Evaluation der Toolchain für 3D-Objektmanipulation in VR-Trainingsumgebungen

Niklas Gerwens und Jonathan Becker
niklas.gerwens@haw-hamburg.de
jonathan.becker@haw-hamburg.de

Hochschule für Angewandte Wissenschaften Hamburg, Deutschland
http://www.besteseitederwelt.de/

Zusammenfassung. In dieser Ausarbeitung geht es um die Evaluation der von uns im Grundprojekt entworfenen Toolchain für 3D-Objektmanipulation in VR-Trainingsumgebungen. Dazu wird zunächst die Integration der Hardware Plattform in das Projekt des Frameworks beschrieben. Danach folgt die Evaluation der Toolchain durch Nutzertests des Frameworks in einem konkreten Trainingsszenario. Dabei werden sowohl für die Plattform als auch für das Framework erfolgte Änderungen und Erweiterungen erläutert. Abschließend wird ein grundlegendes Fazit und ein Ausblick für die Toolchain gegeben.

Schlüsselwörter: Virtual Reality · Mensch-Computer-Interaktion · 3D-Interaktionsdesign · VR-Training · 3D-Objektmanipulation · Toolchain

1 Einleitung

Der Einsatz von Virtual Reality (VR) in verschiedenen Anwendungskontexten hat in letzter Zeit stark zugenommen. So wird auch zunehmend im Trainings- und Schulungskontext VR genutzt, um eine immersive Lernumgebung mit interagierbaren 3D-Objekten zu schaffen. Dabei helfen neuartige Eingabegeräte, welche eine diverse Interaktionsgestaltung ermöglichen, den Lernerfolg der Anwendung zu steigern. Doch ergeben sich aus der Diversität der Eingabegeräte und der konkreten Ziele der Anwendungen auch Probleme für die Entwickler der VR-Umgebungen. So sind die Schnittstellen der Eingabegeräte nicht genormt und bestimmte Interaktionen nicht immer gleich umzusetzen. Die genaue Abbildung einer Interaktion kann in einem Kontext sehr wichtig und in einem anderen zu vernachlässigen sein. Es gilt den richtigen Abstraktionsgrad und geeignete Eingabegeräte zu finden, um Nutzer beim Lernen zu unterstützen, aber die Kosten und den Aufwand möglichst gering zu halten. Da für diese Problemstellung noch keine Vorgehensweisen oder Hilfestellungen für Entwickler existieren, wurde im Rahmen des Grundprojekts[1] eine Toolchain für 3D-Objektmanipulation

[1] https://users.informatik.haw-hamburg.de/~ubicomp/projekte/master2021-proj/beckerGP.pdf

in VR-Trainingsumgebungen entwickelt.
Die Toolchain besteht hauptsächlich aus zwei Komponenten, einer Plattform zur Integration von VR-Eingabegeräten und einem Framework zur uniformen Gestaltung von Interaktionen in VR-Umgebungen. Dabei wurden im Grundprojekt beide Komponenten getrennt voneinander vorgestellt und eine Zusammenführung in Aussicht gestellt. Des Weiteren wurden mögliche Erweiterungen zur Verbesserung beider Komponenten formuliert. Diese Ziele bilden die Grundlage für dieses Projekt. Im Zuge des Projekts werden die beiden Komponenten überarbeitet, erweitert und zusammengeführt sowie der Einsatz der Toolchain in einer konkreten Anwendung aus dem Schulungskontext evaluiert.
Dafür wird zunächst die Integration der Plattform in das Projekt des Frameworks und die dabei erfolgten Änderungen der Plattform beschrieben. Danach wird die Überarbeitung des Frameworks und die Evaluation durch Nutzertests vorgestellt. Abschließend folgt eine Zusammenfassung des Projekts und ein Ausblick auf zukünftige Projekte.

2 Integration der Hardware Plattform

Die rapide Entwicklung von VR-Eingabegeräten ohne einheitliche Schnittstellen oder ausführliche Interaktionsframeworks in den letzten Jahren zwingt VR-Entwickler frühzeitig konkrete Hardware auszuwählen und diese von der ersten Entwicklungsphase an beizubehalten. Dadurch wird eine Untersuchung verschiedener Hardware Typen unter Berücksichtigung des Anwendungskontexts erschwert und häufig auf Techniken zurückgegriffen, welche den aus dem Umgang mit 2D-Welten bekannten Techniken ähneln, aber das Potential der 3D-Umgebung für die Anwendung nicht ausschöpfen. Als Beitrag zu dieser Problemstellung beschäftigt sich der folgende Teil dieser Ausarbeitung mit der Gestaltung einer Plattform zur einfacheren Einbindung von Hardware für die Interaktion mit Objekten in der VR.
Hierbei wurden die im Grundprojekt als weiterführende Ziele formulierten Aspekte aufgegriffen. Einerseits sollte zur Evaluation eine Kombination mit der Frameworks Komponente erfolgen und andererseits sollten notwendige Erweiterungen für ein komplexeres Szenario umgesetzt werden. Dazu wurde die Plattform nachträglich in das Projekt integriert, in dem die Frameworks Komponente zum Einsatz kommt. Diese Integration ist jedoch nur ein Beispiel für die Benutzung der Plattform. Die Qualitätseigenschaften der Plattform, das sie in verschiedenen Anwendungen eingesetzt und um weitere Eingabegeräte erweitert werden kann, sollen auch weiterhin gegeben sein. Im Folgenden wird zuerst der Stand vor der Integration dargestellt, bevor die Durchführung der Integration erläutert und der finale Stand der Plattform hinsichtlich der geforderten Ziele evaluiert wird.

2.1 Projekt pre Plattform

In diesem Kapitel werden hauptsächlich die Aspekte des Projekts thematisiert, welche für die Integration der Hardware Plattform relevant sind. Eine vollständige Beschreibung der Szenarien, der Interaktionsabläufe sowie Implementationsdetails des Frameworks finden sich in Kapitel 3.
In der ursprünglichen Version werden ausschließlich die HTC Vive Controller genutzt, um mit den Objekten in der virtuellen Umgebung zu interagieren. Dabei werden die Position, die Rotation sowie der Zustand des Trigger Knopfes der Controller über das SteamVR Framework für Unity abgefragt und mit Hilfe der SteamVR Controller Modelle dargestellt. Das selbst entwickelte Framework ergänzt diese Darstellung um ein IC Controller Grab Skript auf den einzelnen Controller Modellen, welches bei Druck des Triggers interagierbare Objekte in Reichweite selektiert. Selektierte Objekte reagieren anschließend anhand einer Auswahl an Skripten, die auf das entsprechende Objekt direkt platziert werden. Zumeist folgen sie dem Controller Modell innerhalb von einstellbaren Grenzen, die sich aus dem Interaktionskontext ergeben. Dabei erfolgen Positions- und Rotationsänderungen auf Basis eigener Berechnungen und sind unabhängig von physikalischen Systemen. Im Interaktionsszenario von dem manipulierten Objekt abhängige Objekte werden dann mit Hilfe eines Eventsystems über Änderungen informiert.

