Bauhaus-Universität Weimar
Fakultät Medien
Studiengang Medieninformatik

Konfigurierbare und generische Applikationsinfrastrukturen für 3D-Multi-User-Multi-Display-Umgebungen in der Virtuellen Realität

Bachelorarbeit

Tim Weißker
geboren am 29.04.1993 in Erlangen

Matrikelnummer 110940

1. Gutachter: Prof. Dr. Bernd Fröhlich
2. Gutachter: Junior-Prof. Dr. Florian Echtler

Datum der Abgabe: 30. September 2014

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, dass alle Stellen der Arbeit, die wörtlich oder sinngemäß aus anderen Quellen übernommen wurden, als solche kenntlich gemacht und dass die Arbeit in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegt wurde.

Ort, Datum Tim Weißker

Zusammenfassung

Die interaktive Exploration und Manipulation virtueller Inhalte in einer Gruppe mehrerer individueller Nutzer benötigt *Displays*, welche es ermöglichen, visuelle Rückmeldung über die getätigten Aktionen und Manipulationen zu geben. Oftmals reicht dabei die Darstellungsfläche eines einzigen Displays nicht aus, um alle gewünschten Blickwinkel auf eine simulierte Szene geeignet darzustellen, weshalb zunehmend mehrere Displays in interaktiven *Multi-Display-Umgebungen* integriert werden. In unserem Labor beispielsweise dienen immersive 3D-Leinwand- und 3D-Tabletop-Displays zur perspektivisch korrekten Darstellung virtueller Welten für mehrere Nutzer.

Ziel dieser Arbeit ist es, eine konfigurierbare und generische Applikationsinfrastruktur für solche Multi-Display-Umgebungen im Kontext der Virtuellen Realität zu schaffen. Neben Kriterien der technischen Realisierbarkeit werden dafür zwei Implementierungsstrategien aufgezeigt. Innerhalb des ersten Ansatzes werden alle Displays eines Interaktionsbereichs räumlich kontinuierlich abgebildet, was die physikalische Anordnung der Displays in der virtuellen Welt exakt wiedergibt. Die Alternative ist die standardmäßige separate Abbildung, so dass auf jedem Display ohne zusätzlichen Aufwand unterschiedliche Navigationsinhalte dargestellt werden können.

Beide Implementierungsstrategien werden hinsichtlich ihrer Machbarkeit bezüglich eines definierten Anforderungskatalogs diskutiert. Es zeigt sich, dass der kontinuierlichen Abbildung ein einfacheres mentales Modell zu Grunde liegt, sich jedoch einige der Anforderungen nur durch explizite Eingriffe in die Struktur realisieren lassen. Die separate Abbildung hingegen eröffnet ein breiteres Spektrum an Möglichkeiten, liegt jedoch in ihrer Komplexität deutlich über dem ersten Ansatz.

Danksagung

Ich möchte mich ganz herzlich bei all denjenigen bedanken, welche mich während meines Studiums und bei der Anfertigung dieser Bachelorarbeit motiviert und unterstützt haben.

Insbesondere gilt dieser Dank *André Kunert*, welcher im vergangenen Wintersemester mein Interesse für die Virtuelle Realität geweckt und nun diese Abschlussarbeit betreut hat. Dank seiner Expertise konnte er stets Rückfragen bei Programmierarbeiten oder der Reinschrift beantworten und somit meine Arbeit kontinuierlich unterstützen. Weiterhin danke ich *Alexander Kulik* für seine kritischen Rückfragen und Literaturhinweise, welche meine Arbeit zusätzlich bereichern konnten.

Auch bedanke ich mich bei *Joshua Reibert* und *Ann-Kathrin Tietje*, welche als mit dem Thema vertraute und fachfremde Freunde viel Zeit in die Korrektur meiner Arbeit investiert haben und so einige Inhalt-, Komma- und Rechtschreibfehler eliminieren konnten.

Nicht zuletzt gebührt meinen Großeltern *Helga und Herbert Wagner* und meiner Mutter *Tanja Weißker* Dank, da sie mir während des Studiums nicht nur finanziell, sondern vor allem auch emotional stets zur Seite standen.

Inhaltsverzeichnis

1 Einführung 1
- 1.1 Physikalische und virtuelle Displays . . . 1
- 1.2 Multi-Display-Umgebungen . . . 3
- 1.3 Ziel der Arbeit . . . 3

2 Intelligente Shutterbrillen 6
- 2.1 Technische Voraussetzungen . . . 7
- 2.2 Implementierung . . . 8
  - 2.2.1 Dynamische Slotzuweisung . . . 8
  - 2.2.2 VIP-Nutzer . . . 10
  - 2.2.3 Umgang mit Latenzen . . . 11
- 2.3 Diskussion . . . 13

3 Inhaltliche Anforderungsanalyse 16
- 3.1 Beliebige Raumkopplung von Displays . . . 16
- 3.2 Virtuelle Sichten . . . 18
- 3.3 Überlappende Territorien . . . 19
- 3.4 Kontinuierlicher Raum für Werkzeuge . . . 22
- 3.5 Regelung von Sichtbarkeiten . . . 23
- 3.6 Lokale und entfernte Nutzergruppen . . . 24
- 3.7 Konfigurierbar- und Erweiterbarkeit . . . 24

4 Verwandte Arbeiten 25
- 4.1 Multi-Display-Setups . . . 25
- 4.2 3D-Multi-User-Technologien . . . 27
- 4.3 Interaktionskonzepte . . . 28
- 4.4 Diskussion . . . 30

5 Kontinuierliche Abbildung der Displays des Workspaces 31
5.1 Viewing Setup im Szenegraph . . . 32
5.1.1 Physikalische Displays . . . 32
5.1.2 Virtuelle Displays . . . 35
5.2 Werkzeuge und ihre Repräsentanten . . . 38
5.3 Separate Navigationen . . . 39
5.4 Diskussion . . . 40

6 Separate Abbildung der Displays des Workspaces 43
6.1 Viewing Setup im Szenegraph . . . 44
6.1.1 Physikalische Displays . . . 44
6.1.2 Erweiterung um überlappende Territorien . . . 46
6.1.3 Virtuelle Displays . . . 48
6.2 Werkzeuge und ihre Repräsentanten . . . 48
6.3 Sichtbarkeiten von Repräsentanten . . . 50
6.3.1 Sichtbarkeiten innerhalb der Display-Gruppe . . . 51
6.3.2 Sichtbarkeiten außerhalb der Display-Gruppe . . . 53
6.3.3 Sichtbarkeiten von Szeneelementen . . . 54
6.4 Die aktive Repräsentation von Werkzeugen . . . 55
6.4.1 Allgemeines Entscheidungsverfahren . . . 55
6.4.2 Konkrete Implementierung . . . 57
6.5 Diskussion . . . 59

7 Zusammenfassung und künftige Arbeiten 62

A Anhang 70
A.1 Beispielhafte Konfigurationsdatei . . . 70
A.2 UML-Klassendiagramm . . . 74

1 Einführung

1.1 Physikalische und virtuelle Displays

Die interaktive Exploration und Manipulation virtueller Inhalte benötigt Darstellungsmedien, welche es ermöglichen, visuelle Rückmeldung über die getätigten Aktionen und Manipulationen zu geben. Solche *Displays* können sehr vielfältig sein und sich in ihrer Auflösung, Größe, räumlicher Anordnung und Technologie drastisch unterscheiden. Allgemein lassen sie sich in zwei Kategorien einteilen: *physikalische* und *virtuelle* Displays.

Meist werden bei der Verwendung des Begriffs des Displays die physikalischen Darstellungsmedien assoziiert. Diese bezeichnen optische Ausgabegeräte wie Desktop-Bildschirme, Fernseher, Handy- und Tablet-Touchscreens, Tabletop- und Leinwand-Projektionen sowie viele weitere elektrische Installationen. Physikalische Displays fungieren somit als visuelle Schnittstelle zwischen der realen Welt der Nutzer und der virtuellen Welt der Applikation.

Terrenghi et al. [43] klassifizieren physikalische Displays nach ihrer Größe und benennen diese Kategorien nach amerikanischen Maßeinheiten. So finden sich die portablen Handy- und Tablet-Displays in den Kategorien *Inch-* und *Foot-Size-Displays* wieder, während die Tabletop- und Leinwand-Displays in die Bereiche der *Yard-* und *Perch-Size-Displays* fallen. Noch größere Displays bis hin zum Ausmaß eines ganzen Gebäudes (wie zum Beispiel das Projekt *Blinkenlights* [4]) werden schließlich als *Chain-Size-Displays* eingestuft.

Neben der Größe spielen auch die Stereomodi und -technologien eine wichtige Rolle bei der Unterscheidung physikalischer Displays. Hierbei stehen die klassischen 2D-Displays, wie die meisten Desktop-Bildschirme und Fernseher, den neueren Displays mit eingebauten 3D-Technologien gegenüber. Der 3D-Effekt kann hierbei durch das Tragen von

aktiven Shutterbrillen [20, 23] oder passiven Brillen [10, 38] hervorgerufen werden. Die Wahrnehmung von dreidimensionalen Bildern ohne die Notwendigkeit von speziellen Brillen kann durch Autostereoskopie [29] geleistet werden.

Der Inhalt von 2D-Displays ist in der Regel von mehreren Nutzern gleichermaßen wahrnehmbar, so dass man direkt von einer *Multi-User*-Fähigkeit der Displays sprechen kann. Im dreidimensionalen Raum ist es jedoch erwünscht, die individuelle Perspektive eines Nutzers auf die virtuelle Welt zu bestimmen und anzuzeigen, was von einem *Multi-User-Display* erfordert, jedem Nutzer ein unterschiedliches Stereobild präsentieren zu können. Dies kann durch zeitliche Multiplexverfahren geschehen, bei denen durch das Tragen von aktiven Shutterbrillen die richtigen Bilder gefiltert werden können [20]. Ist diese Voraussetzung nicht gegeben, kann nur ein Nutzer mit perspektivisch korrekten Inhalten versorgt werden und man spricht von einem *Single-User-Display*.

Zusammengefasst beschreibt der Begriff der physikalischen Displays die visuellen Ausgabemedien, welche in der realen Welt lokalisiert und somit greifbar sind. Ihnen gegenüber stehen die *virtuellen Displays*, Darstellungsflächen, welche innerhalb der virtuellen Welt der Applikation verankert sind und Bilder aus derselben oder anderen virtuellen Welten zeigen. In diesem Fall wird zwar das virtuelle Display den Nutzern über ein physikalisches Display dargestellt, kann sich jedoch selbst in seinen Eigenschaften davon unterscheiden. Beispiele für virtuelle Displays sind die klassischen Desktop-Fenster [39], (Foto-)Portale [41, 21] und *Magic Lenses* [11, 44].

Die oben beschriebene Unterteilung der physikalischen Displays kann in ähnlicher Form auch auf die virtuellen Displays angewendet werden. Das Größenkriterium bezieht sich in diesem Fall auf das Ausmaß des Displays innerhalb der virtuellen Welt und ist somit vollkommen unabhängig vom physikalischen Display, auf welchem es angezeigt wird. Die Darstellbarkeit von dreidimensionalen Bildern ist an die Technologien des physikalischen Displays gebunden, dennoch besteht immer die Möglichkeit, zweidimensionale Monobilder anzuzeigen, da diese nur eine Teilmenge des dreidimensionalen Stereofalls sind. Auch die Anzahl an versorgbaren Nutzern ist durch das physikalische Display bedingt. Unterstützt dieses nur einen Nutzer, so überträgt sich diese Beschränkung auch auf das virtuelle Display. Liegt hingegen ein *Multi-User-Display* vor, so können den Nutzern mit Hilfe von entsprechender Applikationslogik auch auf dem virtuellen Display unterschiedliche Inhalte, wie zum Beispiel individuelle Perspektiven auf denselben Szeneausschnitt, angezeigt werden.

1.2 Multi-Display-Umgebungen

Seyed et al. [37] beschreiben *Multi-Display-Umgebungen* als Systeme, welche mehrere physikalische Displays in einem interaktiven Umfeld integrieren. Dies bedeutet, dass die Darstellung einer logischen Applikation nicht auf ein einzelnes physikalisches Display beschränkt ist, sondern in Echtzeit auf die verschiedensten Ausgabemedien innerhalb eines Interaktionsraumes verteilt wird. Dieser kohärente Raum in der realen Welt wird im Verlauf dieser Arbeit als *Workspace* bezeichnet.

Terrenghi et al. [43] kategorisieren Multi-Display-Umgebungen einerseits nach der Größe der beteiligten Displays wie oben beschrieben, andererseits nach den sozialen Interaktionsverhältnissen der Nutzer. So können beispielsweise zwei Menschen in *One-One*-Situationen, kleinere Gruppen von Menschen in *Few-Few*-Situationen oder große Menschenmassen in *Many-Many*-Situationen über mehrere Displays kollaborieren. Kombinationssituationen aus den beschriebenen Fällen, wie zum Beispiel *One-Few*- oder *One-Many*-Präsentationssituationen, sind ebenfalls denkbar.

Etwas weiter gefasst lässt sich die Definition von Multi-Display-Umgebungen durch virtuelle Displays ergänzen. Eine um diesen Aspekt erweiterte Umgebung besteht dann aus mehreren physikalischen Displays im Workspace einerseits, andererseits jedoch zusätzlich aus einem oder mehreren virtuellen Displays in der dargestellten Welt der Applikation.

1.3 Ziel der Arbeit

Im Kontext der Virtuellen Realität werden große physikalische 3D-Multi-User-Displays eingesetzt, um Sichtfenster in die simulierte virtuelle Welt zu bieten. Im Vergleich zur Platzierung von Displays direkt vor den Augen der Nutzer (*head-mounted displays*) bietet dieser Ansatz eine ungestörte Wahrnehmung des eigenen Körpers und weiterer Nutzer. Durch die Kombination mehrerer physikalischer Displays in einer Multi-Display-Umgebung wird das Sichtfenster erweitert und bietet dadurch Raum für zusätzliche Perspektiven und Betrachtungsweisen, um das Immersionsgefühl zu erhöhen. Die Multi-User-Fähigkeit der Displays wird dabei genutzt, um für jeden Nutzer eigene perspektivisch korrekte Bilder zu realisieren. Dazu wird jeden Frame für die aktuelle Kopfposition jedes Nutzers

bezüglich des jeweiligen physikalischen Displays eine entsprechende Projektionsmatrix erstellt. Befinden sich dabei mehrere Nutzer im selben Navigationskoordinatensystem in der virtuellen Welt, entsteht ein *Shared 3D-Space*, in welchem beispielsweise das physikalische Zeigen auf Inhalte in der virtuellen Welt für andere Nutzer nachvollzogen werden kann. Mit virtuellen Displays wird eine physikalische Multi-Display-Umgebung in der simulierten Welt durch zusätzliche Darstellungsmedien ergänzt.

Ziel dieser Arbeit ist es, eine konfigurierbare und generische Applikationsinfrastruktur für Multi-Display-Umgebungen im Kontext der Virtuellen Realität, bestehend aus physikalischen und virtuellen 3D-Multi-User-Displays, zu schaffen. Dazu soll zunächst in Kapitel 2 Augenmerk auf intelligente Shutterbrillen gelegt werden, welche ihre Schaltzeiten entsprechend des betrachteten physikalischen Displays korrekt einstellen. Sind diese technischen Voraussetzungen erfüllt, sollen in Kapitel 3 die inhaltlichen Anforderungen an solch ein System erörtert werden. Kapitel 4 beschäftigt sich anschließend mit verwandten Arbeiten auf dem Gebiet der Multi-Display-Umgebungen. Aus den Betrachtungen ergeben sich zwei grundsätzliche Implementierungsstrategien zur Realisierung der Anforderungen, welche in den Kapiteln 5 und 6 dargestellt werden. In einer abschließenden Zusammenfassung in Kapitel 7 werden die beiden Ansätze nochmals direkt gegenübergestellt und auf künftige Herausforderungen hingewiesen.

Getestet werden die entwickelten Konzepte in einem Workspace mit einem (*Perch-Size*) Leinwand-Display, welches wie von Kulik et al. [20] beschrieben bis zu sechs Nutzer mit Stereobildern versorgt, und einem (*Yard-Size*) Tabletop-Display, welches Stereobilder für bis zu drei Nutzer verarbeiten kann. Abbildung 1.1 zeigt den Aufbau dieser Displays in unserem Labor. Am anderen Ende des Raumes befindet sich ein separater Workspace mit einem (*Perch-Size*) Leinwand-Display für zwei Nutzer, um *Group-to-Group*-Interaktionen [9] zu realisieren. Zur Darstellung der virtuellen Welt der Applikation wird eine weiterentwickelte Version des Virtual-Reality-Frameworks *avangoNG* [6] mit dem Renderer *guacamole* [35] verwendet. Letzterer verwendet Szenegraphen, um die Strukturen der Objekte in der Szene hierarchisch auszudrücken und die Viewing Setups der einzelnen Nutzer zu spezifizieren. Eine Applikation teilt sich dabei auf einen *Server*- und mehrere *Client*-Computer auf. Die Anzahl der Clients entspricht hierbei in der Regel der Anzahl der physikalischen Displays im Setup. Der Server verwaltet die Eingabegeräte und Applikationslogik und baut den Szenegraph korrekt auf, welcher über das Netzwerk zu den Clients gesendet wird, um dort auf den entsprechenden Displays gerendert zu werden. Diese Struktur ist in Abbildung 1.2 am Setup des Labors visualisiert.

Abbildung 1.1: Ein physikalischer Workspace in unserem Labor, bestehend aus einem 3D-Multi-User Leinwand-Display und einem 3D-Multi-User Tabletop-Display auf der linken Seite. Ein weiterer Workspace mit einem 3D-Multi-User Leinwand-Display befindet sich auf der abgewandten Seite des Labors.

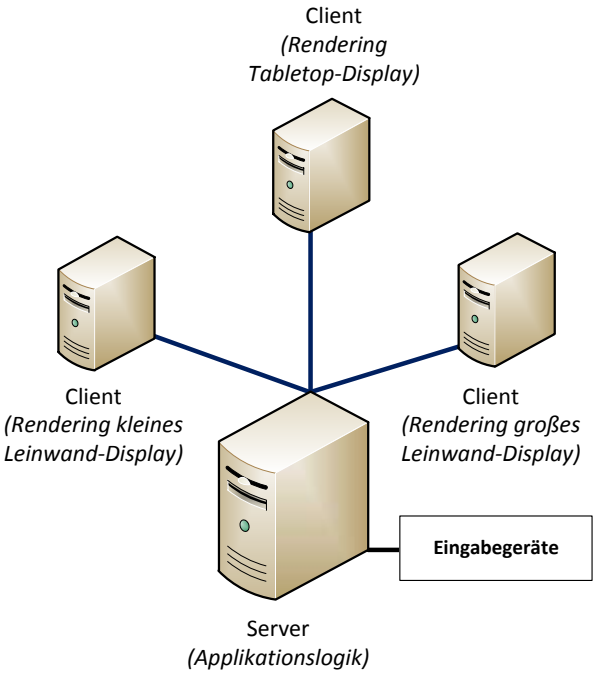


Abbildung 1.2: Verteilung der Verantwortlichkeiten einer Applikation im Setup unseres Labors. Der Server verwaltet die Eingabegeräte und Applikationslogik, baut den Szenegraph auf und sendet diesen über das Netzwerk (blau) zu den Client-Computern, welche die Bilder eines physikalischen Displays rendern.

2 Intelligente Shutterbrillen

Während bestimmte Multi-Display-Umgebungen, wie zum Beispiel Tiled Walls oder die CAVE [12], aus einer Anordnung mehrerer baugleicher Einzeldisplays bestehen, möchte man in anderen Multi-Display-Umgebungen unterschiedliche Eigenschaften der Displays wie Transformation, Auflösung oder physikalische Größe nutzen, um Inhalte gemäß der verschiedenen Affordanzen dieser Eigenschaften zu separieren. So verleitet beispielsweise ein großes, immersives Leinwand-Display dazu, die Welt im Originalmaßstab zu erkunden, während sich ein nach unten geneigtes Tabletop-Display eher für einen Blick von oben auf eine miniaturisierte Welt eignet.

Jedoch bringt die Verwendung von nicht baugleichen Displays auch unerwünschte Eigenschaftsdifferenzen mit sich, die es zu kompensieren gilt. Im Kontext der physikalischen 3D-Multi-User-Displays sind dies insbesondere die Zeiten, zu denen sich die zugehörigen Shutterbrillen öffnen und schließen, um jedem Nutzer im richtigen Moment das entsprechende Stereobild anzuzeigen. Daher ist beim Blickwechsel zwischen den Displays zur Laufzeit eine Anpassung der Shutterkonfiguration nötig. Das in [20] beschriebene 3D-Multi-User-Single-Display-System entscheidet, ob ein Nutzer auf das Leinwand-Display blickt und öffnet die Shutter eines Nutzers, wenn dies nicht der Fall ist. Somit kann sich dieser ohne den Stroboskopeffekt der Shutter im Raum umsehen. Ziel dieses Kapitels ist es, dieses Verfahren auf mehrere physikalische Displays zu erweitern. Das Setup unseres Labors dient hierbei lediglich als Beispiel; die Konzepte sind auf mehrere physikalische Displays analog anwendbar.

2.1 Technische Voraussetzungen

Herkömmliche Mono-DLP-Projektoren drehen ihr Farbrad mit einer Frequenz von 60 Hz. Damit diese um Stereo- und Multi-User-Fähigkeiten wie in [20] erweitert werden können, besteht die Aufgabe darin, durch einen oder mehrere Projektoren die resultierenden 1/60 s ($\approx 16666,6\mu s$) so auszunutzen, dass allen an einem physikalischen Display anwesenden Nutzern ein neues Stereobild präsentiert wird.

Im Falle des Leinwand-Displays werden dafür sechs Zeitabschnitte, sogenannte *Slots* definiert. Innerhalb eines Slots sind die linken und rechten Shutter der Brille eines bestimmten Nutzers geöffnet, so dass dieser mit beiden Augen auf die Leinwand blicken kann. Für den Stereoeffekt kommen Polarisationsfilter zum Einsatz, welche dann die zwei innerhalb dieses Slots präsentierten Bilder trennen, so dass jedes Auge ausschließlich sein eigenes Bild sieht (passives Stereo). Auf diese Weise werden innerhalb der 1/60 s mit Hilfe von sechs Projektoren sechs Stereobildpaare, insgesamt also zwölf Einzelbilder, für sechs Nutzer dargestellt. Das System muss daher die projizierten Inhalte mit 360 Hz umschalten können.

Das Tabletop-Display besteht nur aus einem Projektor, so dass keine Polarisationsfilter zum Einsatz kommen können, weshalb die Trennung der Bilder für das linke und rechte Auge ebenfalls zeitmoduliert vorgenommen werden muss (aktives Stereo). In diesem Fall besteht ein Slot aus zwei Zeitbereichen: im ersten Teil sind die Shutter des linken Auges des jeweiligen Nutzers geöffnet, im zweiten Teil die Shutter des rechten Auges. Diese Einschränkung halbiert die Anzahl der maximal versorgbaren Nutzer auf drei. In Abbildung 2.1 sind die verschiedenen Zeitbereiche der Slots für Leinwand- und Tabletop-Display visualisiert. Es fällt auf, dass die Slots auf beiden Displays nicht gleichmäßig innerhalb der gesamten Zeitspanne aufgeteilt sind. Dies ist darauf zurückzuführen, dass die Projektoren werkseitig mit einem Farbrad ausgetattet sind, welches die drei Farbkanäle eines einzelnen Bildes zeitlich auftrennt. Da das menschliche Auge mehr Grünrezeptoren besitzt, ist der grüne Bereich dieses Farbrads in der Regel etwas kleiner als das rote und blaue Äquivalent. Die Entfernung des Farbrads ermöglicht die Darstellung dreier unterschiedlicher Bilder; die Verdoppelung der Geschwindigkeit erhöht diese Anzahl sogar auf sechs. Dies hat jedoch zur Folge, dass die im ehemals grünen Bereich dargestellten Bilder eine kürzere Darstellungszeit besitzen, was in Abbildung 2.1 sehr gut an Slots 2 und 5 im Leinwand-Display und an Slot 2 im Tabletop-Display nachvollzogen werden kann.

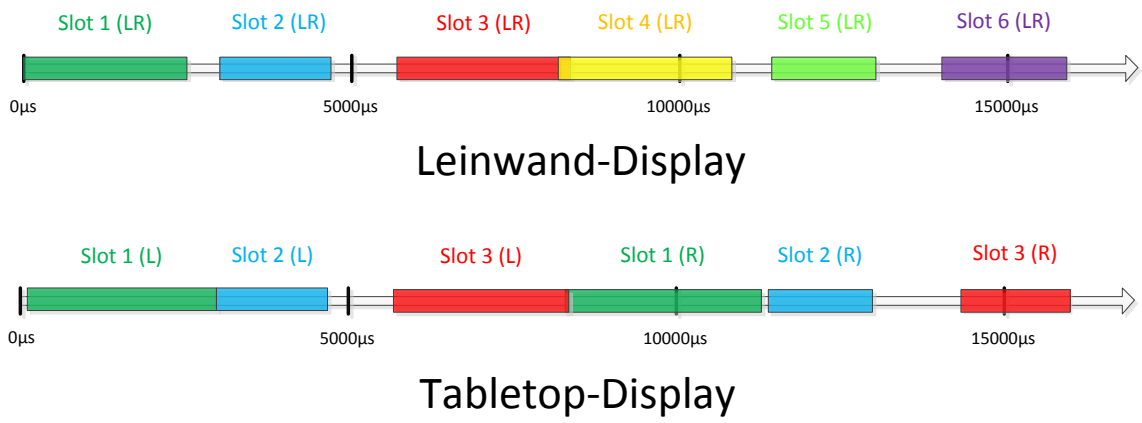


Abbildung 2.1: Optimierte Schaltzeiten der Shutterbrillen für Leinwand- und Tabletop-Display. Jede Farbe visualisiert einen Slot des jeweiligen Displays.

