listing 4.6: Schnittstelle SignalDrawer

```
public interface SignalDrawer {

        public void drawQuadtree(Quadtree qt);

        public void drawBinTree(BinTree bt);

        public void drawSignal(AutSignal autSig);

        public void addUser(UserInfo mtwUser);

        public boolean removeUser(UserInfo mtwUser);

        public Point2D.Double getTouchPosition();
}
```

Die Methoden `drawQuadtree` und `drawBinTree` sind für das Zeichnen eines Quadbaumes bzw. eines Binärbaumes zur Ermittelung der Kameraposition zuständig, `drawSignal` für die Anzeige eines Signals als Farbsequenz. Dabei wird nur eines der drei Elemente gleichzeitig gezeichnet, wenn ein anderes gezeichnet werden soll, müssen die anderen vorher mit null-Objekten aufgerufen werden. Dies sollte auch passieren, wenn gerade kein Benutzer authentifiziert wird, damit auf der Zeichenfläche der Anwendung keine der zuvor erwähnten farbigen Elemente angezeigt werden.

Die Methode `Point2D.Double getTouchPosition()` gibt die erste Position für die Bestimmung der Kameraposition zurück. Dies ist die Koordinate einer Fingerberührung mit der multi-touch Oberfläche, wenn der Benutzer sein mobiles Gerät davor hält. Wenn hierdurch die ungefähre Position bekannt ist, können am Anfang in einem lokalen Teil der Anzeige kleinere Binär- bzw. Quadbäume gezeichnet werden, was den Authentifizierungsvorgang beschleunigt.

Die Methode `void drawIDQuads(Color[] colors)` wird je nach Anwendung, die die Schnittstelle SignalDrawer implementiert, benötigt, um einen User durch eine Farbe

eindeutig zu identifizieren, was z.B. für das in 3.3.2 beschriebene Strategiespiel benötigt wird. Es werden dann mehrere farbige Flächen gezeichnet, und je nachdem, vor welcher eine Person ihr mobiles Gerät hält, bekommt dieser Benutzer dann diese Farbe zugewiesen.

Wenn der Anmeldevorgang für einen Benutzer erfolgreich war, wird er über die Methode `void addUser(UserInfo mtwUser)` hinzugefügt. Abgemeldet wird dieser über sein mobiles Gerät oder von der Authentifizierungsapplikation selber. Es wird dann die Methode `boolean removeUser(UserInfo mtwUser)` aufgerufen. Diese liefert zurück, ob der Benutzer erfolgreich abgemeldet werden konnte, da es sein kann, dass die Anwendung selber schon eine Logik für die automatische Abmeldung von Benutzern hat und dieser bereits abgemeldet sein könnte. Dies wäre z.B. das in 3.3.2 beschriebene Beenden einer Spielrunde eines Spielers, der danach keine spielrelevanten Interaktionen mehr machen kann.

4.3.3.6 Anforderung an die Anwendung

Eine Anwendung, die diesen Authentifizierungsansatz für die in 4.1 beschriebene MTW nutzen möchte, muss die Schnittstellen TuioListener und SignalDrawer implementieren. Über TUIO-Listener wird eine Instanz von der Klasse `TuioClient` eingebunden, die über das Netzwerk Nachrichten von einem Tuio-Server empfängt. Dieser kümmert sich um die Auswertung der aktuellen Berührungen mit der MTW, also der Benutzereingaben. SignalDrawer ist wie schon in 4.3.3.5 beschrieben für die Darstellung von Signalen und der Benutzerverwaltung zuständig. Mehr ist nicht erforderlich, es können sich dann Benutzer anmelden und eine multi-touch Benutzerschnittstelle für Eingaben nutzen.

In Rahmen dieser Arbeit wurde als Anwendung ein Prototyp erstellt, mit dem eine Authentifizierung für Risiko möglich ist. Ein Benutzer kann mit einer Farbe und einer ID registriert werden, anschließend kann er sich jederzeit an der MTW authentifizieren lassen. Dieses wird in der Klasse `Signalshow` realisiert.

