

World3d

Ein Prototyp für eine interaktive virtuelle Umgebung
im Rahmen des Projekts Docs 'n Drugs

Praktikumsbericht
von Jan Fischer

Betreuer:
Torsten Illmann (Abteilung Medieninformatik)
Alke Martens (Abteilung Künstliche Intelligenz)

Ulm, Oktober 2000

Inhaltsverzeichnis

1. Einleitung	3
2. Die Auswahl der passenden Technologie.....	3
3. Die Erstellung der „building blocks“	5
4. Die Konfiguration der Software	5
5. Die Architektur der World3d-Software	6
5.1. World3dMain	6
5.2. WorldAssembler	6
5.3. TextPanel	7
5.4. SlidingDoor, RoomDoor	7
5.5. InfoShape, PickBehavior	7
6. Der Aufbau des Szenengraphen	7
6.1. Zur Bedeutung des Switch-Knotens	8
7. Die Bedienung von <i>World3d</i>	9
8. Die Erweiterung von <i>World3d</i>	12
8.1. Reaktion auf Benutzerinteraktionen	12
8.2. Veränderungen der statischen Grafik-Objekte	13
8.3. Veränderungen der animierten/interaktiven Grafik-Objekte.....	13
9. Ausblick.....	14
Literaturverzeichnis	14

1. Einleitung

Bei der *World3d*-Software handelt es sich um den Prototypen für eine Virtual Reality-Benutzerschnittstelle. Das Ziel ist dabei die interaktive dreidimensionale Darstellung eines Äquivalents für die im Projekt Docs 'n Drugs bereits vorhandene Menüführung. Es soll möglich sein, mittels der thematisch naheliegenden Analogie eines Krankenhauses mit Stationen und Patientenzimmern Einträge aus der vorhandenen Falldatenbank auszuwählen.

Die Entwicklung des *World3d*-Pakets hatte den Charakter einer Machbarkeitsstudie. Es wurde kein großer Wert auf ausgefeilte Grafik oder umfangreiche Funktionalitäten gelegt. Vielmehr sollte *World3d* vor allem die folgenden Anforderungen erfüllen:

- Konfigurierbarkeit (Zahl/Art der Krankenfälle) per XML-Datei
- Dynamische Generierung einer entsprechenden 3D-Umgebung
- Bereitstellung der Möglichkeit von Benutzerinteraktion (Bewegung in der Umgebung und Auswahl von Elementen)



Der Blick in eine von *World3d* dargestellte Station mit zwei Patienten

Angelehnt an die bisherige Philosophie des Projekts Docs 'n Drugs sollte das System vollständig in Java implementiert werden. Daher kam als Basistechnologie nur Java3D [1] in Frage.

Dieses Praktikum wurde vom Autor in Zusammenarbeit mit Matthias Brusdeylins bearbeitet, der vor allem für das XML-Interface und die Benutzernavigation zuständig war. Dieser Bericht konzentriert sich hauptsächlich auf diejenigen Elemente der Software, die vom Autor entwickelt wurden.

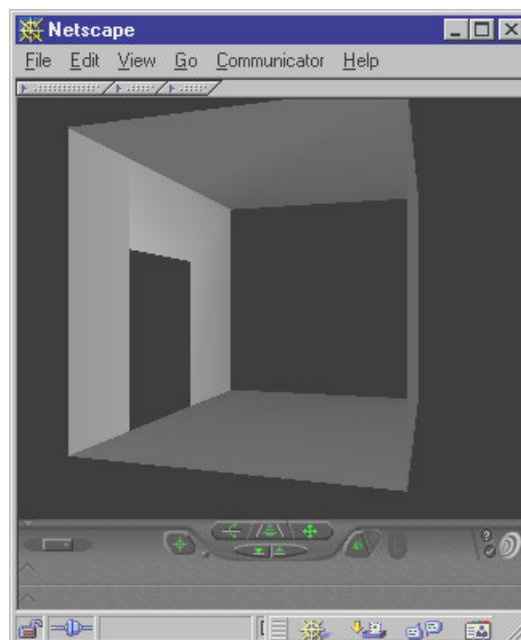
2. Die Auswahl der passenden Technologie

Wie bereits erwähnt, stellt die Java3D Klassenbibliothek die Basis für das *World3d*-Projekt dar. Java3D besitzt eine Reihe entscheidender Vorteile, unter anderem:

- Java3D von Sun ist kostenlos verfügbar.