Die ungleichmäßige Aufteilung der Slots ergibt sich weiterhin aus der experimentellen Anpassung der Schaltzeiten, so dass den Nutzern die maximale Helligkeit gewährt wird, gleichzeitig jedoch der *Ghosting*-Effekt minimal gehalten wird.

Das bestehende 3D-Multi-User-Single-Display-System aus [20] kontrolliert die Zeiten der Shutterbrillen über Funk mit Hilfe der μracoli-Bibliothek [2]. Zur Laufzeit wird anhand des Scherwinkels (*yaw*) zum Leinwand-Display überprüft, ob der jeweilige Nutzer auf die Leinwand blickt oder nicht. Die Shutter werden dann entsprechend eingestellt oder komplett geöffnet. Im Folgenden soll nun die erweiterte Implementierung dieses Verfahrens für mehrere Displays, bei der die Brillen eines Nutzer stets auf das angesehene Display eingestellt sind, erläutert und evaluiert werden.

2.2 Implementierung

2.2.1 Dynamische Slotzuweisung

In Workspaces mit mehreren physikalischen Displays reicht die Betrachtung des Scherwinkels nicht aus, um festzustellen, auf welches der Displays ein Nutzer gerade blickt. Daher werden von jedem im Workspace verfügbaren Display Proxygeometrien erzeugt.

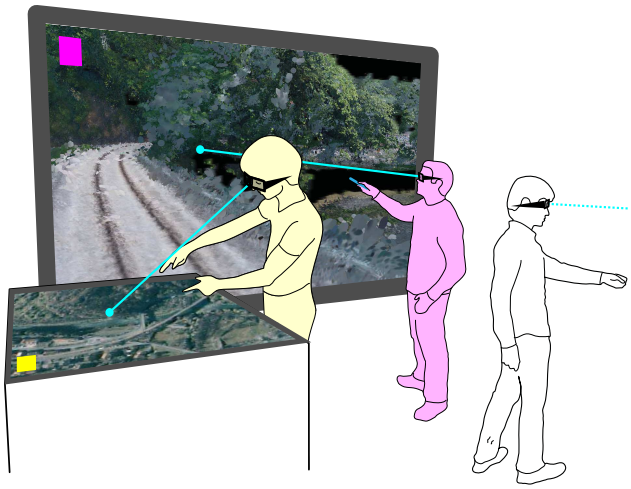


Abbildung 2.2: Die Blickstrahlen der Nutzer werden auf Schnitte mit den Displays geprüft und somit die Zugehörigkeiten bestimmt. Die Zeiten der Shutterbrillen werden entsprechend eingestellt.

Jeder Nutzer definiert durch die Orientierung seiner Shutterbrille einen Strahl, welcher auf Schnitte mit diesen Proxygeometrien analysiert wird. Wird ein Display getroffen, meldet die entsprechende Nutzerinstanz diese Displayzugehörigkeit beim *Slot-Manager*, sofern an diesem Display noch nicht die maximale Nutzeranzahl erreicht wurde. Wird zwei Sekunden lange kein Display geschnitten, so wird dieser Inaktivitätsstatus auch beim Slot-Manager gemeldet. Somit hat der Slot-Manager zu jedem Zeitpunkt einen Überblick über die aktuellen Zugehörigkeiten von Nutzer zu Display. Abbildung 2.2 zeigt eine Beispielsituation, in welcher die Shutterbrille des magenta gefärbten Nutzers auf das Leinwand-Display und die Shutterbrille des gelben Nutzer auf das Tabletop-Display eingestellt ist.

Bei jeder Änderung der Displayzugehörigkeiten ist es Aufgabe des Slot-Managers, den Nutzern entsprechende Slots zuzuweisen. Dafür ist es einerseits nötig, den korrekten Inhalt des Nutzers zur richtigen Zeit auf dem Display zu präsentieren, andererseits müssen die Schaltzeiten der Shutterbrille so gesetzt werden, dass der Nutzer auch genau dieses Bild sehen kann. Bei allen inaktiven Nutzern werden die Shutterbrillen komplett geöffnet, so dass sich diese ohne Flackern im Raum umsehen können. In Abbildung 2.2 ist dies beim weißen Nutzer der Fall.

Oftmals sind an einem Display weniger Nutzer aktiv als durch die Anzahl der Slots technisch möglich wäre. Wenn in einem solchen Fall pro Nutzer nur ein Slot vergeben wird, bleiben die überschüssigen Slots ungenutzt und Helligkeitsleistung geht verloren.

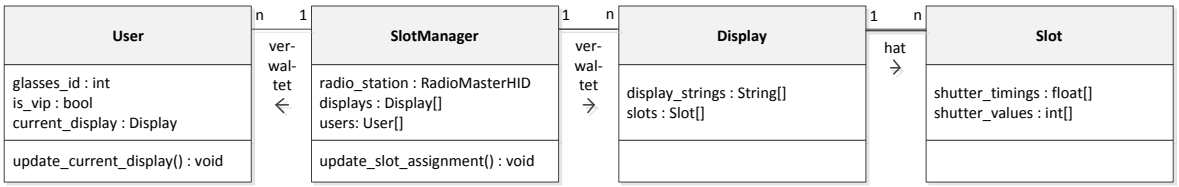


Abbildung 2.3: Benötigte Klassenstruktur zur Verwaltung von Slots mehrerer Displays. Auf Framebasis überprüfen die *User*-Instanzen, welches das aktuell angesehene Display ist und melden eine Änderung bei der *SlotManager*-Instanz des Workspaces. Diese iteriert anschließend über alle verwalteten *Display*s, um den Nutzern die entsprechenden *Slot*-Instanzen zuzuweisen und somit die Schaltzeiten der Brillen einzustellen.

Um diesem Effekt entgegenzuwirken, verteilt der Slot-Manager zunächst je einen Slot pro aktivem Nutzer und vergibt schließlich in derselben Reihenfolge die übrigen freien Slots. Erhält ein Nutzer mehr als einen Slot, so entsprechen die Öffnungszeiten seiner Shutter den Werten des ersten Slots, die Schließzeiten den Werten des letzten zugewiesenen Slots. Das wahrgenommene Bild erscheint somit heller.

In Abbildung 2.3 sind die für den beschriebenen Mechanismus verwendeten Klassen mit ihren Attributen und Funktionen in Form eines UML-Klassendiagramms aufgezeigt. Abbildung 2.4 zeigt eine Sechs-Nutzer-Situation, in der drei Nutzer am Leinwand-Display und zwei Nutzer am Tabletop-Display aktiv sind. Ein weiterer Nutzer blickt aktuell auf keines der Displays und erhält somit auch keine Slots. Da am Leinwand-Display sechs Slots verfübar sind, aber nur drei Nutzer aktiv sind, bekommt jeder Nutzer zwei Slots. Am Tabletop-Display ist ein überschüssiger Slot vorhanden, welcher dem chronologisch ersten Nutzer, in diesem Fall Nutzer 4, zusätzlich zugewiesen wird.

2.2.2 VIP-Nutzer

Manchmal ist eine faire und gleichmäßige Aufteilung der freien Slots, wie soeben beschrieben, nicht erwünscht. Insbesondere bei Besuchen und Vorführungen möchte man dem Gast auf jedem Display die maximal verfügbaren Slots zur Verfügung stellen, so dass dieser stets das hellste Bild sieht. Daher besteht die Möglichkeit, Nutzer als *Very Important Person* (VIP) zu kennzeichnen. In Abbildung 2.3 ist dies durch eine boolesche Variable in

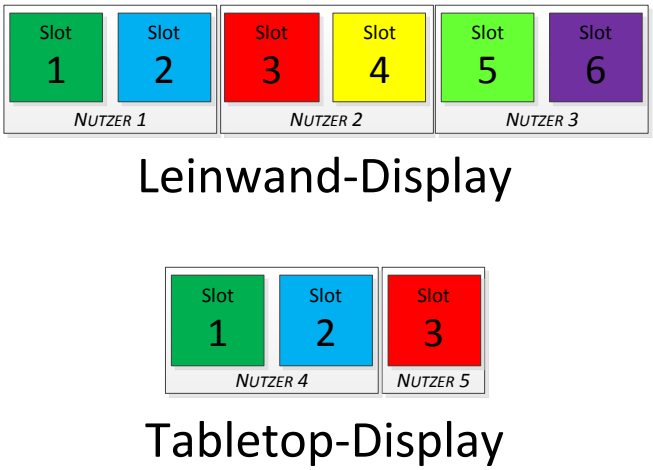


Abbildung 2.4: Slotzuweisung in einer Sechs-Nutzer-Situation, bei der drei auf die Wand, zwei auf den Tisch und einer auf kein Display blickt.

der *User*-Klasse realisiert. Ist diese bei einer oder mehreren Instanzen auf *True* gesetzt, läuft die oben beschriebene Slotzuweisung in einer leicht abgewandelten Variante ab: zunächst werden an jedem Display jedem aktiven Nutzer ein Slot zugewiesen, danach werden die freien Slots ausschließlich an die VIP-Nutzer verteilt.

Abbildung 2.5 zeigt die gleiche Situation wie Abbildung 2.4, nur dass nun Nutzer 1 und Nutzer 5 als VIPs definiert wurden. In diesem Fall bekommt am Leinwand-Display ausschließlich Nutzer 1 die freien Slots, am Tabletop-Display bekommt jetzt Nutzer 5 den einen freien Slot, der im vorherigen Fall noch Nutzer 4 zugeordnet wurde.

2.2.3 Umgang mit Latenzen

In Abschnitt 2.2.1 wurde erläutert, dass bei einer Änderung der Displayzugehörigkeiten einerseits die Bildinformationen auf dem Display angepasst und andererseits die Shutterzeiten geändert werden müssen. Ersteres geschieht durch einen Umbau der Feldverbindungen der entsprechenden Slotknoten im Szenegraph. Die veränderten Shutterzeiten werden über eine USB-Verbindung zum Funksender übertragen, welcher diese anschließend zu den Brillen weiterleitet.

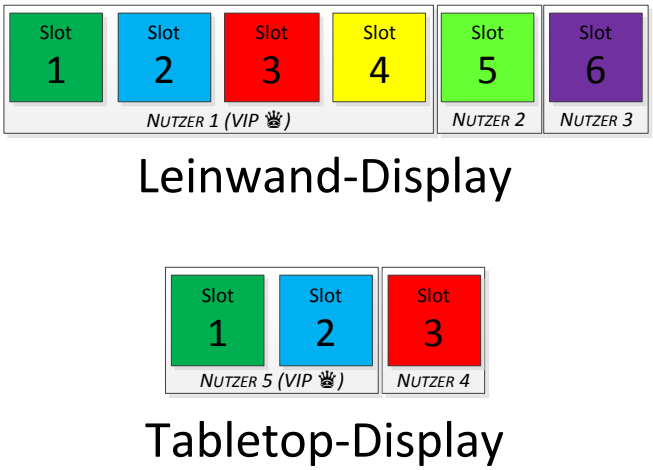


Abbildung 2.5: Slotzuweisung in der Sechs-Nutzer-Situation aus Abbildung 2.4 mit zwei VIPs.

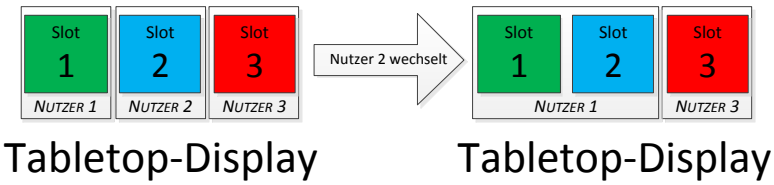


Abbildung 2.6: Wenn Nutzer 2 an ein anderes Display wechselt, sieht Nutzer 1 kurzzeitig auf Grund der Netzwerklatenz ein Bild, welches Nutzer 2 zugeordnet war.

In verteilten Anwendungen kann dies ein Problem darstellen, da die Weiterleitung der veränderten Szenegraphinformationen zum Clientprozess in der Regel länger dauert als die Kommunikation mit den Shutterbrillen. Abbildung 2.6 zeigt einen Fall, in dem sich dieser Unterschied negativ für den Nutzer auswirkt. In diesem Fall haben drei Nutzer an einem Display jeweils einen Slot, jedoch wechselt nun Nutzer 2 zu einem anderen Display. Gemäß der fairen Slotzuweisung bekommt Nutzer 1 den freien Slot vom Slot-Manager zugeteilt. Nun ändern sich seine Shutterzeiten, auf Grund der Netzwerklatenz ist jedoch in Slot 2 für eine kurze Dauer noch das veraltete Bild von Nutzer 2 zu sehen, was als unangenehmes Flackern bei der Umschaltung wahrgenommen wird.

Eine mögliche Lösung für dieses Problem wäre ein Feedback des entsprechenden Clientprozesses an den Server, wenn die jeweilige Änderung angekommen ist. Jedoch beruht

der aktuell implementierte Verteilungsalgorithmus in *avango* auf dem ZMQ- (*Zero Message Queue* [5]) Protokoll, welches nur eine unidirektionale Kommunikation von Server zu Client ermöglicht und somit diesen Lösungsansatz ausschließt.

Die implementierte Lösung prüft daher, wie viele Slots ein Nutzer vor und nach einer Änderung zugewiesen bekommen hat. Erhöht sich die Anzahl der Slots eines Nutzers durch eine Änderung, so wird die Modifikation am Szenegraph sofort vorgenommen, die Einstellung der Shutterzeiten jedoch um eine bestimmte Zeit verzögert. Auf diese Weise ist sichergestellt, dass das entsprechende Bild vorhanden ist, bevor sich die Shutterbrillen auf den verlängerten Zeitbereich einstellen. Verringert sich die Anzahl der Slots eines Nutzers, können sowohl Szenegraph- als auch Shutteränderungen sofort vorgenommen werden, da die Bilder bereits in den verwendeten Slots vorhanden sind.

2.3 Diskussion

Verschiedene Tests mit dem Setup in unserem Labor haben gezeigt, dass das implementierte dynamische Schaltsystem generell funktionsfähig ist. Nutzer können zur Laufzeit von Leinwand zu Tisch und wieder zurückwechseln und der Aktivitäts- und VIP-Status der individuellen Nutzer wird bei der Slotzuweisung berücksichtigt.

Der Ansatz zur Kaschierung der Netzwerklatenzen verhindert einige Fälle, in denen für kurze Zeit falsche Bilder sichtbar sind. Jedoch gibt es immer noch Situationen, in denen für Nutzer die Auswirkungen der Netzwerklatenz feststellbar sind. Abbildung 2.7 zeigt einen ähnlichen Fall wie Abbildung 2.6, jedoch wechselt nun Nutzer 3 zu einem anderen Display. Der freie Slot wird nach wie vor dem ersten Nutzer (Nutzer 1) zugewiesen, was Nutzer 2 dazu zwingt, sich einen Slot nach hinten zu bewegen. In diesem Fall ist jedoch in Slot 3 auf Grund der Netzwerklatenz noch nicht das korrekte Bild angezeigt; ein Flackern wird von Nutzer 2 wahrgenommen.

Um diesem Effekt entgegenzuwirken, dürften frei gewordene Slots nur an Nutzer in Nachbarslots vergeben werden, was wiederum bei Displays mit einer größeren Anzahl an Slots dazu führen kann, dass anwesende VIP-Nutzer nicht die ihn zustehenden Slots zugeteilt bekommen. An dieser Stelle müssten frei gewordene Slots schrittweise zwischen benachbarten Nutzern weitergegeben werden, bis sie beim eigentlichen Zielnutzer angekommen sind.

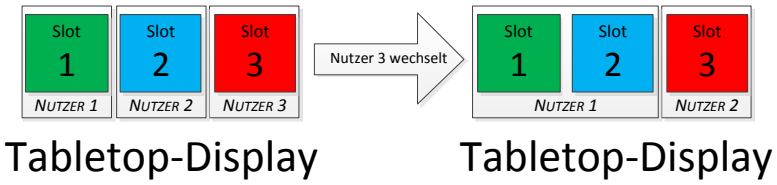


Abbildung 2.7: Durch den Wechsel von Nutzer 3 an ein anderes Display verschiebt sich der Zeitbereich von Nutzer 2 um einen Slot nach hinten, was für diesen als Flackern wahrgenommen wird.

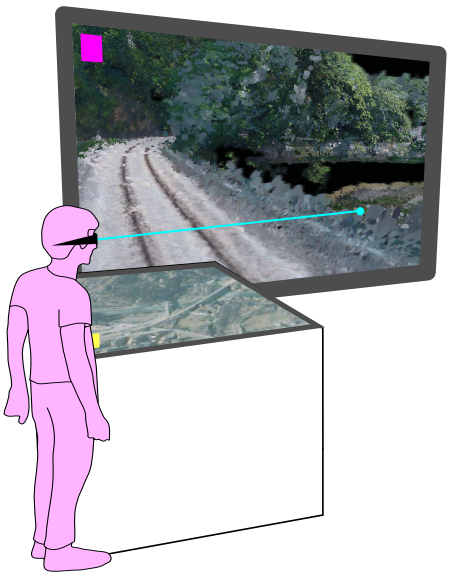


Abbildung 2.8: Der Nutzer kann immer nur den Inhalt eines Displays (hier das Leinwand-Display) korrekt sehen. Beim Blick auf die Kante zwischen den Displays erlebt der Nutzer ständige Schaltsprünge.

Ein weiteres Problem des Systems zeigt sich im ständigen Wechsel der Slotanzahl pro Nutzer an einem Display, wenn andere Nutzer das Display wechseln oder hinzukommen. Zwar ist es fair, die freien Slots zu verteilen, um hellere Bilder zu erhalten, in der Praxis wirken die ständigen und unerwarteten Helligkeitsunterschiede jedoch unangenehm. Eine mögliche Lösung wäre hier die maximale Anzahl der Slots eines Nutzers pro Display auf einen zu beschränken.

Nach Adressierung dieser zwei Problembereiche gibt es weiterhin kritische Situationen, wenn die Displays räumlich nicht klar voneinander separiert sind. Wie in Abbildung 1.1 dargestellt, befindet sich beim Setup unseres Labors das Tabletop-Display zu einem Teil vor dem Leinwand-Display. Begibt man sich nun an die Position hinter dem Tisch, welche in Abbildung 2.8 illustriert ist, so möchte man die Inhalte beider Displays gleichzeitig sehen können, was jedoch aus den beschriebenen technischen Gründen nicht möglich ist. Insbesondere beim Blick auf die Kante zwischen den Displays erlebt man ständige Schaltsprünge, welche unangenehm wirken und die beiden Displays nicht mehr als untereinander zusammenhängende Ausgabegeräte erscheinen lassen.

Eine Lösung dieses Problems kann lediglich auf technischer Ebene vorgenommen werden, indem das System neu konfiguriert wird, so dass die Bilder auf dem Leinwand-Display mit demselben aktiven Stereomodus und denselben Shutterzeiten wie auf dem Tabletop-Display präsentiert werden. Natürlich hat dies zur Folge, dass das Leinwand-Display drei seiner sechs Slots verliert und somit nicht mehr seine volle Leistung erbringen kann. Im Gegenzug dazu können Nutzer dann jedoch ohne Schaltsprünge ihre Blicke zwischen Tabletop- und Leinwand-Display wechseln und in Randgebieten die Inhalte beider Displays korrekt wahrnehmen.

Für die folgenden inhaltlichen Betrachtungen und Implementierungen dieser Arbeit wurde eine solche Neukonfiguration am Setup unseres Labors durchgeführt. In den weiteren Kapiteln wird daher davon ausgegangen, dass es keine Differenzen in den Zeiten der Shutterbrillen für alle am Setup beteiligten physikalischen Displays gibt.

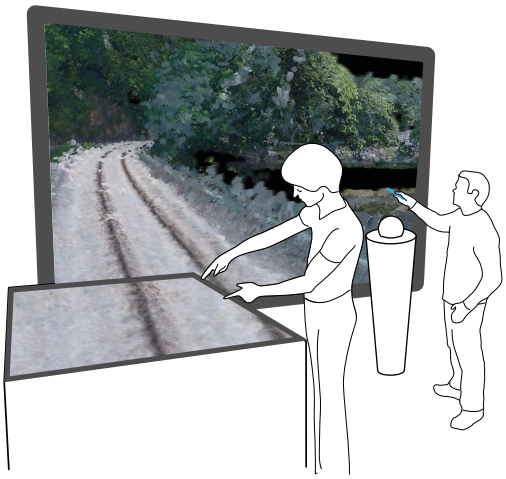
3 Inhaltliche Anforderungsanalyse

In Kapitel 2 wurden die technischen Voraussetzungen an Multi-Display-Umgebungen, bestehend aus physikalischen 3D-Multi-User-Displays, am Beispiel des Setups unseres Labors erläutert und Verfahren zur Kompensation unterschiedlicher Schaltzeiten der Shutterbrillen hergeleitet. Als vielversprechendste Lösung zeigte sich die hardwareseitige Neukonfiguration und somit die Vereinheitlichung der Schaltzeiten, auch wenn damit nur drei der ursprünglichen sechs Nutzerperspektiven auf dem Leinwand-Display zur Verfügung stehen. In diesem Kapitel liegt nun der Fokus auf den inhaltlichen Voraussetzungen und Anforderungen an die Applikationsinfrastruktur für solche Umgebungen. Die herausgearbeiteten Aspekte der Anforderungsanalyse dienen als Basis, um die Mächtigkeit der im Rahmen dieser Arbeit entstandenen Softwarestrukturen einzustufen.

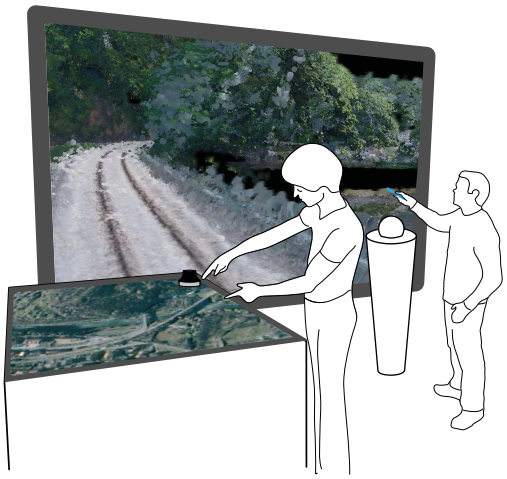
3.1 Beliebige Raumkopplung von Displays

Anforderung 3.1.1: Das System soll dem Entwickler die Möglichkeit geben, die verwendeten physikalischen und virtuellen Displays räumlich beliebig miteinander zu koppeln.

Sind physikalische Displays miteinander gekoppelt, so wird die räumliche Anordnung der Displays im Workspace in der virtuellen Welt exakt abgebildet. Der Inhalt eines Displays wird dadurch analog zum Konzept der CAVE [12] kontinuierlich auf den gekoppelten Displays fortgeführt. Abbildung 3.1(a) zeigt eine Kopplung zwischen dem Leinwand- und dem Tabletop-Display in unserem Labor, in der das Leinwand-Display nach vorne in die virtuelle Welt zeigt, während das Tabletop-Display nach unten auf den Boden gerichtet ist. Die Kopplung virtueller Displays ist ähnlich der Kopplung physikalischer Displays mit dem Unterschied, dass die virtuellen Displays nicht im Workspace verankert sind.



(a) Gekoppelte Displays bilden ihre räumliche Anordnung am angezeigten Standpunkt der virtuellen Welt ab. Szeneinhalte werden somit kontinuierlich über mehrere Displays fortgesetzt. Eine zugeordnete Navigation wirkt sich auf beide Displays aus.



(b) Entkoppelte Displays zeigen unterschiedliche Standpunkte und Skalierungen in der virtuellen Welt und besitzen je eine eigene Navigationsmöglichkeit.

Abbildung 3.1: Kopplung zweier physikalischer Displays

Um das kontinuierliche Verhalten zu erzielen, wird somit die Anordnung der Displayflächen in der virtuellen Welt am dargestellten Standpunkt, ebenfalls in der virtuellen Welt, reproduziert.

Anforderung 3.1.2: Für jede Einheit aus gekoppelten Displays, im Folgenden als *Display-Gruppe* bezeichnet, soll eine individuelle Navigationsmöglichkeit zugeordnet werden können.

Im Falle der Situation aus Abbildung 3.1(a) bilden Leinwand- und Tabletop-Display zusammen eine Display-Gruppe. Auf Grund dieser Kopplung wirkt sich die Eingabe eines Navigationsgeräts auf beide Displays aus. Abbildung 3.1(b) hingegen zeigt eine Situation, in welcher die beiden Displays nicht gekoppelt sind. Somit eröffnet jedes Display eine eigene Display-Gruppe und hat ein zugeordnetes Navigationsgerät, mit welchem sich der jeweilige Standpunkt in der virtuellen Welt modifizieren lässt.

