Institut für Visualisierung und Interaktive Systeme

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Studienarbeit Nr. 2463

Interaktive Steuerung großer Displaywände mittels mobiler Geräte

Julian Falk

Studiengang:	Informatik
Prüfer/in:	Prof. Dr. Thomas Ertl
Betreuer/in:	Dipl.-Inf. Daniel Kauker
Beginn am:	29. Juli 2014
Beendet am:	29. Januar 2014
CR-Nummer:	B.4.2, I.4.1, I.4.6, I.4.7

Kurzfassung

Zur Visualisierung großer Datenmengen werden große Displaywände verwendet. Da die Nutzung von Tastatur und Maus in diesen Fällen nicht praktikabel ist, ist es notwendig, die Handhabung solcher Displaywände zu vereinfachen. In der vorliegenden Arbeit werden zwei Methoden vorgestellt, die mittels mobiler Geräte die Steuerung großer Displaywände erleichtern und verbessern. Bei der ersten Methode wird eine Fotografie eines Teilbereichs der Displaywand erstellt. Mit Hilfe von Bilderkennungstechniken wird aufgrund dieser Fotografie der passende Ausschnitt der Displaywand berechnet und anschließend werden die zugehörigen Daten des Modells auf das Mobilgerät übertragen. Die zweite Steuerungsmethode, die in diesem Dokument vorgestellt wird, nutzt die Bewegungssensoren des Mobilgeräts. Ausgehend vom ganzen Modell kann mit Hilfe der Bewegungssensoren die Perspektive auf das Modell geändert werden, indem die Position des Mobilgeräts verändert wird. Wird das Mobilgerät zum Beispiel um 90° um eine Achse gedreht, so ändert sich die Perspektive auf das Modell am Mobilgerät, während auf der Displaywand weiterhin das ursprüngliche Modell abgebildet ist.

Abstract

To visualize great amounts of data large display walls are used. Due to the fact that keyboard and mouse are not workable in this case it is necessary to simplify the handling of such display walls. In this work two methods are presented, which simplify and thus improve the controlling of large display walls by mobile device. The first method is to take a picture of a subarea of the large display wall. This picture is calculated with image recognition methods to identify the right image section of the display wall followed by the transfer of the associated data of the model to the mobile device. The second method presented in this work uses the movement sensors of the mobile device. Based on the entire model the point of view can be changed by using the movement sensors by changing the position of the mobile device. By turning the mobile device through 90 degrees, for instance the point of view changes on the mobile device while the display wall still shows the original picture.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Das generierte und konsumierte Datenvolumen steigt weltweit exponentiell an. Damit steigt auch die Nachfrage an Werkzeugen, die mit diesen Datenmengen umgehen können. Die Entwicklung solcher Werkzeuge kann mit der Datengenerierung nur schwer mithalten. Ein Beispiel dafür ist, dass eine handelsübliche Kamera Bildauflösungen aufnehmen kann, die die Fähigkeiten der Darstellung durch handelsübliche Displays weit übersteigen.

Für die Forschung ist es wichtig diese Daten auch visuell hochauflösend darstellen zu können, um eine effizientere Arbeitsweise zu ermöglichen. Damit dies erreicht werden kann, müssen Produkte genutzt werden, die den hohen Anforderungen gerecht werden. Ein Resultat dieser Entwicklung sind große Displaywände, die aus mehreren kleineren Displays zusammenstellt werden. Der Zusammenschluss dieser Produkte bringt neben der hohen Auflösung auch neue Herausforderungen mit sich. Die bekannten Steuerungsmöglichkeiten gelangen durch die vom Standard abweichenden Größenordnungen an ihre Grenzen.

Ziel dieser Arbeit ist es Methoden zu erläutern, die die interaktive Steuerung großer und hochauflösender Displaywände erleichtern. Speziell werden in dieser Arbeit zwei Methoden vorgestellt, wie mobile Geräte neue Steuerungsmöglichkeiten bieten können. Mit dem mobilen Gerät soll ein Teilbereich eines auf der Leinwand dargestellten Gesamtmodells lokalisiert und übertragen werden. Bei der ersten Methode wird mittels integriertem Fotosensor und Bilderkennungsmechanismen ein fotografierter Ausschnitt der großen Displaywand erkannt. Mit Hilfe dieser Information werden daraufhin die Daten des zugeordneten Bildausschnittes auf das Handy übertragen. Die zweite Methode beschreibt, wie anhand der Bewegungssensorik die Perspektive zum abgebildeten Modell entsprechend der Bewegung im Raum verändert wird.

Beide Methoden haben zum Ziel, dass auf dem Mobilgerät eine unabhängige reduzierte Sicht des Modellteils entsteht, die die lokale Weiterverarbeitung ohne Beeinträchtigung des Originals ermöglicht.

Die Anschaffung und Unterbringung der großen Displaywände ist mit einem hohen finanziellen Aufwand verbunden und erfordert überdies viel Platz. Dies führt dazu, dass nur wenige Institutionen solche hochauflösenden Displays besitzen. Jeder, der die menschliche Wahrnehmung effizienter ausschöpfen möchte um eine komfortablere Lösung seines Problems zu finden, muss auf eine der wenigen Displaywände zurückgreifen. Der Wunsch Teile der Ergebnisse für weitere Untersuchungen "mitzunehmen", liegt daher nicht fern. Darüber hinaus ist der Versuch mit einem normalen Maus-Cursor einen bestimmten Teilbereich in der hohen Auflösung auszuwählen denkbar schwierig.

Mit Hilfe eines Mobilgeräts, das durch seine geringe Größe und sein geringes Gewicht sehr transportabel ist, kann die erläuterte Problematik gelöst werden. Die Tatsache, dass nahezu jeder Mensch ein Mobilgerät mit großem Funktionalitätsumfang bei sich trägt, begünstigt den Gedanken diese als

Steuerungsgerät für die große Displaywand zu nutzen. Ebenso von Vorteil ist der schnelle Fortschritt in der Applikationsentwicklung, der die Entstehung von Anwendungen für Mobilgeräte zunehmend vereinfacht.

1.1 Verwandte Themen

Neben dem VISUS (Visualisierungsinstitut der Universität Stuttgart) besitzt auch die Hochschule Bonn/Rhein-Sieg in Sankt Augustin eine große Displaywand. Sie ist mit sieben mal drei Metern eine der größten derartigen Displaywände in Deutschland. Mit drei Computern und einem Cluster von zwölf Rechnern mit jeweils drei leistungsstarken Grafikkarten wird die Bildberechnung dafür vorgenommen. Die Steuerung der Displaywand erfolgt mit Hilfe von insgesamt sieben Tracking-Kameras. Mit verschiedenen Gesten steuern die Anwender die Displaywand ohne Eingabegeräte wie Maus und Tastatur, die für eine derartige Anzeigefläche nicht geeignet sind. Die Mitnahme von Informationen aus den angezeigten Daten durch das Erkennen dieser mit einem Smartphone ist nicht Bestandteil der Interaktionsmöglichkeiten. Im Grunde können die von Kameras erfassten Gesten mit denen auf einem Smartphone für die Navigation vorgesehenen verglichen werden. Ebenso gestaltet es sich schwierig, mehrere Benutzer gleichzeitig mit der Displaywand interagieren zu lassen. [OMF]
Auch die Technische Universität Dresden besitzt eine hochauflösende, interaktive Displaywand der Firma MultiTouch Ltd. aus Finnland. Sie ist mit einer Größe von fünf mal 2,5 Metern mit der Displaywand des VISUS vergleichbar. Sie zeichnet sich durch eine berührungsempfindliche Oberfläche aus, die mithilfe von Händen, Fingern und Stiften gesteuert werden kann. Auch bei den hier verfügbaren Interaktionsmöglichkeiten ist ein Portieren von Informationen nicht enthalten. [TU]
Cavens et al. beschreiben in ihrer Arbeit eine Möglichkeit große Displays mit Laserpointern zu steuern. Ihnen ist die Entwicklung einer präziseren und praktikableren Bedienung als der mit Maus und Tastatur gelungen. Dennoch treffen ihre Erkenntnisse nicht die Anforderungen dieser Arbeit. [CVFM02]

Die Androidanwendung, die dieser Arbeit als Basis dient, enthält bereits einfache Interaktionsmöglichkeiten mit der Displayleinwand. Dazu zählt unter anderem das Zoomen, Rotieren und Bewegen des angezeigten Modells auf der Displaywand. Dadurch sind die zugrundeliegenden Techniken zur Datenübertragung und eine Benutzeroberfläche schon vorhanden. Diese gilt es zu erweitern.

1.2 Überblick

Im Kapitel „Grundlagen“ werden die Begriffe der großen Displaywand und speziell der Powerwall definiert und erläutert. Außerdem werden mobile Geräte vorgestellt und deren Vorteile für die Steuerung der Powerwall dargestellt. Darüber hinaus wird der Begriff der Feature Detection erklärt. Im folgenden Kapitel „Ausschnitterkennung“ wird die erste Methode zur Vereinfachung und Verbesserung der Steuerung von großen Displaywänden vorgestellt. Darin werden verschiedene Bilderkennungsmechanismen kurz dargestellt und der verwendete genauer erläutert. Im Anschluss an das Kapitel wird die praktische Durchführung der Ausschnitterkennung vorgestellt. Das Kapitel „Unabhängige

Perspektivenänderung anhand der Bewegungssensoren“ behandelt die zweite Lösungsmethode für die Problematik der Handhabung großer Displaywände. Dazu werden zunächst die verwendeten Sensoren des mobilen Geräts veranschaulicht. Daraufhin wird die Berechnung der neuen Perspektive demonstriert. Anschließend wird auch hier die praktische Umsetzung der Methode vorgestellt.