- Die Integration in Java-Programme erfolgt über eine übersichtliche Java-Klassenschnittstelle.
- Da Java3D eine native Schnittstelle zur OpenGL-Grafikbibliothek besitzt, ist die Darstellungsgeschwindigkeit mehr als ausreichend für ein interaktives System.
- Java3D ist verfügbar für Windows-, Linux- und Solaris-Systeme und bietet somit eine ausreichende Portabilität.
- Java3D stellt die virtuelle Umgebung intern mit Hilfe eines sogenannten Szenengraphen dar, was eine einfache und elegante Manipulation der virtuellen Umgebung erlaubt.

Ein Ziel bei der Erstellung von *World3d* war es, eine möglichst hohe Konfigurierbarkeit zu erhalten. Aus diesem Grunde fiel die Entscheidung, dass die einzelnen Elemente des virtuellen Krankenhauses nicht fest definiert sein sollten, sondern zur Laufzeit aus externen Dateien gelesen werden. Dies ermöglicht es, ein solches als „building block“ bezeichnetes Bauteil in einem externen 3D-Editor zu verändern und dann in *World3d* zu verwenden, ohne dass eine Anpassung des *World3d*-Quellcodes nötig wird.



Darstellung eines „building blocks“ (Stationsgang) von *World3d*

Java3D ermöglicht das einfache Importieren von externen 3D-Dateien, indem eine Reihe sogenannter Loader [2] zur Verfügung gestellt werden. Diese sind zum Teil schon bei der Standard-Distribution vorhanden, zum Teil können sie kostenlos über verschiedene Webseiten bezogen werden. Im Zuge dieser Praktikumsarbeit wurden zunächst einmal eine Reihe solcher Loader auf Ihre Tauglichkeit für das *World3d*-Projekt getestet. Gleichzeitig wurden auch verschiedene Optionen im Hinblick auf Erstellung und Konvertierung der entsprechenden 3D-Dateien untersucht.

Verworfen wurden dabei recht schnell die Loader für das Lightwave- und das Wavefront OBJ-Format. Diese Loader unterstützen entweder einige wichtige Features (wie Texturen oder Materialeigenschaften) nicht, oder es sind keine brauchbaren kostengünstigen Modelle für die entsprechenden Formate verfügbar. Letzten Endes kristallisierte sich das weitverbreitete Vrmf-Format [3] als sinnvollste Lösung heraus. Für dieses gibt es ein spezielles Vrmf97-Modul, das das erwähnte Loader-Interface implementiert.

Problematisch ist allerdings, dass auch dieser Vrl-Loader nicht alle Möglichkeiten des extrem umfangreichen Vrl-Formates und alle möglichen Syntax-Stile unterstützt. Die Vrl-Exporte einiger Modeller und Konvertierer lassen sich somit nicht in Java3D einladen. Daher wurden weiterhin 3D-Tools gesucht, die mit dem Vrl97-Loader zusammenarbeiten. Schließlich kamen folgende Programme zum Einsatz:

- CosmoWorlds von Cosmo Software für die Erstellung der Modelle
- CosmoPlayer, ebenfalls von Cosmo Software, zur Überprüfung der Modelle
- 3D Exploration [4] von X Dimension Software (www.xdsoft.com) zur Betrachtung und Konvertierung der Modelle

Mit diesen Tools war es in Zusammenarbeit mit Java3D möglich, die nötigen „building blocks“ so zu erstellen, dass die Geometrien, aber auch Texturen, Materialeigenschaften und Lichtquellen korrekt importiert werden.

3. Die Erstellung der „building blocks“

Die Bausteine für das virtuelle Krankenhaus wurden vom Autor mit der CosmoWorlds Software erstellt. Insgesamt gibt es sechs solcher Modelle, die sich allesamt im „buildingblocks“-Unterverzeichnis des *World3d*-Projekts befinden:

- Ein Gangstück mit einer Tür (Datei `Gang1.wrl`)
- Ein Gangstück mit zwei Türen (Datei `Gang2.wrl`)
- Das Ende eines Ganges (Datei `GangEnd.wrl`)
- Das unterste Stockwerk (Datei `Level0.wrl`)
- Das Mittelstück der oberen Stockwerke (Datei `LevelHall.wrl`)
- Ein Gangstück zur Darstellung der Stationsinformation (Datei `Statinfo.wrl`)

4. Die Konfiguration der Software

World3d lässt sich in zweierlei Hinsicht konfigurieren. Einerseits können die Vrl-Bausteine angepasst werden, wie in den vorhergehenden Abschnitten beschrieben. Zum anderen muss sich in einer Datei namens `hospital.xml` im *World3d*-Verzeichnis eine Beschreibung der Abteilungen und Patienten des Krankenhauses befinden. Sowohl die „building blocks“ wie auch die XML-Datei werden beim Start der Software einmal eingelesen und dienen der Einstellung der virtuellen Umgebung. Eine genaue Beschreibung des verwendeten XML-Formates liegt im Praktikumsbericht von Matthias Brusdeylins vor.