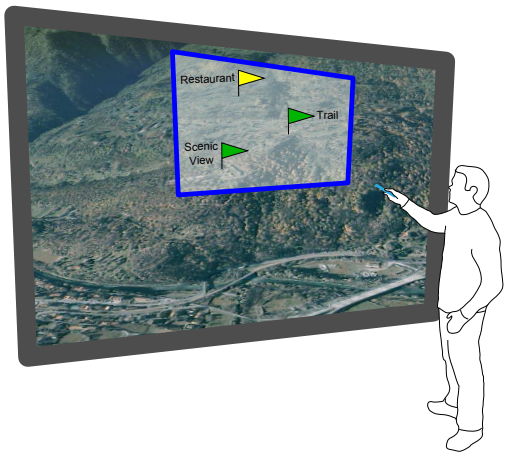
Die in Abbildung 3.1 dargestellten Fälle sind gleichermaßen für virtuelle Displays anwendbar. So können beispielsweise durch das Hinzufügen mehrerer virtueller Displays zu einer Display-Gruppe mehrflächige Portale und somit auch Konstrukte wie *Magic Boxes* [16] realisiert werden.

3.2 Virtuelle Sichten

Anforderung 3.2.1: Das System soll die dynamische Erzeugbarkeit, Austauschbarkeit und Manipulation virtueller Sichten erlauben.

Eine *virtuelle Sicht* ist ein in der virtuellen Welt verankertes alternatives Betrachtungsmedium der Szene, wessen Inhalt sich entweder geometrisch oder semantisch von der auf dem physikalischen Display dargestellten Hauptansicht unterscheidet.

Virtuelle Sichten können durch virtuelle Displays realisiert werden. Im klassischen Portalfall unterscheidet sich dabei die virtuelle Sicht ausschließlich geometrisch von der Hauptansicht. Weiterhin besteht die Möglichkeit, virtuelle Displays als Linsen zu verwenden, so dass die virtuelle Sicht eine andere semantische Darstellungsweise der Szene zeigt. Einen Beispielfall hierfür zeigt Abbildung 3.2(a), in welchem das virtuelle Display geometrisch unveränderten Inhalt anzeigt, semantisch jedoch zusätzlich Hinweise über Sehenswürdigkeiten in Form von Flaggen gibt. Von der Hauptansicht geometrisch und semantisch unterschiedliche virtuelle Sichten durch Kombination von Portalen und Linsen sind ebenfalls denkbar.



(a) Virtuelle Displays als Linsen können zusätzliche Informationen zur Szene geben.

(b) Eine an ein Trackingtarget gebundene WIM erzeugt einen Überblick über die gesamte Szene.

Abbildung 3.2: Mögliche Einsatzgebiete von zusätzlich dargestellten virtuellen Sichten.

Neben den virtuellen Displays kann auch die Instantiierung von Szeneobjekten virtuelle Sichten realisieren. Abbildung 3.2(b) zeigt den Ansatz einer WIM (*World in Miniature* [40]). Dabei wird die gesamte Szene ein zweites Mal in einer deutlich kleineren Skalierung angezeigt und optional an ein Trackingtarget gebunden. Modifikationen der dargestellten Inhalte in einer der beiden Repräsentationen sind ebenfalls in der anderen Abbildung sichtbar. Um andere semantische Inhalte darzustellen, können die instantiierten Szeneobjekte erweitert, reduziert oder in ihrem Erscheinungsbild verändert werden.

3.3 Überlappende Territorien

In Anforderung 3.1.2 wurde definiert, dass jeder Display-Gruppe eine individuelle Navigationsmöglichkeit zugeordnet werden können soll. In diesem Fall wird der Standpunkt der Display-Gruppe in der virtuellen Welt mit Hilfe des Eingabegeräts gesteuert und durch die *Multi-User*-Fähigkeiten der Displays die individuellen Perspektiven jedes Nutzers an diesem Standpunkt zusammen mit den entsprechenden Szeneelementen korrekt dargestellt, um einen *Shared 3D-Space* zu erzeugen. Es gibt jedoch Situationen, in denen diese kohärente Abbildung der Nutzer und der Szene in den Display-Gruppen nicht erwünscht ist und geometrisch oder semantisch aufgehoben werden soll. Da in einem solchen Fall der Inhalt eines Displays nicht mehr für alle Nutzer konsistent ist, spricht man von *überlappenden Territorien* der Nutzer.

Anforderung 3.3.1: Einzelne Nutzer sollen sich von der Hauptnavigation der Display-Gruppe abspalten und auf individuellen Navigationen andere Lokalitäten der Szene wahrnehmen können.

Einer Display-Gruppe können somit mehrere Navigationen zugeordnet sein, zwischen denen die Nutzer wechseln können. Somit befinden sich unter Umständen nicht mehr alle von ihnen am selben Standpunkt in der Szene, so dass sich ihre Territorien geometrisch unterscheiden, jedoch auf Grund der gemeinsam genutzten Displayfläche überlappen.

Ein interessanter Anwendungsfall, in dem Anforderung 3.3.1 erwünscht ist, ergibt sich aus der räumlichen Anordnung der physikalischen Displays im Workspace. Abbildung 3.3(b) zeigt ein solches Szenario am Beispiel unseres Labors. Steht ein Nutzer hinter

dem Tabletop-Display, so dass das Leinwand-Display weiterhin in seinem Sichtfeld liegt, möchte dieser auf Grund seiner aktuellen Position unter Umständen auf dem Leinwand-Display nicht mehr die Hauptnavigation dieser Display-Gruppe sehen. Vielmehr ergibt es Sinn, die zusätzlich zur Verfügung stehende Displayfläche der Wand auszunutzen und dort die Ansicht des Tabletop-Displays fortzuführen. Dies kann auf zwei mögliche Arten geschehen.

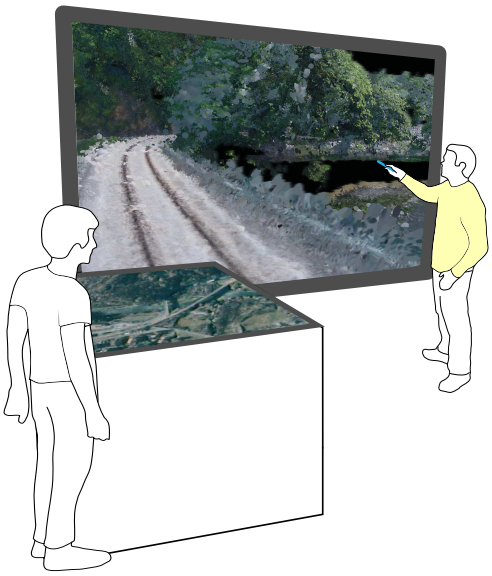
Abbildung 3.3(b) zeigt die erste dieser Möglichkeiten, in der für den betreffenden Nutzer die Navigation auf dem Leinwand-Display so ersetzt wird, dass das Leinwand-Display als zum Tisch räumlich gekoppelt erscheint. Die ursprünglichen Inhalte des Leinwand-Displays werden somit komplett *ausgetauscht*. Die anderen Nutzer werden von diesem Prozess nicht beeinträchtigt und sehen dort weiterhin, wie in Abbildung 3.3(a) gezeigt, die Hauptnavigation der entsprechenden Display-Gruppe.

Anforderung 3.3.2: Das System soll es ermöglichen, für einzelne Nutzer in bestimmten Situationen Inhalte einer Display-Gruppe innerhalb anderer Display-Gruppen fortzuführen, ohne die dort befindlichen Inhalte komplett auszublenden.

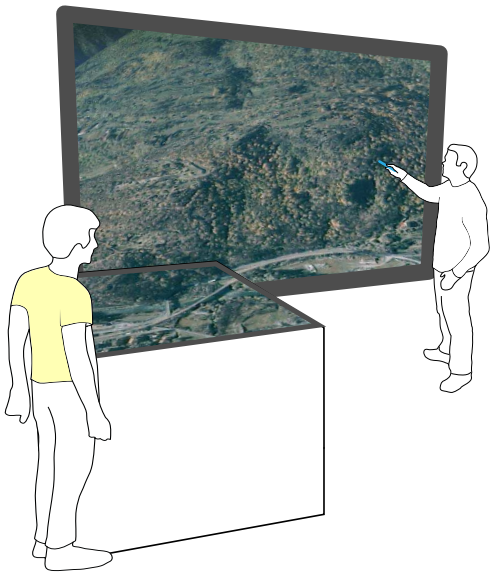
Der soeben beschriebene Ansatz zeigt jedoch seine Schwierigkeiten, wenn die Hauptnavigation des Leinwand-Displays weiterhin von Interesse ist. In diesem Fall ist ein komplettes Ersetzen dieser Navigation nicht erwünscht. Anstattdessen soll, wie in Abbildung 3.3(c) dargestellt, der Inhalt des Tabletop-Displays für bestimmte Nutzer in die Sicht der Hauptnavigation *integriert* werden, ohne diese auszublenden.

Anforderung 3.3.3: Das System soll die Spezifikation von Clipping-Ebenen erlauben, um in bestimmten Situationen Inhalte abzuschneiden.

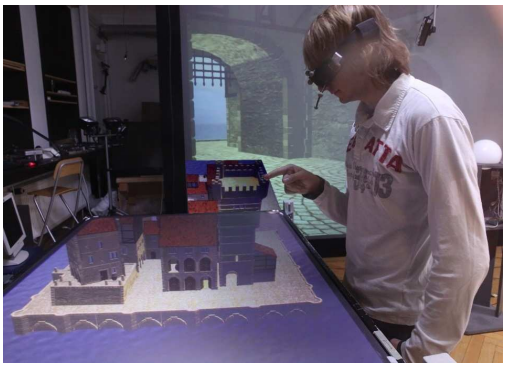
Diese Anforderung ist durch die 3D-Fähigkeit der Displays motiviert und soll die Überlappung von Territorien desselben Nutzers zwischen den Display-Gruppen kompensieren. Abbildung 3.3(d) zeigt eine Situation, in der die Inhalte eines Displays in den Darstellungsbereich eines anderen Displays hineinreichen. Da diese Interferenz störende Effekte hervorrufen kann, sollen solche Inhalte an den Displaygrenzen abgeschnitten werden können.



(a) Der gelb markierte Nutzer sieht auf Leinwand und Tisch die Hauptnavigation der jeweiligen Display-Gruppe.



(b) Im ersten Spezialfall sieht der gelb markierte Nutzer nur auf dem Tabletop-Display dessen Hauptnavigation. Die Navigation des Leinwand-Displays ist so eingestellt, dass die beiden Displays für ihn gekoppelt erscheinen.



(c) *(Fotomontage)* Im zweiten Spezialfall ist es erwünscht, für Nutzer hinter dem Tabletop-Display dessen Ansicht auf dem Leinwand-Display zusätzlich zur Hauptnavigation darzustellen.

(d) Inhalte des Leinwand-Displays interferieren mit dem Darstellungsbereich des Tabletop-Displays und sollen im Überlappungsbereich abgeschnitten werden. Im vorliegenden Fall ragt die Säule des Leinwand-Displays in die Inhalte auf dem Tabletop-Display.

Abbildung 3.3: Überlappende Territorien in einem Setup aus zwei physikalischen, nicht räumlich gekoppelten Displays.

3.4 Kontinuierlicher Raum für Werkzeuge

Die bisher definierten Anforderungen an das Softwaresystem zeigen wichtige Askpekte des Viewing Setups auf. Es soll jedoch den Nutzern weiterhin ermöglicht werden, die wahrgenommenen Inhalte der Szene durch *Werkzeuge* modifizieren und manipulieren zu können. Zusätzlich dazu sollen bestimmte Werkzeugtypen, wie in Anforderung 3.2.1 beschrieben, zur Verwaltung virtueller Sichten beitragen.

Anforderung 3.4.1: Innerhalb eines Workspaces sollen Werkzeuge definierbar sein, welche zur Laufzeit der Applikation von allen Nutzern des Workspaces verwendet werden können.

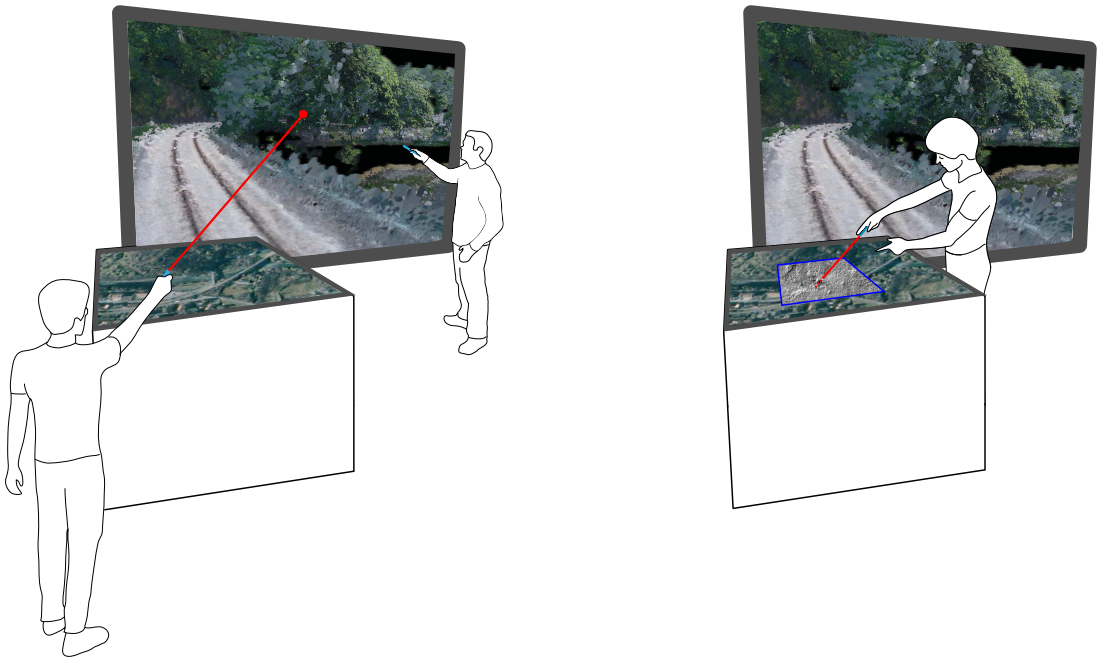
Denkbare Werkzeuge sind an dieser Stelle klassische Beispiele wie Ray-Pointer und Virtual Hands [31], Trackingtargets mit angebundenen virtuellen Sichten (wie zum Beispiel die WIM [12]) und viele mehr. Die Bedienung dieser ist auf den Workspace beschränkt, die entsprechende Werkzeugvisualisierung und die ausgeführten Aktionen sind jedoch für alle Workspaces sichtbar.

Anforderung 3.4.2: Die Werkzeuge sollen kontinuierlich innerhalb des Workspaces funktionieren.

Dies bedeutet, dass Werkzeuge auf Displays aller Display-Gruppen angezeigt werden sollen, unabhängig davon, an welcher Stelle sich in der virtuellen Welt die jeweiligen Navigationen befinden. Dabei impliziert die Anforderung ebenfalls, dass Werkzeuge Objekte der Szene zwischen den Display-Gruppen des Workspaces transportieren können. Abbildung 3.4(a) zeigt einen Fall, in dem ein Strahl auf dem Tabletop-Display beginnt und auf dem Leinwand-Display kontinuierlich fortgesetzt wird, da dieser in allen Display-Gruppen repräsentiert ist.

Anforderung 3.4.3: Manipulationswerkzeuge sollen auch innerhalb virtueller Sichten benutzbar sein.

Dies erweitert die Anforderung des kontinuierlichen Raums für Werkzeuge zur Manipulation von Szeneinhalten auf virtuelle Sichten und erfordert, dass diese in virtuelle Displays, WIMs oder andere Konstrukte zur Realisierung virtueller Sichten transformiert werden können. Im Beispielfall von Abbildung 3.4(b) verwendet ein Nutzer ein Strahlwerkzeug, um in einem Portal eine Manipulation an einer anderen Stelle in der virtuellen Welt vorzunehmen.



(a) Trotz verschiedener Display-Gruppen wird das Strahl-Manipulationswerkzeug kontinuierlich auf dem Leinwand-Display fortgeführt.

(b) Der Strahl behandelt das virtuelle Display nicht als planare Fläche, sondern wird an die angezeigte Stelle transformiert und kann nun dort verwendet werden.

Abbildung 3.4: Zwei Anforderungsszenarien an im Workspace definierte Werkzeuge.

3.5 Regelung von Sichtbarkeiten

Anforderung 3.5.1: Trotz des kontinuierlichen Raums für Werkzeuge soll die Möglichkeit bestehen, Werkzeugvisualisierungen für bestimmte Nutzer und Display-Gruppen zu modifizieren oder unsichtbar zu schalten.

Auch wenn von dieser Möglichkeit nicht immer Gebrauch gemacht werden muss, ergibt diese in bestimmten Situationen durchaus Sinn. Würde in Abbildung 3.4(a) beispielweise der Strahl bereits auf dem Tabeltop-Display ein Objekt schneiden, so wäre es unerwünscht, die Strahlvisualisierung auf dem Leinwand-Display fortzuführen. Vielmehr soll der Strahl nach dem Schnittpunkt abgeschnitten und auf keiner weiteren Display-Gruppe mehr angezeigt werden. Auf Nutzerebene kann es sinnvoll erscheinen, Visualisierungen unterschiedliche Farben in Abhängigkeit des bedienenden Nutzers oder der angezeigten Display-Gruppe zuzuweisen.

Anforderung 3.5.2: Ausgezeichnete Szeneelemente sollen nur auf bestimmten Display-Gruppen oder für bestimmte Nutzer angezeigt werden.

Somit lassen sich beispielsweise ohne die Verwendung von virtuellen Sichten für bestimmte Nutzer oder in bestimmten Display-Gruppen zusätzliche Informationen innerhalb der Szene anzeigen oder ausblenden. Dies kann insbesondere dann hilfreich sein, wenn die beteiligten Nutzer aus verschiedenen Motivationen bzw. in unterschiedlichen Rollen die Szene erkunden und somit an unterschiedlichen Details interessiert sind.

3.6 Lokale und entfernte Nutzergruppen

Anforderung 3.6.1: Das System soll offen für mehrere, nicht zwingend lokale Workspaces sein.

Dies impliziert, dass die Nutzer aller Workspaces als Avatare repräsentiert werden, so dass sich mehrere Parteien in der virtuellen Welt begegnen können (*Group-to-Group-System* [9]). Manipulationen durch Werkzeuge eines Workspaces sollen für die Nutzer anderer Workspaces durch entsprechende Visualisierungen verständlich nachvollziehbar sein. Selbiges gilt für die Erzeugung, Verwendung und Manipulation virtueller Sichten.

3.7 Konfigurierbar- und Erweiterbarkeit

Anforderung 3.7.1: Das System soll konfigurierbar und erweiterbar sein.

Applikationsentwickler haben somit die Möglichkeit, beliebige Setups aus physikalischen Displays, virtuellen Displays und Werkzeugen in Konfigurationsdateien zu spezifizieren. Diese Konfigurationsdateien kapseln die Applikationslogik ab und werden beim Start des Systems automatisch in den benötigten Szenegraph konvertiert. Auf Quelltextebene sollen generische Konzepte wie Vererbung angewendet werden, damit eine leichte Erweiterbarkeit um beispielsweise eigene Werkzeugtypen gegeben ist.

4 Verwandte Arbeiten

Nach einer Darstellung der inhaltlichen Anforderungen an Applikationen für Multi-Display-Umgebungen aus physikalischen und virtuellen 3D-Multi-User-Displays im Kontext der Virtuellen Realität soll in diesem Kapitel auf verwandte Arbeiten zu diesem Thema eingegangen werden. Die Forschung im Bereich der 3D-Multi-User-Displays ist dabei sehr eingeschränkt; die perspektivisch korrekten 3D-Displays für mehr als zwei Nutzer sind sogar unikal an unserer Universität. Es soll daher eine für die beschriebenen Anwendungsfälle relevante Auswahl an verwandten Arbeiten zu Multi-Display-Umgebungen beliebiger Art vorgestellt werden. Ein allgemeiner gehaltener, vom Standpunkt verteilter Nutzer-Interfaces betrachteter Überblick wird von Elmqvist in [13] dargestellt.

4.1 Multi-Display-Setups

Multi-Display-Umgebungen reichen von einfachen privaten erweiterten Desktops auf zwei Displays bis hin zu immersiven Installationen wie der CAVE [12], welche aus vielen physikalisch nebeneinander angeordneten und virtuell gekoppelten Displayflächen besteht.

Neben der CAVE zielen einige weitere Systeme darauf ab, Übergänge zwischen den Einzeldisplays kontinuierlich abzubilden. 1999 präsentierten Rekimoto und Saitoh [33] ein System, welches optische Trackingverfahren nutzt, um mobile Displays innerhalb eines Workspaces zu lokalisieren und somit kontinuierliche Übergänge von Cursor und Fenstern zwischen diesen portablen und zusätzlichen stationären Displays zu ermöglichen. Pirchheim et al. [30] erweitern diese Idee im System *Deskotheque*, indem sie durch Kalibrierung die räumlichen Beziehungen und projektiven Verzerrungen aller Displayflächen analysieren, um diese bei Übergängen von Inhalten zu berücksichtigen. Das System *E-Conic* [27]

berücksichtigt ebenfalls projektive Transformationen und trackt Nutzer und Displays, um perspektivisch korrekte Cursor [28] und Fenster anzuzeigen.

Auch in der Unterhaltungs- und Spieleindustrie finden auf Grund des sinkenden Produktionspreises immer häufiger Systeme mit mehreren Einzeldisplays Anwendung. Ein Beispiel hierfür ist der portable Nintendo DS / 3DS [3], welcher seine zwei Displays nutzt, um die virtuelle Spielewelt entweder kontinuierlich oder separat auf diesen darzustellen. Im letzteren Fall dient das Sekundärdisplay entweder zur Anzeige einer Übersichtskarte oder durch seine Touchfähigkeiten als Eingabegerät für Einstellungen und Aktionen, welche keinem der Buttons auf dem Gerät zugeordnet wurden.

Eine prominente Beschreibung einer weiteren Multi-Display-Umgebung mit potentiell separaten Displayinhalten ist das *i-LAND* [42], welches den Austausch von Informationen zwischen zweidimensionalen Leindwand-Displays, Tabletop-Displays und *CommChairs*, speziellen Sitzmöbeln mit integrierten Displays, ermöglicht. Dabei sind Displays keine kontinuierliche Fortsetzung des Arbeitsbereichs anderer *Roomware*, sondern dienen als individuelle Möglichkeit zur Verarbeitung von Informationen. Forlines et al. [14] erweitern eine klassische Einbenutzeranwendung zur Visualisierung geografischer Daten auf zwei Leinwand- und ein Tabletop-Display. Alle Displays zeigen dabei denselben geografischen Ort aus unterschiedlichen Perspektiven. Der Tisch bietet sich hierfür zur Anzeige der Übersichtskarte an, während die 3D- und gröber gehaltene Straßenkarten-Ansicht auf den Leinwand-Displays dargestellt und durch Touchinput auf dem Tabletop- oder zusätzlichen portablen Touchdisplays gesteuert werden können. Dasselbe physikalische Setup benutzten Forlines und Lilien [15] zwei Jahre später zur Darstellung verschiedener Semantiken visualisierter Moleküle auf unterschiedlichen Displays. Für ein spontanes Aufeinandertreffen von Nutzern beschreiben Nacenta et al. [26] die Umgebung des *Lunch-Table*, in welchem sechs physikalische Displays nebeneinander an einer Wand montiert wurden, um den Eindruck eines größeren Einzeldisplays zu verleihen (*Tiled Wall*). Diese Displays können untereinander beliebig miteinander verkoppelt werden, um gewissen Inhalten einen größeren Teil der gesamten Displayfläche zu gewähren. Die Kontrolle der Inhalte und Kopplungsvorgänge wird dabei an einem Tabletop-Display mit Touchinput vorgenommen.

4.2 3D-Multi-User-Technologien

Agrawala et al. präsentierten im Jahre 1997 die *Two-User Responsive Workbench* [7], eine Weiterentwicklung der *Responsive Workbench* [19], welche die Darstellbarkeit unterschiedlicher Inhalte für zwei Nutzer auf demselben Tabeletop-Display durch zeitliches Multiplexing ermöglicht. Dabei wird ein *Shared 3D-Space* erzeugt, so dass die dreidimensionalen Inhalte für die Nutzer perspektivisch korrekt dargestellt werden, sich jedoch applikationsabhängig in ihrer Semantik unterscheiden können. Das System *IllusionHole* [18] erzeugt durch die geschickte Positionierung mehrerer 3D-Single-User-Displays den Eindruck eines Multi-User-Displays. Die Einzeldisplays werden dabei mit einer Maske so verdeckt, dass drei Nutzer aus verschiedenen Positionen je nur eines der Displays sehen können, womit ihre Bewegungsfreiheit jedoch einschränkt ist. Durch Tracking können dann die perspektivisch korrekten Inhalte berechnet und auf den korrespondierenden Einzeldisplays angezeigt werden. *Volumetrische Displays*, wie zum Beispiel in [17] analysiert, bestehen aus einer drehbaren semi-transparenten Bildfläche, welche durch ihre Rotation Bildpunkte direkt im Raum projizieren kann und somit keine Brillen für den 3D- und Multi-User-Effekt benötigt. Jedoch muss bedacht werden, dass auf Grund der ständigen Drehung kein physikalisches Zeigen innerhalb des erzeugten Shared 3D-Spaces möglich ist, was durch den Einsatz von virtuellen Werkzeugen kompensiert werden kann.