2.2 Integration der Hardware Plattform

Bei der Einbindung der Hardware Plattform in das bestehende Projekt gilt es zwei wichtige Anforderungen zu beachten. Erstens soll von der genutzten Hardware möglichst abstrahiert werden. Der Entwickler soll bei dem Entwurf der Interaktion sich nicht auf einen Typ von Hardware festlegen müssen und die konkrete Hardware innerhalb der Hardware Kategorie soll austauschbar bleiben. Zweitens sollte auf den Einsatz von physikalischen Elementen zur Integration verzichtet werden, da diese vom Framework nicht unterstützt werden.
Konkret müssen die folgenden vier Aspekte für die von der Plattform unterstützten Hardware Kategorien Zusätzliche Artefakte, Hand-tracking basierte Ansätze und Hand-unabhängige Ansätze in das Projekt integriert werden:

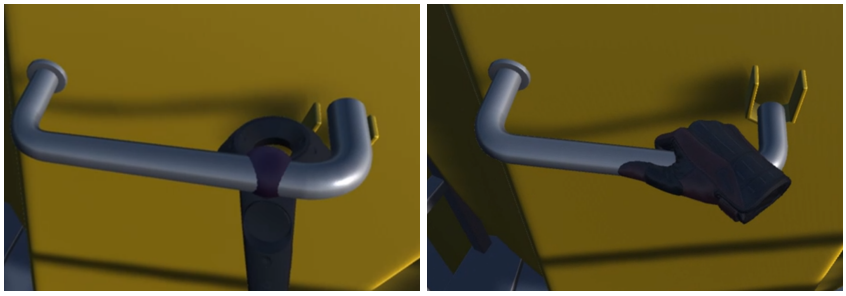
- Darstellung des Nutzer Inputs in der virtuellen Umgebung
- Selektion des zu manipulierenden Objekts
- Trigger Bedingung für die Manipulation
- Objektspezifisches Feedback/Manipulation

Dabei wird das objektspezifische Feedback größtenteils durch das Framework bereitgestellt, während die anderen drei Aspekte mehr unter den Aufgabenbereich der Hardware Plattform fallen. Gemäß der oben aufgeführten Anforderungen sollte die Schnittstelle der Plattform zum Framework unabhängig von der Hardware Kategorie sein. Für die im Grundprojekt geschaffene Plattform und das Framework sind dafür einige Anpassungen nötig, die nach den drei Hardware Kategorien gegliedert in den nächsten Kapiteln erläutert werden. Dabei wird das

Eingangstürszenario, welches in Abschnitt 3.2 beschrieben ist, als Beispiel für eine Integration der Plattform genutzt.

Zusätzliche Artefakte Die Integration von zusätzlichen Artefakten zur Interaktion mit der virtuellen Umgebung des Projekts, unabhängig ob bereits von der Plattform unterstützt oder generell betrachtet, gestaltet sich relativ einfach, da das Framework ursprünglich für den Einsatz der HTC Vive Controller entworfen wurde. Für die Darstellung der zusätzlichen Objekte liefern die meisten Hersteller Modelle entweder in gängigen 3D-Formaten, die von Entwicklungsumgebungen wie Unity unterstützt werden oder als Teil eines Prefabs, welches neben den Modellen auch Skripte beinhaltet, um Position und Rotation der Modelle zu aktualisieren sowie Knopfdrücke und Werte zusätzlicher Sensoren der Artefakte in der Entwicklungsumgebung bereitzustellen. Das Projekt benutzt ein von SteamVR entwickeltes Prefab, welches dynamisch beide Controller Typen der Plattform erkennt und Modelle dieser in der virtuellen Umgebung darstellt. Um Modelle von Artefakten anderer Hersteller einzubinden, würde man das SteamVR Prefab durch ein vom jeweiligen Hersteller veröffentlichtes Prefab/Modell ersetzen.
Im Grundprojekt wurden für die Selektion und Manipulation von Objekten Skripte verschiedener Frameworks innerhalb der Plattform kombiniert. Diese sind jedoch nur für die jeweilige Hardware ausgelegt und können nicht allgemein für andere Eingabegeräte genutzt werden. So funktionieren die Skripte des SteamVR Interaktionsframeworks nur mit den von Steam unterstützten zusätzlichen Artefakten. Das selbst entwickelte Framework hingegen verfolgt mit dem IC Controller Grab Skript einen generischen Ansatz für Controller. Es lässt sich auf beliebige Modelle von zusätzlichen Artefakten ziehen. Für den Ausgangspunkt der Selektion kann ein passender Teil des verwendeten Modells über einen Parameter gesetzt werden. Dieser dient später auch als Punkt, um gegriffene Objekte zu befestigen. Auch die Trigger Bedingung für die Interaktion kann auf ein beliebiges Unity Event eingestellt werden, welches von Vorteil ist, da Hardware, die eine Schnittstelle zu Unity besitzt, zumeist auch eine Einbindung in das Unity Event System bietet. Durch das einfache Design des Event Systems von Unity ist aber auch eine eigene Integration leicht umzusetzen.
Das objektspezifische Feedback, in unserem Beispielszenario das Verhalten der Eingangstür, wird durch das Framework spezifiziert. So folgen Hebel und Tür bei einer erfolgreichen Selektion innerhalb ihrer Rotationsgrenzen dem Objekt, welches sie gegriffen hat. Hierbei ist hervorzuheben, dass die Skripte zur Manipulation von Objekten nicht zwischen Eingabegeräten unterscheiden. Nur die Ausstattung des jeweiligen Modells mit dem IC Controller Grab Skript und eine korrekte Einstellung der Parameter ist erforderlich.
Da diese Schritte für die HTC Vive Controller bereits erfolgt sind, bleiben nur die Valve Index Controller als zusätzliche Artefakte der Plattform zu integrieren. Dafür wird das dynamisch von SteamVR Prefab ausgewählte Modell, um das IC Controller Grab Skript ergänzt, der Ausgangspunkt auf die Mitte der Hand eingestellt und das Unity Event für die Trigger Bedingung auf das Event gesetzt, welches beim Schließen der Hand von SteamVR geworfen wird. Folgend

lassen sich Tür und Türgriff entweder durch Drücken des Triggerknopfes bei den HTC Vive Controller oder durch das Schließen der Hand bei den Vive Index Controllern greifen, wenn die Modelle der Controller abhängig von den jeweiligen Ausgangspunkten sich nahe genug an den Collidern der Interaktionsobjekte befinden.