5. Die Architektur der World3d-Software

Das *World3d*-Projekt besteht aus einer Reihe von Klassen, von denen die meisten als Hilfs- oder Utility-Klassen dienen. Alle Klassen befinden sich dabei im Package `dnd.world3d`. Die zentrale Klasse des Projekts ist `World3dMain`. Diese kann sowohl als Applet als auch als Applikation gestartet werden.

`World3dMain` lädt nach ihrem Aufruf die Konfigurationsdateien und öffnet ein Fenster, in dem die virtuelle Umgebung angezeigt wird. Danach werden die anderen Klassen – in der Regel durch Objektinstanzierung – verwendet, um den Szenengraphen komplett aufzubauen. Ein Überblick über die vom Autor erstellten Klassen findet sich im Folgenden:



Einige wichtige Klassen sollen hier kurz besprochen werden.

5.1. *World3dMain*

Diese Klasse baut unter Verwendung der anderen Klassen den kompletten Szenengraphen auf. Zudem stellt sie die Methode `changeLevel` zur Verfügung, die es ermöglicht, zwischen den Stockwerken des Krankenhauses umzuschalten. Wichtige Knoten des Szenengraphen werden in dieser Klasse noch als Referenzen gehalten.

5.2. *WorldAssembler*

Diese Klasse baut die Teilszenengraphen auf, die je ein Stockwerk des Krankenhauses – d.h. eine Abteilung – beinhalten. Zur Verwendung ist es nötig, ein Objekt der Klasse `WorldAssembler` zu instanzieren. Dieses stellt dann die Methode `buildLevel` bereit, die je einen neuen Teilgraphen mit dem entsprechenden Stockwerk zurückgibt. Die Klasse `WorldAssembler` wird von der Klasse `World3dMain` verwendet.

5.3. *TextPanel*

Die Klasse `TextPanel` macht es möglich, grafische Objekte zu erstellen, die als Schilder dienen. Auf diesen Schildern lassen sich ein- oder mehrzeilige Texte darstellen; dabei sind die Vorder- und Hintergrundfarbe konfigurierbar. Die Schilder lassen sich in der virtuellen Umgebung vom Benutzer anklicken und liefern dann eine Zeichenkette zur Beschreibung und eine numerische ID zurück. Die Darstellung der Texte wurde dabei mit Hilfe der Java2D-API als Texturen realisiert.

Ein `TextPanel`-Objekt stellt quasi einen „Generator“ für solche Schilder dar. Um ein neues Schild zu erzeugen, muss für dieses eine der `createPanel`-Methoden aufgerufen werden.

5.4. *SlidingDoor, RoomDoor*

Diese Klassen dienen dazu, die Türen innerhalb der virtuellen Umgebung darzustellen. Da sie von entsprechenden Java3D-Klassen abgeleitet sind, können sie einfach instanziiert und in den Szenengraphen eingefügt werden. Die Besonderheit einer `SlidingDoor` ist es, dass sie auf den Abstand des Beobachters durch ein Öffnen oder Schließen reagiert („Bewegungsmelder-Effekt“).

5.5. *InfoShape, PickBehavior*

Diese Klassen ermöglichen Benutzerinteraktion im virtuellen Krankenhaus. Durch sie können spezielle grafische Objekte – sogenannte `InfoShapes` – angegeben werden, die angeklickt werden können. Durch einen solchen Klick wird die `processClick`-Methode der `PickBehavior`-Klasse aufgerufen. Somit kann die Anwendung auf die Interaktion reagieren.

Momentan wird nur auf ein Anklicken der Abteilungsknöpfe im Aufzug reagiert – nämlich durch Wechsel des Stockwerks. Ansonsten werden einfach auf der Konsole Beschreibung und numerische ID des `InfoShapes` ausgegeben.