Im Bereich der *Augmented Reality* erzeugt das System *Studierstube* [34] von Schmalstieg et al. den Shared 3D-Space durch semi-transparente *Head-mounted displays*. Dabei werden virtuelle Inhalte an physikalische Panels gebunden und den Nutzern perspektivisch korrekt auf den Displays ihrer Brillen angezeigt. Im Rahmen von Group-to-Group-Interaktionen wurde ebenfalls mit zeitsequentiellen projizierten Stereo-Displayflächen experimentiert, welche jedoch lediglich die Perspektive eines einzelnen Nutzers korrekt anzeigen können. Zusätzliche Nutzer sehen exakt dasselbe Bild und müssen so abhängig von ihrer Position mit Verzerrungen rechnen.

Kulik et al. [20] beschreiben unter dem Stichwort *C1x6* (eine große Projektionswand für sechs Nutzer) die hardwaretechnische Realisierung des 3D-Multi-User Leinwand-Displays in unserem Labor, welches sechs Nutzer mit perspektivisch korrekten Stereobildern versorgen kann. Weiterhin wird auf den Spheron als Eingabegerät und erweiterte Gruppennavigationstechniken eingegangen. Diese können temporär die perspektivisch korrekte Abbildung der Nutzer in der virtuellen Welt aufheben, um dortige Kollisionen mit Hindernissen zu vermeiden. Dabei wird der Effekt des Shared 3D-Spaces aufgehoben, so

dass kein nachvollziehbares physikalisches Zeigen auf Objekte der virtuellen Welt mehr möglich ist.

Das beschriebene Single-Display-Multi-User-Setup wird von Beck et al. [9] durch das zweite, am anderen Ende unseres Labors befindliche, 3D-Multi-User Leinwand-Display in unserem Labor zu einem Telepräsenzsystem erweitert. Nutzer beider Workspaces werden in der virtuellen Welt als Video-Avatare repräsentiert und können gemeinsam die Szene erkunden. Eine an ein Trackingtarget gebundene World in Miniature dient zur Wegplanung und erlaubt die Manipulation der Gruppenpositionen mit Hilfe eines Strahlwerkzeuges.

4.3 Interaktionskonzepte

Interaktionstechniken in Multi-Display-Umgebungen, wie der Informationsaustausch zwischen den Einzeldisplays oder die Erzeugung und Verwaltung virtueller Sichten, ergeben sich einerseits aus der konkreten Beschreibung bestimmter Displaysetups, stellen andererseits jedoch auch einen isoliert davon betrachteten Forschungsschwerpunkt dar.

Zur Klassifizierung von Techniken zum Austausch von Inhalten zwischen den einzelnen Displays in 2D-Multi-Display-Umgebungen stellen Nacenta et al. [24] eine Taxonomie vor. In einer Vorarbeit von Nacenta et al. [25] wurden sechs verschiedene Techniken aus unterschiedlichen Bereichen dieser Taxonomie am Beispiel des Transfers von Inhalten zwischen einem 2D-Single-User-Tablet- und einem 2D-Single-User-Tabletop-Display analysiert. Dabei stellte sich unter anderem heraus, dass durchgeführte Interaktionen dem Nutzer stets Feedback geben und die *Control-Display-Ratio* idealerweise bei eins liegen sollte.

Ein interessantes Konzept, welches diese Anforderung erfüllt, findet sich in der Beschreibung des i-LANDs [42]. *Passenger* sind Objekte in der realen Welt, an welche virtuelle Informationen gebunden und somit nachvollziehbar physikalisch zwischen den Displays transportiert werden können. Auch Reikimoto und Saitoh [33] verwenden dieses Konzept in ihrem getrackten Workspace. Bader et al. [8] verzichten im *Lift-and-Drop*-Ansatz auf die Verwendung von Requisiten und wenden die Trackingverfahren direkt auf die Hände der Nutzer an. Weiterhin ergänzen sie den *Pick-and-Drop*-Ansatz von Rekimoto [32] um

kontinuierliches Feedback auf den Displays während des Vorgangs. Im *LightSpace* von Wilson und Benko [45] kann mit beiden Armen eine Brücke zwischen Displays geschaffen werden, um Inhalte zwischen diesen auszutauschen. Marquardt et al. [22] analysieren bei der Entwicklung des *Proximity Toolkits* zahlreiche proxemische Relationen zwischen Nutzern, Displays und nicht-digitalen Objekten im Workspace. Aus den erlangten Daten werden in auf dem Toolkit aufbauenden Anwendungen semantische Informationen über Beziehungen zwischen den erfassten Entitäten und die daraus resultierenden möglichen Intentionen der Nutzer abgeleitet und im Rahmen der Applikationslogik sinnvoll verarbeitet.

Scott et al. [36] beobachteten bei der Durchführung von zwei Studien das Interaktionsverhalten von Nutzern bei Kollaborationssituationen in einer Gruppe. Dabei zeigten sich drei Formen von *Territorien*: persönliche Territorien eines jeden Nutzers, ein geteiltes Territorium der Gruppe und ein Territorium zur Lagerung aufgabenbezogener und aufgabenunabhängiger Objekte. Insbesondere die Aufteilung in persönliche und Gruppen-Territorien wird von vielen Systemen implementiert, wie zum Beispiel im Konzept der Fotoportale von Kunert et al. [21] zur Erzeugung und Verwaltung virtueller Displays. Dabei wird die Metapher eines Fotoapparats ausgenutzt, um interaktiv Aufnahmen von Szeneinhalten zu erzeugen. Ein Foto kann hierbei privat gehalten, weitergegeben, mit anderen Nutzern geteilt und in Form einer Galerie für die gesamte Gruppe zugänglich gemacht werden. Zum Transfer von Objekten in und aus den virtuellen Displays wird das von 2D-Single-Display-Systemen bekannte *Drag-and-Drop*-Verfahren auf Ray-Pointer im dreidimensionalen Raum angewendet.

Virtuelle Sichten ohne die Verwendung virtueller Displays lassen sich mit der bereits erwähnten *World in Miniature* [40], einem Abbild der virtuellen Welt im Maßstab zur besseren Übersicht, realisieren. In Kombination mit *Passengern* kann sie in Multi-Display-Umgebungen physikalisch im Workspace transportiert werden und somit auf allen Einzeldisplays inspiziert werden.

4.4 Diskussion

Die zwei letzten in Abschnitt 4.2 beschriebenen Arbeiten geben einen Einblick in die bestehenden Multi-User-Systeme unseres Labors mit einem 3D-Multi-User-Display pro Workspace. Auf dem Gebiet der Kombination mehrerer 3D-Multi-User-Displays innerhalb eines Workspaces, wie es bei dem aktuellen Leinwand-Tabletop-Setup in unserem Labor der Fall ist, gibt es noch keine verwandten Arbeiten. Es ist daher Ziel, die bestehenden Technologien als Basis einer Applikation für Multi-Display-Umgebungen innerhalb verschiedener Workspaces zu übertragen und sinnvoll zu ergänzen. Dazu zählen auch die Fotoportale, welche die dynamische Erzeugung und Verwaltung virtueller Displays zur Laufzeit ermöglichen. Die Portalkamera dient hierbei als Passenger und realisiert somit Austausch virtueller Displays zwischen den physikalischen Displays.

Als effektives Manipulationswerkzeug zur Bewegung von Szeneobjekten zeigt sich die 3D-Adaption von Drag-and-Drop mit Hilfe eines Ray-Pointers, da dieser während eines Transformationsvorgangs konstant Feedback gibt und die Bewegungen der Nutzer im Verhältnis 1:1 auf dem Display abbildet. Bei Übergängen zwischen Displays gilt es bei Bedarf, die Objektskalierung entsprechend anzupassen.

Aus den dargestellten Beispielfällen in Abschnitt 4.1 spiegeln sich die unterschiedlichen Displaykopplungsvarianten wieder. Einige Systeme wie Deskotheque oder die CAVE zielen darauf ab, die Displayübergänge geometrisch kontinuierlich erscheinen zu lassen, während die angezeigten Sichtweisen in Systemen wie Forlines' Visualisierungsapplikationen unterschiedlich sind und nur semantisch miteinander zusammenhängen. Auch wenn die genannten Beispiele nicht aus dem Bereich der 3D-Multi-User-Systeme stammen, ergeben sich auf ihrer Basis zwei Implementierungsstrategien für die angestrebte Softwarestruktur dieser Arbeit. Die erste Strategie baut den Szenegraph so auf, dass die virtuelle Welt kontinuierlich auf allen physikalischen Displays des Workspaces abgebildet wird und die Displays somit standardmäßig gekoppelt erscheinen. Innerhalb der zweiten Strategie wird eine separate Abbildung aller physikalischer Displays eines Workspaces im Szenegraph als Standard vorgesehen. In den folgenden zwei Kapiteln 5 und 6 sollen diese Implementierungsstrategien beschrieben und mit der Realisierbarkeit der in Kapitel 3 dargestellten Anforderungen abgeglichen werden.

5 Kontinuierliche Abbildung der Displays des Workspaces

Innerhalb des letzten Kapitels wurden mit Blick auf die verwandten Arbeiten zwei Implementierungsstrategien für die angestrebte Softwarestruktur definiert. In diesem Kapitel soll nun Augenmerk auf den ersten dieser beiden Ansätze gelegt werden, bei welchem alle Displays eines Workspaces in der Applikation räumlich kontinuierlich abgebildet werden und somit standardmäßig gekoppelt erscheinen. Hierzu soll zunächst das entsprechende Viewing Setup für Nutzer und Werkzeuge erläutert und anschließend mit Hilfe der Anforderungen aus Kapitel 3 evaluiert werden.

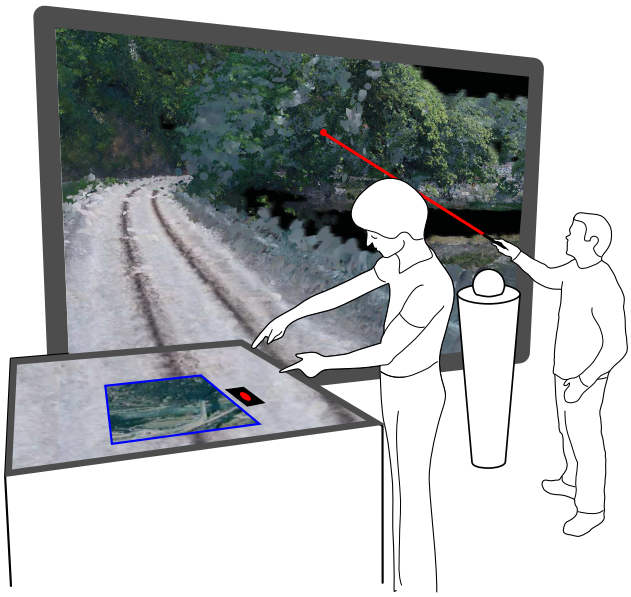


Abbildung 5.1: Ein Workspace mit zwei Nutzern, einem Ray-Pointer, einem physikalischem Leinwand-Display, einem physikalischem Tabletop-Display und einem virtuellen Display (angehangen an eine Portalkamera), welches auf dem Tabletop-Display abgelegt wurde. Ziel ist es, diese Situation im Szenegraph abzubilden.

5.1 Viewing Setup im Szenegraph

Abbildung 5.1 zeigt eine typische Beispielsituation aus zwei physikalischen und einem virtuellen Display. Dem Workspace sind weiterhin zwei Nutzer, eine Portalkamera nach Vorbild der Fotoportale [21] und ein Ray-Pointer zugeordnet. Im Folgenden soll diese Situation im Rahmen des in diesem Kapitel diskutierten kontinuierlichen Ansatzes im Szenegraph abgebildet werden.

5.1.1 Physikalische Displays

In Abbildung 5.2 ist ein Szenegraph dargestellt, aus welchem alle relevanten Informationen zur Berechnung der perspektivisch korrekten Bilder der Nutzer aus Abbildung 5.1 auf den physikalischen Displays bezogen werden können.

Um die physikalischen Displays des Workspaces kontinuierlich abzubilden, so dass diese gekoppelt erscheinen, müssen die entsprechenden softwareseitigen Repräsentanten der Displays, die *Screen-Knoten* des Szenegraphs, mit den spezifischen Transformationsmatrizen im Workspace gefüllt und unterhalb desselben Navigationsknoten angehängt werden. Daher wird zunächst unterhalb des Wurzelknotens ein Navigationsknoten (*workspace_0*) für den Workspace erzeugt, mit einem Eingabegerät (im dargestellten Fall dem Spheron [20]) verbunden und die Screen-Knoten darunter angehängt. Dies hat zur Folge, dass dem Workspace nur eine Navigation zugewiesen werden kann.

Um nun die perspektivisch korrekten Bilder für die Nutzer zu berechnen und auf den Displays anzuzeigen, benötigen auch diese eine softwareseitige Repräsentation. Diese besteht zum einen aus einem Knoten, wessen Transformation mit der Trackinginformation der Shutterbrille des Nutzers verbunden wird und somit seine Kopfposition und -orientierung beinhaltet, zum anderen aus zwei darunterliegenden Knoten zur Repräsentation der Augen, welche im entsprechenden Augenabstand voneinander entfernt sind. Optional können unter dem Kopfknoten des Nutzers noch Geometrien zur Realisierung von Avataren in der virtuellen Welt angehängt werden. Durch die Kopf- und Screen-Knoten sind alle Informationen vorhanden, um individuelle *Model-View-Projection*-Transformationen pro Nutzer und Auge durchführen zu können.

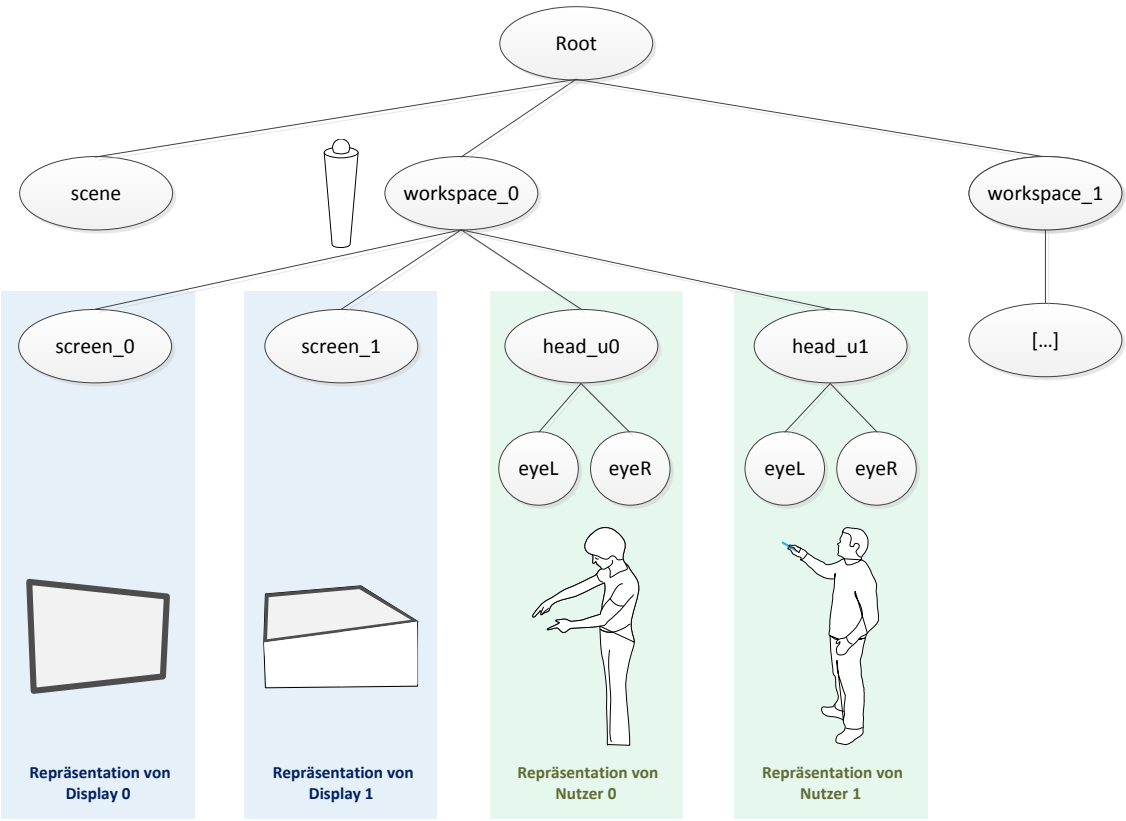


Abbildung 5.2: Szenegraph des Viewing Setups für die Nutzer und physikalischen Displays aus Abbildung 5.1 (durch beliebige Workspaces erweiterbar). Der *scene*-Knoten steht repräsentativ für alle Geometrieknoten der Objekte der Szene.

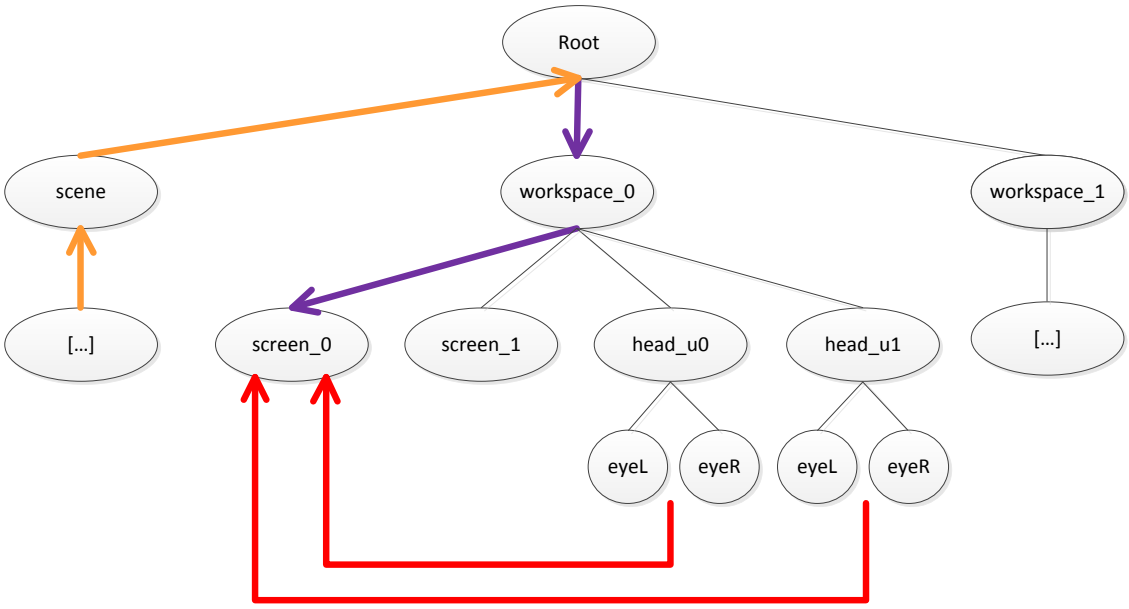


Abbildung 5.3: Berechnung der Matrizen des Viewing Setups für das Leinwand-Display aus Abbildung 5.1. Die *Model*-Transformation (orange) berechnet die Objektmatrizen in Weltkoordinaten, die *View*-Transformation (lila) drückt diese im Koordinatensystem des Screen-Knotens aus. Im Projektionsschritt (rot) wird das Frustum jedes Nutzers aufgebaut und gerendert.

Abbildung 5.3 zeigt diesen Transformationsprozess am Beispiel des Leinwand-Displays aus Abbildung 5.1. Dabei werden zunächst im Rahmen der *Model*-Transformation (orange) alle Objekte der Szene in Weltkoordinaten transformiert, um diese dann mit Hilfe der *View*-Transformation (lila) im Koordinatensystem des Screen-Knotens auszudrücken. Schließlich wird im *Projektionsschritt* (rot) das asymmetrische Kamerafrustum jedes Nutzers mit Hilfe der Augenknoten im Koordinatensystem des Screen-Knotens berechnet. Mit Hilfe des Frustums kann die entsprechende Projektionsvorschrift bestimmt werden und damit die dreidimensionalen Objekte der Szene perspektivisch korrekt auf der 2D-Projektionsfläche des Displays abgebildet werden.

5.1.2 Virtuelle Displays

Da virtuelle Displays nicht in der realen Welt verankert sind, sollen sie nicht nur für die Nutzer eines bestimmten physikalischen Workspaces sichtbar sein. Die Umsetzung eines virtuellen Displays im Szenegraph darf somit nicht unterhalb eines Workspace-Navigationsknotens geschehen. Abbildung 5.4 zeigt eine Erweiterung des Szenegraph aus Abbildung 5.2 im Hinblick auf das virtuelle Display, welches in Abbildung 5.1 auf dem Tisch dargestellt wird.

Unterhalb des Wurzelknotens wird ein logischer Gruppierungsknoten für virtuelle Displays (*virtual_displays*) eingeführt, welcher keine Transformation enthält. Darunter werden die benötigten Knoten für in der Szene vorhandene virtuelle Displays angehängt. Ein Teilbaum zur Realisierung eines solchen virtuellen Displays beginnt ebenfalls mit einem logischen Gruppierungsknoten. An diesem sind zwei Knoten für die Eingangs- (*entry*) und Ausgangstransformation (*exit*) angehängt. Dabei beschreibt die Eingangsmatrix jene Position und Orientierung in Weltkoordinaten, an welcher das virtuelle Display beispielsweise als Textur dargestellt wird. Die Ausgangsmatrix hingegen entspricht der im virtuellen Display abgebildeten Navigation. Im Portalfall unterscheiden sich diese beiden Matrizen, während im Falle der Linse diese hingegen identisch sind.

Damit das virtuelle Display nicht nur ein statisches Bild anzeigt, sondern sich wie ein Sichtfenster zur Ausgangsorientierung verhält, muss das geometrische Verhältnis der Nutzer zur Eingangsmatrix am Ausgang nachgebildet werden. Dazu hängt unter dem Ausgangsknoten analog zur Beschreibung der physikalischen Displays der Screen-Knoten als Repräsentant des virtuellen Displays und die Nutzerrepräsentationen, nur dass in diesem Fall für alle Nutzer aller Workspaces eine Repräsentation vorgesehen wird. Jedoch können in diesem Fall die Transformationsmatrizen der Repräsentation der jeweiligen Kopfknoten im Portal nicht direkt aus den Trackinginformationen verbunden werden. Um die individuelle Betrachtungsperspektive eines Nutzers in ein virtuelles Display hinein abzubilden, muss hingegen das Verhältnis des Kopfes zum Eingang und damit das entsprechende Kamerafrustum am Ausgang nachgebildet werden. Das bedeutet, dass der Offset zwischen Kopf und Eingang berechnet und unter dem Ausgangsknoten verwendet wird:

$$head_wx_uy = entry^{-1} \cdot world_transform(head_uy)$$

Dabei bezeichnet *head_wx_uy* die Transformationsmatrix des Szenegraphknoten von Nutzer y aus Workspace x unter dem Ausgangsknoten des virtuellen Displays, *entry* die Transformationsmatrix des Eingangs des virtuellen Displays und *world_transform(head_uy)* die in `/workspace_x/head_uy` gespeicherte Trackinginformation von Nutzer y aus Workspace x in Weltkoordinaten (in Abbildung 5.4 aus Platzgründen nicht dargestellt).

Somit sind alle Informationen vorhanden, um für jeden Nutzer das perspektivisch korrekte Bild des virtuellen Displays zu berechnen. Dazu werden in Prärenderpässen die Bilder aller virtueller Displays berechnet und in individuelle Texturen gerendert. Wenn im virtuellen Display andere semantische Inhalte dargestellt werden sollen, können entsprechende Einstellungen am Prärenderpass vorgenommen werden. Um die Texturen dann im Haupt-Renderpass an der Eingangstransformation anzuzeigen, werden sie im Szenegraph unter den Eingangsknoten des virtuellen Displays angehängt und so maskiert, dass nur der zugehörige Nutzer diese sehen kann.

Sollen mehrere virtuelle Displays untereinander kontinuierlich abgebildet werden, so dass diese wie beispielsweise bei den *Magic Boxes* [16] gekoppelt wirken, können unterschiedliche Screen-Knoten mit ihren spezifischen Transformationen analog zum physikalischen Displaysetup unterhalb des Ausgangsknotens angehängt werden.

