

SpeechVR – Eine sprachgesteuerte Applikation für OpenGL-Programme

Michael Haller¹, Markus Pargfrieder²

¹Fachhochschule Hagenberg, Medientechnik und –design, Hagenberg, AUSTRIA,
haller@fhs-hagenberg.ac.at

²Natural Communication GmbH, Multimedia Design & Production, Linz, AUSTRIA,
m.pargfrieder@netural.at

Zusammenfassung: *Spracherkennung und Sprachsteuerung sind Themen, die besonders in letzter Zeit das Interesse der IT-Branche stark geweckt haben. Der Einsatz dieser Technologie für Spiele bzw. VR-Umgebungen ist bislang noch nicht intensiver behandelt und erforscht worden. Deshalb wurde an der Fachhochschule Hagenberg im Rahmen einer Diplomarbeit der Prototyp **SpeechVR** entwickelt, der die Sinnhaftigkeit der Sprache als Eingabemedium für eine computergenerierte Welt untersuchen sollte. Dabei sollte in SpeechVR vor allem die Frage geklärt werden, ob durch diese Art der Benutzersteuerung ein höherer Grad an Intuitivität erreicht werden kann und ob sich die Sprache für die Navigation und Interaktion in virtuellen Welten bzw. für Spiele eignet.*

1. Einleitung

Auch wenn die VR-Technologie nichts Neues ist, mangelt es derzeit immer noch an einfach zu bedienenden und intuitiven Eingabegeräten. Und allzu oft hat man den Eindruck, dass gerade bei den Eingabegeräten nichts weiterentwickelt wird – insbesondere dann, wenn es sich um kompliziertere Eingaben in einer VR-Umgebung handelt. Interessant ist dabei das Phänomen, dass auf einer Seite bei der Visualisierung - gerade durch die immer besseren Graphikkarten - der Grad an Realismus immer besser wird, aber auf der anderen Seite bei der Mensch-Maschine-Interaktion wenig Fortschritte zu beobachten sind. In den letzten Jahren ist besonders der Bereich der Spracheingabe und der Spracherkennung immer mehr auf das Interesse der Anwender gestoßen, und es würde nicht verwundern, wenn man sich nun eine

Kombination der beiden Technologien vorstellt. Die derzeitigen 3D-Eingabegeräte reichen von der traditionellen Tastatur über die 3D-Maus (oft auch SpaceMouse genannt), bis hin zum 3D-Joystick. Eine Vielfalt an unterschiedlichen Geräten ist bereits gegeben, aber keines dieser Eingabegeräte erfüllt den Zweck, dass sie einfach und intuitiv zu bedienen sind – geschweige denn, dass sie komplexere Eingaben zulassen. Das größte Problem, welches diese Eingabegeräte meist aufweisen ist, dass sie nur sehr eingeschränkt benutzbar sind und nicht sehr intuitiv anzuwenden sind. Der Datenhandschuh bietet sicherlich eine gelungene Alternative zu den bisher genannten Eingabemedien. Er kann völlig intuitiv verwendet werden, da er die Eingabeinformation direkt von der Hand bezieht, und man daher nie den Blick auf die Tastatur oder auf die Maus richten muss. Eines der größten Nachteile des Datenhandschuhs ist sicherlich der Preis. Aber auch hier gibt es bereits die Firma Essential Reality [1], die eine Billigalternative (Kostenpunkt ca. 129\$) zu den bisherigen Systemen anbietet.

Der Typ des Eingabegerätes ist natürlich eng mit dem Art der Applikation verknüpft. Wenn es sich beim Computerspiel oder bei der VR-Applikation um eine Rennauto- oder Fahrsimulation handelt, dann ist es klar, dass man für das Eingabegerät nichts Besseres zur Verfügung stellen kann, als ein Cockpit, ein Lenkrad und die Fußpedale. Was nimmt man aber bei VR-Applikationen, welche keine so intuitiven Eingabegeräte zulassen? Viele Dinge könnten viel einfacher mit der Sprache erledigt werden. Kombiniert mit den bisher bekannten Eingabegeräten wird die Interaktion in einer VR-Welt sicherlich erleichtert. Im nun folgenden Beitrag wird die Applikation SpeechVR vorgestellt. Dabei handelt es sich um einen Prototypen, bei dem der Versuch gestartet wurde, in einer OpenGL-Szene mittels Spracherkennung eine bessere Interaktion bzw.

Navigation zu ermöglichen. SpeechVR wurde auf Basis der SAPI Speech SDK 5.0 (vgl. [4]) implementiert und läuft auf einem PC.