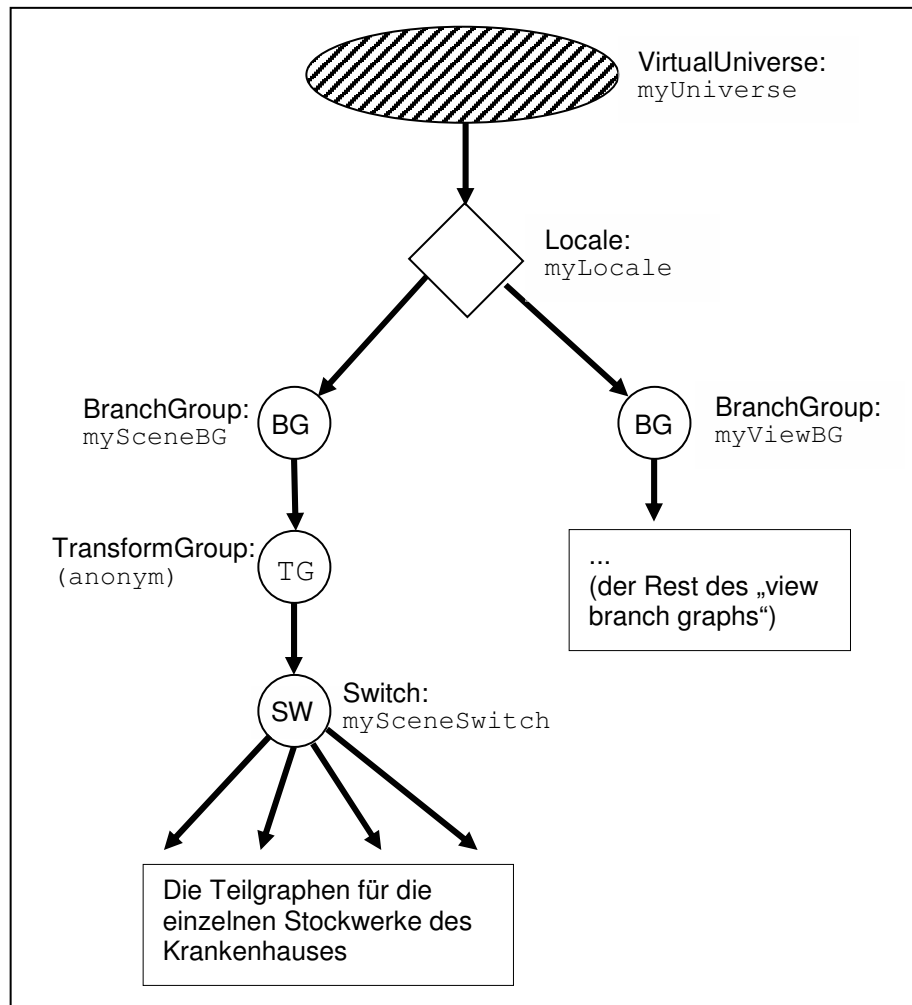
Genauere Informationen zu allen Klassen finden sich auch in der vorhandenen Javadoc-Onlinedokumentation.

6. Der Aufbau des Szenengraphen

Innerhalb der Klasse `World3dMain` wird der Szenengraph, der die grafische Darstellung des kompletten Krankenhauses beinhaltet, aufgebaut. Dabei wird ein standardmäßiger View-Teilgraph erstellt, der die Angaben zu Position und Art des Beobachters enthält. Eine Verwendung der bei Java3D mitgelieferten `SimpleUniverse`-Klasse kam hier nicht in Frage, da der View-Teilgraph zum Zwecke der Benutzernavigation manipulierbar bleiben musste. Eine Referenz auf den View-Teilgraphen wird in der Klasse `World3dMain` in der Variable `myViewBG` gehalten.