Im letzten Kapitel „Zusammenfassung und Ausblick“ werden die Ergebnisse der Arbeit zusammengefasst und ein Ausblick auf die weitere Verbesserung und Vereinfachung in der Steuerung großer Displaywände gegeben.

2 Grundlagen

In diesem Kapitel werden die Grundbegriffe, die für die Durchführung dieser Arbeit notwendig sind, erklärt. Ebenso werden die technischen Grundlagen, die für die praktische Durchführung benutzt werden, erläutert.

2.1 Große Displaywände

Die Wahrnehmungskapazität des Menschen wird unter Verwendung bisher produzierter Displays nicht vollständig ausgelastet. Mit Displaywänden wird versucht, die Auflösung durch die Vielzahl der Geräte weiter zu vervielfachen und die Fähigkeiten des Menschen dadurch besser zu nutzen. Diese Powerwalls finden hauptsächlich Anwendung in der Visualisierung größerer Datenmengen, beispielsweise durch hochauflösende Fotografien oder komplexe Graphen. Diese Variante, die menschliche Wahrnehmung besser auszuschöpfen, ist sehr unflexibel und kostenintensiv.
Die gängigen Eingabegeräte wie Tastatur und Maus sind nur noch bedingt praktikabel. Der Mauszeiger muss durch die erhöhte Auflösung bei gleichbleibender Genauigkeit deutlich mehr Strecke zurücklegen. Entsprechend benötigt der Anwender eine ebenfalls größere Fläche zur Navigation oder eine deutlich angepasste Genauigkeit der Maus, die sich wiederum erschwerend auf die Bewegung der Maus auswirkt, weil der Mauszeiger sensibler auf die Bewegung der Maus durch den Menschen reagiert. Ebenso ist die Tastatureingabe - zum Beispiel bei der Auswahl einzelner Elemente in großen Datenmengen - mühsam.

2.2 Powerwall

Das VISUS (Visualisierungsinstitut der Universität Stuttgart) besitzt eine großflächige Projektionsleinwand, auf der hochauflösende Bilder auf einer Größe von 6 mal 2,25 Meter sowohl in 2D als auch in 3D projiziert werden können. Solche Projektionsleinwände, Powerwalls genannt, werden üblicherweise aus mehreren Standard Projektoren zusammengestellt, wobei die einzelnen Bildpunkte größer dargestellt werden, als bei herkömmlichen Geräten. Die menschliche Wahrnehmung kann jedoch viel mehr aufnehmen als diese recht grobe Ausflösung. Um die Möglichkeiten der Nutzung zu erweitern, muss die Auflösung verbessert werden. Vorallem bei sehr naher Betrachtung und interaktiven Anwendungen ist die bisherige Auflösung nicht ausreichend. Um dieses Problem zu beseitigen werden hochauflösende 4K-Projektoren eingesetzt, die bislang in Kinos Verwendung finden.
Wie auf der Abbildung 2.1 zu sehen ist, besteht die Powerwall aus der Projektionswand mit einer Pixelgröße von 0,56 mm, den Projektoren, die paarweise in fünf Streifen hochkant nebeneinander angeordnet sind, und der Rechnerinfrastruktur. Die paarweise Verwendung der Projektoren erlaubt

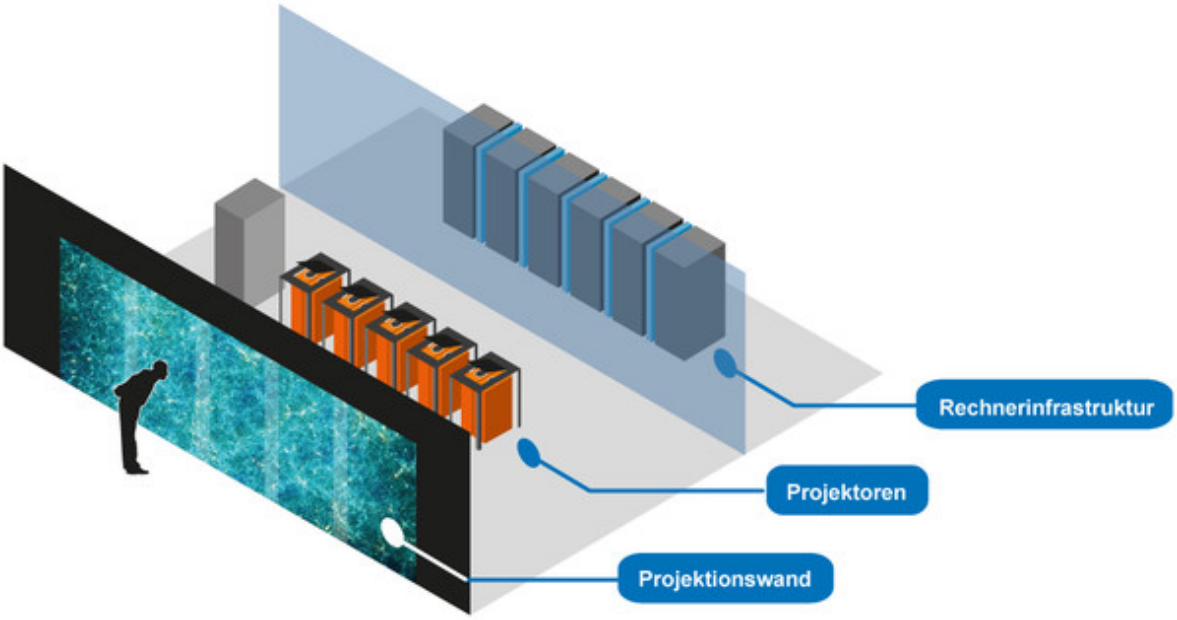

Abbildung 2.1: Der Aufbau der Powerwall des VISUS [Uni15]

die dreidimensionale Darstellung über die INFITEC-Stereo-Kanaltrennung. Jeder dieser Projektoren wird über die Rechnerinfrastruktur mit Bilddaten versorgt. Dafür ist ein Display-Cluster aus zehn Knoten vorhanden mit jeweils zwei Grafikkarten. Für die interaktive Computergrafik ist die Rechenkapazität jedoch noch nicht ausreichend, weswegen diese von weiteren 64 Render-Knoten unterstützt werden. Diese berechnen kleinere Bildausschnitte und senden die Information über ein Hochgeschwindigkeitsnetzwerk an die Display-Knoten. [Uni15]

Die Software zur Steuerung der Powerwall wurde eigens an der Universität Stuttgart entwickelt. Es handelt sich um die Grafikengine MegaMol.

2.3 Mobile Geräte

Mobile Endgeräte sind Geräte, die aufgrund ihres Gewichts und ihrer Größe kaum körperliche Anstrengungen verursachen. Im umgänglichen Sprachgebrauch sind mit mobilen Geräten zeitgemäße und handelsübliche Smartphones sowie Tablets gemeint, die im Alltag verbreitet sind. In dieser Arbeit sind diejenigen Smartphones und Tablets gemeint, die über Kamera und Bewegungssensor verfügen und eine Möglichkeit bieten, Applikationen zu installieren, mit denen auf die eingebauten Sensoren zugegriffen werden kann. Das verwendete Testgerät verwendet eine Version des Betriebssystems Android und verfügt über eine Kamera mit einer Auflösung von bis zu acht Megapixel.

2.4 Feature Detection

Die Feature Detection ist ein Teilbereich der Computer Vision. Ziel ist es, ein Bild zu unterteilen und bedeutungsvolle Bereiche (Features) darin zu finden. Das Auffinden dieser interessanten Punkte oder auch Points of Interest (siehe 2.4.1) übernimmt der Detektor. Das Speichern beziehungsweise das Beschreiben der jeweiligen Bereiche übernimmt der Deskriptor. Über die Beschreibung können dann per Matching die passenden Punkte miteinander verglichen werden. Das Matching ist der Hauptanwendungsbereich der Feature Detection und erstreckt sich von der industriellen Robotik über die Videospielbranche bis hin zu verschiedenen Sicherheitstechniken.

2.4.1 Points of Interests

Die bedeutsamen Stellen, die durch Detektoren der Feature Detection gefunden werden, heißen Points of Interests oder auch Interessenpunkte. Diese Punkte werden vom Detektor erkannt und vom Deskriptor mit einer Bedeutung versehen, um letztendlich durch ein Matching mit anderen Interessenpunkten verglichen zu werden. Das lässt sich mit mit den charakteristischen Merkmalen einer Iris oder eines Fingerabdrucks vergleichen, mit denen ein Sicherheitsabgleich durchgeführt werden kann.

2.5 Verwendete Software

Für die praktische Durchführung wird auf einige bereits programmierte Werkzeuge und Schnittstellen zurückgegriffen, um die Dauer der Durchführung zu verkürzen.