4.3.4 Protokoll zur Authentifikation

Im Folgenden wird ein Überblick über einen erfolgreichen Authentifizierungsvorgang dargestellt. Als Partner sind hier das mobile Gerät als Client und der PC, der die multi-touch Oberfläche zur Ein- und Ausgabe nutzt, als Server beteiligt. Zur Vereinfachung wird hier von einem Authentfizierungsvorgang ausgegangen, bei dem der Server

den Client bereits registriert hat und seine ID und den öffentlichen Schlüssel kennt. Ansonsten werden diese Informationen verschickt, nachdem der Client das aufgenommene Signal in eine Nachricht umgewandelt hat, welche dann seine geheime ID darstellt. Der Server fragt dann nach dem öffentlichen Schlüssel und bekommt diesen vom Client zugeschickt. Anschließend erfolgt wie schon unten beschrieben die Verifizierung der Signatur.

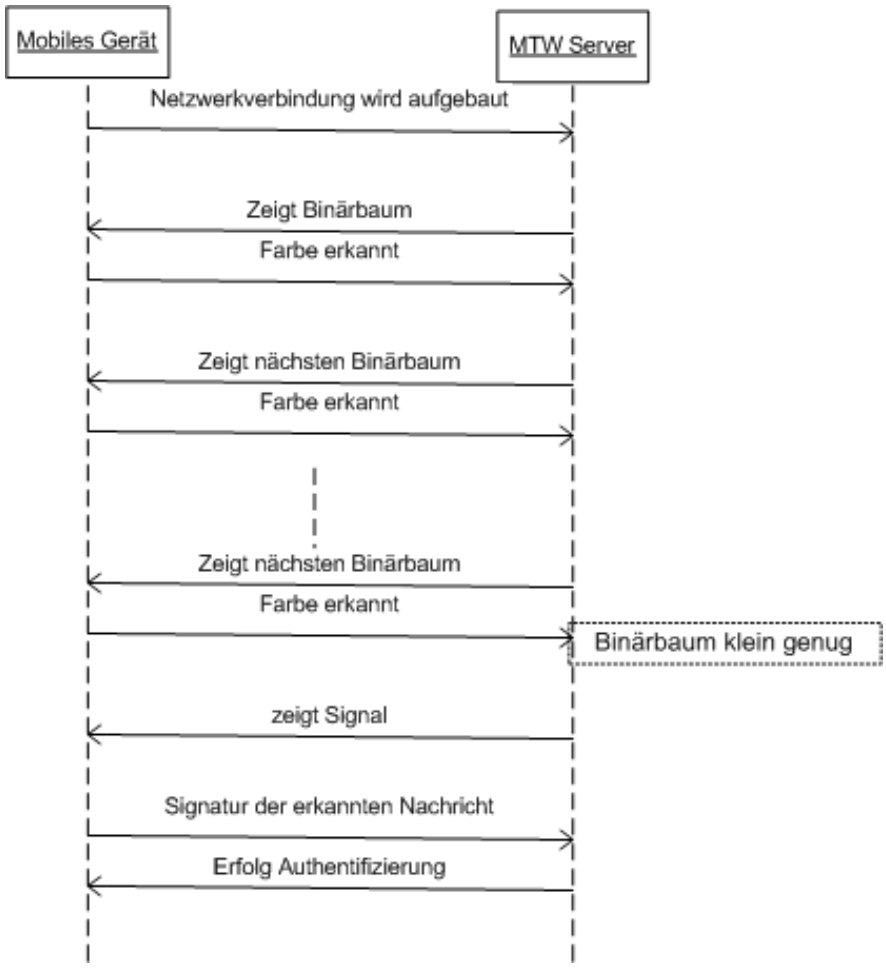

Abbildung 4.8: Ablauf einer Authentifizierung. - Bei dem hier beschriebenen Ablauf wird davon ausgegangen, dass der MTW-Server das mobile Gerät vorher schon registriert hat und schon dessen ID und öffentlichen Schlüssel hat [Eigene Abbildung].

5

Ergebnisse und Ausblick

5.1 Abschließende Diskussion

In diesem Abschnitt wird über die Ergebnisse und den Erfolg der beiden Authentifizierungsmethoden berichtet.