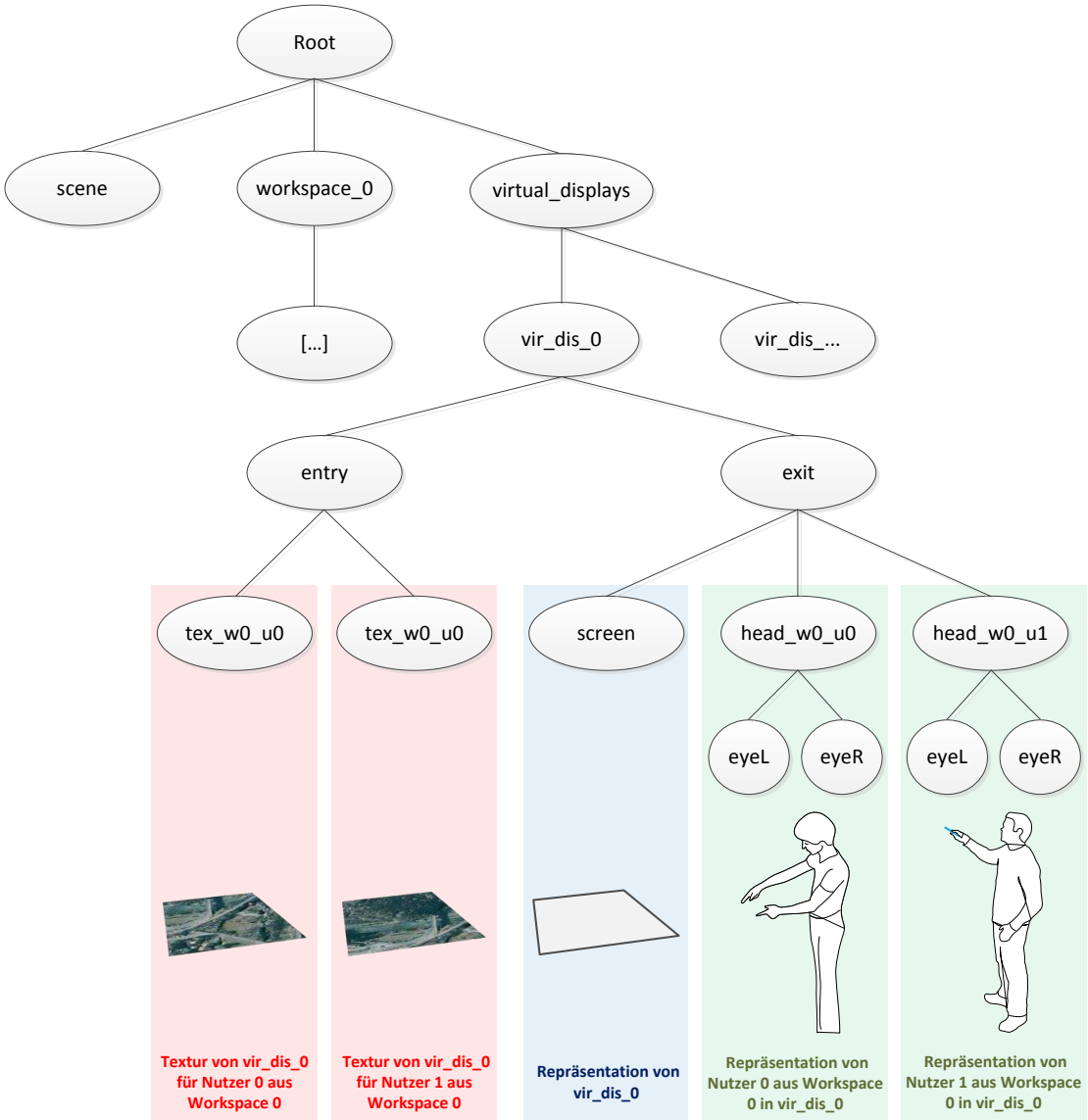


Abbildung 5.4: Abbildung eines virtuellen Displays im Szenegraph. Der dargestellte Szeengraph bezieht sich auf das auf dem Tisch befindliche virtuelle Display aus Abbildung 5.1.

5.2 Werkzeuge und ihre Repräsentanten

Die einzigen Elemente aus Abbildung 5.1, welche im vorherigen Abschnitt noch nicht im Szenegraph abgebildet wurden, sind die Werkzeuge, namentlich Ray-Pointer und Portalkameras. Werkzeuge sind, analog zu den physikalischen Displays und den Nutzern, einem bestimmten Workspace zugeordnet. Daher benötigen sie ebenfalls eine Repräsentation unterhalb des Workspace-Navigationsknotens, wie in Abbildung 5.5 am Beispielfall illustriert ist.

Jedes Werkzeug hat dabei einen Knoten im Szenegraph, wessen Transformationsmatrix mit den Trackingdaten des entsprechenden physikalischen Geräts verbunden wird. Darunter können individuelle Geometrieknoten zur Werkzeugvisualisierung in der virtuellen Welt hängen.

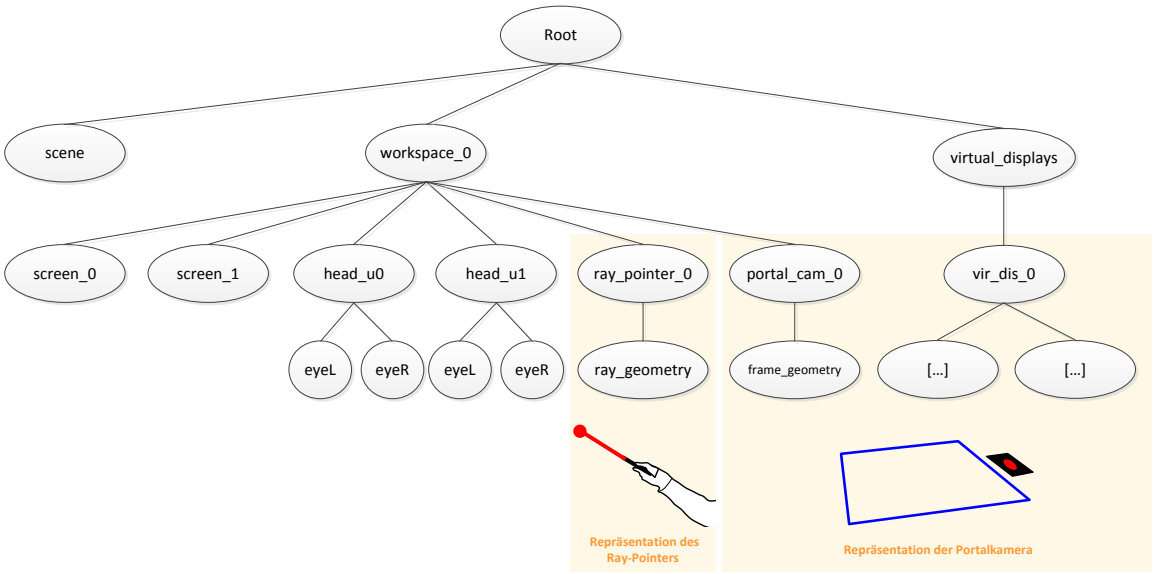


Abbildung 5.5: Repräsentation des Ray-Pointers und der Portalkamera aus Abbildung 5.1 im Szenegraph.

Einen Sonderfall des Werkzeugs stellt dabei die Portalkamera dar. Diese benötigt zusätzlich zur Visualisierungsgeometrie ihres Rahmens ein zugeordnetes virtuelles Display, um die produzierten Aufnahmen darstellen zu können. Da diese im Zuge der Kollaboration auch für andere Workspaces in der virtuellen Welt sichtbar sein sollen, wird das virtuelle Display

analog zur Erläuterung in Abschnitt 5.1.2 oberhalb des Workspace-Navigationsknotens im Szenegraph erzeugt und verwaltet. Der einzige Unterschied besteht darin, dass die Eingangstransformation nicht statisch gehalten, sondern mit den Trackingdaten der Portalkamera verbunden wird, so dass der Eingang stets über dieser erscheint und somit im Workspace frei umhergetragen werden kann.

Die genauen Implementierungsdetails der Funktionen von Portalkamera und Ray-Pointer werden aus Gründen der Übersichtlichkeit in dieser Arbeit nicht genauer erläutert. Sie wurden in Form einer Code-Dokumentation dem Applikationsquelltext beigefügt.

5.3 Separate Navigationen

In Anforderung 3.1.1 wurde beschrieben, dass die Displays des Workspaces beliebig miteinander ver- oder entkoppelbar sein sollen. Im bisher beschriebenen Aufbau ist dies jedoch nicht möglich, da alle physikalischen Displays untereinander kontinuierlich abgebildet werden und somit zu jedem Zeitpunkt gekoppelt sind. Dies hat am Beispiel der Situation aus Abbildung 5.1 zur Folge, dass das Tabletop-Display auf Grund seiner physikalischen Transformation auch in der virtuellen Welt nach unten auf den Boden blickt.

Eine Lösung, welche innerhalb dieses Ansatzes getrennte Navigationen ermöglicht, ist die Verwendung von virtuellen Displays. Dabei wird ein virtuelles Display dynamisch platziert, so dass es stets die komplette Darstellungsfläche eines physikalischen Displays einnimmt. Ein zugeordnetes Eingabegerät kann dann die Navigation (die Ausgangsmatrix) des virtuellen Displays steuern. Im Folgenden soll eine implementierte Strategie zur Realisierung dieses Konzepts mit Hilfe einer Portalkamera geschildert werden.

Die Portalkamera ermöglicht es, Aufnahmen der Szene zu produzieren und diese in Form eines über ihr schwebenden virtuellen Displays anzuzeigen. Für die Portalkamera können im Workspace bestimmte Interaktionsbereiche definiert werden, an welche sie durch einen Buttondruck eine dieser Aufnahmen übergeben kann. Der Interaktionsbereich handhabt die Anzeige seines virtuellen Displays dann separat. In Abbildung 5.6(a) wurde ein solcher Interaktionsbereich in Form eines Quaders über dem Tabletop-Display definiert.

(a) In ausgezeichneten Interaktionsbereichen (hier grün markiert) kann mit einem Buttondruck auf der Portalkamera die Aufnahme übergeben werden.

(b) Die Aufnahme des Interaktionsbereiches wird maximiert dargestellt und ist nun nicht mehr Teil der Portalkamera. Der Inhalt kann mit dem zugeordneten Spacemouse-Navigationsgerät verändert werden.

Abbildung 5.6: Eine Strategie zur Realisierung von getrennten Navigationen, obwohl die physikalischen Displays kontinuierlich abgebildet sind.

Dieser Interaktionsbereich wurde so eingestellt, dass er übergebene virtuelle Displays auf die Größe seiner Bodenfläche, in diesem Fall also die gesamte Displayfläche des Tabletop-Displays, maximiert. Nach einem Buttondruck auf der Portalkamera in Abbildung 5.6(a) wird die Textur des virtuellen Displays dann nachvollziehbar zum Stand in Abbildung 5.6(b) animiert. Die Aufnahme ist nun nicht mehr Teil der Portalkamera, sondern des Interaktionsbereichs. Eine diesem zugeordnete Spacemouse-Navigation kann nun den Inhalt der Aufnahme separat modifizieren. Möchte der Nutzer die Aufnahme zurück in die Portalkamera bewegen, so genügt ein erneuter Buttondruck innerhalb des Interaktionsbereichs.

5.4 Diskussion

Dem im Rahmen dieses Kapitels dargestellten Ansatz liegt ein leicht nachvollziehbares mentales Modell zu Grunde. Durch die kontinuierliche Abbildung der physikalischen Displays benötigen diese, ebenso wie Nutzer und Werkzeuge, nur eine Repräsentation innerhalb des Workspaces. Zusätzliche Nutzerrepräsentationen sind nur bei virtuellen

Displays notwendig. Hier werden alle Nutzer aller Workspaces abgebildet, so dass jeder im virtuellen Display perspektivisch korrekte Inhalte wahrnehmen kann. Virtuelle Displays können sich geometrisch und semantisch durch die Spezifikation unterschiedlicher Eingangs- und Ausgangsmatrizen sowie Einstellungen an den Prärenderpässen unterscheiden.

Abschnitt 5.3 hat bereits gezeigt, dass die Realisierbarkeit der Anforderungen 3.1.1 und 3.1.2 durch das Kopplungsprinzip dieses Ansatzes nicht direkt gegeben sind. Zwar lassen sich alternative Navigationen mit virtuellen Displays realisieren, dennoch resultieren diese durch die Notwendigkeit von Präpässen in einem höheren Renderaufwand. Ist ein bestimmtes physikalisches Display nur dafür ausgelegt, Fläche für einen Interaktionsbereich zu liefern, so können die eigentlichen Objekte der Szene im Rahmen von Anforderung 3.5.2 darauf ausgeblendet werden.

Die nach Anforderung 3.4.1 definierten Werkzeuge sind gut einsetzbar und erscheinen nach Anforderung 3.4.2 auf Grund der kontinuierlichen Abbildung der Displays innerhalb des Workspaces selbst ebenfalls kontinuierlich. So werden beispielsweise Strahlen auf verschiedenen Displays direkt fortgesetzt, so dass gegriffene Objekte ohne Weiteres im kontinuierlichen Raum bewegt werden können. Problematische Fälle ergeben sich, wenn einzelne physikalische Displays zur Darstellung separater Navigationen durch virtuelle Displays genutzt werden. Die Werkzeuge werden hier zwar korrekt oberhalb der virtuellen Displayfläche abgebildet, ignorieren diese jedoch vollständig in ihren Aktionen. Dies zeigt sich beispielsweise bei den Portalkameras daran, dass sie durch die virtuelle Displayfläche hindurch Aufnahmen produzieren und diese bei entsprechendem Neigungswinkeln durch das virtuelle Display abgeschnitten werden. Dieses Verhalten verstärkt die Notwendigkeit von Anforderung 3.4.3, welche durch Werkzeugtransformationen in das virtuelle Display erfüllt werden kann, jedoch zusätzlichen Berechnungsaufwand für Schnitte und Umformungen erfordert. Da durch den Ansatz prinzipiell nur eine Display-Gruppe vorgesehen ist, ist innerhalb dieser die Sichtbarkeit einer Werkzeugvisualisierung generell erwünscht. Im Rahmen von Anforderung 3.5.1 ist es daher lediglich sinnvoll, die Sichtbarkeit von Werkzeugen in virtuellen Displays durch Einstellungen an den Prärenderpässen zu regeln. Bei Schnitten mit Szeneobjekten können Strahlen entsprechend des berechneten Schnittpunkts gekürzt werden.

Bei der Realisierung der überlappenden Territorien scheint es im Rahmen von Anforderung 3.3.2 plausibel, jedoch auch berechnungsintensiv, Inhalte von maximierten virtuellen

Displays so zu instantiieren, dass sie für bestimmte Nutzer auf einem anderen Display des Workspaces nahtlos fortgesetzt werden. Auch die Spezifikation von Clipping-Ebenen für einzelne Nutzer nach Anforderung 3.3.3 ist theoretisch umsetzbar, praktisch jedoch durch den Renderer auf eine Ebene pro Nutzer beschränkt. Wenn sich die Territorien der Nutzer wie in Anforderung 3.3.1 beschrieben geometrisch unterscheiden sollen, zeigt der Ansatz jedoch seine Beschränkungen. Da durch die kontinuierliche Abbildung nur ein Navigationsknoten für den ganzen Workspace exisitiert, sind individuelle pro Nutzer schaltbare Navigationen auf den physikalischen Displays nur durch zusätzliche Transformationsknoten oberhalb der Kopfknoten realisierbar, deren Matrix aber stets im Koordinatensystem des Workspace-Navigationsknotens ausgedrückt werden muss. Im Falle der virtuellen Displays müssen solche vorgeschalteten Knoten zunächst in das Ausgangskoordinatensystem transformiert werden.

Die kontinuierliche Abbildung ist auf eine beliebige Anzahl an Workspaces übertragbar. Weiterhin sind die dargestellten Konzepte durch spezielle Dateien konfigurierbar und offen für Erweiterungen wie weitere Werkzeuge oder zusätzliche Methoden zur Realisierung virtueller Sichten. Anforderungen 3.6.1 und 3.7.1 sind somit erfüllt.

6 Separate Abbildung der Displays des Workspaces

Im vorherigen Kapitel wurde der Ansatz einer kontinuierlichen Abbildung aller physikalischen Displays in der Applikation erläutert und diskutiert. Es zeigte sich, dass der Verwaltungsaufwand überschaubar ist, dennoch einige der definierten Anforderungen nur umständlich realisiert werden können, indem explizit in die Struktur eingegriffen wird. In diesem Kapitel liegt der Fokus nun auf einer zweiten Implementierungsstrategie, bei welcher eine logische Trennung der Displays innerhalb des Szenegraphs vorgenommen wird, obwohl diese physikalisch zum selben Workspace gehören. Im Folgenden werden zunächst die dafür benötigten Strukturen und Mechanismen erläutert, um sie anschließend bezüglich der in Kapitel 3 definierten Anforderungen zu evaluieren.

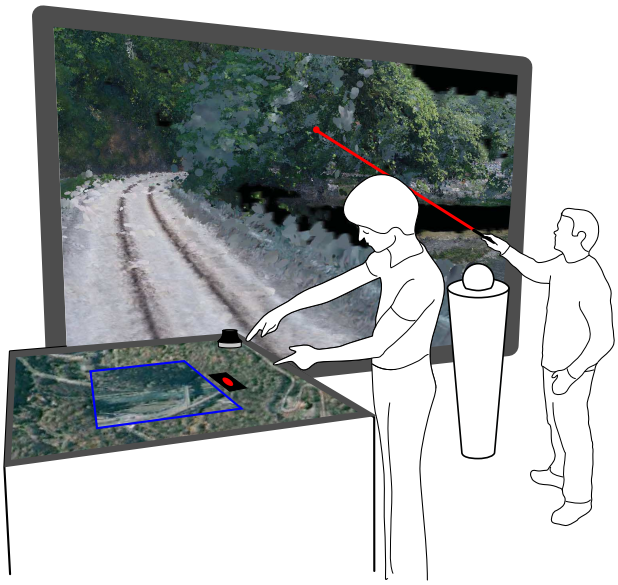


Abbildung 6.1: Workspace-Situation, welche im Folgenden im Szenegraph abgebildet wird. Der Unterschied zu Abbildung 5.1 liegt in den durch den Ansatz gegebenen standardmäßigen separaten Navigationen auf den physikalischen Displays.

6.1 Viewing Setup im Szenegraph

Im Rahmen des Ansatzes der kontinuierlichen Abbildung aus Kapitel 5 wurde das Beispielszenario in Abbildung 5.1 genutzt, um die Repräsentation aller am Setup beteiligten Entitäten innerhalb der Applikation zu erläutern. Um einen Vergleich der beiden Ansätze zu erlauben, soll im Rahmen der folgenden Abschnitte ein ähnlicher Beispielfall zur Erläuterung der Applikationsstruktur dienen.

Der hauptsächliche Unterschied zum vorher beschriebenen Ansatz liegt darin, dass jedes Display eines Workspaces nun standardmäßig einer spezifischen Display-Gruppe zugeordnet ist. Da nach Anforderung 3.1.2 jeder Display-Gruppe eine Navigation zugeordnet werden kann, sind Einzeldisplays innerhalb einer Display-Gruppe kontinuierlich gekoppelt, über die Grenzen der Display-Gruppe hinweg können jedoch unterschiedliche Inhalte angezeigt werden. Ist einer Display-Gruppe nur ein einziges Display zugeordnet, stellt dieses eine eigene Navigationseinheit dar. In Abbildung 6.2 ist eine Übersicht über die im Rahmen dieses Kapitels verwendeten strukturellen Unterteilungen dargestellt. Abbildung 6.1 zeigt einen zu Abbildung 5.1 ähnlichen Beispielfall, in welchem Leinwand- und Tabletop-Display je einer Display-Gruppe mit separaten Eingabegeräten zugeordnet sind.

6.1.1 Physikalische Displays

Abbildung 6.3 zeigt einen Szenegraph, welcher jedes Display der Situation aus Abbildung 6.1 einer eigenen Display-Gruppe zuordnet. Dazu werden zunächst unterhalb des Wurzelknotens für jede Display-Gruppe, je aus einem physikalischen Display bestehend, Navigationsknoten erzeugt. Die Bezeichnung *wx_dgy* bedeutet, dass der entsprechende Knoten die Display-Gruppe y in Workspace x repräsentiert. Separate Eingabegeräte, im vorliegenden Fall ein Spheron und eine Spacemouse, werden mit diesen Navigationsknoten verbunden.

Unter jedem der Navigationsknoten hängen einerseits der Repräsentant (der Screen-Knoten) des jeweiligen Displays, andererseits analog zum vorher beschriebenen Ansatz die Repräsentationen der Nutzer des Workspaces, deren Kopfknoten mit den entsprechenden Trackinginformationen gefüllt sind. Aus der Aufteilung der physikalischen Displays

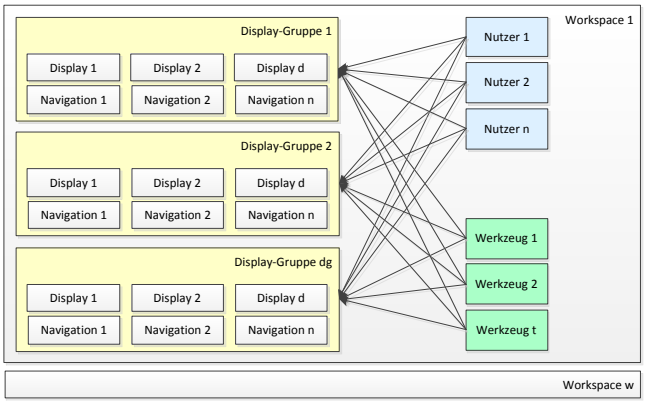


Abbildung 6.2: Übersichtsgrafik über die in diesem Kapitel verwendeten strukturellen Unterteilungen. Workspaces enthalten Display-Gruppen, welchen jeweils eine Menge an (physikalischen oder virtuellen) Displays und Navigationen zugeordnet ist. Nutzer und Werkzeuge sind ebenfalls innerhalb des Workspaces definiert und müssen in jeder Display-Gruppe eine Repräsentation (hier durch Pfeile angedeutet) erhalten.

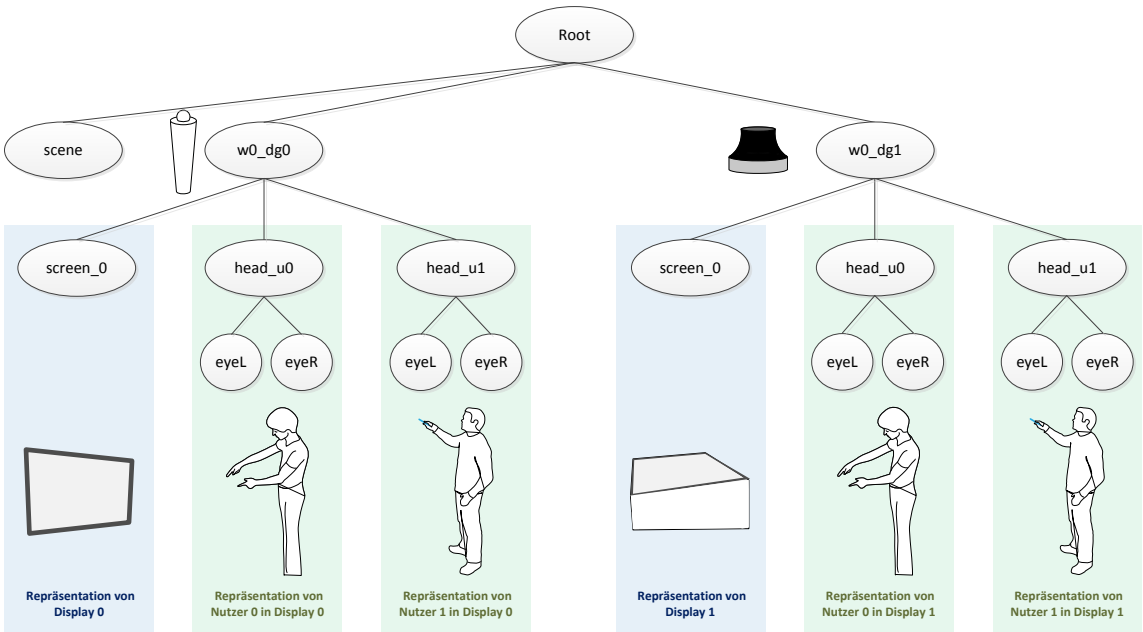


Abbildung 6.3: Szenegraph des Viewing Setups für die Nutzer und physikalischen Displays aus Abbildung 6.1. Intern beginnt die Nummerierung innerhalb jeder Display-Gruppe neu (durch beliebige Workspaces erweiterbar).

des Workspaces in verschiedene Display-Gruppen mit unterschiedlichen Navigationsmöglichkeiten folgt, dass die Nutzer redundant in jeder dieser Display-Gruppen repräsentiert sein müssen.

6.1.2 Erweiterung um überlappende Territorien

Das soeben beschriebene Viewing Setup ermöglicht es nach Anforderung 3.3.1 nicht, für einzelne Nutzer individuelle Navigationen zu definieren, kann aber bezüglich dessen leicht erweitert werden. Hierfür ist es nötig, wie in Abbildung 6.4 dargestellt, für jeden Nutzer innerhalb jeder Display-Gruppe einen Navigationsknoten zu erstellen. Die Bezeichnung dieser Knoten *wx_dgy_uz* beschreiben nun die Navigationseinheiten von Nutzer z in Display-Gruppe y des Workspaces x. Diese Navigationsknoten sind somit Teil der jeweiligen Nutzerrepräsentation geworden. Jede dieser individuellen Navigationen benötigt nun auch eine Displayrepräsentation an der entsprechenden Stelle der virtuellen Welt, weshalb auch diese als Teil der Nutzerrepräsentation aufgefasst werden können. Im Falle mehrerer Nutzer führt dies zu einer redundanten Repräsentation der Displays.