2.5.1 MegaMol

MegaMol ist eine vom Visualisierungsinstitut der Universität Stuttgart entwickelte und eingesetzte Software, die viele Methodiken der Computergrafik, wie zum Beispiel OpenGL Rendering, bereits implementiert hat. Sie ist in C++ realisiert und fungiert als Steuerungsschnittstelle der Powerwall. Im Zuge meiner Arbeit wird diese Software um die Möglichkeit erweitert, mit einem mobilen Endgerät zu kommunizieren und dieses als Steuergerät zu verwenden. Für die Bilderkennung wird hierbei das Open Source Computer Vision Framework OpenCV benutzt.

2.5.2 OpenCV

OpenCV (Open Source Computer Vision Library) ist eine Open Source basierte Software Bibliothek, die mehr als 2500 implementierte und optimierte Algorithmen der Computer Vision und dem maschinellen Lernen beinhaltet. Zusätzlich unterstützt OpenCV verschiedene Programmiersprachen, wie zum Beispiel C++, Java und Python. Diese Arbeit benutzt hauptsächlich die bereits implementierte Variante des SIFT und des SURF Algorithmus, die in Kapitel 3 genauer beschrieben sind.

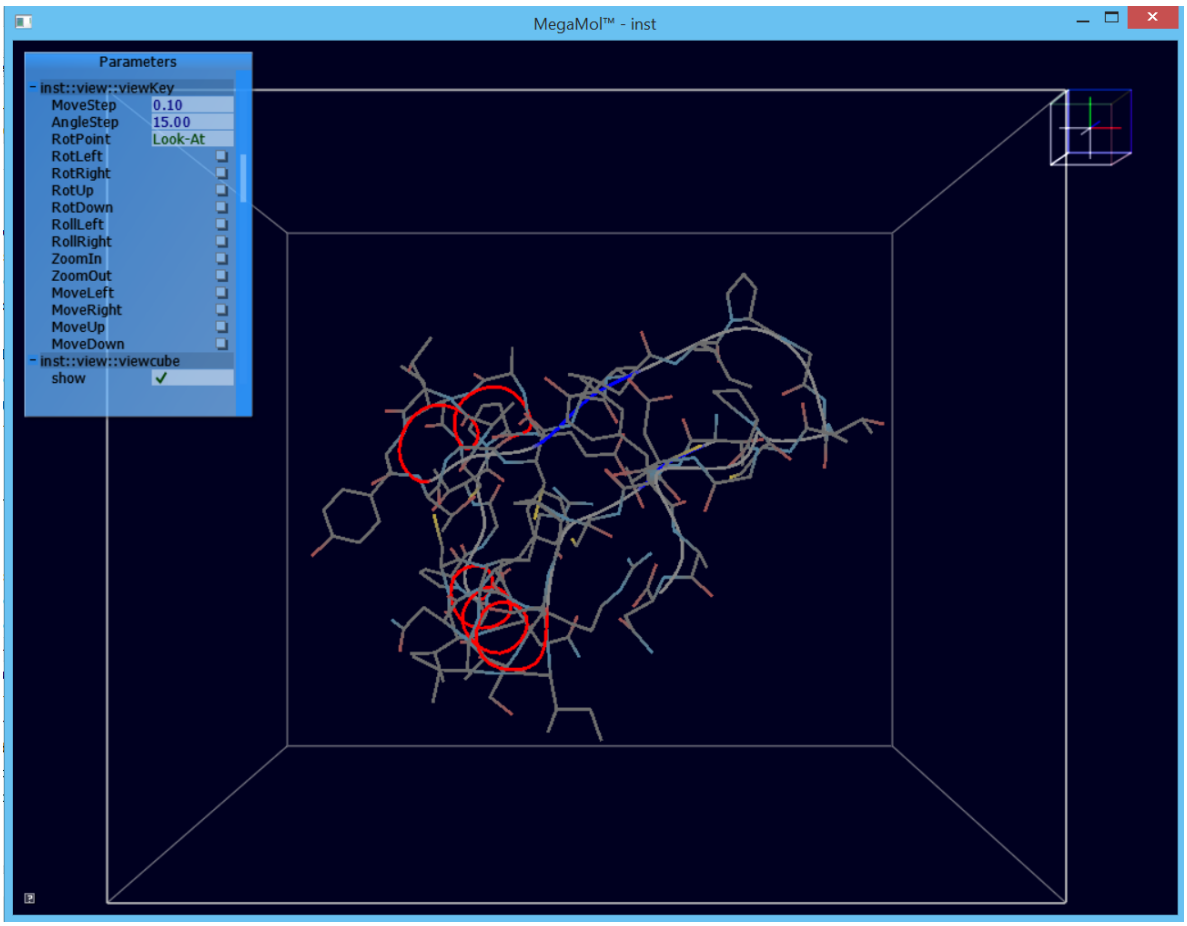

Abbildung 2.2: Screenshot der Software Megamol

3 Ausschnitterkennung

Mit Hilfe von Bilderkennung soll ermittelt werden, welcher Ausschnitt einer Darstellung auf einer Powerwall von einem mobilen Gerät fotografiert wurde. Im folgenden Kapitel werden die Vorraussetzungen erläutert und die zu Verfügung stehenden Algorithmen SIFT und SURF erläutert und getestet.

Für die Anbindung eines mobilen Gerätes an die Powerwall müssen vorher entsprechende programmatische Vorraussetzungen erfüllt sein, auf die hier nur oberflächlich eingegangen wird, weil sie für die praktische Umsetzung in dieser Arbeit größtenteils schon vorhanden waren. Es muss eine Sender-Empfänger-Mechanik auf beiden Seiten implementiert sein, um die fehlerfreie Übertragung von Bildmaterial zu bewerkstelligen. Des Weiteren wird über eine Kompression der Daten entschieden, falls das zu übermittelnde Bildmaterial zu groß ist. Außerdem wird entschieden, ob die vorhandenen Mittel, die benötigt werden, um das Modell auf einem mobilen Gerät anzuzeigen, ausreichend sind, oder ob Funktionalitäten hinzugefügt werden müssen. Weil es nicht gelungen ist, die gewünschten Ergebnisse bei der Bestimmung von Übereinstimmungen zwischen fotografiertem Bildausschnitt und Quellmaterial zu erzielen, wird stattdessen aufgezeigt, welche Möglichkeiten gegeben sind - weil die praktische Umsetzung vor diesem Punkt schon gescheitert war. Auf dem mobilen Gerät liegt eine von der Uni-Stuttgart entwickelte Applikation vor, die schon grundliegende Steuerungsmöglichkeiten, Kommunikation und Anzeigemöglichkeiten des Modells auf der Powerwall implementiert hat. Entsprechend liegen Kommunikationsmöglichkeiten seitens der Powerwall ebenfalls vor.

3.1 Netzwerkübertragung

Die Netzwerkübertragung wird in unserem Fall mit TCP-Sockets realisiert. TCP-Sockets sind eine plattformunabhängige Schnittstelle. Das ist relevant, weil die zugrundeliegenden Systeme des mobilen Gerätes (gängig Java, Objective C) und der Powerwall (hier C++) unterschiedlich sind. Im Zuge der Arbeit wird die Kommunikation mittels TCP-Sockets um eine Dateiübertragung erweitert, welche es ermöglicht, ein Foto direkt nach dem Erzeugen auf dem mobilen Gerät zum Rechnercluster der Powerwall zu schicken.

3.2 Bildkompression

Wenn das Bild des mobilen Gerätes an der Powerwall ankommt, muss das Bild mit einem aktuellen Screenshot der Powerwall verglichen werden. Die Übertragung darf nicht zu lange dauern, weil das fotografierte Segment im Screenshot sonst nicht mehr vorhanden sein könnte. Deshalb muss eine Bildkompression des Fotos vorgenommen werden. In unserem Fall haben wir mit einer eingebauten 8

Megapixel-Kamera Bilder mit einer Auflösung von 3264 × 2448 Pixeln fotografiert. Diese im unkomprimierten RGBA-Format zu speichern hieße, eine Dateigröße von 32 MB zu erreichen. Ursprünglich war eine Kompression in das JPEG-Format mit dem Qualitätsfaktor 60 vorgesehen, um das Foto auf eine Dateigröße zu reduzieren, welche innerhalb einer Sekunde übertragen werden kann. Weil es zu unzureichenden Ergebnissen bei der Ermittlung von Übereinstimmungen gekommen ist, wurde auf die Kompression der Daten verzichtet und die Veränderung des Modells während der Übertragung vermieden.

3.3 Auswahl des Bilderkennungsverfahrens

Das Bild des mobilen Geräts, welches ein Teilbereich des Modells auf der Powerwall darstellt, muss jetzt mit dem Screenshot der zeitgleich die Powerwall repräsentiert verglichen werden. Gängige Methoden lösen das durch das Erkennen markanter, lokaler Regionen (Features), die das Bild gut beschreiben sollen. Diesen Teil übernimmt der Detektor. Wurde dieser Schritt erfolgreich abgeschlossen, übernimmt der Deskriptor. Er soll die gefundenen Features in einer kompakten Form speichern, sodass sie für den Vergleich verwendet werden können. Das Vergleichen der jeweils gespeicherten Features übernimmt dann der Matcher. Dieser vergleicht die Features nach bestimmten Kriterien und gibt eine Menge der zusammenpassenden Features zurück.
Inspiriert von der Kantenerkennung entwickelten C. Harris und M. Stephens einen Eckdetektor, der Ecken als aussagekräftige Features erkennt. Dieser Ansatz ermöglicht es, Drehungen und Verschiebungen relativ zuverlässig zu erkennen und ist bei linearen Helligkeitsveränderungen stabil, weil sich der Gradiend, der für die Eckenerkennung untersucht wird, kaum ändert. [HS88] Diese Methode liefert gute Ergebnisse, sofern die Größe der zu suchenden Teilobjekte bekannt ist. Dies ist für unser Problem nicht ausreichend. David G. Lowe hat 2004 einen neuen Ansatz entwickelt, welcher das Bild in verschiedenen Skalierungen und Glättungen speichert. Daraus resultierend wird die Anzahl gefundener Bedeutungspunkte erhöht und eine Robustheit gegenüber Skalierung geschaffen. Damit sind die Anforderungen erfüllt und diese Methode wird die Grundlage unserer Steuerung.