(a) HTC Vive Controller (b) Vive Index Controller

Abb. 1. Vergleich der Controller Modelle beim Öffnen der Eingangstür

Optisches Tracking Auch bei Optischen Tracking Ansätzen bieten die Hersteller fast immer Modelle und Schnittstellen zur Hardware für die bekannten VR- Entwicklungsumgebungen an. Sind diese erst mal in das Projekt importiert und im Settings Skript der Plattform registriert wird das passende Modell der aktiven Hardware zur Laufzeit geladen. Die Selektion kann durch den Aufbau des IC Controller Grab Skripts auf die gleiche Weise wie bei den zusätzlichen Artefakten erfolgen. Es wird wieder unabhängig vom konkreten Aufbau des Modells nur ein Ausgangspunkt und eine Trigger Bedingung benötigt. Hierbei ist zu beachten, dass im Gegensatz zu einem Controller keine physikalischen Knöpfe oder Drucksensoren bei optischen Tracking Ansätzen vorhanden sind. Deswegen ist die Nutzung einer Gestenerkennung notwendig, um ein Trigger Event bei bestimmten Bewegungen der Hand zu erhalten. Grundsätzlich gibt es für die meisten Hardware Systeme Lösungen für Gestenerkennung von den Herstellern oder Drittparteianbietern. Eine abstrakte Lösung für Gestenerkennung ist durch die Unterschiede in den Modellen und Sensoren der einzelnen Tracking Verfahren jedoch sehr schwierig umzusetzen. Sind die Parameter des IC Controller Grab Skripts gesetzt, kann auch das Feedback der Objekte bei einer erfolgreichen Selektion analog zum Ablauf bei den zusätzlichen Artefakten erfolgen.
Für die Integration der Plattform in das Beispielszenario bedeutet das folgende Schritte. Dem Leap Motion Modell als Teil der Plattform wird das IC Controller Grab Skript hinzugefügt. Das Objekt, welches in dem Hand Modell den Handrücken repräsentiert, wird als Ausgangspunkt definiert. Das Handgeschlossen Event der Leap Motion Gestenerkennung wird als Trigger Event deklariert. An-

schließend kann der Nutzer Tür und Türgriff mit der optisch getrackten Hand greifen und bewegen.

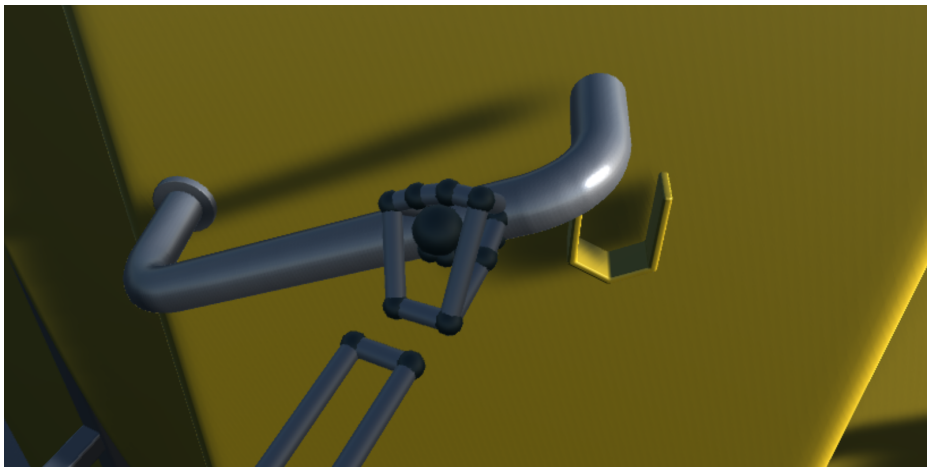


Abb. 2. Leap Motion im Szenario

Hand-unabhängige Ansätze Im Gegensatz zu den auf Artefakten oder Hand-Tracking basierenden Ansätzen gibt es für die Hand-unabhängigen Ansätze noch keine Implementation eines konkreten Hardwaretyps oder eine Schnittstelle wie das IC Controller Grab Skript, die genutzt werden kann. Deswegen muss die Plattform, die mit der Sprachsteuerung bereits eine Hand-unabhängige Interaktionslösung besitzt, um eine Komponente erweitert werden, welche die Sprachsteuerung und andere Hand-unabhängige Eingabegeräte mit den Frameworksskripten zur Objektmanipulation verbindet. Des Weiteren muss wie im Grundprojekt bereits angemerkt aufgrund der erhöhten Komplexität des Projekts auch eine Möglichkeit zur Selektion des gewünschten Interaktionsobjekts geschaffen werden. Die grundsätzliche Funktionalität der Sprachsteuerungsimplementation ist durch die identischen Rahmenbedingungen der Projekte, konkret Windows als Betriebssystem und Unity als Entwicklungsumgebung sowie die HTC Vive als Ausgabegerät, gewährleistet.
Für die Verbindung von Plattform und Framework wurde ein Handler Skript entworfen. Dieses hält einerseits eine Liste an Events, die bei der Eingabe des Nutzers durch das Hand-unabhängige Eingabegerät generiert werden und andererseits eine Liste an Events, welche durch das Framework zur Objektmanipulation angeboten werden. Das Skript wird auf das Interaktionsobjekt platziert und die Listen durch Referenzen auf die entsprechenden Skripte gefüllt. Anschließend kann eine Verknüpfung der Events beider Listen erfolgen. Im Beispielszenario