Auf der anderen Seite wird ein Szenen-Teilgraph in der Variable `mySceneBG` erstellt. In diesem Gruppenknoten befindet sich nur ein einzelner Knoten, nämlich eine Transformationsgruppe. Diese dient nur dem Zweck, das komplette Krankenhaus mit einer uniformen Skalierung zu vergrößern. Der Skalierungsfaktor findet sich in der Konstante `sc`

leFactor der Klasse `World3dMain`. Da diese Transformationsgruppe später nicht mehr benötigt wird, wird sie nicht explizit in einer Referenz gehalten. Innerhalb dieser Transformationsgruppe findet sich dann endlich das eigentliche Krankenhaus in Form eines Switch-Knotens. Der Switch-Knoten – in der Klasse als `mySceneSwitch` abgespeichert – beinhaltet dann alle einzelnen Stockwerke des Krankenhauses als Kinder. Diese werden, wie weiter oben ja schon erwähnt, von einem Objekt des Typs `WorldAssembler` erstellt. Die grafische Darstellung des relevanten Ausschnitts des Szenengraphen sieht so aus:



Darstellung des Wurzelbereichs des Szenengraphen von *World3d*

Die Teilgraphen der einzelnen Stockwerke sind selbstverständlich um ein Vielfaches komplexer und umfassen unter Umständen einige hundert Knoten. Um einen genauen Einblick in ihren Aufbau zu erhalten, empfiehlt sich ein Blick in die Klasse `WorldAssembler`.

6.1. Zur Bedeutung des Switch-Knotens

Wie oben erwähnt, werden alle Stockwerke des Krankenhauses beim Programmstart erzeugt und an `mySceneSwitch` angehängt. Dies ist notwendig, um ein einfaches

und elegantes Umschalten zwischen den Stockwerken zu ermöglichen – nämlich durch Setzen des aktiven Kindknotens für die Switch-Gruppe.

Alternative Verfahren, wie das Aushängen des alten Stockwerks und Einhängen des dynamisch erzeugten neuen Stockwerks in den Szenengraph, haben sich als sehr schwierig oder unmöglich erwiesen. Der Grund dafür ist es, dass es in Java3D nur mit starken Einschränkungen erlaubt ist, einen Szenengraphen, der angezeigt wird (also „live“ ist), zu verändern (s. auch [5]).

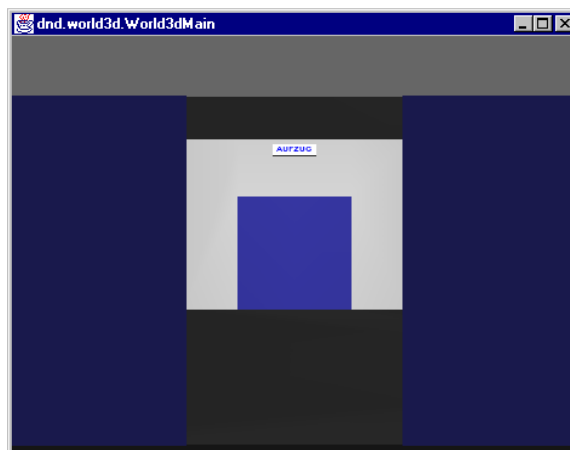
Dies hat natürlich negative Auswirkungen auf den Speicherverbrauch der Anwendung. Selbst bei mittelgroßen Krankenhäusern mit mehr als hundert Patienten hält sich dieser momentan aber noch im Rahmen (ca. 30 – 40 MBytes). In einer späteren Phase des Ausbaus von *World3d* könnte es aber durchaus sinnvoll sein, hier eine Optimierung vorzunehmen.

7. Die Bedienung von *World3d*

Nach dem Start von *World3d* befindet man sich auf dem Vorplatz des Krankenhauses. Dieser soll später noch – je nach vorliegendem Notfall – mit einem Krankenwagen, Notfallhelikopter etc. versehen werden.

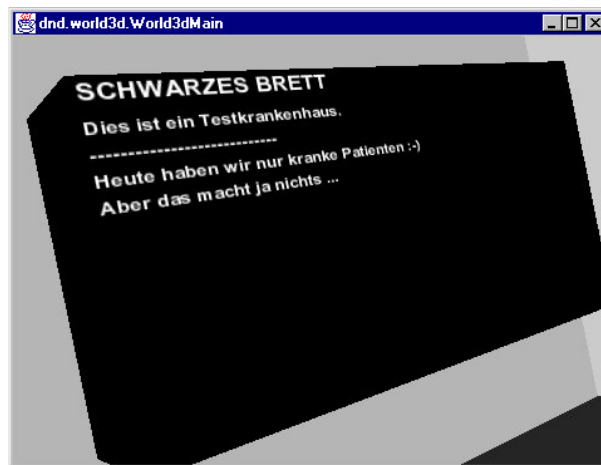
Die Navigation in der virtuellen Umgebung erfolgt – wie aus ähnlichen Anwendungen bekannt – mit Hilfe der Cursortasten. Die Cursortasten links und rechts drehen dabei den Beobachter um die y-Achse. Die Cursortasten oben und unten bewegen ihn vor und zurück. Zu beachten ist, dass zum momentanen Zeitpunkt noch keine Kollisionserkennung erfolgt. Das bedeutet, es ist problemlos möglich, sich auch durch Wände, Türen usw. hindurchzubewegen. Diese Inkonsistenz wird aber in Bälde von Matthias Brusdeylins behoben werden.