2. SpeechVR

SpeechVR wurde im Rahmen einer Diplomarbeit an der Fachhochschule Hagenberg entwickelt [2]. Die Ziele bei dieser Arbeit lagen vor allem darin, zu sehen, ob Spracheingabe bei VR- oder Spiele-Applikationen Sinn macht und wenn ja, wie gut sich die gängigen Systeme mit einer Graphik-API verknüpfen lassen. Die zentrale Frage, ob sich die Spracheingabe als Eingabemedium eignet, stand dabei besonders im Mittelpunkt. Der Prototyp SpeechVR, basierend auf OpenGL und auf SpeechSDK von Microsoft [4], diente dabei als Testapplikation für die Klärung dieser Frage.

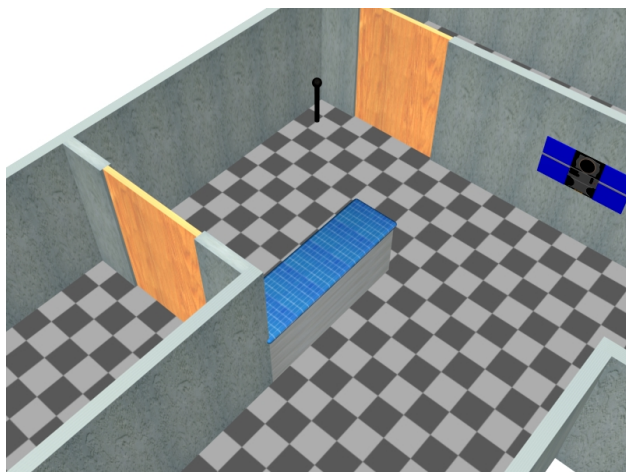


Abbildung 1: In SpeechVR kann der Benutzer mittels Spracherkennung in einer Wohnung navigieren und interagieren.

2.1. Navigation

Die Fortbewegungsmöglichkeiten sollten jenen der realen Welt so weit wie möglich entsprechen. Zu den Basisbewegungen sollten daher auf jeden Fall die Vor- und Rückwärtsbewegung sowie die Links- und Rechtsdrehung gehören. Darüber hinaus können natürlich weitere Möglichkeiten die Benutzbarkeit von SpeechVR wesentlich verbessern. Eine Seitwärtsbewegung kann dem Benutzer beispielsweise die Navigation wesentlich erleichtern. Aber auch die Kontrolle über die Geschwindigkeit, mit welcher er sich bewegen kann, bringt große Vorteile.

Wenn der Benutzer die Möglichkeit hat, seine Fortbewegung zu beschleunigen beziehungsweise zu verlangsamen, so wird eine genauere Bewegung ebenso möglich wie eine sehr schnelle. Besonders der Aspekt der

Fortbewegungsgeschwindigkeit, aber auch aller anderen Aspekte, hängen stark von der gegebenen VR-Welt ab. In einem virtuellen Universum oder im freien Gelände sollte mit einer anderen Geschwindigkeit navigiert werden können, als durch eine Wohnung (vgl. Abbildung 1).

2.2 Interaktion

Im Gegensatz zur Navigation, welche sich allgemein formulieren lässt, ist es sehr schwierig, die Möglichkeiten der Sprachsteuerung in VR-Welten in Bezug auf die Interaktion genau zu beschreiben, da diese sehr stark von den Anforderungen abhängen.



Abbildung 2: Die Interaktion und Navigation in SpeechVR erfolgt über ein Headset.

Gerade die Spracherkennung kann für die Interaktion mit anderen Objekten in einer virtuellen Welt große Vorteile bringen, da komplexe Anweisungen sehr einfach durchgeführt werden können. Bisherige VR-Systeme, bei denen ein Trackingsystem bzw. ein virtueller Joystick eingesetzt werden, weisen einen großen Schwachpunkt auf: Durch die Ungenauigkeit des Trackers zittert die virtuelle Interaktionshand, und der Benutzer ist schließlich nicht damit beschäftigt wie er die **Aufgaben** in einer virtuellen Welt löst, sondern wie er mit der **schlechten Hardware** am besten zurecht kommt und es dabei dennoch schafft, mit zitternder Hand einen virtuellen Gegenstand zu heben. Allzu oft sind die Interaktionen, die wir im alltäglichen Leben ausüben, sehr komplex und es ist somit sehr schwer, diese in einer VR-Welt abzubilden. Wollen wir beispielsweise ein Ventil schließen, welches sich in einer Höhe von drei Meter befindet, dann holen wir uns meist eine Leiter, stellen uns drauf und bedienen das Ventil. Solche für uns alltäglichen Abläufe sind aber in VR nur sehr schwer umzusetzen. Mittels Spracherkennung wäre es nun viel einfacher, solche Szenarien virtuell abzubilden, denn der Benutzer

könnte in unserem Beispiel durch ein einfaches Kommando das entfernte Ventil schließen.