5.1.1 Fingerabdruckerkennung

Im Folgenden werden die Ergebnisse der Fingerabdruckerkennung dargestellt. Zuerst wurden Fingerabdrücke vom mit Silicon beschichteten Teil der in 4.2.1 beschriebenen multi-touch Oberfläche genommen. Hierdurch ließ sich keine Rillenstruktur eines Fingers erkennen. Als nächstes wurden auf den durchsichtigen Teil der Glasscheibe von derselben Person mindestens zehn Fingerabdrücke vom selben Finger, dem rechten Zeigefinger, genommen. Zusätzlich ist noch wichtig, dass genügend Merkmale extrahiert und gespeichert wurden, sonst ist der Fingerabdruck unbrauchbar für eine Verifikation. Dafür wird die durchschnittliche Größe der erkannten Merkmale angezeigt. In Tabelle 5.1 wird dargestellt, aus wie vielen Bildern VeriFinger erfolgreich einen Fingerabdruck erkannt hat.

Es wurde erstmal ein kleiner Stichprobenumfang von fünf Personen angelegt, trotzdem kann das Ergebnis schon als aussagekräftig gesehen werden. In 20% aller Versuche konnte auf einem Bild ein Fingerabdruck erkannt werden. Dies reicht bei weitem nicht aus, um diese Methode zur Authentifikation zu nutzen. Diese müsste im Prinzip bei jeder beliebigen Person funktionieren, was schon hier bei fünf Personen nicht oder fast nicht der Fall ist.

Person	erkannt	nicht erkannt	Merkmale (in KB)
1	1	9	6
2	3	9	5,7
3	4	9	2,5
4	1	12	0,4
5	3	8	3,6
Summe/Durchschnitt	12	47	2,44

Tabelle 5.1: Pro Person ist hier die Anzahl der aufgenommenen Bilder eines Fingers dargestellt, bei welchen die Erkennung als Fingerabdruck funktioniert hat und bei welchen nicht. Außerdem wird die Größe der von VeriFinger gespeicherten Merkmale pro Fingerabdruck dargestellt.

Selbst wenn ein Fingerabdruck erfolgreich erkannt wurde, bedeutet dies noch nicht, dass dieser zur Verifikation eingesetzt werden konnte. Allgemein lässt sich bei VeriFinger sagen, dass hierfür mindestens vier KB an extrahierten Merkmalen erforderlich sind. In vielen Fällen wurden nicht genug Merkmale erfasst, nur bei zwei Personen überhaupt im Durchschnitt mehr als dieser Grenzwert.

In [14] wird erwähnt, dass beim heutigen Stand der Fingerabdruckerkennung die Bildqualität von entscheidender Bedeutung ist, die nachfolgende Extraktion des Fingerabdrucks arbeitet zuverlässig, wenn dieser in ausreichender Qualität vorliegt.

In Bezug auf Kontrast und Beleuchtung, zwei wichtige Kriterien für die Qualität, unterscheiden sich die Bilder unwesentlich wegen gleichen Aufnahmebedingungen. Der Unterschied zwischen den verschiedenen Aufnahmen desselben Fingers liegt alleine in der neuen Ausrichtung des Finger für die nächste Aufnahme.

Wenn zwei Aufnahmen sich nur unwesentlich voneinander unterscheiden, so war es des Öfteren so, dass die eine erkannt wurde und die andere nicht, wie es anhand der Abbildungen 5.15.3 zu sehen ist. Dies ist ein Indikator der grenzwertigen Bildqualität, die hier wesentlich darüber entscheidet, ob VeriFinger einen Fingerabdruck auf einem Bild findet oder nicht. Weiterhin ist bei einer Erkennung meistens eine unvollständige Rillenstruktur erkannt worden 5.2.