Bei Annäherung an die Eingangstür öffnet sich diese und gibt den Blick auf das Innere des Krankenhauses frei.



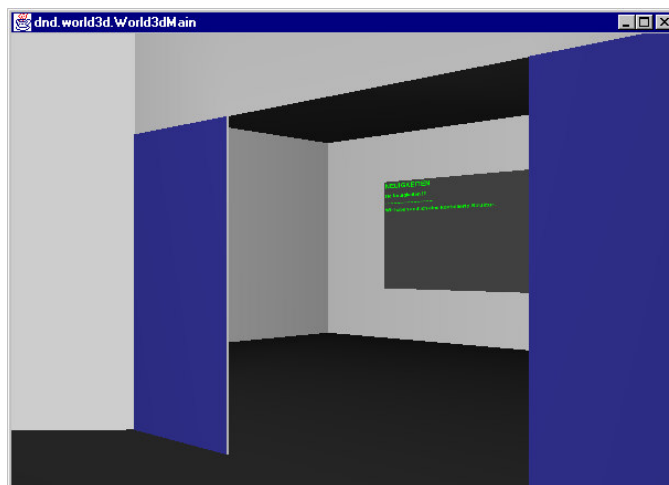
Der Eingang des Krankenhauses

Im untersten Stockwerk finden sich drei relevante Elemente: Das Schwarze Brett zur Linken, der Aufzug gegenüber des Eingangs und der spätere Chatraum zur Rechten.

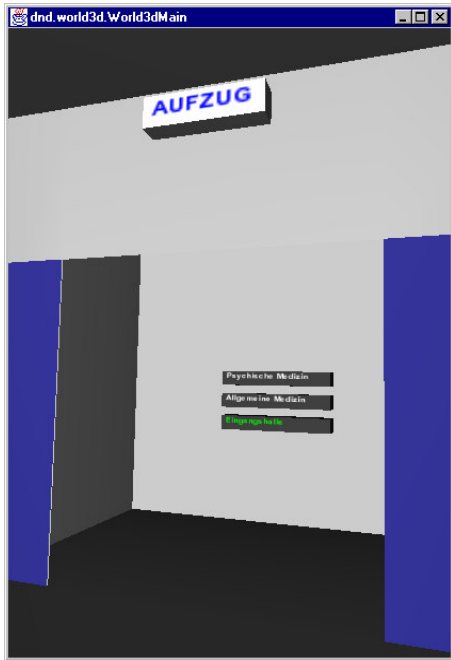


Das Schwarze Brett

Auch im Chatraum findet sich noch mal eine Art Schwarzes Brett – das „Neuigkeiten-Brett“. Beide Schwarzen Bretter lassen sich Anklicken und liefern somit eine Beschreibung zurück. Auf diese Art und Weise könnte man später beispielsweise den Aufruf eines externen Textviewers mit Zusatzinformationen realisieren.