Eine weitere denkbare Anwendung für Spracherkennung und VR hätten wir z.B. in industriellen Lernprogrammen gegeben, in welchen der Benutzer die Möglichkeit hätte, technische Detailinformationen zu diversen Objekten oder Vorgängen abzufragen. Ideal wäre es natürlich, wenn man sich mit *Avataren* unterhalten könnte oder von Avataren Informationen bekommen könnte (vgl. STEVE [5]). Eine derartige Applikation ist allerdings, sollte sie wirklich intuitiv wirken, sehr aufwendig zu gestalten, da die Möglichkeiten ein und dasselbe auf verschiedene Arten zu formulieren derartig weitläufig sind, dass eine komplette Abdeckung dieser nur schwer erreicht werden kann. „Wie spät ist es, bitte?“, „Haben Sie die Zeit für mich?“, „Welche Uhrzeit haben wir, bitte?“ oder „Könnten Sie mir bitte die Zeit sagen.“ sind sehr unterschiedliche Formulierungen, welche aber dieselbe Systemreaktion erfordern würden.

2.3 Sprachgesteuerte Interaktion und Navigation in SpeechVR

Der Benutzer kann in SpeechVR relativ einfach navigieren. Dies erfolgt über die Kommandos „Go forward“, „Go backward“. Zudem kann er über gezielte Befehle die Geschwindigkeit regulieren. Wenn nun der Benutzer sich in einer Bewegungsphase befindet, muss er wiederum ein „Stopp“-Kommando geben, damit die Bewegung aufhört und er zum Stillstand kommt. Leider zeigt sich hier eine kleine Verzögerung, die sich etwas störend auf die Navigation auswirkt.

Natürlich kann der Benutzer sich auch nach rechts bzw. nach links drehen (entweder erfolgt eine kleine Rechts- bzw. Linksdrehung oder eine komplette 90°-Drehung). Die in Tabelle 1 angeführten Befehle können natürlich auch miteinander kombiniert werden. So kann sich der Benutzer z.B. während einer Vorwärtsbewegung mit einem weiteren Befehl eine Rechtskurve einschlagen.

Sprachbefehle unterscheiden sich wesentlich von normalen Tastatur- oder Joystick-Eingaben. Ein Sprachbefehl zur Navigation bleibt nämlich nur solange gültig bis er aufgehoben wird, wohingegen die Befehle, welche über die Tastatur erfolgen, sofort ihre Gültigkeit verlieren, wenn der Benutzer die entsprechende Taste nicht mehr drückt

Kommando	Bedeutung
<i>Move/Go Forward/Backward</i>	Vorwärts- bzw. Rückwärtsbewegung
<i>Turn Right/Left</i>	Drehung
<i>Stop</i>	Stopp die Bewegung

<i>Hard Left/Right</i>	90° Drehung
<i>Speed Up / Slow Down</i>	Geschwindigkeit verstellen
<i>(please) open/close (the) door</i>	Interaktion mit den Türen
<i>(please) use (the) level</i>	Interaktion mit einem Hebel
<i>(please) turn/switch (the) radio on/off / (please) play/stop (the) radio</i>	Interaktion mit dem Radio
...	...

Tabelle 1: Einige interessante Navigations- und Interaktions-Kommandos von SpeechVR.

2.4 SpeechVR in Aktion

In *SpeechVR* wird eine Szene dargestellt, die in ihrer grundsätzlichen Gestaltung einer herkömmlichen Wohnung entspricht. Die verschiedenen Räume werden durch Türen, welche zur Startzeit der Applikation allesamt geschlossen sind, voneinander getrennt. Mit Hilfe eines Sprachbefehls (vgl. Tabelle 1) wird es dem Benutzer ermöglicht, Türen zu öffnen.



Abbildung 3: In SpeechVR lassen sich alle Türen mittels Sprachkommandos öffnen bzw. schließen.