Dieser Aufbau gibt dem Applikationsentwickler die Freiheit, im Rahmen der beschriebenen Fälle der überlappenden Territorien oder der erweiterten Gruppennavigationstechniken in [20] einzelnen Nutzern andere Navigationen zuzuweisen. Dazu wird innerhalb jeder Display-Gruppe, wie bereits in Abbildung 6.2 angedeutet, eine Menge an verfügbaren Navigationen definiert. Aktuell implementierte Navigationen sind *Steering Navigations*, welche mit Hilfe eines Eingabegeräts bedient werden können, oder *Static Navigations*, welche eine konstante Position und Orientierung in der Szene anzeigen, wie es beispielsweise bei einigen virtuellen Displays benötigt wird. Abbildung 6.5 zeigt den benötigten Datenfluss innerhalb der Klasse *SteeringNavigation*. Eine Sub-Klasse von *6DOFDevice* bestimmt die relativen Werte eines Eingabegeräts und leitet diese an das *InputMapping* weiter, welches die relativen Werte auf die absolute Navigationsmatrix akkumuliert. Zur Realisierung von Schwerkrafteffekten wird diese Matrix durch Strahlschnitte mit der Szene vom *GroundFollowing* korrigiert, zurück zum *InputMapping* gesendet und dann in einem Attribut der Basisklasse *Navigation* gespeichert. Alle Nutzerrepräsentationen überprüfen, welche Navigationen der eigenen Display-Gruppe zugeordnet sind und verbinden eine der angebotenen Navigationsmatrizen mit ihrem Navigationsknoten. Damit sind im Rahmen der überlappenden Territorien unterschiedliche Navigationen pro Nutzer innerhalb

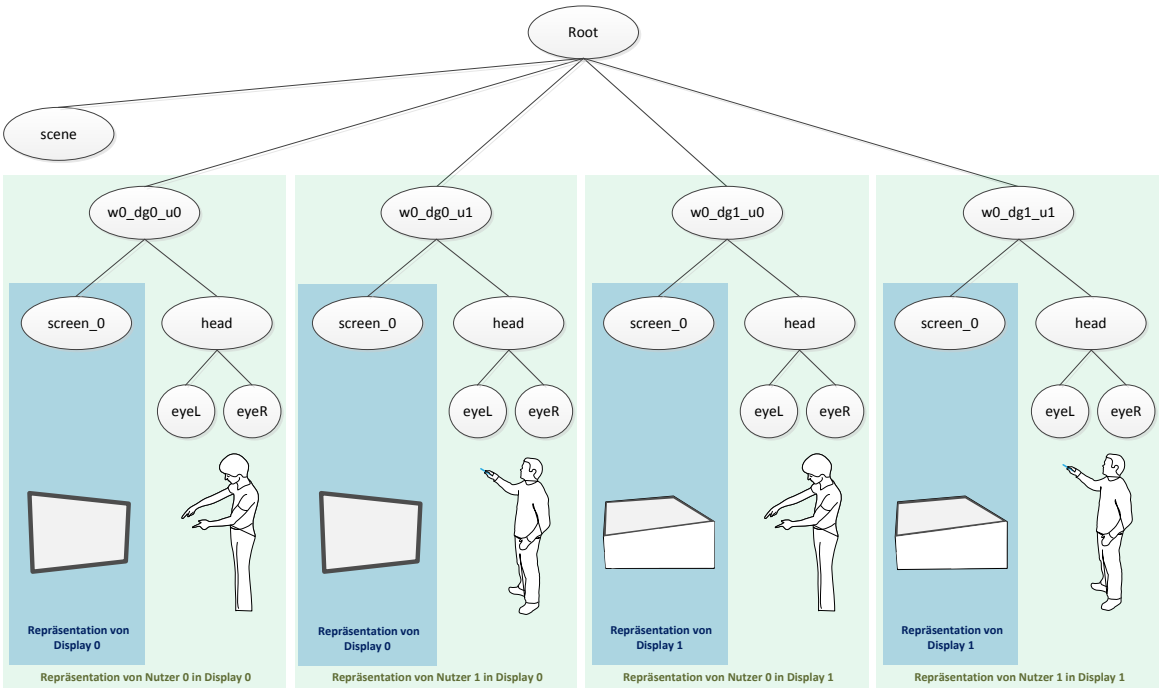


Abbildung 6.4: Erweiterung des Viewing Setups aus Abbildung 6.3 um individuelle Navigationsknoten pro Nutzer. Jede Nutzerrepräsentation im Szenegraphen besteht nun zusätzlich aus einem Navigationsknoten und einer Display-Repräsentation.

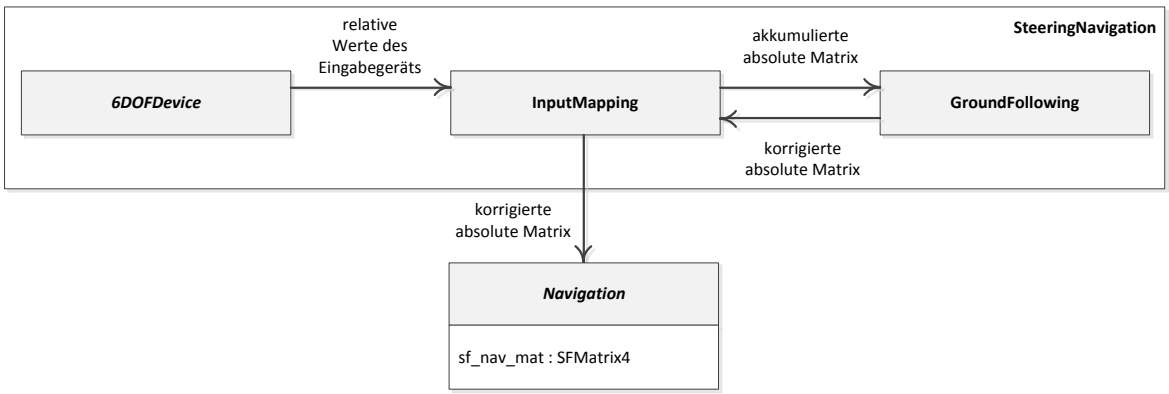


Abbildung 6.5: Datenfluss innerhalb der Klasse *SteeringNavigation*, welche von *Navigation* erbt. Die relativen Werte eines Eingabegeräts werden auf die absolute Navigationsmatrix akkumuliert und diese bezüglich der Schwerkraft korrigiert. Das Ergebnis wird in das Feld *sf_nav_mat* der Basisklasse geschrieben.

der Display-Gruppe realisierbar, der in Abschnitt 6.1.1 dargestellte Fall wird jedoch nicht ausgeschlossen. Um eine einheitliche Navigation innerhalb der Display-Gruppe für alle Nutzer zu realisieren, müssen lediglich alle Nutzerrepräsentationen dieselbe Navigationsmatrix verbinden.

Ein implementierter Beispielfall auf dem großen Leinwand-Display unseres Labors ordnet zunächst allen Nutzern standardmäßig eine Spheron-Navigation zu. Zusätzlich dazu liegt ein getrackter Controller bereit und kann auf Knopfdruck aktiviert werden. In diesem Fall wechseln alle Nutzer im Umkreis von 0.5 m des Eingabegeräts von der Spheron-Navigation zur Controller-Navigation und können somit individuell die Szene erkunden. Mit einem erneuten Knopfdruck kann zurück zur Hauptnavigation gewechselt werden.

6.1.3 Virtuelle Displays

In Abschnitt 5.1.2 wurde erläutert, dass virtuelle Displays von allen Workspaces gleichermaßen gesehen werden können. Da innerhalb der separaten Abbildung die Displays auf mehrere Display-Gruppen verteilt sein können, erweitert sich der beschriebene Mechanimus zur Erzeugung und Verwaltung virtueller Displays um die zusätzlichen Nutzerrepräsentationen, die der Ansatz mit sich bringt. Im Rahmen des in Abschnitt 5.1.2 vorgestellten Verfahrens muss daher unterhalb des Ausgangsknotens des virtuellen Displays für alle Workspaces das Verhältnis jedes Nutzers in jeder Display-Gruppe bezüglich der Eingangstransformation abgebildet werden, so dass der Inhalt des virtuellen Displays auf jedem physikalischen Display für alle Nutzer perspektivisch korrekt angezeigt werden kann.

6.2 Werkzeuge und ihre Repräsentanten

Da im Rahmen des in Abschnitt 6.1 hergeleiteten Viewing Setups jeder Nutzer potentiell auf einer individuellen Navigation die Szene erkunden kann, ist es notwendig, Werkzeugrepräsentanten nun ebenfalls innerhalb jeder Nutzerrepräsentation zu definieren, um die Werkzeuge nach Anforderung 3.4.2 kontinuierlich innerhalb des Workspaces erscheinen zu lassen.

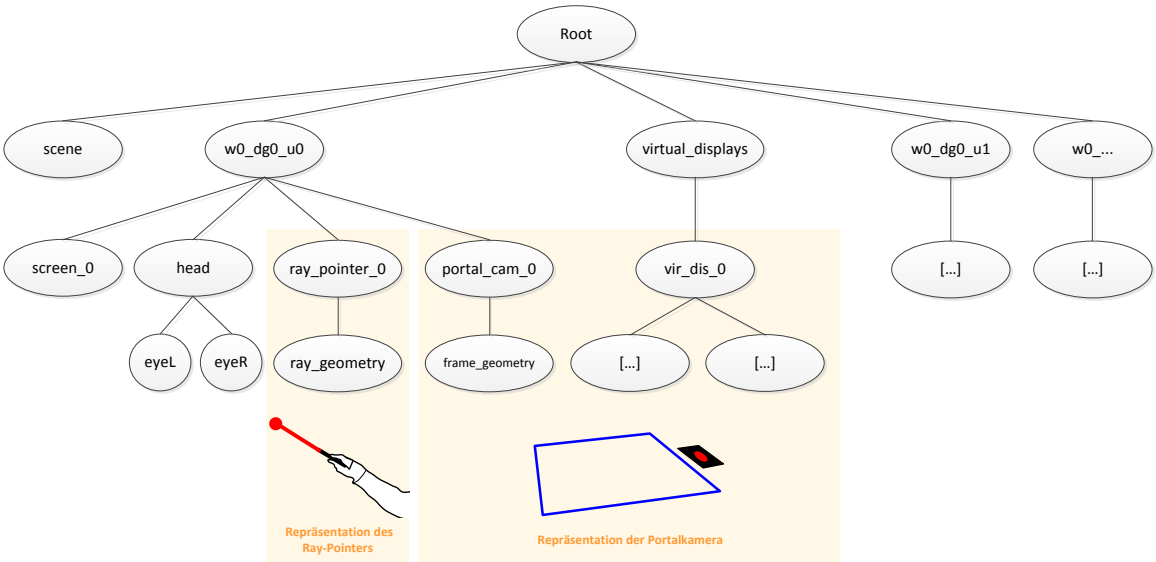
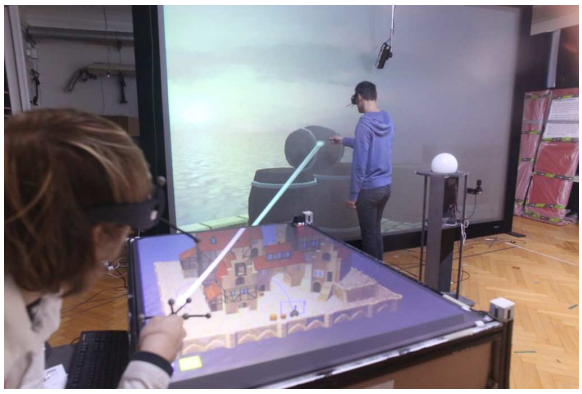


Abbildung 6.6: Repräsentation des Ray-Pointers und der Portalkamera für einen Nutzer in einer Display-Gruppe. Die Werkzeugrepräsentanten aller anderen Nutzerrepräsentationen wurden aus Platzgründen in dieser Grafik nicht dargestellt.

Dies hat zur Folge, dass der eine logische Ray-Pointer aus Abbildung 6.1 nun Repräsentanten für zwei Nutzer in zwei Display-Gruppen, insgesamt also vier Stück, benötigt. Diese Repräsentanten werden, wie in Abbildung 6.6 illustriert, je unter einen der Navigationsknoten angehängt, so dass sie an jedem dargestellten Standpunkt der Szene vorhanden sind und angezeigt werden können. Dasselbe Verhalten gilt auch für die Portalkamera, ist jedoch bei ihr etwas aufwändiger, da für jeden Repräsentanten zusätzlich ein virtuelles Display erzeugt und verwaltet werden muss.

Durch die multiple Repräsentation derselben Werkzeuge in allen Nutzerrepräsentationen jeder Display-Gruppe kann die in Anforderung 3.4.2 angestrebte Kontinuität zwischen den Display-Gruppen gewährleistet werden. Abbildung 6.7 zeigt hierzu zwei Beispielfälle aus unserem Labor. Es ist jedoch zu beachten, dass zwar Repräsentanten eines Werkzeugs für alle Nutzerrepräsentationen aller Display-Gruppen vorgesehen werden, jedoch nicht alle zwangsläufig immer sichtbar sind, da diese unter Umständen aktuell nicht von Bedeutung sind. Sollen die Werkzeuge nach Anforderung 3.4.3 auch in virtuellen Displays funktionieren, müssen diese auf Grund ihrer vom Workspace unabhängigen Verwaltung Werkzeugrepräsentanten aller Werkzeuge aller Workspaces besitzen.



(a) Die Visualisierung eines Ray-Pointers beginnt auf dem Tabletop-Display und wird in der Display-Gruppe des Leinwand-Displays fortgesetzt.

(b) Das virtuelle Display der Portalkamera wird nicht durch den Displaywechsel beeinträchtigt, so dass der Inhalt auf beiden Displays kontinuierlich wahrgenommen werden kann.

Abbildung 6.7: Durch Repräsentanten für alle Nutzer in allen Display-Gruppen werden Werkzeuge kontinuierlich visualisiert.

6.3 Sichtbarkeiten von Repräsentanten

Wie in den vorherigen beiden Abschnitten erläutert wurde, erfordert der Ansatz der separaten Abbildung eine höhere Anzahl an Nutzer-, Display- und Werkzeugrepräsentationen als die kontinuierliche Abbildung. Wenn den einzelnen Repräsentanten jedoch Geometrien zur Visualisierung in der virtuellen Welt zugeordnet sind, so wie es bei den Nutzern durch die Avatare und bei den Werkzeugen der Fall ist, wird die Szene sehr schnell mit einer hohen Anzahl an Geometrien gefüllt, die nicht unter allen Umständen sichtbar sein sollen. Vielmehr sollen Regeln definierbar sein, nach denen die Sichtbarkeiten einzelner Repräsentanten geschaltet werden können.

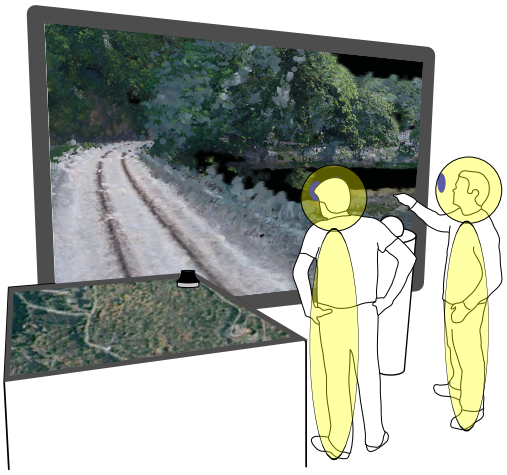
Diese Regelung kann auf zwei Ebenen erfolgen. Zunächst muss innerhalb jeder Display-Gruppe entschieden werden, welche der Repräsentanten für welche Nutzer sichtbar sein sollen. Eine Ebene höher kann anschließend bestimmt werden, welche der Repräsentanten innerhalb der Display-Gruppe nach außen für andere Display-Gruppen in diesem bzw. in weiteren Workspaces angezeigt werden.

Im Rahmen dieser Arbeit wurde sich dazu entschlossen, ein festgelegtes Regelwerk auf die Sichtbarkeiten innerhalb der Display-Gruppe anzuwenden, da dieses Verhalten in der Regel stets identisch sein sollte. Die Sichtbarkeitsregeln nach außen hingegen sind nicht klar definiert und somit in Konfigurationsdateien offen gehalten.

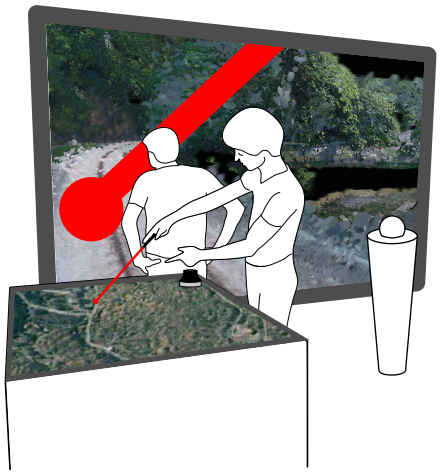
6.3.1 Sichtbarkeiten innerhalb der Display-Gruppe

Avatare helfen, den Standpunkt der zugehörigen Nutzer und ihrer Display-Gruppe in der virtuellen Welt nachzuvollziehen. Wie in Abbildung 6.8(a) dargestellt wurde, ist es innerhalb der Display-Gruppe jedoch nicht hilfreich, die eigene und andere Repräsentationen derselben Display-Gruppe zu sehen, da sich in diesem Fall die menschlichen Körper in der realen Welt und ihre Avatarrepräsentationen durch die 3D-Fähigkeit des zugehörigen Displays überschneiden. Daher werden im Beispielfall von Abbildung 6.8(d) die Avatare des Leinwand-Displays auf diesem ausgeblendet. Wechselt jedoch ein Nutzer nach Anforderung 3.3.1 innerhalb einer Display-Gruppe die Navigation (wie im Beispielfall am Ende von Abschnitt 6.1.2), so ist wiederum für die anderen die Anzeige seines Avatars erwünscht, um seine Aktionen nachvollziehen zu können. Daher überprüft das implementierte Verfahren für jede Display-Gruppe, welche Nutzer die virtuelle Welt auf welcher Navigation erkunden und schaltet die Sichtbarkeiten der Avatare so, dass für Nutzer innerhalb einer Display-Gruppe Avatare auf derselben Navigation ausgeblendet werden.

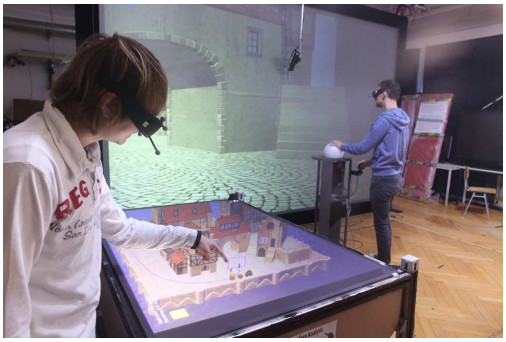
Auch bei den Werkzeugen ergeben sich kritische Situationen, in denen die Sichtbarkeiten von Repräsentanten innerhalb der Display-Gruppe geregelt werden müssen. Dazu muss zunächst bestimmt werden, welcher Nutzer das Werkzeug gerade bedient. Im Folgenden wird dieser als dem Werkzeug *zugeordnet* bezeichnet. Es soll dann für alle Nutzer innerhalb der Display-Gruppe nur die Werkzeugrepräsentation des zugeordneten Nutzers angezeigt werden, damit seine Aktionen für alle nachvollziehbar und die anderen Repräsentanten des Werkzeugs, welche eventuell auf Grund unterschiedlicher Navigationen an anderen Stellen der Szene sein können, aktuell nicht von Bedeutung sind. Das implementierte Verfahren modelliert jeden Nutzer im Workspace als Linie, welche von seiner Kopfposition senkrecht zum Boden verläuft und ermittelt in Form des kürzesten Punkt-zu-Linie-Abstands, welcher Nutzer aktuell einem Werkzeug zugeordnet ist. Die Sichtbarkeiten der Repräsentanten innerhalb der Display-Gruppe werden danach entsprechend gesetzt. Um diese Nutzerzuordnung nachvollziehbar zu gestalten, werden bei geteilten Navigationen Farbcodierungen eingesetzt. Der zugeordnete Nutzer sieht hierbei die Visualiserung des Werkzeugs in Farbe, während die restlichen Nutzer eine weiße Werkzeugrepräsentation wahrnehmen.



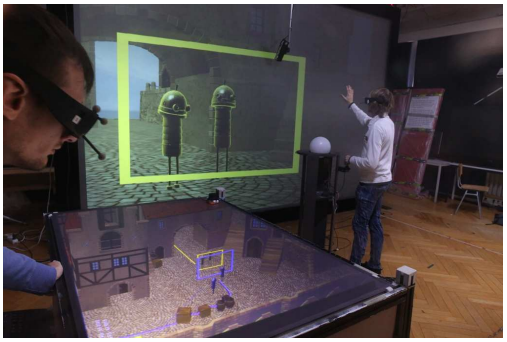
(a) Ein Nutzer soll innerhalb einer Display-Gruppe nicht die Avatarrepräsentation eines anderen Nutzers sehen, wenn beide die selbe Navigation teilen, da sich sonst der menschliche Körper mit dem Avatar überschneidet.



(b) Werden Manipulationen durch Ray-Pointer am Tabletop-Display vorgenommen, so kann es sinnvoll sein, die Strahlrepräsentation auf dem Leinwand-Display anzuzeigen. In anderen Fällen möchte man den großen Strahl jedoch ausblenden können.



(c) Die Spur eines Navigationsgeräts wird aus Gründen der Nachvollziehbarkeit in der Miniatur auf dem Tabletop-Display angezeigt, ist jedoch auf dem Leinwand-Display nicht zwingend von Interesse.



(d) Auf dem Leinwand-Display werden die eigenen Avatare (blau) ausgeblendet, um Überschneidungen mit den menschlichen Körpern zu vermeiden. Die Avatare anderer Display-Gruppen (gelb) werden jedoch auf beiden Displays dargestellt.

Abbildung 6.8: Beispielfälle, in denen Sichtbarkeitsregelungen von Repräsentanten vorgenommen werden müssen.

6.3.2 Sichtbarkeiten außerhalb der Display-Gruppe

Nachdem die Sichtbarkeiten innerhalb der einzelnen Display-Gruppen geklärt sind stellt sich die Frage, welche der Repräsentanten für andere Display-Gruppen, eventuell auch in anderen Workspaces, angezeigt werden. Würden in Abbildung 6.8(a) beispielsweise alle Avatare in allen Display-Gruppen sichtbar sein, so könnte man auf dem Tabletop-Display in der Miniaturansicht wie in Abbildung 6.8(d) erwartungskonform die Avatare der Display-Gruppe des Leinwand-Displays sehen, umgekehrt jedoch auch auf dem Leinwand-Display die großskalierten Avatare des Tabletop-Displays, was nicht zwingend erwünscht ist.

Bei den Werkzeugen ist diese Situation ähnlich. Abbildung 6.8(b) zeigt einen Beispielfall, bei dem die Strahlrepräsentation des Tabletop-Displays auf dem Leinwand-Display sichtbar ist und somit als sehr groß skalierte Geometrie erscheint. Während dies in bestimmten Situationen als visuelle Hilfestellung sinnvoll sein kann, mag es in anderen Situationen störend wirken.

Der implementierte Ansatz erlaubt daher eine freie Konfiguration der Sichtbarkeiten von Repräsentanten zwischen Display-Gruppen. Dazu wird jeder Display-Gruppe ein so genanntes *Visibility Tag* zugeordnet. Für jedes Werkzeug und jeden Nutzer kann dann anschließend eine Sichtbarkeitstabelle (*Visibility Table*) spezifiziert werden, in der festgelegt ist, welche Display-Gruppe mit einem bestimmten Visibility Tag welche Repräsentanten des Nutzers oder Werkzeugs aus anderen Display-Gruppen sehen können. Ein Beispiel einer solchen Spezifikation ist in Tabelle 6.1 dargestellt.

	Wand	Tisch
Wand	X	False
Tisch	True	X
Andere	True	False

Tabelle 6.1: Beispielhafte Sichtbarkeitstabelle eines Nutzers oder Werkzeugs. Im Beispielsetup aus Abbildung 6.1 wurden die zwei Visibility Tags „Wand" und „Tisch" an das Leinwand- und das Tabletop-Display vergeben.

Zur Vereinfachung wurden im Beispielsetup aus Abbildung 6.1 die Visibility Tags „Wand" und „Tisch" an das Leinwand- und Tabletop-Display vergeben. Pro Zeile der Tabelle

werden die Sichtbarkeiten von Repräsentanten für Display-Gruppen mit dem links angegebenen Visibility Tag spezifiziert. Dafür wird in jeder Spalte in Form eines Wahrheitswertes festgelegt, ob Display-Gruppen des im Zeilentitel spezifizierten Visibility Tags Repräsentanten von Display-Gruppen des im Spaltentitel angegebenen Visibility Tags sehen. Im Falle der ersten Zeile beispielsweise wird spezifiziert, dass auf dem Leinwand-Display keine Repräsentanten des Tabletop-Displays gesehen werden können. Wäre diese Tabelle einem Strahlwerkzeug zugeordnet, so würde die Situation des großen Strahls aus Abbildung 6.8(b) unterbunden werden. Ist in der Tabelle der Wert *True* spezifiziert, so werden im Nutzerfall alle Avatare der Display-Gruppen der Spaltenüberschrit angezeigt, im Werkzeugfall hingegen nur der Repräsentant des zugeordneten Nutzers in dieser Display-Gruppe, da nur dieser aktuell relevant ist.

Die Sichtbarkeitswerte von Repräsentanten des eigenen Visibility Tags wurden in der Tabelle nicht spezifiziert und mit einem X markiert. Sie ergeben sich durch das Navigationskriterium, welches in Abschnitt 6.3.1 erläutert wurde, auch wenn sich mehrere Display-Gruppen dasselbe Visibility Tag teilen. Für alle Display-Gruppen, deren Visibility Tag keine eigene Zeile in der Sichtbarkeitstabelle einnimmt, wird die Zeile „Andere“verwendet; bei einer fehlenden Spalte wird standardmäßig der Wert *False* angenommen.

6.3.3 Sichtbarkeiten von Szeneelementen

Die soeben beschriebenen Sichtbarkeitstabellen für Werkzeug- und Nutzerrepräsentation können auf Szeneelemente verallgemeinert werden. Hierbei kann der Applikationsentwickler in Form einer eindimensionalen Sichtbarkeitsliste spezifizieren, auf welchen Display-Gruppen mit bestimmten Visibility Tags ein Objekt angezeigt werden soll. Beispielhafte Anwendungsfälle hierfür sind die in Abbildung 6.8(c) dargestellten Spuren von Navigationsgeräten und die Informationsflaggen aus Abbildung 3.2(a), welche ausschließlich innerhalb eines virtuellen Displays angezeigt werden sollen.