befindet sich das Handler Skript auf der Eingangstür und hält Referenzen auf das Sprachsteuerungsskript mit den registrierten Sprachbefehlen und das ICRotation Skript der Tür, welches Events besitzt, um die Tür in einen offen oder einen geschlossen Zustand zu setzen. So können die Events von der Sprachsteuerung bei der Erkennung der Befehle "Tür öffnen" und "Tür schließen" mit den jeweiligen Events des ICRotation Skripts verknüpft werden.
Damit sich bei einem erfolgreichen Sprachbefehl nicht alle Türen in der Anwendung öffnen oder schließen sondern nur die Eingangstür muss eine Selektion dieser erfolgen. Zu diesem Zweck wurde die Plattform um eine Raycast Komponente erweitert. Ausgehend von der Blickrichtung des Nutzers wird bei Eingabe des Nutzers auf eine einstellbare Entfernung geprüft ob ein Objekt das Handler Skript besitzt. Ist dies der Fall kann das zugeordnete Event für die Objektmanipulation ausgelöst werden. Der Vorteil von Selektion durch Raycast ist die Unabhängigkeit von spezifischen Eingabegeräten, welcher gerade bei den Hand-unabhängigen Ansätzen zur Geltung kommt, da diese sehr unterschiedliche Merkmale aufweisen können. Dafür ist die Selektion bei verdeckten Objekten oder kleinen Objekten mit wenig Distanz zwischen ihnen schwierig. Für diese Situationen könnte eine Hardware spezifischere Lösung bessere Ergebnisse liefern. Der Mehraufwand ein Dialogsystem für die Auswahl von Interaktionsobjekten zu entwerfen oder die Objekte mit konkreten Namen zu versehen, könnte die Nutzererfahrung der Sprachsteuerung verbessern.

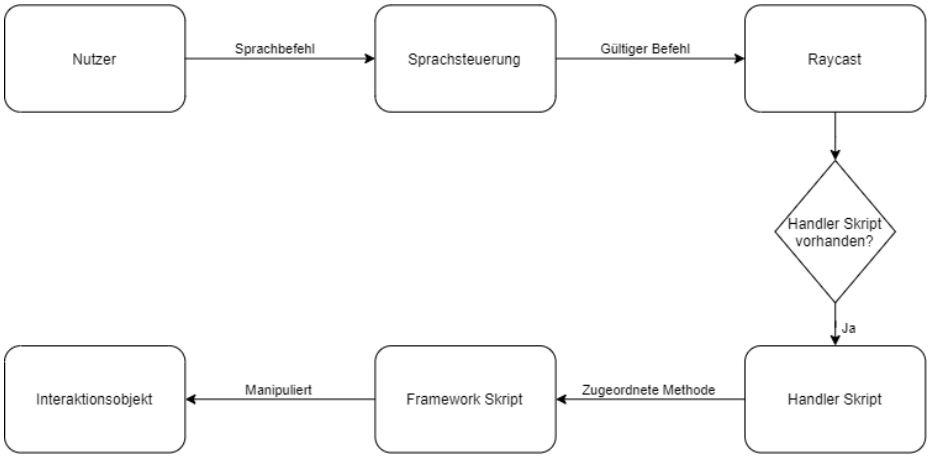


Abb. 3. Ablauf der Sprachsteuerung mit Raycast und Framework

2.3 Fazit und Ausblick

Im Rahmen dieses Projekts ist es erfolgreich gelungen die Plattform in das Projekt des Frameworks zu integrieren. Dabei wurde die Plattform um einige Features wie die Raycast Komponente für die Sprachsteuerung ergänzt, damit die bereits unterstützte Hardware in dem komplexeren Szenario und in Kombination mit den Skripten des Frameworks benutzbar ist. Das Ergebnis wurde innerhalb des Beispielszenarios der Eingangstür veranschaulicht.
Die Integration dient jedoch nur als Beispiel wie die Plattform eingebunden werden kann. Durch den Fokus auf möglichst abstrakte Schnittstellen für die jeweiligen Hardwaretypen ist die Plattform weiterhin um neue Eingabegeräte erweiterbar und auch in andere Projekte integrierbar. So können gerade in Kombination mit dem Framework beliebige Controller oder Hand-Tracking basierte Ansätze durch das IC Controller Grab Skript eingebunden werden. Auch Hand-unabhängige Ansätze lassen sich durch das generische Handler Skript der Plattform leicht integrieren. Dabei werden wie im Beispiel der Eingangstür zu sehen nur die Komponenten der Plattform und des Frameworks benötigt, um ein Objekt interaktive zu gestalten. Ein Transfer auf andere Unity Projekte ist demnach unabhängig von anderen Komponenten möglich.
In der Theorie lassen sich durch den Aufbau der Plattform grundsätzlich alle neuen Eingabegeräte einbinden. Aufgrund eines Mangels an zur Verfügung stehender Hardware konnte gerade im Bereich der Hand-unabhängigen Ansätze jedoch noch kein weiterer Test erfolgen. Dies könnte vor allem interessant werden wenn das Eingabegerät nicht Event basiert sondern kontinuierliche Eingaben für das Objekt sendet, die unabhängig von der Position des Nutzers sind. Des Weiteren erfolgt der Prozess die Software der konkreten Eingabegeräte einzubinden und entsprechende Referenzen bei den Skripten der Plattform zu setzen fast vollständig manuell. Eine dynamischere Unterstützung des Nutzers bei der Initialisierung neuer Eingabegeräte könnte erstrebenswert sein.
In diesem Projekt erfolgt keine Wertung welches Eingabegerät am Besten oder am Natürlichsten für den Nutzer ist. Eine solche Wertung ist sehr vom Kontext und Ziel der konkreten Anwendung abhängig. Die Plattform an sich soll den Anwender die Integration von Eingabegeräten zur Interaktion mit Objekten in einer virtuellen Umgebung erleichtern. Eine Evaluation der eingebauten Hardware im Anwendungskontext ist nach erfolgter Integration dann vom Anwender durchzuführen.

3 Evaluation des Frameworks

Im Grundprojekt wurde ein Framework für Interaktionen in VR entworfen, aufgebaut und auf seine Funktionen getestet. Im vorhergehenden Abschnitt wurde gezeigt, dass es möglich ist das Framework mit anderen Controllern zu verwenden. In diesem Abschnitt wird die Verwendung des Frameworks zur Erstellung und Nutzung von Szenarien bewertet. Die Auswertung der Entwicklerseite bezieht sich dabei auf das Erweitern der vorhandenen Interaktionen und die Verknüpfung von singulären Interaktionen zu komplexen Interaktionen. Die Seite der Virtual Reality Nutzer beschäftigt sich mit dem Ease of use der Anwendung, dem Transfer zwischen den Welten und der Orthogonalität der Interaktionen.