Alle Schwingtüren können mit demselben Befehl geöffnet werden, wobei die Unterscheidung, welche Tür der Benutzer nun eigentlich meint, über die Entfernung zu den einzelnen Türen getroffen wird. Wie im Abschnitt 3.3 beschrieben wird, verfügen alle Objekte, welche für eine Interaktion zur Verfügung stehen, also auch Türen, über eine Art *Aktionsradius*, mittels welchem eine Zuordnung des Befehls erfolgt. Neben den bereits erwähnten Schwingtüren befindet sich in der Szene auch eine

Schiebetür, welche allerdings nicht direkt geöffnet werden kann. Zu diesem Zwecke befindet sich an einer Seite der Tür ein Hebel, welchen man über Sprachangabe steuern kann und dadurch die Tür entsprechend öffnet oder schließt. Die Möglichkeit, Hintergrundmusik für die Szene zu schaffen, wurde über ein Radio realisiert, welches sich unter anderem ein- und ausschalten lässt. Der Benutzer kann einen Track anhalten bzw. wieder fortsetzen. Zum Abspielen stehen mehrere verschiedene Lieder zur Verfügung, zwischen welchen mit diesen Befehlen hin- und hergeschaltet werden kann. Falls das Ende der Liste von Liedern erreicht wird und der Benutzer den nächsten Song abzuspielen wünscht, so wird wieder beim ersten begonnen.



Abbildung 4: Mit der magische Puppe *Magic Puppet* kann man in SpeechVR sehr eingeschränkt kommunizieren.

In der Szene befindet sich auch eine kleine magische Puppe, die *Magic Puppet*, welche in der Lage ist, dem Benutzer in einigen Bereichen helfend unter die Arme zu greifen. Die Befehle, welche an diese Puppe gerichtet werden können, sind natürlich sehr einfach. Der Benutzer kann durch Eingabe eines Sprachbefehls die *Magic Puppet* dazu auffordern, sich vorzustellen. Weiters kann man sie bitten, die Szene zu überprüfen und danach dem Benutzer die Anzahl der geöffneten Türen anzugeben. Schlussendlich ist die Puppe auch in der Lage, alle Türen in SpeechVR auf einen Schlag zu öffnen bzw. zu schließen.

Natürlich sind auch einzelne Befehle zur Steuerung der Applikation selbst mittels Spracheingabe möglich. Durch Eingabe des Kommandos „*Game Command Help*“ wird z.B. ein Hilfe-Menü eingeblendet, welches einen Überblick über die gängigsten und wichtigsten Befehle geben soll. Durch Öffnen dieses Menüs wird die Möglichkeit zu navigieren und zu

interagieren vorübergehend gesperrt, und es wird erst dann wieder für den Benutzer aktiviert, wenn das Menü geschlossen wird.



Abbildung 5: Die Abfrage zum Beenden in SpeechVR.

3. Implementierung

SpeechVR ist eine Desktop-Anwendung, welche mit OpenGL realisiert wurde. In den nun folgenden Abschnitten wird die Spracherkennungs-Engine und die zugrundeliegende Implementierung genauer erklärt.

3.1 Die Spracherkennungs-Engine

Auf der Suche nach verschiedenen Spracherkennungs-Technologien stößt man immer wieder auf zwei sehr bekannte Namen in der Computerbranche. IBM und Microsoft sind zwar nicht die einzigen Firmen, die sich mit dieser Technologie auseinandersetzen und Produkte in diesem Bereich anbieten, aber diese beiden tun dies bereits seit den Anfangsjahren der Spracherkennung. Allerdings unterscheiden sich die Vorgehensweisen dieser beiden Anbieter doch sehr deutlich. Während IBM mit der kommerziellen Software *IBM ViaVoice* [6] ein fertiges Anwenderpaket auf den Markt bringt, stellt Microsoft zu diesem Thema ein Software Development Kit (*Microsoft Speech SDK 5.0*) als Entwicklungsumgebung für alle Programmierer im Internet zur freien Verfügung [4].

3.2 Microsoft Speech SDK und SpeechVR

Die Sprachsteuerung und -eingabe unterscheidet grundsätzlich zwei verschiedene Arten der Erkennung:

- *Dictation* und
- *Command & Control*.

Erstes meint das Diktieren von freiem Text in beliebigen Applikationen wie Texteditoren oder E-Mail-Programmen, während zweites auf einem definierten Wortschatz beruht. SpeechVR basiert auf Command & Control, wobei die möglichen Steuerungsbefehle zur Navigation und Interaktion in einem speziellen XML-File definiert werden. Erkannte Schlüsselwörter werden in weiterer Folge mit den entsprechenden Befehlen verknüpft. Ein einfaches Beispiel für die Definition eines solchen Signalwortes kann wie folgt aussehen:

```
<RULE ID="RULE_MOVEMENT" >
  <O>-please</O>
  <L>
    <P>move</P>
    <P>go</P>
  </L>
  <L PROPID="RULE_MOVEMENT">
    <P VAL="NAV_FORWARD">forward</P>
    <P VAL="NAV_BACKWARD">backward</P>
  </L>
</RULE>
```

Der Tag <P> umklammert ein Schlüsselwort, welches unbedingt vorkommen muss, um die Regel „RULE_MOVEMENT“ auszulösen. Der „Listen“-Tag ermöglicht es, unterschiedliche Signalwörter zu verknüpfen. Die genaue Auswahl, welches nun tatsächlich ausgesprochen wurde, kann dann über den Wert des Elements (z.B.: „NAV_FORWARD“) erfolgen. Der Tag <O> hingegen umklammert Worte die nur optional sind. „Please move forward“ führt ebenso zu einer Systemreaktion wie „Go backward“. Zeichen wie „-“, „+“ oder auch Wildcards können auch verwendet werden und haben unterschiedliche Auswirkungen. In diesem Falle meint das „-“ vor dem „please“, dass auch ähnlich klingende Worte akzeptiert werden.