Insgesamt lässt sich sagen, dass die Erkennungsraten und die Größe der erkannten Merkmale zu gering sind, um zuverlässig Personen an einer multi-touch Oberfläche

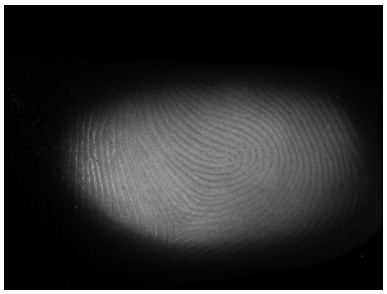

Abbildung 5.1: *Aufnahme 7 vom rechten Zeigefinger von Person 1 [Eigene Abbildung].*

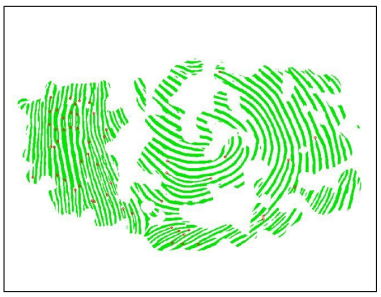

Abbildung 5.2: *Die erkannte Rillenstruktur von dieser Aufnahme [Screenshot aus VeriFinger].*

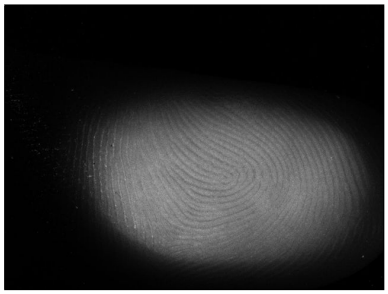

Abbildung 5.3: *Aufnahme 8, derselbe Finger und dieselbe Person. Es konnte kein Fingerabdruck erkannt werden [Eigene Abbildung].*

authentifizieren zu können. Deswegen wurde in diese Richtung nicht weiter geforscht, stattdessen wurde der Fokus auf die andere Methode in dieser Arbeit gelegt.

5.1.2 Mobiles Gerät

Die Authentifizierung mit dem mobilen Gerät ist zuverlässig und sicher. Durch die Reduktion auf maximal drei verwendete Farben werden diese gut erkannt, was das Hauptproblem bei der Umsetzung dieser Methode gewesen ist. Der ganze Vorgang dauert bei Vorwissen der Position des Mobiltelefons ungefähr vier Sekunden, was akzeptabel ist [23]. Hierbei muss eine Berührung mit der multi-touch Oberfläche in der Nähe des mobilen Gerätes existieren.

Die Sicherheit ist durch die geheime Aufnahme eines Farbsignals und der anschließenden Signierung dessen gegeben. Wenn die Nachricht als Farbsignal übertragen wird, wird diese auf einer Fläche angezeigt, die komplett vom mobilen Gerät verdeckt ist. Dadurch hat eine andere Person keine Möglichkeit, diese optisch zu erfassen, z.B. diese dann mit der eigenen Kamera abzufilmen. Des weiteren könnte jemand versuchen, die übertragene Nachricht abzufangen und versuchen, sich dann mit dieser auszugeben. Dies scheitert daran, dass erstens nur in der Subregion um das mobile Gerät, welches diese Nachricht abgefilmt hat, die neuen Zugriffsrechte auf die multi-touch Benutzerschnittstelle gewährt wird. Zweitens wird nur die Signatur verschickt, d.h. man kann hiermit nur die Geheimnachricht verifizieren, aber nicht ändern, da bei einem solchen

Versuch mit dem zugehörigen öffentlichen Schlüssel die Nachricht nicht verifiziert werden kann. Da die Nachricht aus dem Geheimsignal und der ID des Benutzers besteht, ist auch gewährleistet, dass diese Nachricht vom richtigen mobilen Gerät gesendet wurde. Somit ist man gegen IP-Spoofing geschützt. Ein Man-in-the-middle-Angriff ist zwecklos, da nur die Signatur über das WLAN verschickt wird. Wie schon zuvor erwähnt, macht jede Änderung diese ungültig. Es könnte höchstens die Nachricht mit der Signatur abgefangen werden und nicht weitergesendet werden, was aber nicht die Sicherheit der Anwendung, wofür die multi-touch Benutzerschnittstelle genutzt wird, gefährden würde.

5.2 Ausblick

In diesem Abschnitt wird dargestellt, was im Rahmen dieser Arbeit nicht realisiert werden konnte, aber Prinzipiell möglich ist oder sein kann und was nicht möglich ist.