Der Chatraum mit dem Neuigkeiten-Brett



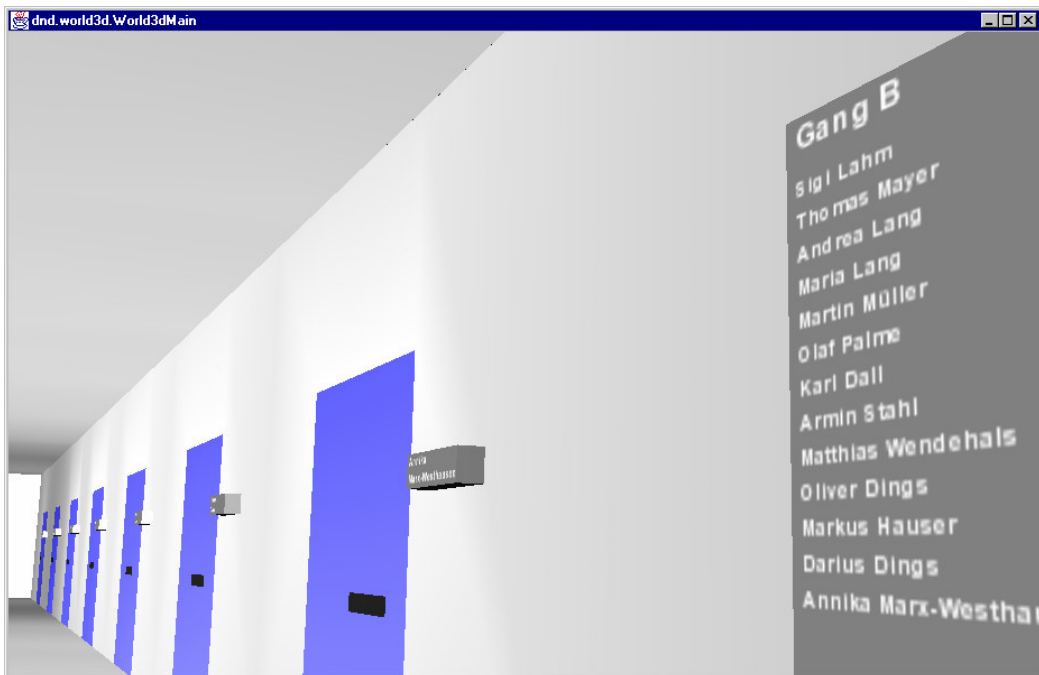
Der Aufzug

Das zentrale Bedienelement der World3d-Umgebung ist der Aufzug. In ihm findet sich für jede Abteilung des Krankenhauses ein Knopf. Ein Klick auf einen dieser Knöpfe führt zu einem Wechsel des aktuellen Stockwerks.

Natürlich befindet sich in jedem der Stockwerke ein Aufzug, der entsprechend funktioniert.

In den Abteilungen finden sich dann – in Gänge aufgeteilt – die Patientenzimmer, von denen jedes für einen Fall steht. Die Gänge werden dabei festverdrahtet als „Gang A“, „Gang B“ und „Gang C“ bezeichnet. Zu Beginn jeden Ganges hängt rechts eine Übersicht über alle in diesem Gang liegenden Patienten. Zudem steht neben jeder Patiententür der zugehörige Name.

Es ist möglich, sowohl die Namensschilder, wie auch die Patiententüren, aber auch die Übersichtslisten anzuklicken.



Der Blick in einen Abteilungsgang

Dies soll später die Hauptfunktion von *World3d* ermöglichen. Durch einen Klick auf die Tür oder das Namensschild eines Patienten soll der zugehörige Fall geladen und angezeigt werden.

Wie *World3d* entsprechend erweitert werden kann, wird im folgenden Abschnitt diskutiert.

8. Die Erweiterung von *World3d*

Wie eingangs erwähnt, handelt es sich bei *World3d* um einen Prototypen, dessen Quellcode aber auch die Basis für die weitere Entwicklung darstellen soll. Daher ist es besonders interessant zu betrachten, wie solche Erweiterungen oder Veränderungen vor sich gehen müssen.

Allgemein kann gesagt werden, dass die *World3d*-Software den Charakter einer Anwendung hat. Viele Erweiterungen – gerade umfassendere – lassen sich daher nicht durch bloße Instanzierung oder Vererbung der vorhandenen Klassen bewerkstelligen. Vielmehr wird es oft nötig sein, den vorhandenen Quellcode „intern“ anzupassen.

8.1. Reaktion auf Benutzerinteraktionen

Eine der vordringlichsten Aufgaben bei der Erweiterung von *World3d* wird es sein, für eine angemessene Reaktion auf Benutzereingaben zu sorgen. Hauptsächlich ist es wichtig, beim Klick auf die Tür oder das Namensschild eines Patientenzimmers den entsprechenden Fall anzuzeigen.

Die Reaktion auf Mausklicks findet – wie oben bereits angesprochen – in der Klasse `PickBehavior` statt. Hier gibt es die Methode `processClick`, die als Event-Handler fungiert. Um nun ein angepasstes Verhalten zu erzeugen, kann man diese Methode entweder im Quellcode von `PickBehavior` verändern, oder aber in einer abgeleiteten Klasse überschreiben.

```
public void processClick(InfoShape where)
{
    // Print InfoShape information
    System.out.println(where.getDescr() + " " + where.getID());
}
```

Die `processClick`-Methode der Klasse `PickBehavior`

Die Methode `processClick` bekommt als Parameter die `InfoShape` übergeben, auf die der Klick erfolgt ist. Für diese geben die Methoden `getDescr()` und `getID()` die Beschreibung bzw. die numerische ID zurück.