3.3 Interactivity Spheres

Die Zuordnung der Sprachbefehle zu bestimmten Objekten stellt eine Problematik dar, welche in einem Konzept der Kollisionserkennung ihre Lösung findet. Für eine Interaktion relevante Objekte werden dabei mit sog. „Interactivity Spheres“ [7] umgeben, welche eine Art Aktionsradius definieren.

Nur wenn der Benutzer sich innerhalb einer solchen Kugel befindet, kann das Objekt, zu welchem diese Kugel gehört, Befehle „erkennen“ und richtig interpretieren. Spheres von Objekten, welche auf dieselben Befehle reagieren (z.B. Türen-Öffnen), sollten

einander nicht überschneiden, da sonst eine eindeutige Zuordnung nicht mehr möglich ist.

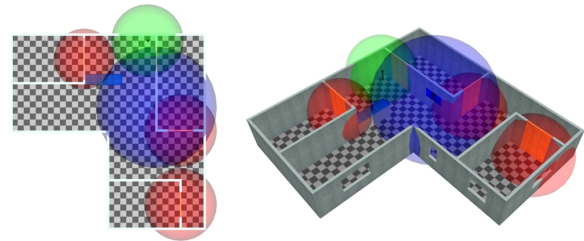


Abbildung 6: Die Interactivity Spheres der einzelnen Objekte in SpeechVR.

Dieses Konzept trifft sich auch sehr gut mit dem Realismus einer Sprachsteuerung, denn auch in der „wirklichen“ Welt ist die Hörbarkeit der menschlichen Stimme an die Entfernung gebunden.

3.4 Kollisionserkennung

Die Kollisionserkennung ist wesentlicher Bestandteil jeder VR-Applikation, da ohne ihr die Glaubwürdigkeit der Szene sehr schnell verloren gehen kann. Wenn der Benutzer eine Wand vor sich sieht, wird er sie nur dann als existent akzeptieren, wenn er sie nicht durchdringen kann. In SpeechVR kommt die Kollisionserkennung zusätzlich ungeübten Anwendern entgegen, da falsche oder zu hektische Eingaben zur Navigation meist in einer Ecke des Raumes enden und so der unkontrollierten Bewegung Einhalt geboten wird.

Kollisionserkennung kann auf unterschiedliche Arten realisiert werden, welche sich im Wesentlichen in Komplexität und Rechenaufwand unterscheiden:

- Bounding Spheres
- Bounding Boxes
 - AABB-Test
 - OBB-Test
- Polygongenaue Kollisionstest

Im Falle der Bounding Spheres werden alle für eine Kollision relevanten Objekte mit einer möglichst kleinen Kugel umgeben, welche aber auf jeden Fall noch das gesamte Objekt fasst. Die Frage der Kollision wird somit auf eine Abfrage auf Überschneidung der Kugeln reduziert. Diese Methode zeigt sich sehr leicht realisierbar und äußerst rechnerfreundlich, allerdings gibt es eine Menge von Objekten, welche sich hierfür nicht eignen. Eine lang gezogene Wand, zum Beispiel, erscheint mit einer umgebenden Kugel wesentlich größer entlang der Längsseite als für eine VR-Szene gut ist.

In diesem Falle würde sich das Konzept der Bounding Boxes anbieten, bei welchem jedes Objekt mit einem Quader anstatt einer Kugel umgeben wird. Der kleinstmögliche Quader, welcher das Objekt noch

vollständig umschließt kann in einer beliebigen Orientierung im Raume stehen, weshalb dieses Konzept auch *OBB-Konzept* (Oriented Bounding Boxes) genannt wird. Eine wesentliche Erleichterung in der Abfrage auf mögliche Überschneidungen kann man allerdings erreichen, wenn dafür Sorge getragen wird, dass nicht der kleinstmögliche Quader gesucht wird, sondern der kleinste, welcher entlang der drei Raumachsen ausgerichtet ist. Das dadurch erreichte *AABB-Konzept* (Axis-Aligned Bounding Boxes) zeichnet sich durch eine raschen und relativ einfache Abfrage aus. Dieses Konzept kam auch in SpeechVR zum Einsatz, da darauf geachtet werden konnte, dass auch alle relevanten Objekte entlang der Raumachsen ausgerichtet sind, es also keine schräg stehenden Wände oder Türen gibt.