3.1 Erweiterung des Frameworks

Das im Grundprojekt erstellte Framework ist bereits mit einigen Standard Interaktionen ausgestattet, die für unterschiedliche Situationen konfiguriert werden können. Andere Elemente können durch die Events der Interaktionsobjekte über deren Zustandsänderungen benachrichtigt werden. Dieses erlaubt es die Umgebung auf Interaktionen reagieren zu lassen und diese miteinander zu verknüpfen. Für das Szenario wurde das Framework um die folgenden Interaktionen erweitert.

IOJHebel wird von ICRotation abgeleitet. Diese Klasse soll Interaktionen mit Objekten wie Türen und Hebeln erlauben. Dazu soll nach dem Greifen die Positionsänderung des Controllers in Relation zur Rotationsachse des Objekts auf das Objekt übertragen werden. Auch soll es die Möglichkeit geben, dass das Interaktionsobjekt wieder in seinen Ausgangszustand zurückkehrt nachdem es losgelassen wurde.
Durch das Erben von der Klasse ICRotation sind folgende Elemente vorhanden: In den Parametern lassen sich die Freiheiten in beide Rotationsrichtungen angeben und die Punkte, an denen die Events für Offen/Nicht-Offen und Geschlossen/Nicht-Geschlossen ausgelöst werden. Das Auslösen dieser Events wird bei jedem Setzen des Rotationswertes geprüft. Zusätzlich sind die Events für Greifen/Loslassen vorhanden, da ICRotation diese erbt.
Was ergänzt werden muss ist die Grab Methode zur Initialisierung der Winkelberechnung und die Release Methode, um je nach Einstellung das Objekt wieder Inactive zu setzen oder Running, damit es in die Ausgangsstellung zurückkehrt. Ebenso muss die Update Methode implementiert werden, um die Winkelberechung kontinuierlich während des Greifens auszuführen und das Zurückkehren in die Ausgangsposition im Running Zustand einzuleiten.
Die Methode OBJNameConventionIsFullfild() muss über das Interface OBJWithNameConvention implementiert werden und hilft die Namenskonvention der Gameobjekte zu überprüfen. Die letzte zu implementierende Methode ist CalculatePositionAndRotation() sie wird von ICRotation aufgerufen wenn ein neuer Rotationswert gesetzt wird und aktualisiert die Position und Rotation des Objekts in der Szene.

Die entstandene Klasse ist dabei klein und Verständlich geblieben und hat eine reduzierte Chance Fehler in das Projekt zu bringen.

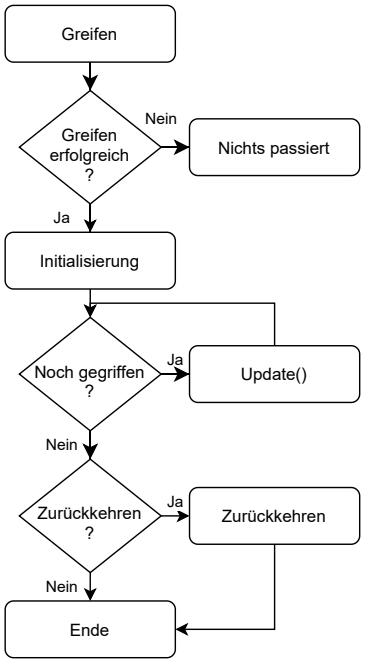


Abb. 4. Darstellung des Integrationsprozesses von IOJHebel

IOJKnob wurde auch von ICRotation abgeleitet. Da von der selben Klasse wie bei IOJHebel geerbt wird ist der Inhalt auch sehr ähnlich. Die großen Unterschiede liegen hier in der Berechnung der Rotationsänderung, die direkt an die Vorwärtsachse des greifenden Controllers gekoppelt ist.
Die Implementierung dieser Klasse war damit vergleichbar einfach wie bei der IOJHebel Klasse zu realisieren.

IORolltor erbt direkt von IOJGrabable. Das Ergebnis soll ein Rolltor sein, das in der Höhe verschoben werden kann. Das Objekt soll auch Auskunft über seinen aktuellen Status geben und Events auslösen, wenn es die Schwelle für ganz Offen oder Zu in eine der Richtungen überschreitet. Im gegriffenem zustand soll das Rolltor sich gleichzeitig mit dem Controller bewegen. Bei dieser Änderung müssen entsprechend auch die Events ausgelöst werden. Zusätzlich wird danach noch der grafische Aspekt angepasst, indem die zum Rolltor gehörenden Objekte Verschoben und skaliert werden, um den Öffnungsgrad wiederzugeben.

3.2 Aufbau Komplexer Interaktionen

Bei dem Szenario handelt es sich um eine Schulung für Techniker von Windenergieanlagen. Das Szenario besteht insgesamt aus 5 unabhängigen Schauplätzen mit über 70 interaktiven Elementen, die sich in 7 Abschnitte teilen. Aus diesen Abschnitten soll auf drei der Komplexen Interaktionen und den Aufbau des Tutorials genauer eingegangen werden.

Die Eingangstür bietet Zugang zum Turm. Sie ist durch einen großen Hebel verriegelt, der von beiden Seiten aus bedient werden kann. Der Ablauf für das Szenario beginnt mit dem Entriegeln der Tür gefolgt vom Öffnen der Tür. Auf der anderen Seite gilt es diese wieder zu schließen und zu Verriegeln. Ein besonderer Punkt an dieser Stelle ist, dass durch das Öffnen und Schließen der Tür die Sound Kulisse zwischen draußen und drinnen umgeschaltet wird.

Abb. 5. Darstellung des Greifens der Tür

Das Szenario besteht aus zwei direkt interaktiven Elementen. Der Hebel an der Tür und die Tür selber. Der Zustandsraum besteht aus dem Griff und der Tür, jeweils mit geschlossen und nicht geschlossen als Zustände. Eine Positionskomponente für den VR Nutzer ist zusätzlich notwendig, um die Sound Kulisse auf drinnen oder draußen schalten zu können.
Für die beiden aktiven Interaktionsobjekte können Hebel Objekte Verwendet werden. Sie erlauben die Rotation um eine Achse wobei der Rotationsbereich eingestellt werden kann. Ebenso lässt sich der Zustand abfragen und beim Zustandswechsel werden Events ausgelöst. Die Skripte müssen dafür lediglich auf die Objeckte angewendet und konfiguriert werden. Für die Position muss abzufragen sein auf welcher Seite der Tür der Nutzer sich befindet, dieses kann direkt durch Kollisionsabfragen gelöst werden.

Vom Code her besteht das Szenario aus zwei Teilen. Der Erste ist für das Entriegeln und Verriegeln der Tür zuständig. Um die Zustände zu realisieren, Abbildung: 6, kann auf die Events zurückgegriffen werden. Es gibt dafür drei Dinge Übergänge die ausgelöst werden können.

	Tür offen	Tür zu
Griff offen	Tür entriegelt	Tür entriegelt
Griff zu	Tür entriegelt	Tür verriegelt

Abb. 6. Darstellung der Zustände von der Tür

Der Übergang zu Entriegelt wird durch ein nicht mehr geschlossen Event vom Griff ausgelöst und setzt den Status der Tür von Disabeld zu Inactive und sorgt danach dafür, dass die Tür dann mit dem Controller gegriffen wird, der den Griff greift, damit diese auch bewegt werden kann.
Der Übergang zu Verriegelt kann durch das Event für das Schließen der Tür oder des Hebels ausgelöst werden, wenn beide Objekte zu diesem Zeitpunkt als geschlossen gelten. Dabei wird die Tür losgelassen, vollständig geschlossen und auf Disabeld gesetzt.
Der dritte Teil wird ausgelöst wenn die Tür entriegelt ist und der Griff gegriffen wird. Dann wird auch wieder die Tür mit dem selbem Controller gegriffen, damit diese auch mit bewegt werden kann.
Das Umschalten des Sounds wird immer über das Event für das Schließen oder nicht mehr Geschlossen sein der Tür geprüft. Beim Schließen wird die Position des Nutzers abgefragt und entsprechend auf drinnen oder draußen umgeschaltet. Beim Öffnen wird immer auf draußen Sound umgeschaltet.
Die Interaktion der Tür ließ sich Mithilfe der Events ohne Anpassung des vorhandenen Codes implementieren. Die Implementierung dazu konnte sich vollständig auf die minimalen Zustandsübergänge beschränken was den Code klein und Übersichtlich hält.

Der Sicherungsschlitten ist das zentrale Objekt, wenn es um die Benutzung von Leitern geht und ist ein wichtiger Bestandteil der persönlichen Sicherung. Immer wenn der Nutzer Leitern nutzen will, muss dieser in ein Sicherungsseil

eingehängt werden, um das persönliche Abstürzen zu verhindern. Deswegen steht der Sicherungsschlitten bei der Interaktion mit Leitern im Mittelpunkt, die an fünf unterschiedlichen stellen vorkommt. Wenn es darum geht die Ebene zu wechseln muss dieser an dem Sicherungsseil eingehängt sein und nach oben oder unten geschoben werden, um einen wechsel der Etage auszulösen.

Abb. 7. Darstellung des Greifen des Sicherungsschlittens

Die Komponenten der Interaktion mit dem Sicherungsseil besehen aus dem Sicherungsseil, welches entlang der Leiter über alle verbundenen Etagen geht, dem Sicherungsschlitten und einer Positionskomponente, um das aktuelle Stockwerk und die angrenzenden zu ermitteln.
Das Sicherungsseil dient als Auslöser, es ist ein IOJGrabable Objekt, das Events für das Greifen und Loslassen hat, um den Sicherungsschlitten an diesem erscheinen zu lassen, wenn dieser noch nicht eingehakt ist. In allen anderen Fällen hat es keine Funktion. Der Schlitten selber ist auch ein IOJGrabable, damit er vom Nutzer selektiert werden und er Grab und Release Events senden kann.
Der Sicherungsschlitten, wenn nicht gegriffen, gleitet auf dem Sicherungsseil entlang, wobei er immer zum Nutzer zeigt. Dabei wird seine Position und Rotation von dem kontrollierenden Skript gesetzt. Wenn der Sicherungsschlitten gegriffen ist, nimmt er immer die Position am dichtesten zum Controller auf dem Seil ein. Wenn der Controller während des Greifens zu weit vom Seil entfernt wird, wird der Sicherungsschlitten vom Seil entfernt. Wird dieser über eine bestimmte Distanz nach oben oder unten geschoben, wird versucht die Teleportation ins nächste Stockwerk, darüber oder drunter, auszulösen. Dieses ist dadurch bedingt, dass der Nutzer in dem von der Positionskomponente beschriebenem Feld steht, damit die Positionsdaten für die Teleportation ausgelesen werden können. Beim Auslösen der Teleportation wird der Sicherungsschlitten automatisch losgelassen und mit in die nächste Etage geschoben.
In dieser Komponente wurde mehr als nur Code für Zustandsübergänge benötigt. Es war auch aktiver Code nötig für die Positions- und Rotations-Updates des

Sicherungsschlitten wie auch für die Überprüfung der Konditionen der Deaktivierung des Schlittens und des Auslösen der Teleportation. Durch das Auslagern der Events in die IOJGrabable Objekte hat aber auch dieser Code saubere Zustandsübergänge, die einfach zu trennen sind.

Der Aufzug dient zum Überwinden des größten Teils des Anstiegs. Dieser unterliegt wie auch die Leitern gewissen Sicherheitsanforderungen und Prozeduren, die zur Nutzung durchgeführt werden müssen. Dieses ist in reduzierter Form folgend abgebildet. Um den Aufzug zu erreichen, muss als erstes ein Tor in einer Absperrung durchquert werden. Der zweite Schritt ist das Öffnen des Rolltors, gefolgt von dem Aushaken der Sicherungskette, um einsteigen zu können. Nach dem Einsteigen muss diese wieder eingehängt und das Rolltor wieder geschlossen werden bevor der Aufzug über einen Drehknopf in die nächste Etage gefahren werden kann.