6.4 Die aktive Repräsentation von Werkzeugen

Im vorherigen Abschnitt 6.3 wurde ein Verfahren vorgestellt, welches die Sichtbarkeiten von Nutzer- und Werkzeugrepräsentanten für die eigene und andere Display-Gruppen regelt. Wenn nun ein Button auf einem der Werkzeuge gedrückt wird, beispielsweise um beim Ray-Pointer einen Dragging-Vorgang zu initiieren oder bei der Portalkamera eine Aufnahme zu erstellen, muss jedoch zusätzlich geklärt werden, aus welchem der zahlreichen Repräsentanten die benötigte Transformationsmatrix verwendet wird. Dieser Repräsentant wird im Folgenden als die *aktive Repräsentation* eines Werkzeuges bezeichnet. Zur Bestimmung der aktiven Repräsentation wurde ein generischer Algorithmus implementiert, dessen Funktionsweise im Folgenden dargelegt werden soll.

6.4.1 Allgemeines Entscheidungsverfahren

Abbildung 6.9 zeigt ein UML-Sequenzdiagramm zum implementierten Verfahren zur Bestimmung der aktiven Repräsentation eines Werkzeugs. Dieses Ablaufschema wurde in abstrakten Basisklassen *Tool* und *ToolRepresentation* verankert. Die Klasse *ToolRepresentation* stellt hierbei die konkreten Werkzeugrepräsentanten eines Nutzers in einer Display-Gruppe dar, während die Klasse *Tool* nur einmal auf Workspacebene existiert und die Werkzeuglogik verwaltet. Diese Unterteilung ist in Abbildung 6.10 in Form eines vereinfachten UML-Klassendiagramms dargestellt.

Wird eine Anfrage nach der aktiven Repräsentation gestellt, so stellt das entsprechende Objekt der Subklasse von *Tool* zunächst fest, welcher der aktuell zugeordnete Nutzer ist. Anschließend wird über jede Display-Gruppe des Workspaces iteriert und die Weltkoordinaten des dortigen Repräsentanten des zugeordneten Nutzers abgefragt. Im Folgenden wird in einer Analyse entschieden, ob der Repräsentant als aktive Repräsentation in Frage kommt und somit in eine Kandidatenliste aufgenommen wird. Nach der Iteration über alle Display-Gruppen wird aus der Kandidatenliste über die aktive Repräsentation entschieden und diese zurückgegeben.

Der Aufbau der Kandidatenliste und die Auswahl der aktiven Repräsentation aus dieser ist werkzeugspezifisch und somit von Tool-Subklasse zu Tool-Subklasse unterschiedlich. Die Schritte wurden daher in Abbildung 6.9 rot markiert.

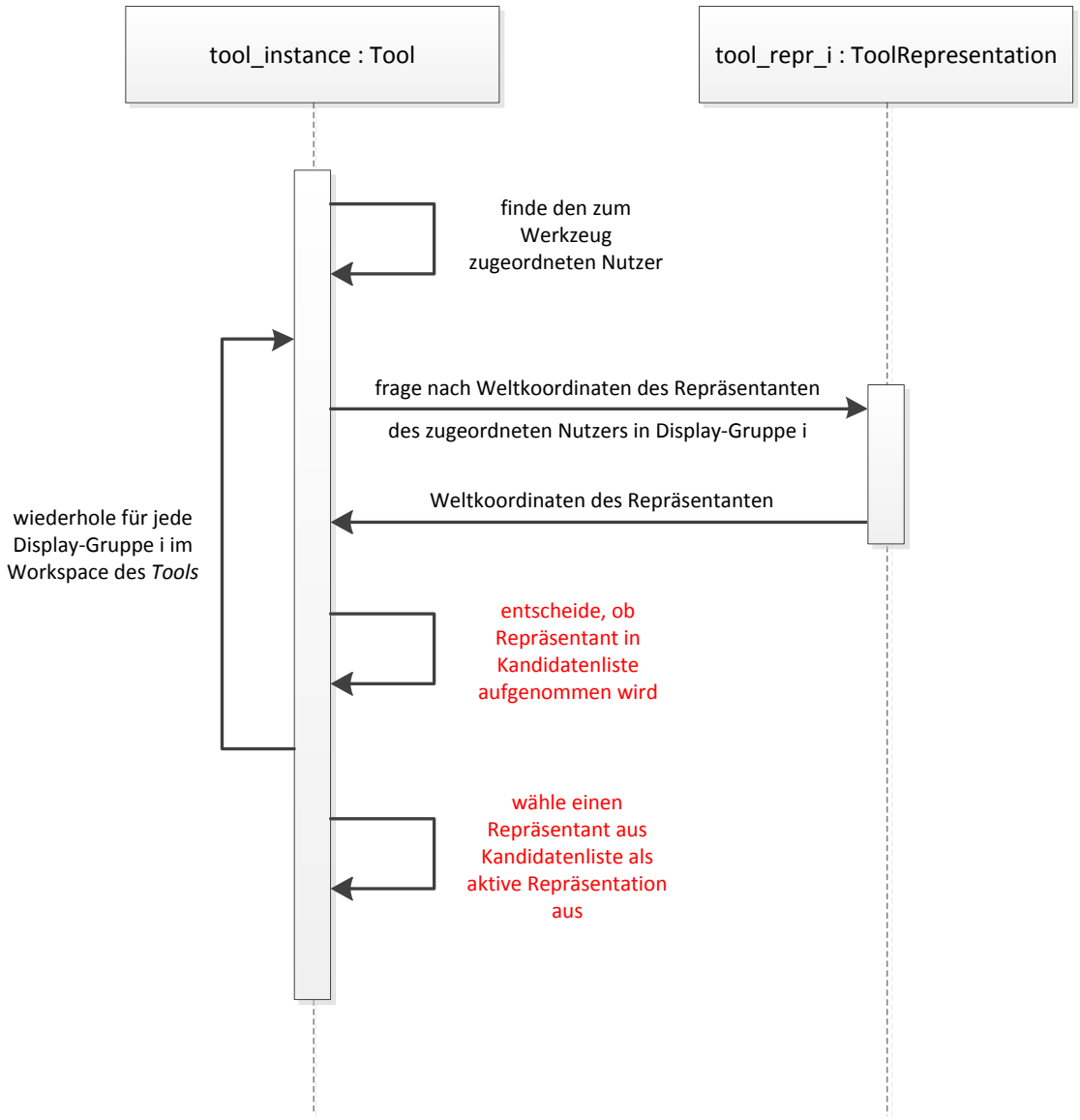


Abbildung 6.9: UML-Sequenzdiagramm zur Bestimmung der aktiven Repräsentation eines Werkzeugs. Der Ablauf wurde in abstrakten Basisklassen *Tool* und *ToolRepresentation* implementiert. Rot markierte Schritte sind offen zur Spezifikation durch die jeweiligen abgeleiteten Klassen.

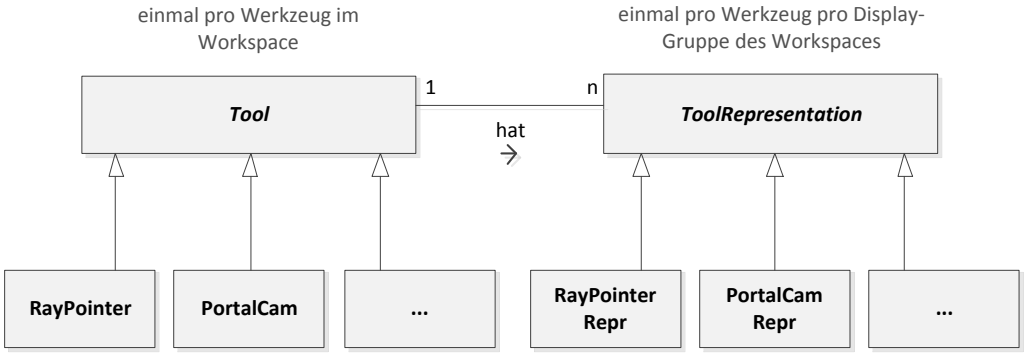


Abbildung 6.10: Klassenstruktur für Werkzeuge. Jedes Werkzeug eines Workspaces benötigt eine Instanz einer Sub-Klasse von *Tool* zur Verwaltung der Werkzeuglogik und pro Display-Gruppe einen Repräsentanten, realisiert durch eine Instanz einer Sub-Klasse von *ToolRepresentation*.

Eine mögliche Implementierung der werkzeugspezifischen Funktionen wird im folgenden Abschnitt 6.4.2 am Beispiel der Ray-Pointer und Portalkameras beschrieben.

6.4.2 Konkrete Implementierung

Ray-Pointer

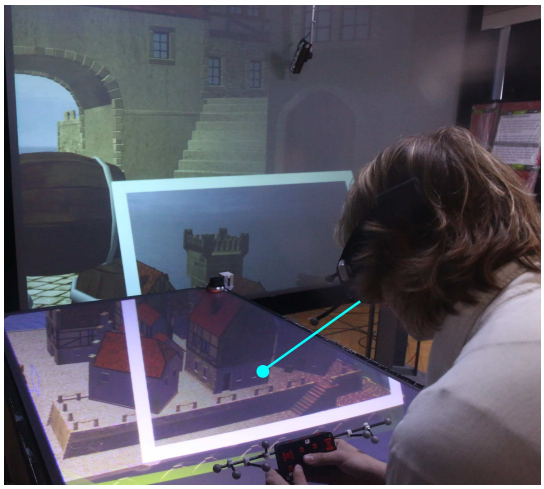
Bei Ray-Pointern ist es von Interesse, welche Objekte in der Szene geschnitten werden, so dass auf diesen beispielsweise ein Dragging-Vorgang ausgelöst werden kann. Beim Erstellen der Kandidatenliste wird daher überprüft, ob ein Objekt der Szene vom jeweiligen Strahlrepräsentanten geschnitten wird. Im Falle eines Schnittpunkts wird getestet, ob dieser im Kamerafrustum des dem Werkzeug zugeordneten Nutzers liegt, da er sonst für diesen nicht von Bedeutung ist. Alle Repräsentanten mit sichtbaren Schnittpunkten werden anschließend in die Kandidatenliste aufgenommen. Zuletzt wird aus der Liste der Kandidaten derjenige Repräsentant mit der kürzesten Schnittdistanz als aktive Repräsentation zurückgegeben. Wurden für keinen Repräsentanten sichtbare Schnittpunkte ermittelt, so ist die aktive Repräsentation zum Auswertungszeitpunkt nicht existent.

Abbildung 6.11: Die Strahlrepräsentanten in beiden Display-Gruppen schneiden je ein Objekt. Da beide Schnittpunkte im entsprechenden Kamerafrustum des zugeordneten Nutzers liegen, wird auf den nächsten Schnitt überprüft und somit der Tabletop-Repräsentant als aktive Repräsentation ausgewählt. Der Strahl wird entsprechend des Schnittpunkts gekürzt, so dass dieser auf dem Leinwand-Display nicht mehr zu sehen ist.

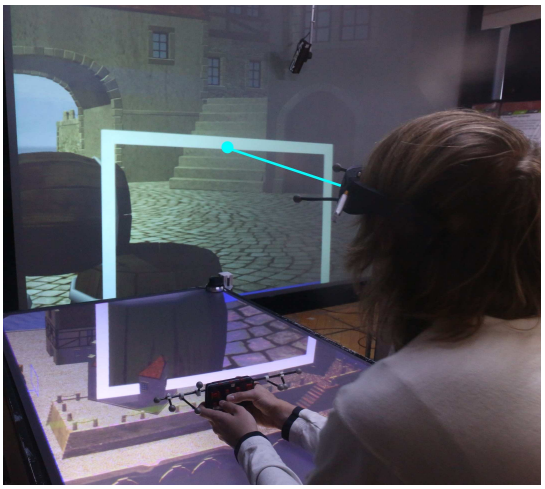
In Abbildung 6.7(a) beispielsweise schneidet lediglich der Repräsentant in der Display-Gruppe des Leinwand-Displays ein Objekt und wird somit als aktive Repräsentation ausgewählt. Im Fall von Abbildung 6.11 schneiden hingegen der Leinwand- und Tabletop-Repräsentant des Strahls je ein Objekt. Da sowohl der Schnittpunkt mit dem Fass (Leinwand) als auch der Schnittpunkt mit dem Haus (Tabletop) innerhalb des entsprechenden Kamerafrustums des zugeordneten Nutzers liegen, wird der dem Nutzer nächstliegende Schnitt bestimmt und somit der Tabeletop-Repräsentant als aktive Repräsentation ausgewählt.

Portalkamera

Wird bei der Portalkamera der Auslösebutton gedrückt, muss die aktive Repräsentation zur Anfertigung der Aufnahme festgestellt werden. Die dafür implementierten Entscheidungsmechanismen werden in Abbildung 6.12 illustriert. Auf Framebasis wird für jeden Nutzer wie in Abschnitt 2.2.1 beschrieben das Display bestimmt, auf welches er aktuell blickt.



(a) Blickt der zugeordnete Nutzer nach unten, so wird der Repräsentant des Tabletop-Displays als aktive Repräsentation ausgewählt. Die Miniatur wird kontinuierlich in der Repräsentation der Portalkamera auf dem Leinwand-Display fortgesetzt.



(b) Blickt der zugeordnete Nutzer nach oben, so wird der Repräsentant des Leinwand-Displays als aktive Repräsentation ausgewählt. Die Fässer werden kontinuierlich in der Repräsentation der Portalkamera auf dem Tabletop-Display fortgesetzt.

Abbildung 6.12: Entscheidungsverfahren zur Bestimmung der aktiven Repräsentation am Beispiel der Portalkamera.

Schneidet der Nutzer kein Display mit seinem Blickstrahl, wird das zuletzt betrachtete Display gespeichert, so dass auch Aufnahmen ohne Sichtkontrolle erzeugt werden können. In die Kandidatenliste wird dann nur der Repräsentant der zugehörigen Display-Gruppe aufgenommen, welcher dann anschließend als aktive Repräsentation zurückgegeben wird.

6.5 Diskussion

Die in diesem Kapitel vorgestellte separate Abbildung der Displays des Workspaces resultiert in einem höheren Verwaltungsaufwand an Szenegraphknoten, welche durch zusätzliche Applikationslogik verwaltet werden müssen. Um separate Navigationen pro Display-Gruppe zu ermöglichen, müssen Nutzer in jeder Display-Gruppe repräsentiert sein. Wird die Navigationsmöglichkeit auf Nutzerebene realisiert, so ist zusätzlich eine redundante Repräsentation der physikalischen Displays notwendig, was jedoch lediglich eine Verallgemeinerung der bereits vorliegenden Struktur ist. Um die Inhalte

von virtuellen Displays korrekt darzustellen, müssen die Nutzerrepräsentationen aller Display-Gruppen aller Workspaces unterhalb des Ausgangsknotens abgebildet werden.

Innerhalb dieses Kapitels wurde davon ausgegangen, dass jeder Display-Gruppe standardmäßig nur ein Display zugeordnet wird und die Begriffe „Display-Gruppe" und „Display" somit synonym verwendet werden können. Im Rahmen der Realisierung von Anforderungen 3.1.1 und 3.1.2 ist es jedoch denkbar, mehrere Displays zu einer Display-Gruppe hinzuzufügen, so dass dann, analog zur kontinuierlichen Abbildung, unterhalb der Navigationsknoten mehrere Screen-Knoten hängen und die Displays somit gekoppelt erscheinen. Es ist also festzustellen, dass der in Kapitel 5 beschriebene Ansatz implizit in dieser Implementierungsstrategie enthalten ist und für den Applikationsentwickler für beliebige physikalische und virtuelle Displays definierbar ist.

Im Rahmen der überlappenden Territorien sind durch die individuellen Navigationsknoten pro Nutzer alle benötigten Freiheiten gegeben, um Anforderung 3.3.1 zu realisieren. Auch die Instantiierung von Szeneobjekten im Rahmen von Anforderung 3.3.2 ist anwendbar, wenn auch zusätzlich berechnungsintensiv. Die Anzahl von Clipping-Ebenen aus Anforderung 3.3.3 ist wie bereits erwähnt durch den Renderer auf eins beschränkt.

Damit die nach Anforderung 3.4.1 vorhandenen Werkzeuge auf Basis von Anforderung 3.4.2 kontinuierlich innerhalb des Workspaces funktionieren, erfordert der Ansatz Algorithmen zur Bestimmung der aktiven Repräsentation, um entsprechende Aktionen durchführen zu können. Die beschriebenen Tabellen regeln die Sichtbarkeiten von Werkzeugen und Avataren in Display-Gruppen nach Anforderung 3.5.1. Die frameweise Bestimmung der aktiven Repräsentation erlaubt es im Falle des Strahlwerkzeugs, den Strahl entsprechend des ausgewählten Schnittpunkts zu kürzen und nicht in anderen Display-Gruppen fortzuführen. Eine Verallgemeinerung auf Sichtbarkeitslisten kann die Darstellung von Szeneelementen in Display-Gruppen nach Anforderung 3.5.2 steuern.

Das nahtlose Dragging von Objekten zwischen verschiedenen Display-Gruppen ist nicht wie im kontinuierlichen Fall direkt realisierbar, sondern erfordert einen Wechsel in ein anderes Navigationskoordinatensystem und eine Änderung der aktiven Repräsentation. Sollen Objekte bereits vor dem Wechsel in andere Display-Gruppen hineinragen, so ist eine Instantiierung des betroffenen Objekts in der Display-Gruppe des Ziel-Displays

notwendig. Die Transformation von Werkzeugen in virtuelle Sichten nach Anforderung 3.4.3 ist durch zusätzliche Werkzeugrepräsentanten ebenfalls abbildbar, jedoch im Rahmen des Ansatzes nicht essentiell, um Werkzeuge in alternativen Navigationen funktionieren zu lassen.

Durch die Spezifikation allgemeiner Verfahren in abstrakten Basisklassen ist die Implementierung nach Anforderung 3.7.1 offen für Erweiterungen wie weitere Werkzeuge oder andere Regeln zur Auswahl der aktiven Repräsentation. Die Konfigurierbarkeit wird durch Dateien zur Spezifikation von Workspaces, Displays, Display-Gruppen, Nutzern, Werkzeugen und entsprechenden Sichtbarkeitstabellen geleistet. Eine beispielhafte Konfigurationsdatei dieses Ansatzes ist im Anhang in Abschnitt A.1 gegeben.

7 Zusammenfassung und künftige Arbeiten

Physikalische Multi-Display-Umgebungen erweitern die Darstellungsfläche eines Einzeldisplays im Workspace, um das Sichtfenster in die virtuelle Welt zu vergrößern und den Nutzern somit ein höheres Immersionsgefühl zu verleihen. Dynamisch erzeugbare virtuelle Displays eröffnen die Möglichkeit, Szeneinhalte festzuhalten, an anderen Stellen der virtuellen Welt zu verankern und somit die Multi-Display-Umgebung durch weitere Territorien hilfreich zu ergänzen. Im Rahmen einer Anforderungsanalyse wurde aufgezeigt, welche Funktionalitäten in solchen interaktiven Umgebungen von Bedeutung sind. Displays sollen miteinander verkoppelt werden können, so dass die angezeigten Inhalte kontinuierlich fortgeführt werden. Besteht keine solche Kopplung, können unterschiedliche Navigationen auf den Einzeldisplays dargestellt werden. Konkrete Anwendungsfälle erfordern verschiedene Kopplungsszenarien der am Setup beteiligten physikalischen und virtuellen Displays. Die im Rahmen dieser Arbeit vorgestellten Implementierungsstrategien nähern sich den Anforderungen von zwei unterschiedlichen Seiten.

Die standardmäßige kontinuierliche Abbildung betrachtet alle Displays als räumlich gekoppelt und realisiert separate Navigationen durch virtuelle Displays in Interaktionsbereichen. Individuelle Navigationen auf Nutzerebene können durch dem entsprechenden Kopfknoten vorgeschaltete Transformationen erreicht werden; diese müssen jedoch entweder im Koordinatensystem des Workspace-Navigationsknotens oder des Ausgangsknotens des virtuellen Displays ausgedrückt werden. Dies stellt jedoch einen Eingriff in die durch den Ansatz gegebene Struktur dar. Werkzeuge können direkt unterhalb des Workspace-Navigationsknotens angehängt werden, benötigen jedoch Transformationen in die virtuellen Displays alternativer Navigationen, um erwartungskonforme Manipulationen durchführen zu können.

Die separate Abbildung der Displays sieht zunächst jedes Display als eigene Display-Gruppe und ordnet dieser einen eigenen Navigationsknoten zu, fordert dadurch jedoch auch eine redundante Repräsentation der Nutzer. Um Kopplungen zu realisieren, können unproblematisch mehrere Displays einer Display-Gruppe zugeordnet werden. Wird die Navigationsmöglichkeit auf Nutzerebene verallgemeinert, so müssen zusätzlich zu den Nutzern auch die Screen-Knoten redundant abgebildet werden. Ebenso müssen Werkzeuge für alle Nutzer in allen Display-Gruppen Repräsentanten besitzen. Zur Verwaltung dieser werden Algorithmen zur Regelung der Sichtbarkeit und der aktiven Repräsentation verwendet.

Die Implementierungen und Evaluierungen mit dem Setup unseres Labors haben gezeigt, dass trotz des höheren Verwaltungsaufwands der Ansatz der separaten Abbildung überschaubar bleibt und effizient zu verwenden ist. Alternative Navigationen können ohne zusätzliche virtuelle Displays und ohne Verletzungen der Struktur direkt realisiert werden; dennoch besteht die Möglichkeit, durch die Hinzunahme von Displays zu einer Display-Gruppe Kopplungen leicht abzubilden. Die vielen Repräsentanten von Nutzern und Werkzeugen werden hinsichtlich definierter Sichtbarkeitsregeln verwaltet und Algorithmen zur Bestimmung der aktiven Repräsentation komplettieren den kontinuierlichen Raum für Werkzeuge, obwohl die Display-Gruppen und die dortigen Nutzerrepräsentationen unterschiedliche Navigationsmatrizen besitzen können. Es wurde sich daher dazu entschlossen, den Ansatz der separaten Abbildung als Basis für künftige Arbeiten auf dem Gebiet der 3D-Multi-User-Multi-Display-Umgebungen in unserem Labor einzusetzen. Im Anhang (Kapitel A) sind hierfür noch ein paar nützliche Hinweise dargestellt.

Im Rahmen künftiger Arbeiten in unserem Haus soll das Tabletop-Display des Labors durch Touchinput erweitert werden. Daraus ergibt sich die Herausforderung, den zweidimensionalen Input mehrerer Nutzer auf die dargestellte dreidimensionale virtuelle Welt in Form einer angemessenen Navigationsmetapher anzuwenden. Dabei muss bedacht werden, dass in gekoppelten Situationen der Touchinput nicht nur den Inhalt eines einzelnen Displays beeinflusst. Im Rahmen des EU-Projekts *3D-Pitoti* [1] soll auf den in dieser Arbeit entwickelten Strukturen aufbauend eine Anwendung zur Erkundung prähistorischer Felsenkunst, virtuell in Form von Punktwolken dargestellt, entwickelt werden. Dabei soll der Fokus einerseits auf der soeben erwähnten Navigation mit Hilfe von Touchinput liegen, andererseits den Nutzern durch vorschaubasierte Gruppennavigationstechniken mit virtuellen Displays helfen, interessante Orte der Szene zu lokalisieren

und zu bereisen.

Weiterhin soll künftig von den abstrakten Avatarrepräsentationen der Nutzer zu Gunsten von Video-Avataren standardmäßig abgesehen werden. Dies erfordert auf Grund des physikalischen Setups der Kameras eine Regelung der Avatare auf Workspace- anstatt auf Nutzerebene, da die Kameras den gesamten physikalischen Workspace und somit nicht explizit einzelne Nutzer erfassen. Die Sichtbarkeit der Video-Avatare soll hierbei nicht nur mit booleschen Werten in der Sichtbarkeitstabelle geregelt werden. Vielmehr soll die verwendete Displayfläche der Avatare für einen Betrachter berechnet und ab einem bestimmten Schwellenwert die Video-Avatare durch abstrakte Avatare ersetzt werden, um Renderlast zu sparen. Ein ähnlicher Anwendungsfall zeigt sich bei den virtuellen Displays, bei welchen ab einem bestimmten kleinen Größenwert das perspektivisch korrekte Bild durch ein statisches Monobild ersetzt werden kann.

Die Bestimmung des zugeordneten Nutzers eines Werkzeugs ist durch den aktuell implementierten Punkt-zu-Linie-Abstand an ein einziges Kriterium gebunden, welches unter Umständen nicht für eine korrekte Bestimmung ausreicht. In Zukunft soll Augenmerk auf ein mehrstufiges Verfahren gelegt werden, welches mit Hilfe von Kriterien der *Proxemik* [22] den zugeordneten Nutzer eines Werkzeugs zuverlässiger bestimmen und dessen Intentionen ableiten kann. Die bestehende Implementierung unterstützt die Werkzeuge Ray-Pointer und Portalkamera, wurde aber wie erläutert erweiterbar gehalten. Künftig sollen daher weitere Werkzeuge mit ihren eigenen Repräsentanten implementiert werden. Denkbar ist hier beispielsweise ein Lampenwerkzeug, welches ähnlich einer Taschenlampe Szeneinhalte (auch an entfernten Stellen durch ein virtuelles Display hindurch) beleuchtet, um diese unter gesonderten Illuminationsbedingungen analysieren zu können. Ein Repräsentant im Szenegraph bestünde in diesem Fall zusätzlich zu seinem Transformationsknoten aus einem einzelnen *Spotlight*-Knoten.