3.4 Scale invariant feature transform (SIFT)

SIFT besteht nach Lowe aus folgenden vier Prozessstufen [Low04]:

- Scale-space Extremstellen Erkennung
- Genaue Keypoint Lokalisierung
- Ausrichtungsbestimmung
- Keypoint Deskriptor

Im folgenden wird SIFT genauer hinsichtlich dieser Prozessstufen beleuchtet.

3.4.1 Scale-space Extremstellen Erkennung

Die Erzeugung eines Space Scales dient als Basis. Das Bild wird durch mehrfaches Reduzieren der Samplerate in Ebenen unterteilt. Pro reduzierter Abtastrate entsteht eine Okatave, wie Abbildung 3.1 zu sehen ist. Innerhalb jeder Oktave wird bei gleichbleibender Abtastrate mit dem Gaußfilter

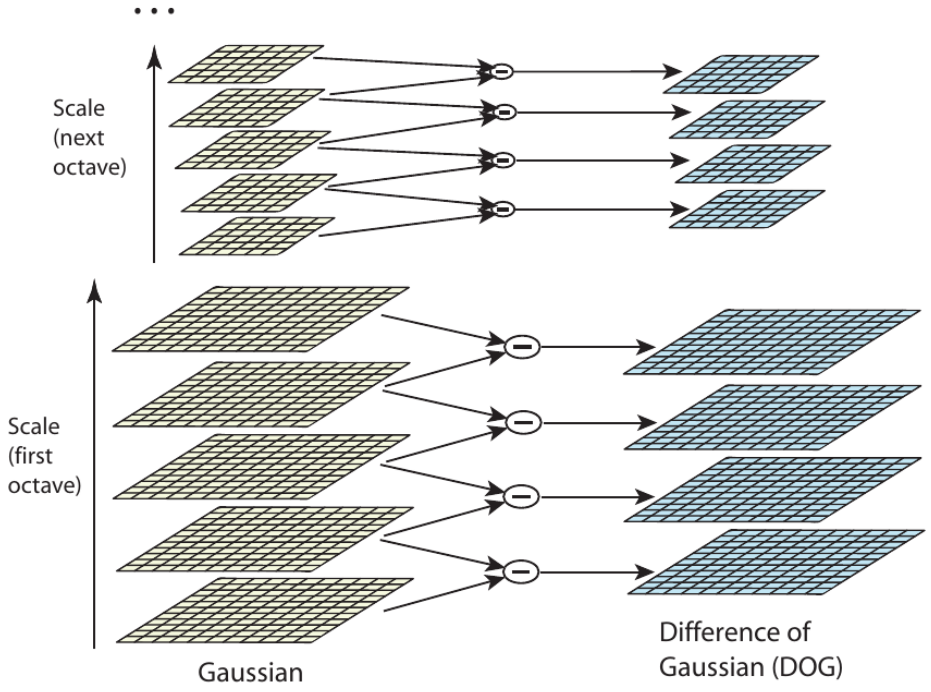

Abbildung 3.1: Skizze des Scale Spaces nach Lowe [Low04]

(Gleichung 3.1) das Bild k-fach geglättet. Die aufeinanderfolgenden Resultate werden voneinander subtrahiert (siehe).

(3.1)

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$$

Wobei σ die Reichweite des Filters angibt.
D. Lowe erwähnt in seiner Arbeit, dass $\sigma = \sqrt{2}$ für jedes seiner getesteten Bilder ausreichend gute Ergebnisse erzielt hat.

Nach dem Erzeugen des Scale-Spaces wird die Nachbarschaft auf Extrema untersucht. Das heißt, für einen Punkt werden die acht Punkte auf der gleichen Ebene um ihn herum sowie die neun über und unter ihm auf den benachbarten Glättungsebenen betrachtet.

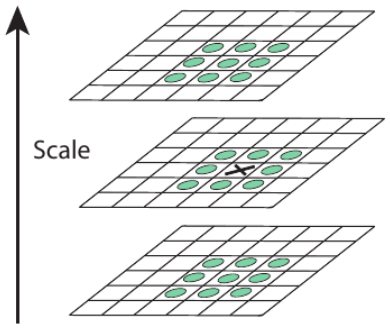

Abbildung 3.2: Veranschaulichung der Nachbarschaft eines Interessenpunkts [Low04]

3.4.2 Genaue Keypoint Lokalisierung

Bei der Untersuchung auf Extremstellen können schlecht geeignete Punkte auftreten. D. Lowe beschreibt eine starke Trefferwarscheinlichkeit von Kanten durch die Differenz der geglätteten Bilder. Dies wird mit einer Annäherung der Extremwerte anhand der Taylorreihe gelöst, womit diese Punkte ausgeschlossen werden können. Zusätzlich werden dadurch auch Keypoints ausgeschloßen, die in Gebieten gefunden werden, welche eigentlich durch ihren niedrigen Kontrast uninteressant wären.

3.4.3 Ausrichtungsbestimmung

Die Invarianz gegenüber Drehungen wird durch die Bestimmung der Orientierung jeder Extremstelle erreicht. Diese wird dadurch bestimmt, indem der Gradient um den Extrempunkt betrachtet wird und für jeden Nachbarn die Orientierung des Gradienten in ein Histogramm eingetragen wird. Der höchste Ausschlag des Histogramms gibt die Ausrichtung der Extremstelle vor.

3.4.4 Keypoint Descriptor

Der Teil der Erkennung (Detection) der Keypoints ist mit den bisher genannten Teilschritten abgeschlossen. Darauf folgend wird der Schritt des Beschreibens und Speicherns eines Keypoints (Description) geklärt. Basierend auf dem Ansatz der Orientierungsbestimmung wird bei der Beschreibung die Gradientorientierung für die 16x16 (ein von D. Lowe empirisch ermittelter Wert) Nachbarschaft berechnet. Dafür werden 4x4 Gradientenorientierungen zusammengefasst und in einem Histogramm hinterlegt, dass die gesammten 360° in 8 Bereiche gliedert. Daraus resultiert ein Vektor mit 4x4x8=128 Dimensionen, genannt SIFT Descriptor.

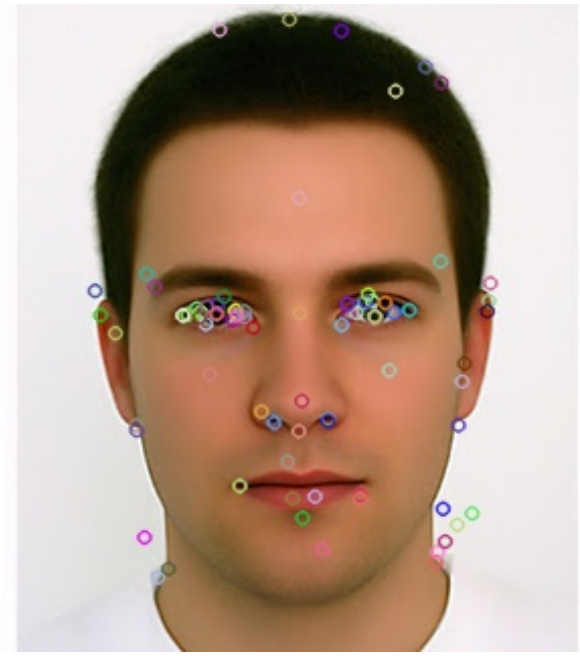

Abbildung 3.3: Gefundene Keypoints mit dem OpenCV Sift Detektor

3.4.5 Matching

Mit den bisherigen beschriebenen Techniken ist es möglich, von einem einzelnen Bild Keypoints zu bestimmen. Der nächste Schritt wäre, die Keypoints zweier mit SIFT analysierten Bilder zu vergleichen und die passenden Zusammenhänge zu finden. D. Lowe berechnet dazu die euklidische Distanz jedes Deskriptors. Er definiert, dass die nächsten Nachbarn eines Keypoints diejenigen sind die sich mit ihrer berechneten euklidischen Distanz am nächsten bei ihm befinden. Dadurch können die Distanzen der Nachbarn zweier Punkte miteinander verglichen werden. Sind die Differenzen zu groß werde sie nicht als Paar eliminiert.

Das Opensource Framework OpenCV (Open Computer Vision) enthält eine bereits implementierte Version von SIFT. Aus diesem Grund wird in dieser Arbeit die praktische Durchführung auf die Benutzung des Frameworks reduziert.

3.5 Speeded up robust features (SURF)

2006 verfassten Bay et al. eine Arbeit über eine schnellere Art der Merkmalserkennung und Beschreibung genannt SURF (Speeded up robust features). Sie beschreiben eine 3-5 fache Verschnellerung der Methode - durch Änderungen im SIFT Algorithmus. [BTG06]

In dieser Arbeit wird bei der praktischen Ausführung zum Teil auf diese Variante eingegangen, weil bei der Implementierung der Variante des OpenCV Frameworks Probleme aufgetreten sind.