Abb. 8. Darstellung des Aufzugs

Die Komponenten sind ein Tor das selbst schließend ist, das Rolltor sowie die Sicherungskette, welche sich an einem einem Ende greifen lässt. Wenn die Kette in der Nähe des Einsteckplatzes losgelassen wird, springt sie in eine vorgegebene Position zurück. Der letzte interaktive Gegenstand ist der Drehknopf, der automatisch in seine Position zurückkehrt und sich in beide Richtungen drehen lässt.
Das Tor lässt sich durch ein IOJHebel Skript erstellen, welches auch die Optionen für die Rückkehr in die Ausgangsposition bietet. Das Rolltor lässt sich als IOJRolltor realisieren. Das Ende der Sicherungskette ist ein IOJPickable Objekt,

das frei bewegt werden kann. Der Drehknopf verhält sich wie ein IOJKnob.
Das Einrasten des Endes der Sicherungskette geschieht über das Event für das Loslassen. Dabei wird der Abstand zum Steckplatz überprüft und wenn dieser klein genug ist, das Einrasten ausgelöst. Dieses wird dann als Zustand gespeichert bis das Objekt wieder aufgenommen wird.
Der Notwendige Code für die Bewegung wird von den Events des Drehknopfes ausgelöst und überprüft dann, ob das Rolltor geschlossen ist sowie auch das Ende der Sicherungskette im Sockel steckt. Wenn beides zutrifft wird eine Teleportation in die andere Etage ausgelöst.
Damit ist der Code für die übergreifende Interaktion in zwei Methoden als Reaktionen auf Events beschrieben und bleibt trotz der komplexen Teilinteraktionen übersichtlich.

Das Tutorial Da Kenntnisse zum Umgang mit den Eingabegeräten der Anwendung nicht vorausgesetzt werden können, bietet es sich an den Nutzern die Option zu bieten ein Tutorial zu absolvieren. Das in diesem Projekt entwickelte Tutorial lässt sich im Hauptmenü einschalten und ist in den Start des Szenarios integriert. So können die Nutzer der Anwendung sich mit den Controllern direkt in der Anwendungsumgebung vertraut machen. Dabei werden sowohl Highlighting Skripte benutzt, um das jeweils nächste Interaktionsobjekt in der korrekten Handlungsreihenfolge hervorzuheben, als auch Pfeile eingeblendet, welche die Bewegungsrichtung bestimmter Interaktionsobjekte visuell demonstrieren. Außerdem wird der Trigger Knopf der Controller mit einem roten Material versehen, welches sich deutlich von der Farbe des restlichen Controllers abhebt.

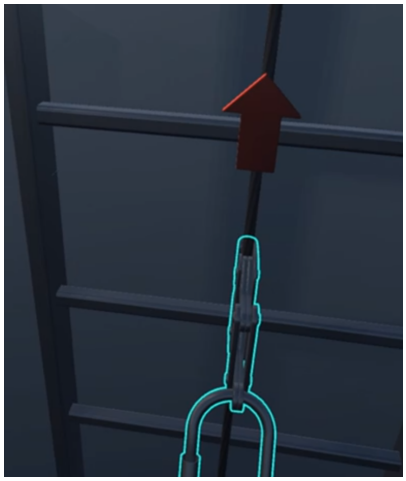


Abb. 9. Tutorial am Sicherungsschlitten

Die korrekte Handlungsreihenfolge und die Logik wann welches unterstützende Merkmal aktiviert werden soll findet sich in dem Tutorial Skript. Dieses abonniert die Events der Interaktionsobjekte des Tutorials, die bei einem erfolgreichen Durchlauf auftreten und hält Referenzen auf alle Highlighting Skripte und Pfeile des Tutorials. Tritt ein abonniertes Event auf, wird eine vorher definierte Methode aufgerufen. In diesen Methoden werden dann die Highlighting Skripte und Pfeile des vorherigen Schrittes deaktiviert und anschließend die Skripte und Pfeile des nächsten Schrittes aktiviert. Damit die Schritte des Tutorials nicht in beliebiger Reihenfolge erfolgen können, wird eine Counter Variable genutzt, die nach jedem korrekten Schritt hochgezählt wird. Nur wenn der Counter dem in der Methode angegebenen Wert entspricht, wird diese nach Auslösung des Events durch die Interaktion des Nutzers mit dem entsprechenden Objekt ausgeführt.
Das Tutorial beschränkt sich auf die ersten Interaktionsschritte des Szenarios und zeigt anhand einiger Interaktionsobjekte, hauptsächlich der Eingangstür und dem Sicherheitsschlitten, wie der Nutzer mit den Controllern die Objekte der virtuellen Umgebung beeinflussen kann. Des weiteren wird durch die vordefinierte Reihenfolge der Interaktionsschritte eine Orientierungshilfe für Erstanwender gegeben. Bei wiederholten Durchläufen kann das Tutorial im Hauptmenü abgeschaltet werden, um auch die ersten Schritte ohne Hilfestellung trainieren zu können.

3.3 Auswertung

Die Auswertung ist zweigeteilt, sie gliedert sich in die Bewertung von Einrichtung und Erstellung von Szenarien und die Evaluation durch Testdurchläufe von Nutzern.

Einrichtung und Erstellung von Szenarien Das Einbinden von weiteren Interaktionen hat sich als einfach herausgestellt. Mit der vorgegebenen Struktur waren die zu implementierenden Teile und Fehler einfach zu beheben. Auf Grund der Singularität der Interaktionen sind dabei keine Konflikte aufgetreten.
Interaktionen nicht singulärer Art, die komplexes Verhalten erfordern, lassen sich nur begrenzt ohne weiteren Code abbilden. Wenn dieser Code aktive, verschiedene Interaktionen zusammenbringen soll hat sich die vorhandene Event Struktur als hilfreich und praktikabel erwiesen. Mithilfe der Events war es einfach Zustandsänderungen auszulösen und alle Code Pfade zu überblicken, da diese direkt an Interaktionen hingen. Aufgrund der gemeinsamen Basis aller Interaktionen waren die wichtigen Features alle vollständig und orthogonal vorhanden und ließen sich einfach nutzen.
Das Fehlerhandling hat sich auch beim Zusammenstellen mehrerer Interaktionen nicht als Problem herausgestellt. Durch die einfache Trennung ist es möglich Übergänge mit Preconditions abzusichern und Fehler ließen sich einfach finden. Das häufigste Problem ist das Fehlen von Verlinkungen oder falsche Parameter im Inspektor von Unity. Diese Fehler sind aber durch die automatische Validierung der konfigurierbaren Felder schnell zu beheben.