Nicht zuletzt wäre es wünschenswert, die renderseitige Beschränkung der Anzahl der Clipping-Ebenen aufzuheben, so dass Displayinhalte präziser in Form von Volumen abgeschnitten werden können, um die Überlagerung mit anderen Displayflächen zu vermeiden.

Literaturverzeichnis

[1] 3D-Pitoti - 3D acquisition, processing and presentation of prehistoric European rock-art . http://3d-pitoti.eu/. Besucht am: 09.09.2014.

[2] μracoli - The μController Radio Communication Library . http://www.nongnu.org/uracoli/. Besucht am: 04.07.2014.

[3] Nintendo DS / 3DS . http://www.nintendo.com/3ds. Besucht am: 03.09.2014.

[4] Project Blinkenlights . http://blinkenlights.net/. Besucht am: 01.09.2014.

[5] Zero Message Queue - Code Connected . http://zeromq.org/. Besucht am: 04.07.2014.

[6] AvangoNG - Cross-platform software framework for virtual reality. http://avango.org/, 2011. Besucht am: 31.08.2014.

[7] M. Agrawala, A. C. Beers, I. McDowall, B. Fröhlich, M. Bolas und P. Hanrahan. The Two-user Responsive Workbench: Support for Collaboration Through Individual Views of a Shared Space. *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, 327–332, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[8] T. Bader, A. Heck und J. Beyerer. Lift-and-drop: Crossing Boundaries in a Multi-display Environment by Airlift. *Proceedings of the International Conference on Advanced Visual Interfaces*, AVI '10, 139–146, New York, NY, USA, 2010. ACM.

[9] S. Beck, A. Kunert, A. Kulik und B. Fröhlich. Immersive Group-to-Group Telepresence. *IEEE Transactions on Visualization and Computer Graphics*, 19(4):616–625, April 2013.

[10] L. Beiser. Anaglyph stereoscopy, September 1981. US-Patent 4,290,675.

[11] E. A. Bier, M. C. Stone, K. Pier, W. Buxton und T. D. DeRose. Toolglass and Magic Lenses: The See-through Interface. *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, 73–80, New York, NY, USA, 1993. ACM.

[12] C. Cruz-Neira, D. J. Sandin, T. A. DeFanti, R. V. Kenyon und J. C. Hart. The CAVE: Audio Visual Experience Automatic Virtual Environment. *Communications of the ACM*, 35(6):64–72, Juni 1992.

[13] N. Elmqvist. Distributed User Interfaces: State of the Art. In J. A. Gallud, R. Tesoriero und V. M.R. Penichet, *Distributed User Interfaces*, Human-Computer Interaction Series, 1–12. Springer London, 2011.

[14] C. Forlines, A. Esenther, C. Shen, D. Wigdor und K. Ryall. Multi-user, Multi-display Interaction with a Single-user, Single-display Geospatial Application. *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*, UIST '06, 273–276, New York, NY, USA, 2006. ACM.

[15] C. Forlines und R. Lilien. Adapting a Single-user, Single-display Molecular Visualization Application for Use in a Multi-user, Multi-display Environment. *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '08, 367–371, New York, NY, USA, 2008. ACM.

[16] A. Fuhrmann und E. Gröller. Real-time Techniques for 3D Flow Visualization. *Proceedings of the Conference on Visualization '98*, VIS '98, 305–312, Los Alamitos, CA, USA, 1998. IEEE Computer Society Press.

[17] T. Grossman, D. Wigdor und R. Balakrishnan. Multi-finger Gestural Interaction with 3D Volumetric Displays. *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*, UIST '04, 61–70, New York, NY, USA, 2004. ACM.

[18] Y. Kitamura, T. Konishi, S. Yamamoto und F. Kishino. Interactive Stereoscopic Display for Three or More Users. *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, 231–240, New York, NY, USA, 2001. ACM.

[19] W. Krüger, C.-A. Bohn, B. Fröhlich, H. Schüth, W. Strauss und G. Wesche. The Responsive Workbench: A Virtual Work Environment. *IEEE Computer*, 28(7):42–48, Juli 1995.

[20] A. Kulik, A. Kunert, S. Beck, R. Reichel, R. Blach, A. Zink und B. Fröhlich. C1x6: A Stereoscopic Six-user Display for Co-located Collaboration in Shared Virtual Environments. *Proceedings of the 2011 SIGGRAPH Asia Conference*, SA '11, 188:1–188:12, New York, NY, USA, 2011. ACM.

[21] A. Kunert, A. Kulik, S. Beck und B. Fröhlich. Photoportals: Shared References in Space and Time. *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work; Social Computing*, CSCW '14, 1388–1399, New York, NY, USA, 2014. ACM.

[22] N. Marquardt, R. Diaz-Marino, S. Boring und S. Greenberg. The Proximity Toolkit: Prototyping Proxemic Interactions in Ubiquitous Computing Ecologies. *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, 315–326, New York, NY, USA, 2011. ACM.

[23] J. Mentz und S. Caldwell. Method of stereoscopic 3d viewing using wireless or multiple protocol capable shutter glasses, August 2010. US-Patent App. 12/699,685.

[24] M. Nacenta, C. Gutwin, D. Aliakseyeu und S. Subramanian. There and Back again: Cross-Display Object Movement in Multi-Display Environments. *Journal of Human-Computer Interaction*, Ausgabe 24, 170–229, 2009.

[25] M. A. Nacenta, D. Aliakseyeu, S. Subramanian und C. Gutwin. A Comparison of Techniques for Multi-display Reaching. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, 371–380, New York, NY, USA, 2005. ACM.

[26] M. A. Nacenta, M. R. Jakobsen, R. Dautriche, U. Hinrichs, M. Dörk, J. Haber und S. Carpendale. The LunchTable: A Multi-user, Multi-display System for Information Sharing in Casual Group Interactions. *Proceedings of the 2012 International Symposium on Pervasive Displays*, PerDis '12, 18:1–18:6, New York, NY, USA, 2012. ACM.

[27] M. A. Nacenta, S. Sakurai, T. Yamaguchi, Y. Miki, Y. Itoh, Y. Kitamura, S. Subramanian und C. Gutwin. E-conic: A Perspective-aware Interface for Multi-display Environments. *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, UIST '07, 279–288, New York, NY, USA, 2007. ACM.

[28] M. A. Nacenta, S. Sallam, B. Champoux, S. Subramanian und C. Gutwin. Perspective Cursor: Perspective-based Interaction for Multi-display Environments. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, 289–298, New York, NY, USA, 2006. ACM.

[29] K. Perlin, S. Paxia und J. S. Kollin. An Autostereoscopic Display. *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, 319–326, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[30] C. Pirchheim, M. Waldner und D. Schmalstieg. Deskotheque: Improved Spatial Awareness in Multi-Display Environments. *IEEE Virtual Reality 2009*, 123–126, März 2009.

[31] M. Pouliquen, A. Bernard, J. Marsot und L. Chodorge. Virtual hands and virtual reality multimodal platform to design safer industrial systems. *Computers in Industry*, 58(1):46 – 56, 2007.

[32] J. Rekimoto. Pick-and-drop: A Direct Manipulation Technique for Multiple Computer Environments. *Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology*, UIST '97, 31–39, New York, NY, USA, 1997. ACM.

[33] J. Rekimoto und M. Saitoh. Augmented Surfaces: A Spatially Continuous Work Space for Hybrid Computing Environments. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '99, 378–385, New York, NY, USA, 1999. ACM.

[34] D. Schmalstieg, A. Fuhrmann, G. Hesina, Z. Szalavári, L. M. Encarnaçäo, M. Gervautz und W. Purgathofer. The Studierstube Augmented Reality Project. *Presence: Teleoperators and Virtual Environments*, 11(1):33–54, Februar 2002.

[35] S. Schneegans, F. Lauer, A.-C. Bernstein, A. Schollmeyer und B. Fröhlich. guacamole - An Extensible Scene Graph and Rendering Framework Based on Deferred Shading. *Proceedings of the 7th Workshop on Software Engineering and Architectures for Realtime Interactive Systems*, SEARIS '14, Minneapolis, MN, USA, März 2014.

[36] S. D. Scott, M. S. T. Carpendale und K. M. Inkpen. Territoriality in Collaborative Tabletop Workspaces. *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work*, CSCW '04, 294–303, New York, NY, USA, 2004. ACM.

[37] T. Seyed, C. Burns, M. Costa Sousa und F. Maurer. From Small Screens to Big Displays: Understanding Interaction in Multi-display Environments. *Proceedings of the Companion Publication of the 2013 International Conference on Intelligent User Interfaces Companion*, IUI '13 Companion, 33–36, New York, NY, USA, 2013. ACM.

[38] D.M. Sheiman. Stereoscopic viewing system and glasses, Mai 1988. US-Patent 4,744,633.

[39] D. C. Smith, C. Irby, R. Kimball, B. Verplank und E. Harslem. Perspectives on the computer revolution. chapter Designing the Star User Interface (1982), 261–283. Ablex Publishing Corp., Norwood, NJ, USA, 1989.

[40] R. Stoakley, M. J. Conway und R. Pausch. Virtual Reality on a WIM: Interactive Worlds in Miniature. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95, 265–272, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

[41] S. L. Stoev und D. Schmalstieg. Application and Taxonomy of Through-the-lens Techniques. *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, VRST '02, 57–64, New York, NY, USA, 2002. ACM.

[42] N. A. Streitz, J. Geißler, T. Holmer, S. Konomi, C. Müller-Tomfelde, W. Reischl, P. Rexroth, P. Seitz und R. Steinmetz. i-LAND: An Interactive Landscape for Creativity and Innovation. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '99, 120–127, New York, NY, USA, 1999. ACM.

[43] L. Terrenghi, A. Quigley und A. Dix. A taxonomy for and analysis of multi-person-display ecosystems. *Personal and Ubiquitous Computing*, 13(8):583–598, 2009.

[44] J. Viega, Matthew J. Conway, G. Williams und R. Pausch. 3D Magic Lenses. *Proceedings of the 9th Annual ACM Symposium on User Interface Software and Technology*, UIST '96, 51–58, New York, NY, USA, 1996. ACM.

[45] A. D. Wilson und H. Benko. Combining Multiple Depth Cameras and Projectors for Interactions on, above and between Surfaces. *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, 273–282, New York, NY, USA, 2010. ACM.

A Anhang

In Kapitel 7 wurde beschrieben, dass die Implementierungsstrategie der separaten Abbildung der Displays des Workspaces auf Grund ihrer Mächtigkeit als Basis für künftige Projekte im Bereich der Multi-Display-Umgebungen in unserem Labor dienen soll. In diesem Kapitel sollen daher ein paar nützliche Anmerkungen für Applikationsentwickler zum Umgang mit der ausgearbeiteten Softwarestruktur dargelegt werden.

A.1 Beispielhafte Konfigurationsdatei

Die im Folgenden angehängte Konfigurationsdatei spezifiziert das Display-Setup unseres Labors. Innerhalb eines Workspaces werden dabei separate Display-Gruppen für Leinwand- und Tabletop-Display erzeugt und durch einen zweiten Workspace erweitert, welcher nur eine Display-Gruppe mit einem weiteren Leinwand-Display besitzt. Den Display-Gruppen werden entsprechende Navigationen zugeordnet, deren Spur durch eine Sichtbarkeitsliste in ihrer Anzeige beschränkt wird. Auf Workspaceebene werden Nutzer und Werkzeuge mit ihren Sichtbarkeitstabellen definiert. Zuletzt wird einer weiteren Display-Gruppe ein virtuelles Portaldisplay zugewiesen, welches einen statischen Standpunkt in der Szene abbildet. Im Quelltext wurden alle offensichtlichen oder zu vorherigen Betrachtungen analogen Stellen durch *[...]* gekürzt.

```
#!/usr/bin/python

## @file
# Contains workspace, display, navigation, display group and user configuration classes.

# import guacamole libraries
# [...]

# import framework libraries
```

```
# [...]

## Create Workspaces first ##
# parameters: name, tracking transmitter offset
vr_lab_rear = Workspace('VR-Lab-Rear'
                        , avango.gua.make_trans_mat(0.0, 0.043, 0.0))
vr_lab_front = Workspace('VR-Lab-Front'
                         , avango.gua.make_trans_mat(0.0, 0.043, 0.0))

workspaces = [vr_lab_rear, vr_lab_front]

## Create Navigation instances ##
# visibility list
# format: {A : bool, ...}
# interpretation: does display with tag A see the movement trace of the navigation
trace_visibility_list_dlp_wall_nav = {  "dlp_wall"  : False
                                      , "lcd_wall" : False
                                      , "table" : True
                                      , "portal" : False
                                     }

trace_visibility_list_table_nav = { [...] }

trace_visibility_list_lcd_wall_nav = { [...] }


spheron_navigation = SteeringNavigation()
spheron_navigation.my_constructor( STARTING_MATRIX = avango.gua.make_trans_mat(0, 0, 15) * \
                                                     avango.gua.make_rot_mat(0, 0, 1, 0)
                                 , STARTING_SCALE = 1.0
                                 , INPUT_DEVICE_TYPE = 'NewSpheron'
                                 , INPUT_DEVICE_NAME = 'device-new-spheron'
                                 , NO_TRACKING_MAT = avango.gua.make_trans_mat(0, 1.75, 1.6)
                                 , GROUND_FOLLOWING_SETTINGS = [True, 0.75]
                                 , INVERT = False
                                 , TRACE_VISIBILITY_LIST =
                                     trace_visibility_list_dlp_wall_nav
                                 , DEVICE_TRACKING_NAME = 'tracking-new-spheron'
                                 , REACTS_ON_PORTAL_TRANSIT = True)

spacemouse_navigation = SteeringNavigation()
spacemouse_navigation.my_constructor( [...] )

xbox_navigation = SteeringNavigation()
xbox_navigation.my_constructor( [...] )

old_spheron_navigation = SteeringNavigation()
old_spheron_navigation.my_constructor( [...] )

## Create Display instances. ##
large_powerwall = LargePowerwall()
touch_table_3D = TouchTable3D()
```

```
small_powerwall = SmallPowerwall()

displays = [large_powerwall, touch_table_3D, small_powerwall]

## Create display groups ##
vr_lab_rear.create_display_group( DISPLAY_LIST = [large_powerwall]
                                , NAVIGATION_LIST = [spheron_navigation, xbox_navigation]
                                , VISIBILITY_TAG = "dlp_wall"
                                , OFFSET_TO_WORKSPACE = avango.gua.make_trans_mat(0, 0, 1.6)
                                    )

vr_lab_rear.create_display_group( DISPLAY_LIST = [touch_table_3D]
                                , NAVIGATION_LIST = [spacemouse_navigation]
                                , VISIBILITY_TAG = "table"
                                , OFFSET_TO_WORKSPACE = avango.gua.make_trans_mat(0.79,
                                    -0.96, 1.96) * \
                                                        avango.gua.make_rot_mat(-90, 0, 1,
                                                            0) )

vr_lab_front.create_display_group( DISPLAY_LIST = [small_powerwall]
                                , NAVIGATION_LIST = [old_spheron_navigation]
                                , VISIBILITY_TAG = "lcd_wall"
                                , OFFSET_TO_WORKSPACE = avango.gua.make_trans_mat(0, 0, 1.6)
                                    )

## Create users ##
# visibility table
# format: {A : { B : bool, ...}, ... }
# interpretation: does display with tag A see avatar of user in displays with tag B?
avatar_visibility_table = {
          "dlp_wall"  : {"table" : False, "lcd_wall" : True, "portal" : False}
        , "table" :     {"dlp_wall" : True, "lcd_wall" : True, "portal" : False}
        , "lcd_wall" :  {"dlp_wall" : True,  "table" : False, "portal" : False}
        , "portal" :    {"dlp_wall" : True, "table" : False, "lcd_wall" : True} }


vr_lab_rear.create_user( VIP = False
                       , AVATAR_VISIBILITY_TABLE = avatar_visibility_table
                       , HEADTRACKING_TARGET_NAME = 'tracking-dlp-glasses-1'
                       , EYE_DISTANCE = 0.065)

vr_lab_rear.create_user( [...] )

vr_lab_rear.create_user( [...] )


vr_lab_front.create_user( [...] )

vr_lab_front.create_user( [...] )

## Create tools ##
```

```
# visibility table
# format: {A : { B : bool, ...}, ... }
# interpretation: does display with tag A see representation of tool in displays with tag B?
tool_visibility_table = {
          "dlp_wall"  : {"table" : True, "portal" : False}
        , "table" :     {"dlp_wall" : True, "portal" : False}
        , "lcd_wall" :  {"dlp_wall" : True, "table" : False, "portal" : False}
        , "portal" :    {"dlp_wall" : True, "table" : False, "lcd_wall" : True} }

vr_lab_rear.create_ray_pointer( POINTER_TRACKING_STATION = 'tracking-dlp-pointer1'
                              , POINTER_DEVICE_STATION = 'device-pointer1'
                              , VISIBILITY_TABLE = tool_visibility_table)

vr_lab_rear.create_portal_cam(  CAMERA_TRACKING_STATION = 'tracking-portal-camera-32'
                             ,  CAMERA_DEVICE_STATION = 'device-portal-camera-32'
                             ,  VISIBILITY_TABLE = tool_visibility_table)

## Create portal navigations. ##

# specification of virtual displays and assignment to display groups
# virtual display groups belong to no workspaces, they are added to the list
    portal_display_groups

portal_1_nav = StaticNavigation()
portal_1_nav.my_constructor(STATIC_ABS_MAT = avango.gua.make_trans_mat(-12.0, 17.3, -7.0)
                          , STATIC_SCALE = 1.0)


portal_1 = Portal(PORTAL_MATRIX = avango.gua.make_trans_mat(-23.0, 1.3, 21.0)
                , WIDTH = 4.0
                , HEIGHT = 2.6
                , VIEWING_MODE = "3D"
                , CAMERA_MODE = "PERSPECTIVE"
                , NEGATIVE_PARALLAX = "False"
                , BORDER_MATERIAL = "data/materials/White.gmd"
                , TRANSITABLE = True)

## Create virtual display groups ##
portal_1_dg = DisplayGroup(ID = None
                         , DISPLAY_LIST = [portal_1]
                         , NAVIGATION_LIST = [portal_1_nav]
                         , VISIBILITY_TAG = "portal"
                          )

portal_display_groups = [portal_1_dg]
```

A.2 UML-Klassendiagramm

Während im vorherigen Abschnitt die Konfigurierbarkeit der Applikation mit Hilfe von speziellen Dateien dargestellt wurde, sollen nun die etwas tiefer liegenden Strukturen beleuchtet und erläutert werden. Abbildung A.1 zeigt hierfür ein stark vereinfachtes UML-Klassendiagramm. Um den Fokus auf die Klassen und ihre Zusammenhänge zu legen, wurde bewusst auf die meisten Attribute und Methoden verzichtet. Diese wurden in Form einer Code-Dokumentation, welche Teil des Quelltexts ist, separat beschrieben. Um dem Applikationsentwickler jedoch einen Einstieg in die Programmlogik zu geben, sollen im Rahmen dieser Ausarbeitung kurz die Verantwortlichkeiten der einzelnen Klassen dargestellt werden.

Um den Szenegraph zu Beginn der Applikation korrekt aufzubauen, analysiert der *ApplicationManager*, welchen *Workspace*s welche *DisplayGroup*-Instanzen in der Konfigurationsdatei zugeordnet wurden. Eine *DisplayGroup* wiederum hält mehrere *PhysicalDisplay*s und *VirtualDisplays*, um die Inhalte der virtuellen Welt darauf anzuzeigen. Weiterhin gehören zu jeder *DisplayGroup* ein oder mehrere *Navigation*en. Diese können entweder 6-DOF-*SteeringNavigation*s, welche mit Hilfe eines *6DOFDevice*s, *InputMapping*s und *GroundFollowing*s ihre Matrizen berechnen, oder konstante *StaticNavigations* sein, welche eine festgelegte, nicht-modifizierbare Orientierung der Szene repräsentieren.

Für jeden in der Konfigurationsdatei dem *Workspace* zugeordneten *User* werden zu Applikationsbeginn *UserRepresentation*s erzeugt. Diesen sind Avatargeometrien zur Visualisierung des Nutzers in der *DisplayGroup* zugeordnet. Weiterhin sind sie dafür verantwortlich, aus den *Navigation*en der zugehörigen *DisplayGroup* eine auszuwählen und ihren entsprechenden Szenegraphknoten damit zu verbinden. Für im *Workspace* instantiierte Sub-Klassen von *Tool*, namentlich *RayPointer* und *PortalCam*, wird für jede *UserRepresentation* eine *ToolRepresentation* erzeugt. Wenn die aktive Repräsentation benötigt wird, führt die entsprechende *Tool*-Instanz das in Abschnitt 6.4 beschriebene Verfahren durch.

Um die Sichtbarkeiten der Repräsentationen nach Abschnitt 6.3 zu regeln, erben die Klassen *User* und *Tool* von *VisibilityHandler2D*. Diese abstrakte Klasse gibt vor, dass eine zweidimensionale Sichtbarkeitstabelle, wie in Tabelle 6.1 dargestellt, definiert sein muss. Die Regelung der Sichtbarkeiten aller Repräsentationen wird dann durch den Aufruf einer

speziellen Funktion geleistet. Ähnlich dazu erlaubt die abstrakte Klasse *VisibilityHandler1D* die Regelung der Sichtbarkeiten von Szeneelementen mit Hilfe von Sichtbarkeitslisten, welche in Abschnitt6.3.3 kurz erläutert wurde. Die Klasse *TraceLines*, welche zur Visualisierung der Spur von *SteeringNavigation*s dient, macht von dieser Funktionalität Gebrauch.

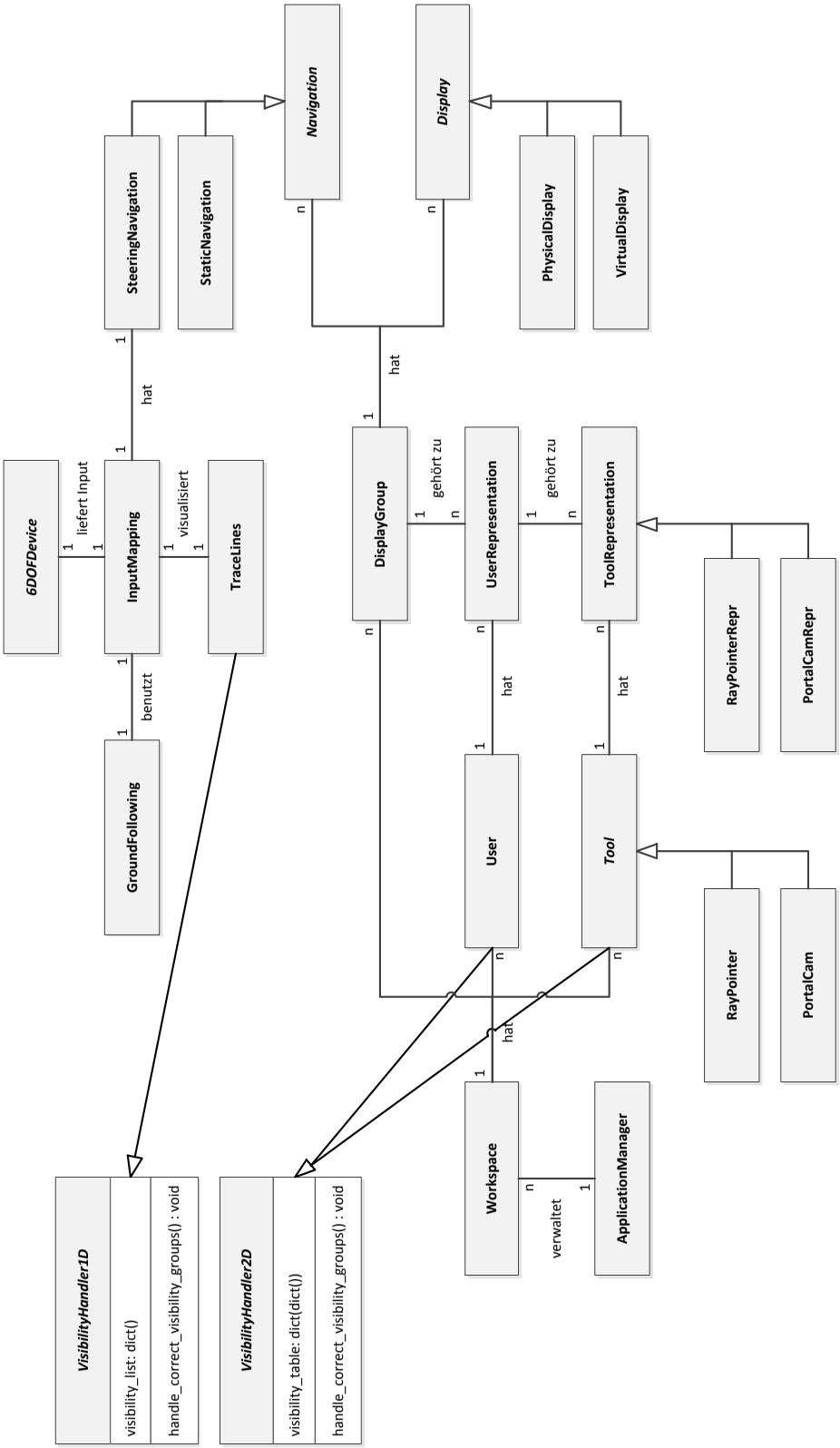


Abbildung A.1: Vereinfachtes Klassendiagramm der Implementierung der separaten Abbildung der Displays des Workspaces.