SURF unterscheidet sich in folgenden Punkten von SIFT:

- Für das Finden von Points of Interests verwendet SURF einen Detektor basierend auf einer Hesse-Matrix.
- Anstelle des Gaußfilters wird ein Mittelwertfilter eingesetzt
- Der Deskriptor besteht aus 64 Dimensionen anstatt 128

Panchal et al. sprechen in ihrer Gegenüberstellung von SIFT und SURF von einer annähernd guten Performanz bei deutlicher Erhöhung der Geschwindigkeit. [PMP13]

3.6 Ermittelung des Bildausschnittes

Das nächste Ziel ist es, zu ermitteln, welcher Bereich fotografiert wurde. Dazu müssen die Eckpunkte des Bildes auf die Leinwandkoordinaten transformiert werden. Naheliegend wäre hier, die homografische Matrix zu finden. Für die Bestimmung der Matrix werden acht korrekte Trefferpaare benötigt. An diesem Punkt scheiterte der Praxisversuch. Wie im Bild zu sehen ist, gibt es im Vergleich zu wenig Übereinstimmungen, um zuverlässig eine Homographische Matrix zu bestimmen.

3.7 Praktische Durchführung

Die praktische Durchführung wurde in C++ am Clienten und in Java am Androidgerät entwickelt. Da die mobilen Geräte nur über begrenzte Rechenleistung verfügen, werden die Bilderkennungsmethoden auf dem Servercluster berechnet. Die Javaentwicklung im Zuge dieser Arbeit beschränkt sich auf die Benutzung des Kamerasensors und die Übertragung des Fotos zum Server. Wie bereits in Kapitel 3.1 erwähnt, wurde die Bildübertragung mit TCP - Sockets realisiert. Aufgrund von Schwierigkeiten bei der Implementierung wird nachträglich von der Komprimierung der Fotodatei abgesehen. Dies sorgt für eine längere Übertragungsdauer. Die Fotodaten sind im RBGA Format, bei dem jeder Pixel 4 Bytes Speicherplatz belegt. Jeweils 1 Byte für den Rot-, Blau und Grünwert und einen zusätzlichen Byte für die Transparenz. Ziel ist sowohl den Screenshot als auch das empfangene Foto möglichst im Hauptspeicher direkt zu verwenden, um den Datenmüll durch gespeicherte Bilddateien zu vermeiden und Rechenzeit zu sparen. Der Screenshot wird mittels OpenGL-Methode vom Framebuffer der Grafikkarte ausgelesen und in den Hauptspeicher kopiert. Nachdem der Screenshot im Hauptspeicher angekommen ist und somit beide Bilder zur Weiterverarbeitung zur Verfügung stehen, startet der Bilderkennungsmechanismus. Zuerst muss der Detektor, wie bereits in Kapitel 3.4 beschrieben, die Keypoints erkennen und beschreiben. Zum Vergleich der beiden Detektoren (SIFT beziehungsweise SURF) analysieren beide Methoden dasselbe Bild. Das Ergebnis ist in Abbildung 3.4 zu sehen.

Die SIFT-Variante findet hier, entgegen der Erwartungen, im Vergleich mit dem SURF-Algorithmus deutlich weniger Interessenpunkte. Um die Möglichkeit zu klären, ob die SIFT-Variante zwar weniger Interessenpunkte findet, diese qualitativ aber besser sind, wird für den nächsten Test ein Teilbild aus dem Bild entnommen. Beide werden von den jeweiligen Detektoren auf Keypoints analysiert. Die Interessenpunkte von Teilbild und Ursprungsbild werden einem Matcher übergeben. Damit die Ergebnisse aussagekräftig sind, muss dieser Matcher der gleiche sein.

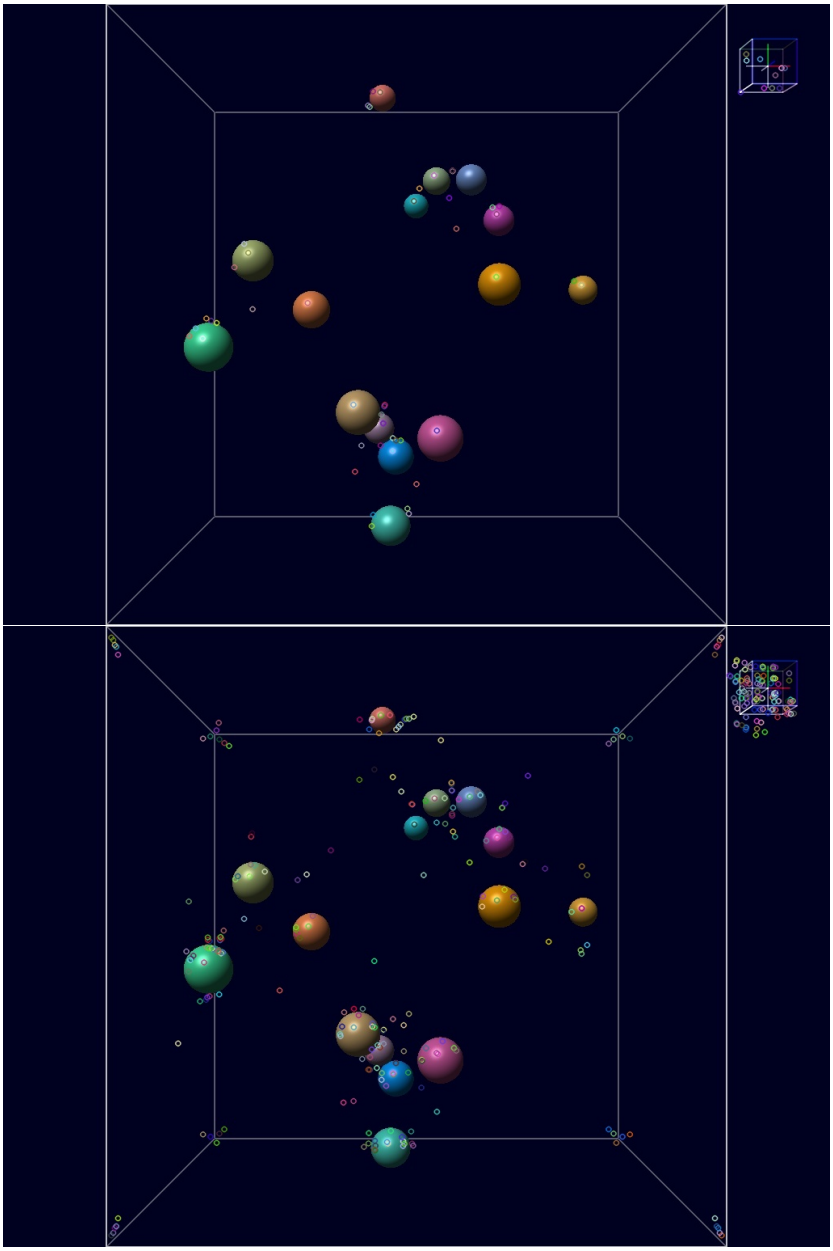

Abbildung 3.4: OpenCV SIFT und SURF-Detektor Keypoints bei einer Moleküldarstellung auf der Powerwall

Die Abbildung 3.5 zeigt eine höhere Anzahl gefundener Übereinstimmungen, wie an der Anzahl paralleler Linien zu sehen ist. Zu Anschaulichkeitszwecken wurde für den Test ein männliches Gesicht gewählt, da hier deutlicher zu erkennen ist, wo das Teilbild seinen Ursprung hat. Als Ergebnis dieses Tests wird für den weiteren Verlauf der SURF Algorithmus verwendet. Sind die Matchingpaare gefunden, muss der Bildausschnitt auf dem Screenshot gefunden werden. Dazu muss die homographische Matrix ermittelt werden, wie bereits in Kapitel 3.6 erklärt. Um diese zu finden, werden die Matching-

paare verwendet, wobei falsche Paare die Transformation verfälschen. Daher gilt es möglichst viele der fälschlich zusammengestellten Matchingpaare vorher auszuschließen. Im vorliegenden Fall muss eine Eigenschaft der Features gefunden werden, die auch bei perspektivischen Änderungen stabil ist. Das heißt, dass Gemeinsamkeiten wie die Distanz der Punkte im Bild oder der Winkel der Linien keine zuverlässige Aussagen treffen können, sobald das Teilbild zum Beispiel um 90° gedreht ist. In unserem Fall wird auf die euklidische Distanz des Deskriptorvektors zurückgegriffen. Übersteigt die Differenz der Distanzen zweier Vektoren einen gewissen Wert, werden sie als falsches Paar erkannt und eliminiert. Damit kann überprüft werden, ob die ermittelte homografische Matrix korrekt ist. Mit ihrer Hilfe werden die Eckpunkte des Teilbildes auf das Ursprungsbild transformiert und durch Linien verbunden. Diese Vorgehensweise soll bei erfolgreicher Berechnung der homografischen Matrix genau den Teilbereich des Bildes eingrenzen, der erkannt werden soll.

In Abbildung 3.6 sind beide Varianten am Bild eines Schmetterlings mit einer hohen Anzahl bedeutsamer Stellen getestet worden, um sicherzugehen, dass ausreichend korrekte Paare vorhanden sind.