5.2.1 Fingerabdrücke

Die Authentifizierung durch Fingerabdrücke, wie sie im Rahmen dieser Bachelor-Arbeit durchgeführt wurde, ist nicht brauchbar. Das Problem ist die geringe Bildqualität, der Kontrast spielt hier eine entscheidende Rolle, es konnten oft Fingerabdrücke nicht erkannt werden. Obwohl viele Fingerabdrucksensoren auf dem Prinzip von FTIR basieren, funktioniert dieses Prinzip bei einer multi-touch Oberfläche wesentlich schlechter. Eine Ursache könnte die geringere Auflösung, bedingt durch die Kamera, sein. Eine andere die Entfernung des Fingers zur Kamera und dem Umstand, dass nur reflektierte Infrarotstrahlung erfasst wird, dadurch lassen sich keine Bilder mit gutem Kontrast aufnehmen. Beide Ursachen lassen sich wegen des FTIR-Prinzips nicht ohne weiteres beheben.

5.2.2 Mobiles Gerät

Der ganze Authentifizierungsvorgang dauert von der Kamerapositionserkennung bis zum verifizieren der übertragenen Signatur ca. vier Sekunden. Dieser Zeitraum ist noch akzeptabel, für eine wirklich spontane Authentifikation könnte dieser noch gesenkt werden. Das Hauptproblem ist hierbei die zuverlässige Erkennung der Farben selber bzw. von mehr als den drei verwendeten Farben. Wenn dies für vier Farben geschieht, könnte

man zur Auffindung der Kameraposition Quadbäume verwenden, um diese Position im Vergleich zu Binärbäumen doppelt so schnell zu finden. Weiterhin könnte die Nachricht durch mehr Zeichen statt nur durch Binärcode codiert werden und wäre bei gleichem Inhalt kürzer.

Da die Applikation für das Handy-Betriebssystem Android programmiert wurde, ist sie leicht auf andere Endgeräte portierbar. Bald kommen für dieses Betriebssystem neue Endgeräte auf den Markt, es könnte dann sein, dass diese über eine verbesserte Kamera verfügen, die die Geschwindigkeit der Authentifikation steigert. Weiterhin könnte die nächste Version von Android über erweiterte Einstellungen für die Kamera verfügen, so dass z.B. der automatische Weißabgleich deaktiviert werden kann.

Alles in allem wurde eine zuverlässige Methode zur Benutzer-Authentifizierung an einer multi-touch Benutzerschnittstelle entwickelt, die noch Potential zur Verbesserung aufweist.

Referenzen

[1] Thomas Baudel and Michel Beaudouin-Lafon. **Charade: remote control of objects using free-hand gestures**. *Commun. ACM*, **36**(7):28–35, 1993. 2

[2] Roland Bless, Stefan Mink, Erik-Oliver Blass, Michael Conrad, Hans-Joachim Hofand Kendy Kutzner and Marcus Schöller. *Sichere Netzwerkkommunikation*. Springer Berlin Heidelberg, 2005. 8

[3] Kurt E. Brassel, Andrej Vckovski and Eberhard Schmitt. **Informatisierung der Geographie**. *Geographica Helvetica*, 1998. 3

[4] Bill Buxton. **Multi-Touch Systems that I Have Known and Loved**. 2009. 7

[5] Paul Dietz and Darren Leigh. **Diamond Touch: A Multi-User Technology**. In *UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology*. 14th annual ACM symposium on User interface software and technology, 2001. 10

[6] Alan Esenther and Kent Wittenburg. **Multi-user Multi-touch Games on DiamondTouch**. **Volume 3814/2005**:315–319, 2005. 10

[7] John Florence, Kathleen Hornsby and Max J. Egenhofer. **The GIS Wall-Board: Interactions with Spatial Information on Large-Scale Displays**. In *International Symposium on Spatial Data Handling*, 1996. 9, 13

[8] Bundesamt für Bevölkerungsschutz und Katastrophenhilfe. **Feuerwehr-Dienstvorschrift 100**. http://www.bbk.bund.de, 1999. 25

[9] Bundesministerium für Forschung und Bildung. **SoKNOS**. www.soknos.de, 2009. xi, 24

[10] Sven Furhmann, Alan M. MacEachren, Jianwei Dou, Ke Wang and Adrian Cox. **Gesture and Speech-Based Maps to Support Use of GIS for Crisis Management: A User Study**. 2005. 9, 13

[11] Jefferson Y. Han. **Low-cost multi-touch sensing through frustrated total internal reflection**. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, 2005. 9

[12] Ken Hinckley, Randy Pausch, Dennis Proffitt and Neal F. Kassell. **Two-handed virtual manipulation**. *ACM Trans. Comput.-Hum. Interact.*, **5**(3):260–302, 1998. 4

[13] Shahram Izadi, Steve Hodge, Stuart Taylo, Dan Rosenfeld, Nicolas Villar, Alex Butler and Jonathan Westhues. **Going beyond the display: a surface technology with an electronically switchable diffuser**. In *UIST '08: Proceedings of the 21st annual ACM symposium on User interface software and technology*, 2008. 12, 19

[14] Anil K. Jain, Patrick Flynn and Arun A. Ross. *Handbook of Biometrics*. Springer, 2008. 7, 56

[15] M. Kaltenbrunner, T. Bovermann, R. Bencina and E. Costanz. **TUIO: A Protocol for Table-Top Tangible User Interfaces**. 2005. 33

[16] Myron K. Krueger. *Artificial Reality*. 1983. 3

[17] Andrea Leganchuk, Shumin Zhai and William Buxton. **Manual and cognitive benefits of two-handed input: an experimental study**. *ACM Trans. Comput.-Hum. Interact.*, **5**(4), 1998. 2

[18] Macazin. **Apple iPhone 3G-Verkauf knackt bald die 12 Millionenmarke**. http://www.macazin.de/apple/apple-iphone-3g-verkauf-knackt-bald-die-12-millionenmarke/271454/, 2008. 2

[19] Alan M. MacEachren and Isaac Brewer. **Developing a cenceptual Framework for Visually-enabled**. *International Journal of Geographical Information*, 2004. 4, 9, 13

[20] Rene Mayrhofer, Mike Hazas and Hans Gellersen. **Shake well before use: authentication based on accelerometer data**. Technical report, Lancaster University, 2006. 12

[21] Kento Miyaoku, Suguru Higashino and Yoshinobu Tonomura. **C-blink: a hue-difference-based light signal marker for large screen interaction via any mobile terminal**. In *UIST '04: Proceedings of the 17th annual ACM symposium on User interface software and technology*, pages 147–156. ACM, 2004. 11, 12

[22] YellowCat Project. **KeyTool IUI**, 2009. 46

[23] Johannes Schöning. **Video zum Authentifizierungsvorgang**. http://www.youtube.com/watch?v=7NCm8WxrtWY, 2009. 57

[24] 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, 2008. x, 2

[25] Johannes Schöning, Michael Rohs and Antonio Krüger. **Spatial Authentication on Large Interactive Multi-Touch Surfaces**. In Schoening, editor, *Tabetop 2008: Adjunct Proceedings of IEEE Tabletops and Interactie Surfaces*, 2008. 12

[26] Johannes Schöning, Michael Rohs and Antonio Krüger. **Using Mobile Phones to Spontaneously Authenticate and Interact with Multi-Touch Surfaces**. In *AVI 2008: Workshop on designing multi-touch interaction techniques for coupled private and public displays*, May 2008. 11

[27] Ted Selker. **Touching the future**. *Commun. ACM*, **51**(12):14–16, 2008. 2

[28] NEURO technology. **VeriFinger**, 2009. 35

[29] Daniel Vogel and Ravin Balakrishnan. **Interactive public ambient displays: transitioning from implicit to explicit, public to personal, interaction with multiple users**. In *UIST '04: Proceedings of the 17th annual ACM symposium on User interface software and technology*, pages 137–146, 2004. 11, 13

[30] Andrew D. Wilson. **TouchLight: an imaging touch screen and display for gesture-based interaction**. In *ICMI '04: Proceedings of the 6th international conference on Multimodal interfaces*, 2004. 12, 13, 19

[31] Andrew D. Wilson and Raman Sarin. **BlueTable: connecting wireless mobile devices on interactive surfaces using vision-based handshaking**. In *GI '07: Proceedings of Graphics Interface 2007*, 2007. 11