Das letzte Verfahren der Kollisionserkennung (Polygon-Kollisionstest) stellt natürlich das genaueste allerdings auch komplexeste und rechenaufwendigste dar. Dabei wird Polygon für Polygon getestet, ob eine Kollision auftritt. Eine Kombination verschiedener Konzepte führt allerdings zu den besten Ergebnissen, wenn eine genauere Erkennung notwendig wird. Dabei wird zuerst zum Beispiel mit Bounding Spheres gearbeitet und nur im Falle einer Kollision auf dieser Ebene, wird dann auf eine genaueres Konzept zurückgegriffen. Sukzessiv kann man sich so auf die objektgenaue Ebene „hinunterarbeiten“ und bleibt laufzeitfreundlich.

3.5 Die „AI-Engine“

Die AI-Engine stellt ein Konzept dar, welches aus der Spielprogrammierung stammt [8]. Einfach ausgedrückt, dient sie dazu, den unterschiedlichsten Objekten einer Szene eine einfache Kommunikationsschnittstelle zur Verfügung zu stellen und das Verhalten dieser Objekte durch eine Verknüpfung mit verschiedenen Zuständen zu definieren, wobei es sich bei letzterem um eine Art „state machine“ handelt. Es können unterschiedlichste Typen von Objekten definiert werden, von denen jeder seine individuellen Zustände („states“) und dafür festgelegte Verhaltensmuster besitzt. Ein sog. *Message-Router*, stellt das Kommunikationsinterface dar. Jedes Objekt hat die Möglichkeit, Nachrichten abzusetzen, welche den eigentlichen Nachrichteninhalt und zusätzliche Informationen, wie Absender, Empfänger, Zustellzeit, usw. enthalten. Der Message-Router übernimmt die Aufgabe der korrekten und zeitgerechten Zustellung dieser Nachrichten, auf welche die adressierten Objekte im gegebenen Falle mit einer Änderung ihres Zustandes und somit ihres Verhaltens reagieren können.

3.6 Von der Sprache zum Ergebnis

Im Folgenden soll nur ein kurzes Kommunikationsbeispiel in SpeechVR angeführt werden, wobei die Aufgabenstellung darin liegt, eine Tür zu öffnen, welche sich nur über einen Hebel bedienen lässt. Das ganze soll natürlich über die Sprache erfolgen:

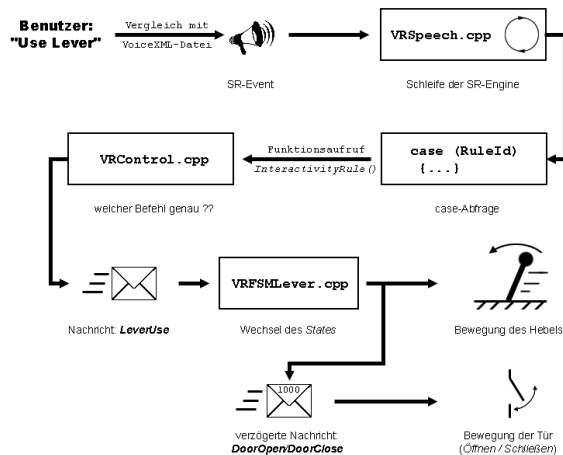


Abbildung 7: Kommunikationsbeispiel aus SpeechVR

Der entsprechende Befehl lautet „Use Lever“, welcher nachdem ihn der Benutzer ausgesprochen hat, zunächst einmal von der Spracherkennungs-Engine im definierten Wortschatz gesucht wird. Wenn das Kommando dort gefunden wurde, wird der Befehl entsprechend seiner ID weitergeleitet. Dies bewirkt schließlich, dass eine Nachricht abgesetzt wird, welche an den Hebel adressiert ist. Dieser befindet sich gerade im Zustand „vorne“ oder „hinten“. Für beide Zustände ist eine Reaktion auf eine ankommende Nachricht „Use Lever“ definiert, welche jeweils einen Wechsel in den anderen Zustand bewirkt. Beim Wechsel des Zustandes wird allerdings nicht nur die Animation des Hebels in der Szene veranlasst, sondern auch noch eine weitere, diesmal aber verzögerte Nachricht an die Tür gesendet, wobei die Verzögerung der Animationsdauer des Hebels entspricht. Je nachdem in welchen Zustand der Hebel nun wechselte, lautet die Nachricht für die Tür „öffnen“ oder „schließen“. Die Tür ihrerseits befindet sich ebenfalls in einem von zwei Zuständen („offen“ oder „geschlossen“). Sollte nun eine Nachricht „öffnen“ ankommen, so wird im Falle einer geschlossenen Tür der Zustand gewechselt. Dieser Wechsel des Zustands hat nun wiederum eine Animation in der Szene zur Folge.