Die Bilderkennung funktioniert für den optimalen Fall, wie Abbildung 3.6 zeigt. In unserem Anwendungsfall sind die Bedingungen erschwert und das System scheitert an diesen, wie Abbildung 3.7 gut zu sehen ist. Die Verfeinerung des Systems, um auch schlechteren Bedingungen standhalten zu können, wird in Kapitel 5.1 behandelt.

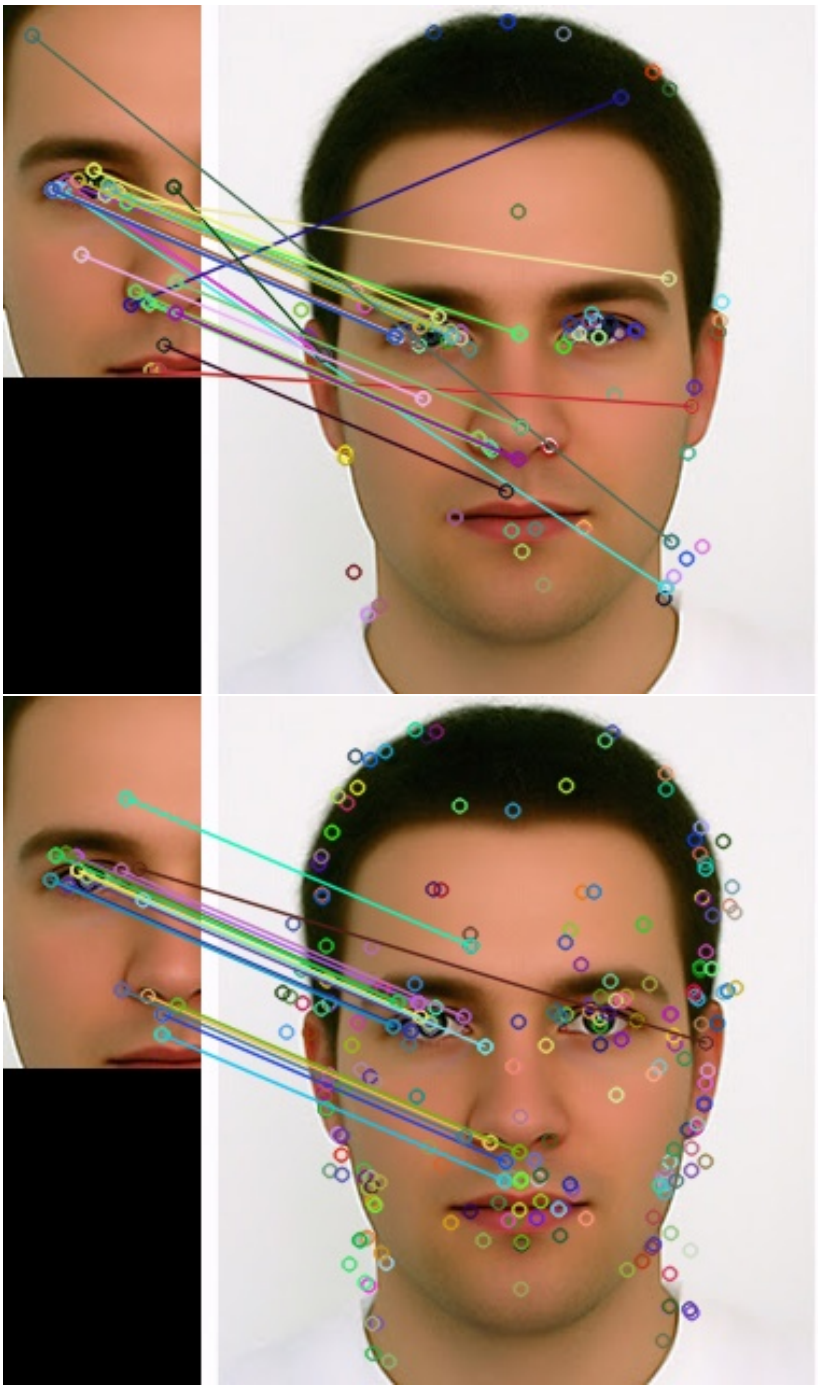

Abbildung 3.5: OpenCV SIFT und SURF-Keypointmatching mit einem Teilbild ohne Transformation aus dem Orginal entnommen. Die Kreise verdeutlichen die Keypoints und die Linien die ausgewählten Matchingpaare.

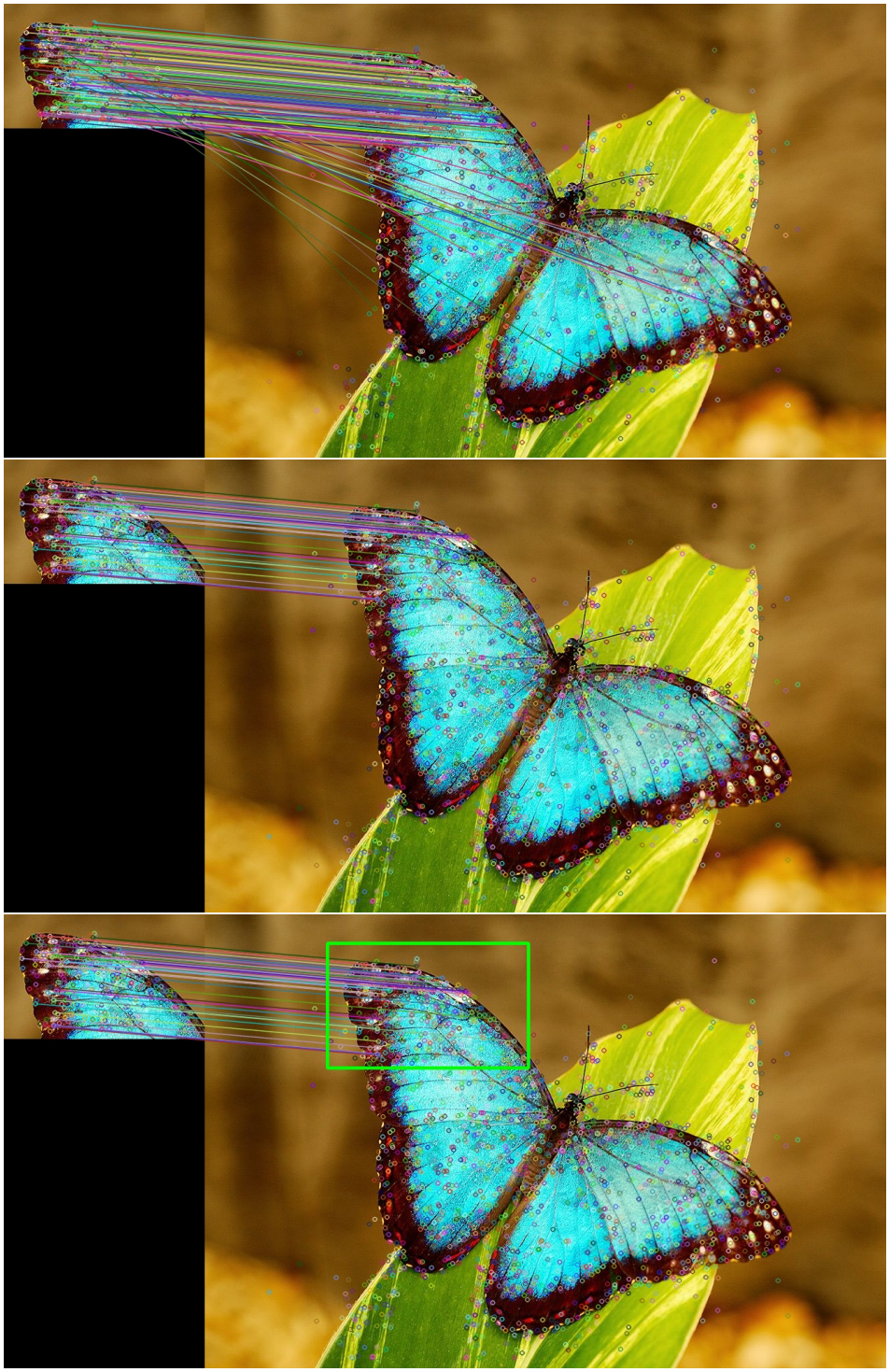

Abbildung 3.6: OpenCV SURF Keypointmatching. Oben dargestellt sind alle ermittelten Matchingpaare. Mittig sind die schlechten Paare eliminiert und unten sind die Teilbildgrenzen in grün eingezeichnet, die mit der errechneten homografischen Matrix bestimmt wurden.