Sollte die Option gewählt werden, dass man `processClick` in einer abgeleiteten Klasse überschreibt, ist noch darauf zu achten, dass dann `World3dMain` entsprechend angepasst werden muss.

```
private void buildUniverse() {
    (...)
    // The picking behavior
    PickBehavior myPick;
```

```

(...)

// Create new picking behavior
myPick = new PickBehavior(canvas, mySceneBG,
    new BoundingSphere(new Point3d(0, 0, 0), 5000));
(...)

}

```

In der Methode `World3dMain.buildUniverse()` wird das `PickBehavior` instanziiert

Denn in `World3dMain` wird innerhalb der Methode `buildWorld` eine Instanz von `PickBehavior` angelegt, um dieses zu aktivieren. Bei Verwendung einer anderen Klasse muss diese Stelle also entsprechend umgeschrieben werden.

Zur Identifikation der angeklickten `InfoShape` dient die numerische ID, die diese liefert. In der Javadoc-Dokumentation für die Klasse `WorldAssembler` findet sich eine Liste der bisher verwendeten IDs.

8.2. *Veränderungen der statischen Grafik-Objekte*

Um die statischen Grafik-Objekte, d.h. die „building blocks“, zu verändern, reicht es, wenn die zugehörigen Vrlml-Dateien angepasst werden. Dabei ist allerdings zu beachten, dass das grundsätzliche Layout - wie die Position der Türöffnungen oder der freie Platz für die Schwarzen Bretter - nicht verändert werden darf. Denn die interaktiven/animierten Elemente, die im Programm erzeugt werden, werden „festverdrahtet“ positioniert. Somit bestünde die Gefahr, dass bei einer vollständigen Änderung des Bauplans beispielsweise plötzlich Türen im Raum schweben oder gar nicht mehr zu erreichen sind.

Zudem ist noch zu beachten, dass auch die von Matthias Brusdeylins noch zu erstellende Kollisionserkennung eventuell fest definierte Begrenzungen verwenden wird. Auch diese wären – etwa beim Hinzufügen weiterer Hindernisse, z.B. Möbelstücke – dann natürlich anzupassen.

8.3. *Veränderungen der animierten/interaktiven Grafik-Objekte*

Sämtliche animierten bzw. interaktiven Elemente werden zur Laufzeit in der Software erzeugt. Der Grund dafür ist, dass es mit großem Aufwand verbunden wäre, den Szenengraphen einer mittels Loader eingelesenen Vrlml-Datei nachträglich entsprechend zu manipulieren. Die in Kapitel 5 als „Klassen für grafische Objekte“ beschriebenen Klassen dienen dazu, die notwendigen Türen, Schilder usw. zu erzeugen. Wenn also das Aussehen oder die Funktionalität solcher Elemente verändert werden soll, sind Änderungen im zugehörigen Quellcode nötig.

9. Ausblick

Im Großen und Ganzen kann *World3d* als ein durchaus erfolgreicher Prototyp angesehen werden. Es ist demonstriert worden, dass die gestellte Aufgabe mit vertretbarem Aufwand in Java3D realisiert werden kann.

Für die Zukunft wären eine ganze Reihe sinnvoller Erweiterungen denkbar, beispielsweise die Folgenden:

- Die vollständige Integration in das Docs 'n Docs Projekt, vor allem durch den Zugriff auf die Falldatenbank.
- Die Verschönerung der grafischen Darstellung, vor allem durch Verbesserung der „building blocks“.
- Die Integration eines Chatraums, u.U. auch als 3D-Chat mit entsprechenden Figuren, die sich im Krankenhaus bewegen können.
- Eine Optimierung der grafischen Darstellung, etwa durch die Anwendung einfacher Occlusion Culling-Verfahren.

Literaturverzeichnis

- [1] *Java 3D API Home Page* bei <http://www.java.sun.com/products/java-media/3D/index.html>
- [2] *J3D – File Loader Archives* bei <http://www.j3d.org/utilities/loaders.html>
- [3] *Web3D Consortium* bei <http://www.vrml.org/>
- [4] *3D Exploration* bei <http://www.xdsoft.com/explorer/>
- [5] *Detaching a BranchGroup or Behavior* bei <http://www.j3d.org/faq/manipulating.html#detach>