4. Ergebnisse

Basierend auf der Funktionalität und der Handhabung des Prototypen SpeechVR lassen sich einige Aussagen treffen.

Die Spracherkennung ist eine sehr sinnvolle und hilfreiche Eingabehilfe für VR-Szenen darstellt. Die **Interaktion** mit den verschiedensten Objekten scheint mit Hilfe eines Sprachbefehls sehr intuitiv zu sein. Die Illusion einer virtuellen Welt wird durch die Spracheingabe kaum gestört, da der Blick des Benutzers voll und ganz auf das Ausgabegerät (Bildschirm oder HMD) gerichtet bleiben kann. Sind die verschiedensten Befehle einmal erlernt, ist es auch einem ungeübten Anwender sehr leicht möglich, diese Funktionalitäten von **SpeechVR** zu nutzen.

Die **Navigation** zeigt sich in diesem Zusammenhang etwas differenziert. In ihrem Fall gelten zwar dieselben Vorteile wie bei der Interaktion, allerdings kommt hier ein weiterer Aspekt zu tragen, welcher sich als sehr störend erwies. Das Arbeiten mit einer Sprachsteuerung ist mit einer zeitlichen Verzögerung unumgänglich verbunden, da eben eine gewisse Zeit benötigt wird, um den gesprochenen Befehl zu interpretieren und korrekt zuzuordnen. Im bestmöglichen Fall muss zumindest der Befehl fertig ausgesprochen sein, ehe das System darauf reagieren kann. Diese zeitliche Verzögerung wirkt sich allerdings äußerst störend auf die Navigation in der VR-Szene aus. Ein exaktes Navigieren wird dadurch sehr erschwert. Wenn der Benutzer sich in einer Rechtsdrehung befindet und diese zu beenden wünscht, dann muss er im Falle von SpeechVR „*Stopp*“ als Befehl eingeben. In der Zeit in welcher der Benutzer diesen Befehl ausspricht und dieser dann in weiterer Folge vom System richtig erkannt und interpretiert wird, ist die Drehung bereits so weit fortgeschritten, dass sie in die Gegenrichtung korrigiert werden muss. Durch intensives Üben mit dem System kann der Benutzer zwar diesem Effekt entgegenwirken, indem er den Befehl etwas früher kundtut, allerdings scheint es nicht möglich zu sein, diese Erscheinung wirklich in den Griff zu bekommen.

Im Prototypen wurden zum Erleichtern der Navigation einige Hilfsbefehle eingeführt, wie zum Beispiel das schrittweise Bewegen, die Kontrolle über die Fortbewegungsgeschwindigkeit oder eine fix definierte 90°-Drehungen, welche alle zusammen den Benutzer zwar unterstützen können, doch eine gewisse Schwierigkeit beim Navigieren nicht einfach verschwinden lassen können.

Gerade die Kombination von Spracheingabe mit anderen Eingabegeräten (z.B. Dataglove) erscheint bei der Navigation sehr sinnvoll. Auf einem auf den Benutzer abgestimmtes System lässt sich sehr komfortabel und vor allem intuitiv mit der VR-Szene umgehen, wobei die Mannigfaltigkeit der zur Verfügung stehenden

Sprachbefehle schon bei diesem Prototypen die meisten Erstbenutzer sehr beeindruckte. Ohne Anlegen eines eigenen Stimmenprofils wird ein Arbeiten mit dem System allerdings schon um einiges problematischer, da nicht mehr gewährleistet werden kann, dass jeder Sprachbefehl gleich auf beim ersten Mal erkannt wird. So kommt es mitunter unnötigen Verzögerungen, die vor allem die Navigation sehr stören. Bei der Interaktion ist diese Latenz bei weitem nicht so störend, weil dort nicht unbedingt sofort eine Aktion ausgelöst werden muss und es den meisten Benutzern nicht sonderlich auffällt.

Sehr komplexe Navigationen (wie z.B. „Gehe ins Wohnzimmer“, „Öffne den Schalter oben rechts“, etc.) sind natürlich mittels Spracheingabe sehr einfach umzusetzen und es macht Sinn, dies nicht mit den bisherigen Eingabegeräten zu tun.