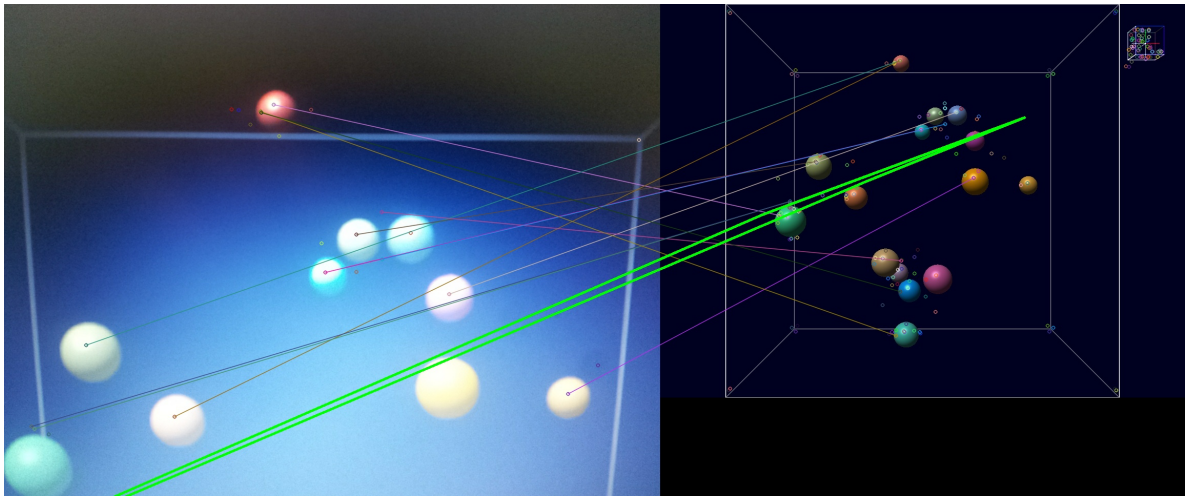

Abbildung 3.7: Objekterkennung mit zu vielen schlechten Paaren. Es wird eine fehlerhafte Homografische Matrix berechnet, was durch die fehlerhaft eingezeichneten grünen Randlinien gezeigt wird.

4 Unabhängige Perspektivenveränderung anhand der Bewegungssensoren

Der heutige Produktstandard von mobilen Geräten beinhaltet verschiedene Bewegungssensoren. Die dort üblichen Programmiersprachen Java, C-Sharp und objective C bieten einfache Schnittstellen, die eine schnelle Implementierung ermöglichen. Damit wird es möglich, diese Sensoren als Steuerung zur Änderung der Perspektive zu verwenden. Der Gyroskopsensor und der lineare Beschleunigungssensor bieten sich an, um die Bewegungen die im Raum gemacht werden aufzunehmen und die Perspektive auf das Modell der Bewegung im Raum anzupassen. Da die mehrere Benutzer auf die Powerwall schauen, ist es nicht sinnvoll die Änderung an der Powerwall selbst vorzunehmen, sondern nur an der Repräsentation des Modells auf dem mobilen Gerät.
In unserem Fall wird die Android / Java Seite beispielhaft genauer erläutert. Die praktische Umsetzung unterscheidet sich in den anderen Platformen nur unwesentlich. [Goo15]

4.1 Softwarearchitektur der Sensoren

Das Sensorframework besteht aus vier Objekten, die für das Benutzen der Sensoren notwendig sind:

- Das Sensor-Objekt beeinhaltet Informationen über seinen Typ.
- Das Sensorevent-Objekt enthält die Sensordaten, welcher Sensor es erzeugt hat, den Entstehungszeitpunkt und die Genauigkeit der Daten.
- Das Sensoreventlistener-Objekt bestimmt, ob die Daten beim Ändern der Genauigkeit oder der Sensordaten an sich abgefangen werden sollen.
- Das Sensormanager-Objekt verbindet Sensoren und Listener. Sie bietet Methoden, um Sensoren zu registrieren und sie zu kalibrieren.

4.2 Auswahl der Sensoren

Ziel der Steuerung soll es sein, dass der Benutzer sich mit dem mobilen Gerät um die Powerwall bewegt und sich die Perspektive des Modells sich entsprechend der Bewegung im Raum auf dem

Display des Tablets anpasst. Für die Bewegung entlang der Koordinatenachsen wäre der Lineare Beschleunigungssensor geeignet. Er misst die Beschleunigung des Gerätes entlang den Achsen, welche als Translation der Perspektive interpretiert werden kann. Das Gyroskop misst die Drehung entlang den Achsen und eignet sich für die Berechnung der Rotation der Perspektive im Raum.

4.2.1 Linearer Beschleunigungssensor

Der Lineare Beschleunigungssensor ist ein synthetischer Sensor, da er den Hardware-basierten Beschleunigungssensor benutzt und die Erdanziehung auf Softwarebasis ausgleicht. Er misst die Beschleunigung, mit der das Mobilgerät in x-,y- und z-Achsenrichtung bewegt wird und gibt die Werte in der Einheit m/s² zurück. Die Ausrichtung

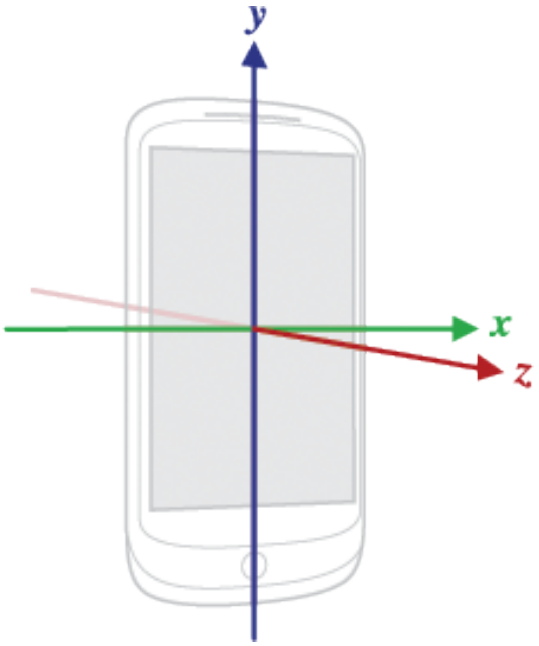

Abbildung 4.1: Zeigt das Koordinatensystem des mobilen Geräts und die Richtungen in denen die linearen Beschleunigungen gemessen werden. [Goo15]

4.2.2 Gyroskop

Das Gyroskop ist ein Messgerät mit dem Fluchtkräfte einer Rotation gemessen werden können. In den gängigen mobilen Geräten ist ein Gyroskop verbaut, das die Kräfte um die x-, y- und z-Achse misst. Der Messwert wird in rad/s zurückgegeben.

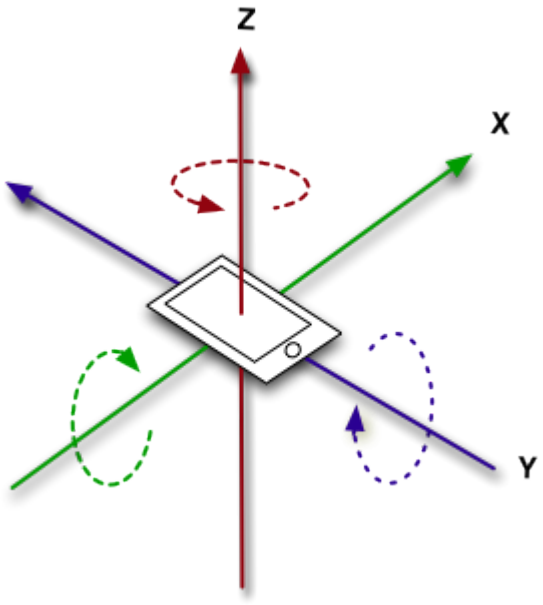

Abbildung 4.2: Zeigt das Koordinatensystem des mobilen Geräts und die Richtungen in denen die Rotationen gemessen werden. [Goo15]

4.3 Berechnung der neuen Perspektive

Für die Berechnung der Szene gibt es 3 Transformationsmatrizen: Die Model- , die View- und die Projectionmatrix. Modelmatrix transformiert die Objektkoordinaten in das Weltkoordinatensystem. Die Viewmatrix positioniert die Welkoordinaten aus der Sicht des Betrachters und die Projektionsmatrix transformiert die Szene so, dass die Perspektive mit den Eigenschaften der Kamera übereinstimmt. Die Matrizen werden miteinander multipliziert und ergeben die Szene. Gängig ist auch die zusammenfassung der Model- und Viewmatrix zur Model-Viewmatrix (OpenGL).

$$\mathbf{MVP} = \mathbf{M} \cdot \mathbf{V} \cdot \mathbf{P}$$

Wobei

- **M** Modelmatrix
- **V** Viewmatrix
- **P** Projectionmatrix
- **MV** Model-Viewmatrix
- **MVP** Model-View-Projectionmatrix

4.4 Übertragung

Damit auf dem mobilen Gerät auch die gleichen perspektivischen Einstellungen muss die MVP Matrix übermittelt werden. Um die Sichtkoordinaten unabhängig zu verändern, müsste der Client die MVP-Matrix mit der inversen Projectionmatrix multiplizieren. Deshalb werden MV- und P-Matrix getrennt übermittelt. Und auf dem Clienten verrechnet. So kann die MV- Matrix ohne großen Aufwand verändert werden.

4.5 Transformationsmatrix

Die Werte, die der Sensormanager aufnimmt werden innerhalb einer Transformationsmatrix T verrechnet, die für jedes Sensorevent mit der Modelview-Matrix multipliziert wird.

$$\mathbf{MV}_{neu} = \mathbf{MV}_{alt} \cdot \mathbf{T}$$

4.5.1 Translation

Die lineare Bewegung ist wie folgt zu berechnen:

Da die Sensoren die Beschleunigung messen, muss sie vorher erst in Gleichung 4.1 für $a(t)$ eingesetzt werden, um den zurückgelegten Weg zu berechnen. Die Zeit t ist die Zeit zwischen den zwei Messungen. Sie muss bei jeder Messung neu ermittelt werden.