Testdurchläufe von Nutzern Insgesamt wurde das Framework von drei Gruppen an Nutzern getestet, welche unterschiedliche Vorbedingungen mitbringen. Die Gruppen und die Ergebnisse der jeweiligen Durchläufe werden im Folgenden genauer beschrieben.
Die erste Gruppe, 5 Personen, konnte direkt nach der Nutzung befragt werden und war mit dem verwendetem Szenario bekannt. Diese Gruppe war der Befragung nach auch bereits mit Computerspielen und zum Teil mit Virtual Reality vertraut. Insgesamt kamen sie gut mit dem Szenario zurecht. Interaktionen, die eine direkte Translation aus der Realen Welt waren, hatten dabei keine Erklärung benötigt. Nachdem sich mit der Funktionsweise der Virtual Reality an der Tür vertraut gemacht wurde, wussten die Teilnehmer aufgrund des übertragbaren Verhaltens die folgenden Interaktionen instinktiv zu bedienen. Die Eingangstür, die auch die Eröffnung für das Szenario ist, gibt aufgrund ihres Aussehens die Funktion der Teile, Hebel und Tür, vor. Ebenso ist die Benutzung dieser von der Realität vertraut. An dieser Stelle klappt die Übertragung zwischen den Welten sehr gut.
Interaktionen die kein Gegenstück zur realen Welt haben mussten einmalig erklärt werden, da ihre Funktion nicht aus den Gegebenheiten erkannt werden konnten. Das Besteigen der Leiter ist hierfür ein Beispiel, an diesem musste der Sicherungsschlitten eingehängt und dann nach oben oder unten verschoben werden, um ein Wechsel des Stockwerks auszulösen. Nach einer einmaligen Erklärung und Anwendung war diese Abstraktion, für das wechseln der Etage, kein Problem. Da sich die Einzelinteraktionen Oorthogonal zu den anderen bedienen ließen, musste nur das übergreifende Konzept erklärt werden. Im weiteren Verlauf wurden Situationen zur Wiederholung der Technik ohne Probleme erkannt.
Die zweite Gruppe bestand aus dem Szenario fremden Personen. Mit der Bedienung der Interaktionen hatte auch diese keine Probleme. Aufgrund der Gegebenheit, dass ihnen das Szenario unbekannt war, haben diese aber eine Anleitung für die übergreifenden Interaktionen benötigt, um die nächste Interaktion in Reihe zu erkennen wie bei der Leiter oder dem Aufzug.
Die dritte Gruppe bestand aus unbeobachteten Rückmeldungen über das Tutorial zur Einweisung in das Interaktionsschema. Dieses bestand aus dem Hervorheben der Taste am Controller und der zu greifenden Interaktion. Bei der Leiter gab es zusätzlich den Hinweis in der Form eines Pfeils nach oben neben dem Sicherungsschlitten nachdem dieser eingehängt war. Die Rückmeldungen ergaben, dass das Tutorial gut in das Interaktionsschema einweist und die abstrakte Interaktion des Leiterkletterns erklären kann.

Fazit In dieser Ausarbeitung wurde gezeigt, dass das im Grundprojekt entworfene Framework für Interaktionen den Anforderungen entspricht.
Das Einbinden von neuen Interaktionen war ohne Problem möglich und wurde von den vorhandenen Templates gut unterstützt. So konnten mit wenigen Schritten neue Interaktionen implementiert werden. Durch die Eigenschaft, das einzelne Interaktion unabhängig voneinander sind, ließen diese sich mit geringem Aufwand konfigurieren und Fehler schnell finden. In der aktuellen Version ist das Konfiguri-

eren mit Expertenwissen über den Aufbau der Interaktionen verbunden, da diese direkt über Zahlenwerte ohne visuelle Rückmeldung konfiguriert werden. Eine Visuelle Darstellung der Parameter und eine interaktive grafische Konfiguration könnten an dieser Stelle das Framework für Nicht-Experten zugänglicher machen.
Das Erstellen der komplexen Interaktionen wurde ebenfalls gut durch die vorhandene Eventstruktur unterstützt. Da die neuen Interaktionen Erweiterungen anderer Interaktionen sind, waren diese bereits in die Eventstruktur eingebunden. Die entsprechende Vorarbeit im Grundprojekt war damit in diesem Punkt hilfreich und erfolgreich.
Die Nutzung der Interaktionen in der Virtual Reality hat sich positiv bemerkbar gemacht. Die Übertragung zwischen Realität und Virtual Reality war bei den ähnlichen Interaktionen ohne Hindernisse oder Reality Gap möglich. Interaktionen mit symbolischem Abstraktionsgrad mussten in ihrer übergreifenden Funktion einmalig erklärt werden, aufgrund der herrschenden Orthogonalität der Interaktionen war die Nutzung selber in allen Fällen gelungen. Das Tutorial hat an dieser Stelle gezeigt, das diese Stellen auch ohne externe Intervention gelöst werden können.

4 Zusammenfassung und Ausblick

In diesem Projekt wurde die Möglichkeit der Verknüpfung der im Grundprojekt erstellten Komponenten unter Beweis gestellt. Die bereits von der Plattform eingebundene Hardware ließ sich in das Framework ohne Probleme integrieren. Dabei wurden einige Verbesserungen wie die Erweiterung der Sprachsteuerung um eine Raycast Komponente zur Selektion in dem komplexeren Szenario vorgenommen. Bei dem Design der Schnittstelle von Hardware Plattform und Framework wurde darauf geachtet ein hohes Level an Abstraktion beizubehalten, um weitere Eingabegeräte möglichst leicht einzubinden und die Toolchain in anderen Kontexten nutzen zu können.
Die Verwendung des Frameworks für die Erstellung von Interaktionen und die Nutzung in Schulungsszenarien wurde erfolgreich getestet. Das Framework ließ sich problemlos um neue Interaktionen erweitern und die Eventstruktur hat es auf einfachem Wege erlaubt Interaktionen zu komplexeren zu verknüpfen. Die Rückmeldung der Testpersonen bezüglich der Erlernbarkeit der Interaktionen, die vom Framework bereitgestellt werden, sowie zur Vermittlung der anwendungsspezifischen Lerninhalte durch die virtuelle Lernumgebung war positiv.
Weiterführend könnte die Einbindung von neuartigen Eingabegeräten und Controllinterfaces in die Toolchain getestet werden. Besonders im Bereich der Handunabhängigen Ansätze existieren Eingabegeräte, die vollkommen unterschiedliche Eigenschaften zu den bereits Implementierten aufweisen. Auch grafische Interfaces zur Konfiguration und Verknüpfung von Interaktionen sowie bei der Einbindung von neuer Hardware könnten eine sinnvolle Erweiterung der Toolchain sein. Des Weiteren könnte der Einsatz der Toolchain in einem anderen Kontext untersucht werden, um die Übertragbarkeit der Toolchain zu überprüfen.