5. Vergleichbare Systeme

Derzeit gibt es zwar viele Bereiche, wo Spracherkennung erfolgreich eingesetzt wird – ein Einsatz dieser Technologie im Graphik- bzw. VR-Bereich ist jedoch neu. Ein vergleichbares System beschreibt Hundertmark in [2]. Bei dieser Arbeit werden technische Geräte visualisiert und der Benutzer erhält Zusatzinformationen, die er über Spracheingabe nach Belieben bekommen kann.

Eine weitere - sehr interessante - Anwendung ist VRIO. Dabei handelt es sich um ein visuelles Debugging-Programm, bei welchem über Spracheingabe Parallelprogramme untersucht werden können. Die Idee, die die Autoren verfolgen, besteht darin, dass man mittels CAVE bzw. ImmersaDesk Parallelprogramme besser visuell untersuchen kann. Die Steuerung und Navigation sollte zudem nicht mittels Joystick oder 3D-Mouse erfolgen, sondern über Spracheingabe gelöst werden [3].

6. Resümee und Ausblick

Spracherkennung alleine wird sicherlich nicht die derzeitigen VR-Eingabeprobleme beseitigen. Jedoch kann man mittels Spracherkennung eine noch intuitivere Eingabe ermöglichen. Die Autoren sind der Auffassung, dass erst durch eine Kombination mit den herkömmlichen Eingabemedien die optimale Grundlage geschaffen werden kann. Der Leser stelle sich nur vor, dass er mittels Gestik und Sprache ein System steuert, wo er mit der Hand auf einen Gegenstand zeigt und mittels Sprache die Steuerbefehle erteilt. In einem solchen Szenario können vor allem sehr komplexe Bewegungen oder Aktionen

mittels der Sprache bewältigt werden, während eher einfachere Interaktionen mit den herkömmlichen Eingabegeräte gemacht werden können.

SpeechVR hat gezeigt, dass eine einfache Navigation mittels Spracherkennung wenig Sinn macht, sofern man wirklich nur mit der Sprache versucht zu navigieren. Unter einer einfachen Navigation verstehen die Autoren z.B. einfache Vorwärts- oder Rückwärtsbewegungen. Komplexere Navigationen (z.B. „Gehe in die Küche“, „Setz Dich auf den Stuhl“, „Schau aus dem Fenster“) lassen sich natürlich mittels Spracheingabe optimal lösen. Der nächste Schritt von SpeechVR wird es sein, das System von einem Desktop-Version auf eine HMD-Version zu portieren, um die Vorzüge unseres Programmes anhand einer VR-Applikation zu testen. Geplant ist auch der Einsatz und insbesondere die Kombination mit unterschiedlichen Eingabegeräten. Ein weiteres Ziel ist es, die Komplexität der Sprache auszubauen. Man stelle sich bloß vor, man könne sich mit virtuellen Personen in der Szene richtig unterhalten. Da sich die Spracherkennung in den unterschiedlichen Gebieten stark weiterentwickeln wird, wird auch die Entwicklung im Bereich „Virtual Environments“ nicht lange auf sich warten lassen.

7. Quellenverzeichnis

[1] Essential Reality, <http://www.essentialreality.com>, Stand Juli 2001.

[2] J. Hundertmark, *Sprachgesteuerte Navigation in virtuellen Welten*, Diplomarbeit, Universität Paderborn, <http://www.uni-paderborn/diplomarbeiten/hundertmark/abstract.html>, Stand: Juli 2001.

[3] D. Kranzlmüller, B. Reitinger, I. Hackl, J. Volkert, *Voice Controlled Virtual Reality and Its Perspectives for Everyday Life*, Technischer Bericht, GUP, Johannes Kepler Universität Linz, 2001.

[4] Microsoft, *Recent Speech Developments at Microsoft*, Technical Report, SpeechSDK 5.0, <http://www.microsoft.com/speech>, Stand: Juli 2001.

[5] J. Rickel, W. Lewis Johnson, *STEVE: A Pedagogical Agent for Virtual Reality*, Proceedings of the 2nd International Conference on Autonomous Agents (Agents'98), ACM Press, 332-333, 1998.

[6] Edward D. Shockley, *Advances in Human Language Technologies: An IBM White Paper*, <http://www-4.ibm.com/software/speech/dev/hltwp.html>, Stand: Juli 2001.

[7] M. Pargfrieder, *Sprachgesteuerte Möglichkeiten der Navigation und Interaktion in virtuellen Welten*, Diplomarbeit 2001, Fachhochschule Hagenberg, Medientechnik und -design.

[8] S. Rabin, *Designing a General Robust AI Engine*, Game Programming Gems, ISBN 1-58450-049-2, Charles River Media, 2000.