(4.1)

$$s(t) = \ddot{a}(t) = \dot{v}(t) * t + v_0 = a(t) * \frac{t^2}{2} + v_0 * t + s_0, mit\ s_0 = 0$$

Zusätzlich wird die Anfangsgescheindigkeit v_0 berechnet. Diese ist die Endgeschwindigkeit der letzten Messung, berechnet wie in Gleichung 4.2.

(4.2)

$$v(t) = \dot{a}(t) = a(t) * t + v_0$$

Diese sind die Ausgangswerte die in die Matrix 4.3 als x,y und z gesetzt werden, welche dann die Transformationsmatrix der Translation ergibt.

(4.3)

$$\mathbf{T} = \begin{Bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{Bmatrix}$$

Wobei

- x ist der berechnete Weg in x-Achsenrichtung
- y ist der berechnete Weg in y-Achsenrichtung
- z ist der berechnete Weg in z-Achsenrichtung

4.5.2 Rotation

Die Rotation wird wie folgt berechnet:

Rotation um die X-Achse:

(4.4)
$$\mathbf{R}_x = \begin{Bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(x) & -sin(x) & 0 \\ 0 & sin(x) & cos(x) & 0 \\ 0 & 0 & 0 & 1 \end{Bmatrix}$$

Rotation um die Y-Achse:

(4.5)
$$\mathbf{R}_y = \begin{Bmatrix} cos(y) & 0 & sin(y) & 0 \\ 0 & 1 & 0 & 0 \\ -sin(y) & 0 & cos(x) & 0 \\ 0 & 0 & 0 & 1 \end{Bmatrix}$$

Rotation um die Z-Achse:

(4.6)
$$\mathbf{R}_z = \begin{Bmatrix} cos(z) & -sin(z) & 0 & 0 \\ -sin(z) & cos(z) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{Bmatrix}$$

Wobei x,y,z die gemessenen Werte des Gyroskops sind.

Da die Entfernungen in die einzelnen Achsenrichtungen pro Sensorevent sehr klein sind, kann die Frage nach der Reihenfolge der Multiplikation hinweggesehen werden und wie folgt ausgerechnet werden:

$$\mathbf{T}_R = \mathbf{R}_x \cdot \mathbf{R}_y \cdot \mathbf{R}_z$$

Die Rotation der Modelview Matrix bedeutet, dass sich die Kameraperspektive um das zu drehende Objekt verändert. Dies hat zur Folge, dass sobald sich das zu sehende Objekt nichtmehr im Mittelpunkt befindet. Daher muss das Model zuerst ins Koordinatenzentrum zurück transformiert werden, um es dort zu drehen. Dannach wird es wieder zurück an die Ausgangsposition geschoben. Dies hat den Effekt, dass wenn das Objekt vorher schon gedreht wurde, dennoch in die Drehrichtung entsprechend der Bewegung des Geräts entspricht. Bei einer Drehung um 180° wären diese sonst Spiegelverkehrt und somit unintuitiv.

4.6 Probleme bei der Durchführung

Jeder Sensor hat ein Rauschverhalten, dass in unserem Fall dafür sorgt, dass das Modell langsam "wegschwebt". Das heißt, es entsteht ein Fehler der sich aufsummiert und das Modell aus dem Sichtbereich verschiebt. Dies wäre durch eine Kalibrierung der Sensoren vermeidbar. Ebenso ist darauf zu achten, dass sich x- und y- Achsen vertauschen, sobald das Mobile Gerät in der vertikalen Ansicht verwendet wird, wie es in der MegaMol-Applikation der Fall ist.

5 Zusammenfassung und Ausblick

Die Anforderungen an diese Arbeit ist Methoden zu konzipieren, die eine flexible Handhabung der visualisierten Daten ermöglichen, gleichzeitig mehrere Benutzer mit ihr interagieren können und die Steuerung im Vergleich zur Tastatur und Maus vereinfachen. In dieser Arbeit wurde präsentiert, wie dies mit der Verwendung handelsüblicher mobiler Geräte realisiert werden kann. Dazu soll das Mobilgerät einen Teilbereich eines Modells, das auf einer großen Displayleinwand dargestellt wird, auswählen können. Dieser Bereich des Modells soll in reduzierter Form zum Mobilgerät übertragen werden, damit unabhängig von der Leinwand auf dem Mobilgerät weitergearbeitet werden kann. Diese Möglichkeit wird in zwei Varianten dargestellt.
Die Steuerung großer Displayleinwände wird durch die Entwicklung beider Methoden komfortabler. Die Arbeit zeigt, dass es möglich ist, mobile Geräte zur Steuerung großer Displaywände zu nutzen. Die Methode Teilbereiche des Modells über Feature Detection zu erkennen, hat bisher keine Erleichterung gebracht. Das konzipierte System zeigt noch Schwächen bei geringerer Verschlechterung der Bedingungen. Schon der Test eines perspektivisch veränderten Teilbilds führt zu einer fehlerhaften Erkennung des Objekts. Das fotografierte Bild und der Screenshot weichen so stark voneinander ab, dass SIFT und SURF nicht ausreichend passende Gemeinsamkeiten finden können.
Die Variante dagegen, über Bewegungssensoren den gewünschten Teilbereich zu ërlaufen", erzielt gute Ergebnisse. Sie ist in der Theorie nicht so praktisch wie der Bilderkennungsansatz, dafür ist er deutlich einfacher in der Implementierung und in der Berechnung, wodurch ein schnelleres Ergebnis erreicht wird.

Ausblick

Beide Ansätze haben noch Probleme in ihrer Durchführung. Der Bilderkennungsansatz hat in der Ausführung nur schlechte Ergebnisse erzielt. Die schlechten Ergebnisse hängen von mehreren Faktoren ab, an denen in zukünftiger Arbeit weiterentwickelt werden kann. Mit entsprechender Verbesserung der Faktoren ist es denkbar diese Methode mit guten Ergebnissen praktisch umzusetzen. Ein Faktor, der weiterentwickelt werden kann, ist die Kalibrierung der Helligkeit und der Farbwerte mit der Powerwall. Dadurch kann eine höhere Übereinstimmung im Matchingprozess erreicht werden. Ein weiterer Faktor ist die zuverlässigere Eliminierung falscher Matchingpaare. Die euklidische Distanz ist eine recht zuverlässige Eigenschaft, aber noch nicht zu hundert Prozent zuverlässig. Ein weiterer Faktor ist die Auswahl der Matchingpaare zur Bestimmung der homografischen Matrix. OpenCV nutzt für die Auswahl ein Medianverfahren, über das im Zuge dieser Arbeit keine Tests gemacht wurden. Denkbar ist, dass ein spezielleres Auswahlverfahren für bestimmte Fälle ebenfalls eine Verbesserung bringen kann. Des Weiteren führt der technologische Fortschritt zu Verbesserungen an den Kamerasensoren, was ebenfalls zu einer stetigen Verbesserung der Bedingungen führt.

Dem Bewegungssensoransatz fehlt eine zuverlässige Variante, um dem Rauschen der Sensoren entgegenzuwirken. Für zukünftige Arbeiten kann dies mit einer vorherigen Kalibrierung verhindert werden. Der Sensor wird bewegungslos eine bestimmte Zeit gemessen, die Sensorwerte werden gemittelt und bei jeder Berechnung subtrahiert. Dies entspricht dem Fotografieren mit geschlossener Blende, um das Grundrauschen des Kamerasensors zu ermitteln.

Literaturverzeichnis

[BTG06] H. Bay, T. Tuytelaars, L. V. Gool. SURF: Speeded Up Robust Features. Technischer Bericht, ETH Zurich, Katholieke Universiteit Leuven, 2006. (Zitiert auf Seite 19)

[CVFM02] D. Cavens, F. Vogt, S. Fels, M. Meitner. Interacting with the Big Screen: Pointers to Ponder. In *CHI '02 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '02, S. 678–679. ACM, New York, NY, USA, 2002. doi:10.1145/506443.506542. URL http://doi.acm.org/10.1145/506443.506542. (Zitiert auf Seite 8)

[Goo15] Google. Sensors Overview | Android Developer, 2015. URL http://developer.android.com/guide/topics/sensors/sensors_overview.html. (Zitiert auf den Seiten 6, 27, 28 und 29)

[HS88] C. Harris, M. Stephens. A combined Edge and Corner Detector. Technischer Bericht, Plessey Research Manor, United Kingdom, 1988. (Zitiert auf Seite 16)

[Low04] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. Technischer Bericht, Computer Science Departmant University of British Columbia, 2004. (Zitiert auf den Seiten 6, 16, 17 und 18)

[OMF] P. Odrich, A. Mörer-Funk. (Zitiert auf Seite 8)

[PMP13] S. K. S. P. M. Panchal, S. R. Panchal. Comparison of SIFT and SURF. Technischer Bericht, University of Baroda, India, 2013. (Zitiert auf Seite 20)

[TU] TU Dresden. (Zitiert auf Seite 8)

[Uni15] Universität Stuttgart. Technischer Aufbau: Visualisierungsinstitut Stuttgart, 2015. URL http://www.visus.uni-stuttgart.de/institut/visualisierungslabor/technischer-aufbau.html. (Zitiert auf den Seiten 6 und 12)

Alle URLs wurden zuletzt am 17. 01. 2015 geprüft.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift