Ein hierarchisches Verfahren zur effizienten Kollisionsvermeidung in einer interaktiven graphischen Entwurfsumgebung

Marko Merkler und Ipke Wachsmuth

1. Übersicht

In dieser Arbeit wird ein Verfahren zur effizienten Kollisionsvermeidung bei der Manipulation (Translation, Rotation, Skalierung) von Graphikobjekten in virtuellen Räumen dargestellt. Dieses für das VIENA-System (Wachsmuth & Cao, 1995) entwickelte Verfahren wird durch eine Hierarchie von vereinfachten Objektmodellen und speziell für die einzelnen Modelle entwickelten Kollisionstests realisiert. Die einzelnen Objektmodelle setzen sich aus den konvexen geometrischen Raumprimitiven Sphäre, Zylinder und Quader zusammen. Die Grundidee ist dabei die vollständige Umhüllung des zu betrachtenden Objektes bzw. der Objektkomponente durch einen Raumprimitiv der entsprechenden Hierarchiestufe.

Die Kollisionstests operieren auf den einzelnen Raumprimitiven und werden zu einer Aussage über den Kollisionszustand des Gesamtobjektes integriert. Die entwickelten Kollisionstests dienen einerseits als Vortests, um möglichst viele Objekte bzw. Komponenten als potentielle Kollisionspartner für einen nachgeschalteten Kollisionstest auf höherer Hierarchiestufe auszuschließen, andererseits werden sie direkt zur Transformationsrealisation verwendet und beschreiben dann eine potentielle Ziellage der eigentlichen Objekte im VIENA-Raum.

Aufgrund der großen Ausdehnung der Objektmodelle Sphäre und Zylinder erlauben in dem vorgestellten Verfahren lediglich die Bounding-Box-Modelle die Beschreibung von erlaubten Zielpositionen.

Das Verfahren arbeitet nach dem Prinzip des "Step and Check" (Baraff, 1992). Die Kollisionstests operieren jeweils nur auf gleichartigen Objektmodellen. Kollisionstests, die unterschiedliche Raumprimitive gegeneinander auf Kollision testen, wurden bislang nicht entwickelt. Eines der Hauptziele war die Implementierung der Kollisionstests für strukturierte Komponenten-Bounding-Boxen, um die räumliche Struktur der Objekte gegenüber dem Modell der einfachen Objekt-Bounding-Boxen besser beschreiben zu können. Durch die hierarchische Staffelung von groben und feineren Kollisionstests stellt dieses Modell einen guten Kompromiß zwischen erforderlichem Rechenaufwand auf der einen Seite und Qualität der räumlichen Auflösung auf der anderen Seite dar, und es erlaubt eine befriedigende physikalisch basierte Interaktion mit den virtuellen Gegenständen des Anwendungsszenarios.

2. Vergleich der verschiedenen Objektmodelle

Im diesem Abschnitt soll zunächst ein Qualitätsvergleich für verschiedene Objektmodelle angestellt werden.
Allgemein gilt, daß mit der Steigerung der räumlichen Auflösung der Modelle auch der Berechnungsaufwand für den entsprechenden Kollisionstest steigt. Die Güte eines Objektmodells muß auf das zugrundeliegende Objekt bezogen werden.
Ein bezogen auf räumliche Maßstäbe gutes Modell gibt dabei die *Struktur des Objektes* möglichst gut wieder, während bezogen auf Zeitaspekte vor allem in echtzeitfähigen Systemen ein gutes Modell verknüpft ist mit einem *schnellen Kollisionstest*.
Ein einfaches Maß für die Güte der räumlichen Auflösung der entwickelten Modelle ist das Modellvolumen.

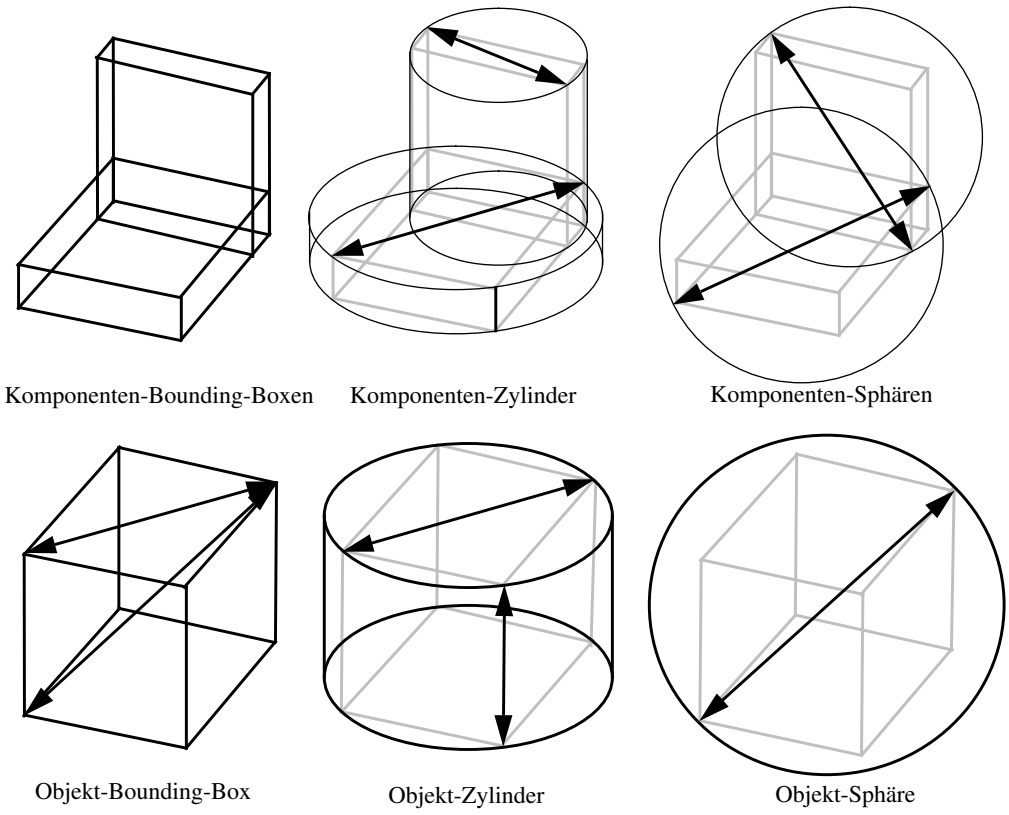


Abbildung 1: Vergleich verschiedener Objektmodelle
(hier z.B. für einen Stuhlsitz)

Je besser das Modellvolumen mit dem eigentlichen Objektvolumen übereinstimmt, desto besser ist auch das räumliche Modell. Dabei muß für Objektmodelle, die zur Kollisionsdetektion eingesetzt werden sollen, garantiert sein, daß das Modellobjekt, welches das eigentliche Objekt bei dem Kollisionstest repräsentieren soll, dieses auch vollständig umschließt. Die in dem VIENA-System verwendeten Objektmodelle werden in Abbildung 1 dargestellt. Ausgangsmodell aller in diesem System verwendeten Objektmodelle zur Kollisionsdetektion sind die Komponenten-Bounding-Boxen. Diese werden direkt aus den Polygondaten des Objektes abgeleitet. Dabei erhält jede Objekt-Komponente eine eigene umschreibende Bounding-Box (siehe Kapitel 3).
Somit stellt das Modell der Komponenten-Bounding-Boxen in dem entwickelten System die beste Annäherung an das eigentliche Objekt dar. Durch die Aufspaltung des i.a. konkaven Objektes in eine Menge von Bounding-Boxen wird automatisch eine einfache, aus konvexen Geometrieprimitiven bestehende Hülle erzeugt. Dies hat zur Folge, daß mit einfachen Geometrietests auf den einzelnen Bounding-Boxen eine Aussage über den Kollisionszustand des Gesamtobjektes möglich ist. Dies ist deswegen von großem Vorteil, da einfache Kollisionstests, die direkt auf konkaven Körpern operieren können, derzeit nicht bekannt sind.

Die Objekt-Bounding-Box eines Objektes wird durch die acht extremen Eckpunkte aller objekteigenen Komponenten-Bounding-Boxen definiert. Sie bildet eine Einhüllende der Komponenten-Bounding-Boxen. Während in dem Modell der Komponenten-Bounding-Boxen noch Informationen über objekteingeschlossene Freiräume verfügbar sind, geht diese Information auf der Hierarchiestufe der Objekt-Bounding-Boxen verloren. Es ist hier prinzipiell nicht mehr möglich, auf einzelne Objektbestandteile zuzugreifen. Betrachtet man die im VIENA-Anwendungsszenario vorgefundenen Objekte, so nehmen die Volumina der objektieingeschlossene Freiräume, dh. die Volumendifferenz von Objekt-Bounding-Box und der Summe der entsprechenden Komponenten-Bounding-Boxen, durchschnittlich mehr als 80% des Volumens der Objekt-Bounding-Box ein (Abbildung 2).

Die geringere räumliche Auflösung in dem Modell der Objekt-Bounding-Boxen ermöglicht jedoch einen schnelleren Gesamtkollisionstest, da weniger Bounding-Boxen gegeneinander auf Kollision getestet werden müssen, wobei in beiden Fällen ein Trennebenentest als elementarer Kollisionstest fungiert. Dieser Test entscheidet für eine Ebene, die durch Angabe eines Normalenvektors und eines Ebenenpunktes beschrieben wird, und einen Punkt, dessen Lage durch einen Ortsvektor gegeben, ob der Punkt sich diesseits der Ebene, auf der Ebene oder jenseits der Ebene befindet.

Dies kann leicht mittels des Skalarproduktes des Normalenvektors und des Differenzvektors der beiden Punkte entschieden werden. Somit kann bestimmt werden, ob der Testpunkt innerhalb, auf oder außerhalb der zugehörigen Bounding-Box liegt.

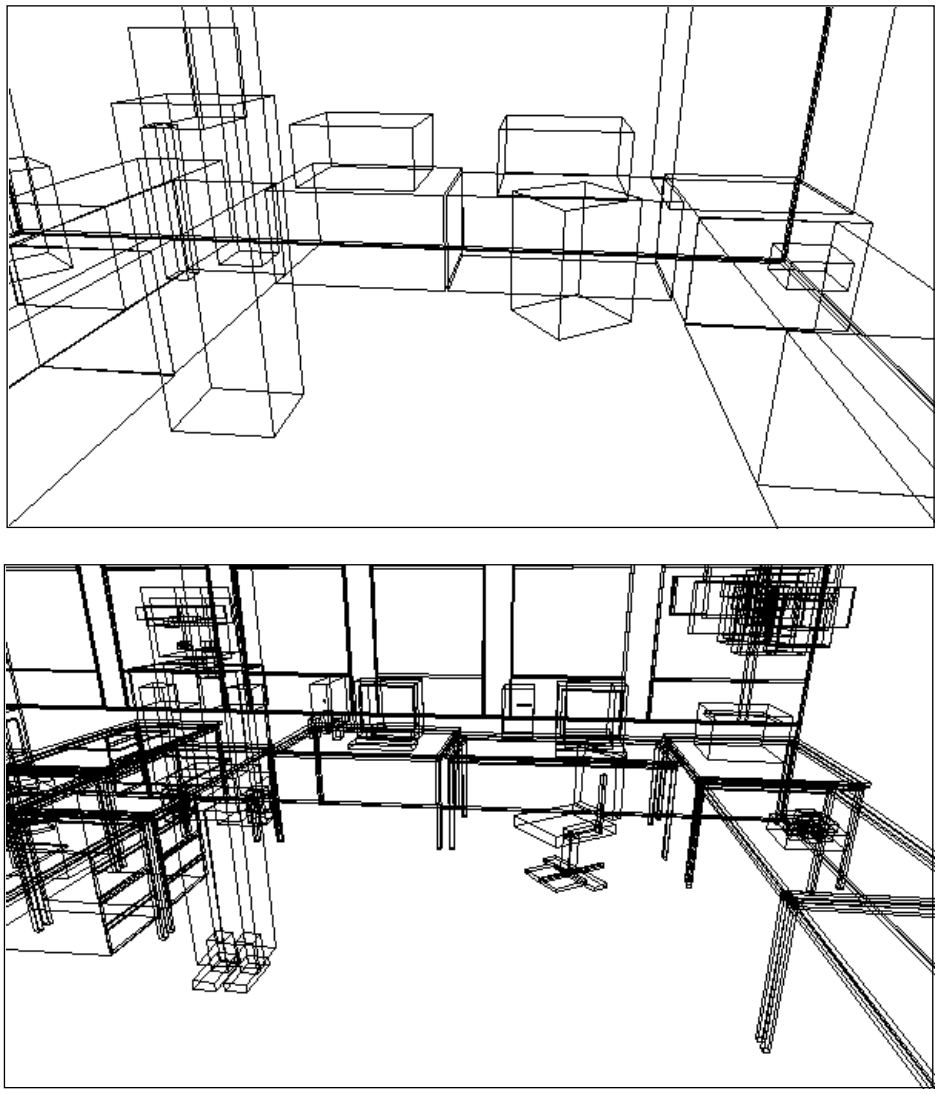


Abbildung 2: Räumliche Auflösung der Objektmodelle Objekt-Bounding-Box und Komponenten-Bounding-Box

Betrachtet man den Kollisionstest für zwei Objekte Objekt 1 und Objekt 2, wobei Objekt 1 aus n_1 Komponenten und Objekt 2 aus n_2 Komponenten zusammengesetzt sein sollen, so sind im Objektmodell der Objekt-Bounding-Boxen maximal zwei Bounding-Box-Kollisionstests durchzuführen, während im Fall der Komponenten-Bounding-Boxen schlimmstenfalls $2 \bullet n_1 \bullet n_2$ Tests vorgenommen werden müssen.
Werden in einer Szene mit m_2 Hindernisobjekten, die insgesamt aus n_2 Komponenten zusammengesetzt sind, in einer Transformation m_1 Objekte mit insgesamt n_1 Komponenten bewegt, so ergibt sich die Anzahl A der maximal benötigten Bounding-Box-Kollisionstests im Fall der Objekt-Bounding-Boxen zu

$$A = (2 \cdot m_1 \cdot m_2),$$

im Fall der Komponenten-Bounding-Boxen analog zu

$$A = (2 \cdot n_1 \cdot n_2).$$

Momentan besteht die exemplarische VIENA-Szene aus 33 Objekten, die wiederum aus insgesamt 381 Komponenten zusammengesetzt sind.
Um mit dem implementierten Trennebenentest zwei Bounding-Boxen auf Kollision zu testen, müssen im Extremfall alle acht Eckpunkte beider Bounding-Boxen gegen alle sechs Seitenflächen der jeweils anderen Bounding-Box getestet werden, jedoch nur, bis eine Trennebene für alle acht Eckpunkte der anderen Bounding-Box gefunden wird.
Somit müssen insgesamt schlimmstenfalls $6 \bullet 8 \bullet 2 = 96$ einzelne Trennebenentests für zwei Bounding-Boxen durchgeführt werden. Die Anzahl der durchzuführenden Bounding-Box-Tests kann jedoch durch die rechengünstigen und effizienten Vortests erheblich reduziert werden.
Die Objektmodelle der Komponenten-Zylinder und der Komponenten-Sphären werden aus den Daten der entsprechenden Komponenten-Bounding-Boxen abgeleitet. Aufgrund der Geometrie bzw. der Größe einiger Objekt-Komponenten des aktuellen Szenarios eignen sich diese Modelle nicht zur Ermittlung kollisionsloser Objektpositionen. Dieses gilt für die Objekt-Zylinder und die Objekt-Sphären entsprechend. Dabei stellen die Zylinder gegenüber den Sphären bez. der räumlichen Objektauflösung die besseren Modelle dar.
Im Fall der Zylinder und Sphären ist es ausreichend, sich auf einen Kollisionstest pro Primitivenpaar zu beschränken, während im Fall der Bounding-Boxen maximal zwei Tests zur Klassifikation notwendig sind.

3. Die Berechnung der Bounding-Boxen

Die Berechnung der Geometriedaten der Bounding-Boxen für die Kollisionstests wird von einem Anpasser übernommen. Dieser nimmt in der Agentenstruktur des VIENA-Systems eine besondere Rolle ein, da er als Schnittstelle zum Graphiksystem fungiert und somit als Bindeglied zwischen dem technischem System zur Graphikausgabe und dem Agentensystem angesehen werden kann. Der Anpasser ist auf das Graphiksystem GL zuschnitten. Er analysiert die von dem Graphiksystem SOFTIMAGE erzeugten Szenendaten, die nach einer Konvertierung in einer Datei als strukturierter Baum im ASCII-Format vorliegen. Die einzelnen Objekte sind dabei in einer Boundary-Repräsentation dargestellt.
Bei der Analyse der Szenendatei wird der Objektbaum traversiert, und durch Kombination von Klammerverwaltung und Detektion von Schlüsselwörtern wird die Struktur der Szene erkannt. Bei diesem Vorgang werden die Polygondaten, welche die Objekte beschreiben, und die Angaben für Translation, Rotation und Skalierung der Objekte in eine eigene Datenstruktur übernommen. Diese Datenstruktur wird zur Visualisierung der Szene benutzt. Andere Informationen, die ebenfalls für den Umgang mit der Szene benötigt werden, wie Objektnamen, Farbeigenschaften, Beleuchtung der Szene und Kameradaten werden ebenfalls in der Struktur vermerkt.
Diese Visualisierungs-Datenstruktur wird ebenfalls bei der Berechnung der Bounding-Boxen verwendet. Sie wird durchsucht, und für jedes Modell, welches in der Struktur gefunden wird und eigene Geometriedaten besitzt, wird ein Minimum-Maximum-Test durchgeführt. Bei diesem Test werden die Koordinaten aller Polygonpunkte nacheinander untersucht und die in dem entsprechenden Modell jeweils (lokal) minimalen und maximalen Koordinatenkomponenten $(x_{min},y_{min},z_{min})$ und $(x_{max},y_{max},z_{max})$ in der Struktur gespeichert. Sie dienen als Eckpunkte der Bounding-Box, die dieses Modell repräsentiert. Ein Modell kann in diesem Zusammenhang entweder eine Objektkomponente (ein Blatt der Baumstruktur) oder ein aus Komponenten zusammengesetztes Gesamtobjekt (ein Sub-Baum des Szenenbaumes) sein.
Die Vorgehensweise bei dem hier beschriebenen Vorgehen entspricht prinzipiell dem Generieren der Objekt-Bounding-Box aus den Daten der zugehörigen Komponenten-Bounding-Boxen, das weiter unten verdeutlicht wird (siehe Abbildung 4).
Bei dem beschriebenen Verfahren wird davon ausgegangen, daß sich nur in den Blättern des Baumes eigene Polygondaten befinden. Dies ist bei der Konstruktion von neuen Szenenobjekten zu beachten und kann in dem Programm SOFTIMAGE, welches zur Objektmodellierung benutzt wurde, mit Hilfe der Anzeigeoption „Schema“ überprüft und gegebenenfalls korrigiert werden.

Polygondaten, die sich nicht in Blättern des Baumes befinden, werden bei der Berechnung der Bounding-Boxen nicht berücksichtigt.

Die eben genannte Forderung stellt aber keine Einschränkung bei der Konstruktion von Objekten dar, da diese i.a. bottom-up von Raumprimitiven und Teilobjekten ausgehend durch Verknüpfung konstruiert werden, so daß die geforderte Struktur bei dem Konstruktionsvorgang automatisch entsteht. Sollte es dennoch geschehen, daß nichtterminalen Knoten des Baumes eigene Polygondaten zugeordnet sind, so kann in SOFTIMAGE mit Hilfe der Option *Schema* direkt in die Baumhierarchie eingegriffen werden.
Des weiteren ist bei der Konstruktion neuer Objekte darauf zu achten, daß die körpereigenen Koordinatensysteme korrekt in das Weltkoordinatensystem des VIENA-Szenarios integriert werden.
Das neu konstruierte Objekt ist in den Ursprung der Gesamtszene zu bewegen, um objekteigenes Koordinatensystem und Weltkoordinatensystem zur Deckung zu bringen.
In diesem Zustand muß auf das neue Szenenobjekt die SOFTIMAGE-Operation *Freeze* angewendet werden. Dadurch werden alle objektinternen (komponentenspezifischen) Translationen und Rotationen auf Null geeicht und alle objektinternen Skalierungen auf Eins gesetzt, ohne daß sich die Gestalt des Objektes verändert.

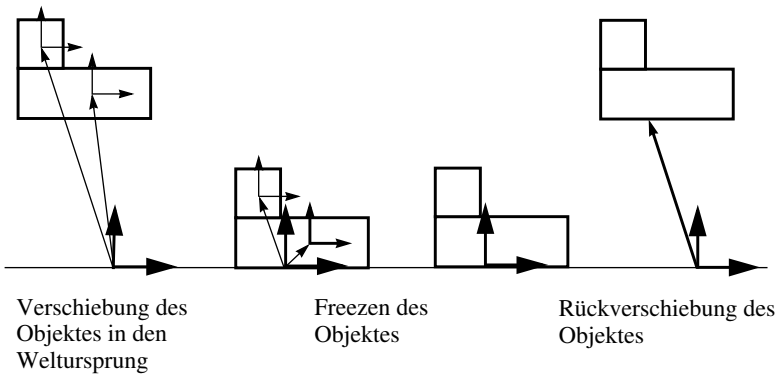


Abbildung 3: Freezen eines Objektes am Ursprung des Weltkoordinatensystems

Die Darstellung des meist aus verschiedenen Komponenten bestehenden Objektes, welches bis jetzt für jedes Unterobjekt ein eigenes Koordinatensystem besessen hat, bezieht sich nun auf den Ursprung des Weltkoordinatensystems. Die Daten aller Objektkomponenten beziehen sich ebenfalls auf diesen Nullpunkt.
Nun kann das Objekt zu seinem Standort in der Szene gebracht werden. Abbildung 3 soll diesen Vorgang veranschaulichen.

Die neue Objektposition wird durch die Vektoren *Rotation* und *Translation* bezüglich des Ursprungs des Weltkoordinatensystems angegeben. Durch die Darstellung der Komponentendaten und Objektdaten in einem einzigen Koordinatensystem ist gewährleistet, daß Änderungen bez. Translation, Rotation und Skalierung für jedes Objekt nur für das Objekt selber, nicht aber für alle Komponenten des Objektes einzeln verwaltet werden müssen, da diese für Objekt und Komponenten gleich sind.

Durch die entsprechenden Einträge für das Gesamtobjekt kann direkt auf Lage, Orientierung und Abmessung jeder Objektkomponente geschlossen werden, ohne daß die entsprechenden Daten der Vektoren für die komponenteneigene Koordinatensysteme transformiert werden müssen. Dadurch vereinfacht sich die Verwaltung der Objekte.

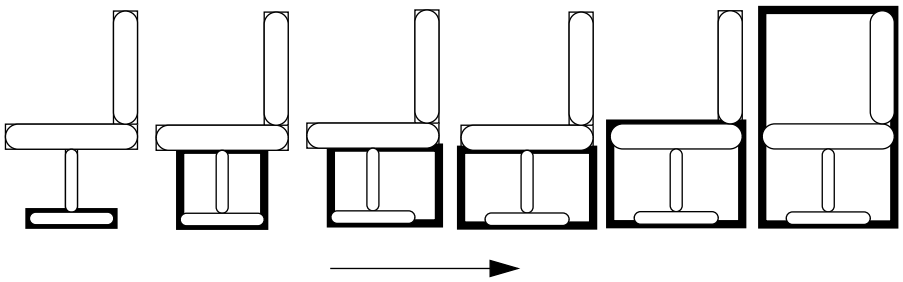


Abbildung 4: Expansion der Komponenten-Bounding-Boxen zu der Objekt-Bounding-Box

Für die Berechnung der Objekt-Bounding-Box eines Modells werden in einem Minimum-Maximumtest die Polygonkoordinaten aller zu dem Modell gehörenden Submodelle (Komponenten) betrachtet (Abbildung 4). Dabei wird die jeweilige Bounding-Box für jedes Objekt im Verlauf der Berechnung gleichsam expandiert, bis sie das Gesamtobjekt einschließt. Die extremen Koordinatenkomponenten über alle Submodelle dienen also als Eckpunkte der Objekt-Bounding-Box des betrachteten Modells. Diese Vorgehensweise ist analog zu der Berechnung der Komponenten-Bounding-Boxen aus den Polygondaten der Graphikmodelle.

Die Speicherung der Daten, die bei der Zusammenfassung einzelner Submodelle (z.B. bei dem Modell Stuhl die Komponenten Sitzfläche und Rückenlehne) zu einer gemeinsamen "maximalen" Bounding-Box entstehen, ist nicht unbedingt sinnvoll, da die entstehende Struktur abhängig von der Anordnung der Modelle in dem Szenenbaum ist. Die Hierarchie dieser "Zwischenboxen" ist nicht klar strukturiert und somit für eine physikalisch basierte Transformationsrealisation unbrauchbar.

Die berechneten Bounding-Box-Daten werden in die Struktur einer Kommunikationsliste umgewandelt und zur Verwaltung an den Buchhalter gesendet.
Die momentane Szenenbeschreibung enthält wie erwähnt 33 Objekte, die insgesamt aus 381 Komponenten zusammengesetzt sind. Um den Kommunikationsaufwand gering zu halten, werden aus diesem Grund dem Buchhalter die Informationen über die Komponenten-Bounding-Boxen nicht mitgeteilt, sondern nur die Daten über Kameraposition, Lichtpositionen und die Koordinaten der Objekt-Bounding-Boxen der Objekte. Die Informationen über Komponenten-Bounding-Boxen werden nur der Raumagentur mitgeteilt. Sie werden von den in dem VIENA-System vorhandenen Raumagenten benötigt, um die von ihnen geforderten Szenentransformationen zu berechnen. Die einzelnen Raumagenten fordern deshalb jeweils zu Beginn jeder Sitzung einmalig eine Kommunikationsliste mit allen Bounding-Box-Informationen vom Anpasser an. Die Daten werden von den einzelnen Raumagenten in einer statischen Datenstruktur abgelegt.
Zur Berechnung der aktuellen Objektdaten wird vor jeder Transformation eine Liste mit den Daten der Objekt-Bounding-Boxen der aktuellen Szene vom Buchhalter angefordert. Mit Hilfe dieser Daten werden dann die Positionen der Komponenten-Bounding-Boxen der aktuellen Szene ermittelt. Dies ist möglich, da die Vektoren für Translation, Rotation und Skalierung für die Komponenten-Bounding-Boxen und die Bounding-Box des entsprechenden Objektes gleich sind.
Die Koordinaten der Bounding-Boxen können zur Generierung von anderen Grundprimitiven wie Zylindern und Kugeln benutzt werden, die dann als Hilfsmodelle für eine schnelle Kollisionsdetektion eingesetzt werden können.
Die Generierung der Bounding-Boxen im Anpasser ist systembezogen und momentan auf eine mit SOFTIMAGE modellierte und in das ASCII-Format konvertierte Szenenbeschreibung zugeschnitten. Bei einem Systemwechsel müßten die im Anpasser realisierten Berechnungsverfahren für die umgebenden Volumina modifiziert werden.

Die anderen Agenten sollen dabei von der Wahl des unterliegenden technischen Systems unabhängig sein. Aus diesem Grunde bauen die anderen Agenten autonome, vom technischen System unabhängige Datenstrukturen auf. Bei einem Wechsel des Modellers muß also nur der Anpasser modifiziert werden, der dann mit dem restlichen System durch die unveränderten internen Strukturen des VIENA-Systems interagiert.

Aus diesem Grunde erfolgt die Berechnung der aus den Bounding-Box-Daten abgeleiteten Größen im eigentlichen Raumagenten. Dadurch wird gewährleistet, daß der Raumagent weitgehend unabhängig vom Modeller arbeiten kann, und der

Kommunikationsaufwand wird reduziert. Dabei kommt es jedoch nicht zu Inkonsistenzen mit den Daten der zentralen Datenbasis, da die entsprechenden Objektmodelle für jede Transformation neu aus den vom Buchhalter angeforderten aktuellen Daten der Szenenbeschreibung generiert werden.
Da die meisten Modeller Objektdaten in Form eines strukturierten Baumes liefern, können die Grundalgorithmen zur Bounding-Box-Berechnung übernommen werden, während die Anpasserkomponente, die für das Parsen der Szenenbeschreibung verantwortlich ist, stark modifiziert werden muß.
Bei einem Wechsel des darstellenden Graphiksystems (momentan GL) müssen hier ebenfalls nur Änderungen im Anpasser vorgenommen werden, während das restliche Agentensystem von dem Systemwechsel unberührt bleibt.

Nachdem die verschiedenen Objektmodelle und die Berechnung der Bounding-Boxen durch den Anpasser dargestellt wurden, sollen im folgenden Kapitel nun die Vorgänge, die zur Realisation eines Transformationsauftrages im VIENA-System unter Berücksichtigung der verfeinerten Kollisionsvermeidung notwendig sind, besprochen werden.

4. Ablauf der Transformationsrealisation

4.1 Die Initialisierungsphase

Der Raumagent benötigt die Information, mit welchen Objektmodellen er Transformationen realisieren soll und welches Bezugskoordinatensystem zu wählen ist. Diese Informationen werden ihm als Aufrufparameter übergeben. Dabei sind die möglichen Parameter für die Objektmodelle:

objekt : Der Raumagent benutzt für die Realisation der Transformationen die Objekt-Bounding-Boxen.

komponente : Der Raumagent verwendet für die Realisation der Transformationen die Komponenten-Bounding-Boxen.

anytime : Der Raumagent benutzt für die Realisation der Transformationen in Abhängigkeit der ihm noch zur Verfügung stehenden Rechenzeit das Objektmodell Objekt-Bounding-Boxen und, sofern noch Zeit ist, das Modell der Komponenten-Bounding-Boxen.

Mögliche Belegungen für die Wahl des Referenzkoordinatensystems sind:

deiktisch : Die Transformationen werden bez. der Sichtweise des Benutzers, die mit der Kamera-Perspektive zusammenfällt, realisiert.

intrinsisch : Falls die zu transformierenden Objekte intrinsische Eigenschaften besitzen, so werden diese zur Instantiierung eines entsprechenden Koordinatensystems ausgenutzt.

Zum Beginn einer Sitzung fordert jeder Raumagent mit der Funktion `daten_vom_anpasser` sämtliche Daten über die Bounding-Boxen der Szene an.
Zur vollständigen Beschreibung einer Bounding-Box im VIENA-System werden folgende Angaben benötigt: Translationsvektor, Rotationsvektor, Skalierungsvektor und die Vektoren der Bounding-Box-Eckpunkte, die auf der Raumdiagonalen liegen. Da die Grundflächen aller Bounding-Boxen parallel zu der x-z-Ebene des VIENA-Raumes ausgerichtet sind, genügen diese Angaben, um die restlichen Daten der Bounding-Box herzuleiten.
Um die Bounding-Box-Daten an das eigentliche virtuelle Objekt zu koppeln, werden der Objektname und die Objektklasse mit den entsprechenden Daten gemeinsam verwaltet. Die vom Anpasser gelieferten Daten werden mit der Prozedur `einlesen_anpasserdaten` in einem statischen Speicherbereich abgelegt. Dieser Vorgang muß nur einmal beim Start des Raumagenten erfolgen. Danach ist der Raumagent bereit, Aufträge entgegenzunehmen.
Erhält die Raumagentur einen Verschiebeauftrag, so prüfen die einzelnen Agenteninstanzen den Auftragsinhalt auf syntaktische Korrektheit.
Ist der Auftrag syntaktisch korrekt, so wird von jedem Mitglied der Raumagentur ein Angebot erstellt. Zum derzeitigen Projektstand kann die Raumagentur bedingt durch die unterschiedlichen Referenzsysteme und die verschiedenen Berechnungsmodelle aus maximal sechs unterschiedlichen Agenteninstanzen bestehen. Der von Karsten Otto implementierte Raumagent (Otto, 1994) kann nicht ohne weiteres verwendet werden, da auf Komponenten aufliegende Objekte nicht als solche klassifiziert werden. Eine entsprechende Anpassung ist jedoch möglich.
Der Koordinator stellt mittels der im Angebot aufgeführten Vertrauensparameter fest, welcher Raumagent im bisherigen Verlauf der Sitzung am erfolgreichsten gearbeitet hat und erteilt diesem den Zuschlag. Somit ist immer jeweils nur ein Raumagent aktiv.
Um den Auftrag bearbeiten zu können, fordert der aktive Raumagent die aktuellen Daten über Objekt-Bounding-Boxen und Kamerasicht an, da aus Konsistenzgründen das aktuelle Szenenwissen nur im Buchhalter gespeichert werden darf. Die Anforderung der Daten erfolgt mit dem Aufruf der Funktion `daten_vom_buchhalter`.

Nach dem Empfang der Daten wird eine temporäre Speicherstruktur mit allen Objektdaten aufgebaut. Dieses geschieht in der Funktion `buchhalterdaten_einlesen`. Dabei werden die aktuellen Daten der Objekt-Bounding-Boxen vom Buchhalter zum Aktualisieren der zu Sitzungsbeginn vom Anpasser angeforderten Bounding-Box-Daten über die Objekte und Komponenten benutzt. Dies ist leicht möglich, weil die Vektoren für Rotation, Translation und Skalierung für Komponenten und Gesamtobjekt gleich sind, da sie auf einen gemeinsamen Referenzpunkt bezogen sind. Die Verknüpfung der Daten geschieht mit Hilfe der Funktion `erstellen_komponentenliste`. Als Resultat entsteht eine verkettete Liste mit Objekten und dazugehörigen Unter-objekten (Komponenten), deren Bounding-Box-Daten in Weltkoordinaten vorliegen.

Die Szenenobjekte werden über den Zeiger `next` mit dem jeweiligen Nachfolger verknüpft, die Verbindung der Objekte zu den Komponenten wird über den Zeiger `sub` realisiert. Je nach Auftrag werden eine Objektliste für das zu transformierende Objekt (ref1), je eine Liste für die in dem Befehl evtl. angegebenen Referenzobjekte (ref2, ref3) und eine Liste aller restlichen Objekte (koll_liste), die potentielle Kollisionspartner für das zu transformierende Objekt darstellen, erstellt.
Die angegebenen Referenzobjekte sind für die Ermittlung der Richtung der gewünschten Transformation erforderlich. Da sie ebenfalls potentielle Kollisionsobjekte sein können, werden sie an die Liste möglicher Kollisionsobjekte angehängt.
Nach dem Aufbau der aktuellen Szenenrepräsentation ist der Raumagent nun bereit, mit der eigentlichen Transformationsberechnung zu beginnen.

4.2 Berechnung der Transformation

Im ersten Schritt der Berechnung wird anhand der Funktion `support_test` geprüft, ob einem Objekt durch die Ausführung des Auftrages die Unterstützung entzogen wird. Ist dies der Fall, so ist die Transformation nicht möglich, und es wird eine entsprechende Fehlermeldung generiert.
Falls sich auf dem zu bewegenden Objekt andere Objekte befinden, so müssen diese mittransformiert werden. Mit der Funktion `support` wird die Liste möglicher Kollisionsobjekte nach aufliegenden Objekten durchsucht. Da es vorkommen kann, daß in einer Sitzung Objekttransformationen sowohl mit dem Objektmodell der Komponenten-Bounding-Boxen, als auch mit den Objekt-Bounding-Boxen realisiert wurden, müssen aufliegende Objekte gemäß beider Objektmodelle erkannt werden, um Inkonsistenzen zu vermeiden.

Ein Objekt 1 befindet sich auf einem anderen Objekt 2, wenn die Höhe der Objektunterseite von Objekt 1 mit der Höhe der Objektoberseite der Objekt-Bounding-Box von Objekt 2 oder einer Oberseite einer Komponenten-Bounding-Boxen des Objektes zusammenfällt und sich der Schwerpunkt des Objektes 1 über dem Objekt 2 befindet. Für das zweite Kriterium wird getestet, ob sich der Objektschwerpunkt von Objekt 1 auf den Innenseiten der vier Seitenflächen von Objekt 2 befindet.
Werden aufliegende Objekte gefunden, so werden sie aus dieser Liste entfernt, da sie für das zu bewegende Objekt keine Kollisionspartner darstellen können, und dem zu bewegenden Objekt als Nachfolgeobjekte angehängt.
Dieses Verfahren kann angewandt werden, da es sich bei den Objekten nicht um veränderliche Objekte handelt, bei denen sich die Anordnung der objekteigenen Komponenten verändern kann. Im Fall veränderlicher Objekte müßte auch ein Kollisionstest zwischen den objekteigenen Komponenten und den aufliegenden Objekten durchgeführt werden.
Zu diesem Zeitpunkt der Berechnung liegen alle erforderlichen Informationen über die Szene vor.
In Abhängigkeit von der gewünschten Transformationsart wird nun mit der eigentlichen Transformationsberechnung begonnen. Dazu wurden verschiedene Funktionen entwickelt, die gemäß des Auftragsinhalts versuchen, mögliche Zielpositionen zu instantiieren (Otto, 1994).
Um zu garantieren, daß nur physikalisch sinnvolle Zielpositionen als Ergebnis der Transformation realisiert werden, müssen die möglichen Zielpositionen durch einen Kollisionstest verifiziert werden. Der in (Otto, 1994) beschriebene Raumagent realisiert diese Tests für eine speziell konstruierte Bounding-Box, die das zu transformierende Objekt und die aufliegenden Objekte umschließt. Durch dieses Verfahren entsteht unter Umständen eine sehr große Gesamt-Bounding-Box, die den von den eigentlichen Objekten eingenommenen Raum nur sehr schlecht beschreibt.
Dies kann dazu führen, daß Transformationen nicht physikalisch sinnvoll realisiert werden. Wird beispielsweise Objekt 1 auf Objekt 2 gestellt und Objekt 2 danach nach links verschoben, so ist eine anschließende Verschiebung nach rechts evtl. nicht möglich, da für diese Transformation eine gemeinsame Bounding-Box für beide Objekte berechnet wird, die weit größer sein kann, als die Bounding-Box von Objekt 2, die für die Realisation der Anfangslage von Objekt 2 verantwortlich war (siehe Abbildung 5).

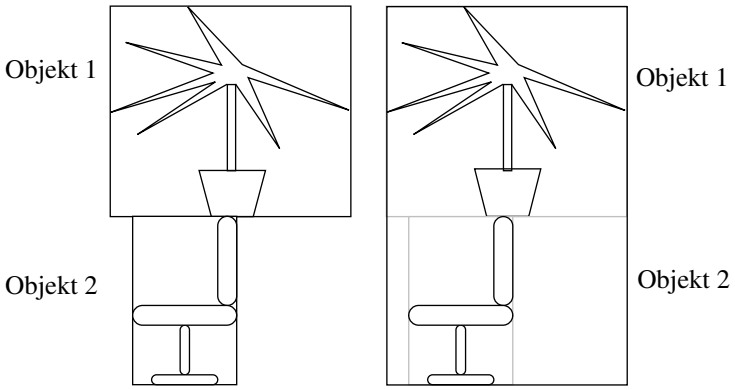


Abbildung 5: Vergleich von Objekt-Bounding-Boxen (links)
und Mehr-Objekt-Bounding-Boxen (rechts)

Um dieses unerwünschte Verhalten zu vermeiden, stehen dem Raumagenten zwei erweiterte Rechenmodelle zur Verfügung, die Objekt-Bounding-Boxen und die Komponenten-Bounding-Boxen, welche die räumlichen Eigenschaften der Objekte besser approximieren.

Der Ablauf der Berechnung für die Kollisionsvermeidung wird durch die Aufrufparameter gesteuert. Im Fall der *Objekt-Bounding-Boxen* bzw. der *Komponenten-Bounding-Boxen* als Arbeitsmodell wird ausschließlich das angegebene Modell zur Transformationsrealisation verwendet. Dabei dienen jeweils die Objektmodelle niedrigerer Granularitätsstufe als günstige Vortests, um die Menge der potentiellen Kollisionsobjekte für die aufwendigeren nachgeschalteten Kollisionstests zu reduzieren.

Für den Fall des *anytime-fähigen Arbeitsmodells* wird im ersten Schritt versucht, mit Hilfe des Modelles der Objekt-Bounding-Boxen eine Lösung für die gestellte Aufgabe zu finden. Steht nach diesem Rechenabschnitt gemäß der im Auftrag festgelegten maximal erlaubten Rechenzeit noch Zeit zur Verfügung, so wird versucht, mit Hilfe des Objektmodells der Komponenten-Bounding-Boxen eine bessere Lösung zu finden.

Die einzelnen Funktionen, die zur Berechnung potentieller Zielpositionen entwickelt wurden, sind in (Otto, 1994) beschrieben und werden auch für den hier vorgestellten Raumagenten benutzt.

Um den Agenten auf die neuen Objektmodelle umzustellen, waren weitreichende Änderungen in den programminternen Datenstrukturen erforderlich. Des weiteren wurden verschiedene neue Kollisionstests implementiert. Diese Änderungen sollen nun besprochen werden.

Vorgehensweise

Die einzelnen Funktionen, die für das Auffinden einer gemäß des Auftragsinhaltes intendierten erlaubten Zielposition zuständig sind, erhalten als Aufrufparameter das zu verschiebende Objekt mit evtl. aufliegenden Objekten, die Referenzobjekte, die Liste der potentiellen Kollisionsobjekte und die Information über das aktuell verwendete Objektmodell zur Kollisionsdetektion. Dabei sind die Referenzobjekte zur Spezifikation der vom Benutzer gewünschten Zielposition/Zielrichtung erforderlich. Gleichzeitig stellen sie potentielle Kollisionsobjekte dar.
Im ersten Arbeitsschritt wird aus der qualitativen Richtungsangabe ein konkreter Wert in Form eines Vektors berechnet. Dabei kann es sich je nach Befehlsart um einen Rotationsvektor, einen Translationsvektor oder einen Skalierungsvektor handeln. Mit Hilfe der Funktion `copy_objektliste` wird eine Kopie des zu verschiebenden Objektes einschließlich der evtl. aufliegenden Objekte angelegt. Für diese Kopie wird mit Hilfe der Funktion `aktualisieren_objektliste` die durch den ermittelten Vektor beschriebene Transformation instantiiert. Das Ergebnis dieser versuchsweisen Transformation wird agentenintern gehalten und nicht visualisiert. Für die versuchsweise transformierte Objektgruppe wird nun mit Hilfe von Kollisionstests versucht, die aktuelle Position zu verifizieren. Nur wenn durch die Transformation kein Kollisionszustand erzeugt wurde und alle bewegten Objekte durch ein anderes Szenenobjekt unterstützt werden, ist durch die Transformation eine erlaubte Objektkonstellation erzeugt worden, die als Ergebnis an den Buchhalter bzw. den Auftraggeber zurückgegeben werden kann.
Liefert der Kollisionstest als Ergebnis eine Kollisionssituation, oder wird für die Objekte keine Unterstützung gefunden, so ist die versuchsweise instantiierte Objektanordnung nicht erlaubt und muß modifiziert werden, indem ein neuer Transformationsvektor berechnet und für die Testobjektgruppe instantiiert wird.
Diese Vorgehensweise ist an das sog. Verfahren des *Step and Check* angelehnt (Baraff, 1992). Dabei wird jedoch der zeitlich-räumliche Verlauf bei der Transformationsrealisation nicht beachtet, da es im Anwendungsfall lediglich notwendig ist, eine erlaubte Zielposition zu finden. Die Bereiche des Raumes, die beim Auffinden dieser Objektanordnung überschritten werden und die i.a. keinen erlaubten Zustände darstellen, sind dabei nicht von Bedeutung.
Für den Fall der Modifikation einer Translation wird in den meisten Funktionen eine Bereichssuche orthogonal zu dem versuchsweise instantiierten Translationsvektor durchgeführt, bevor die Größe des Vektors selber verändert wird.

Eine Besonderheit ergibt sich bei der Transformation eines Objektes 1 auf ein anderes Objekt 2. Bei der Realisation dieser Transformation wird für das Objektmodell der Komponenten-Bounding-Boxen zuerst versucht, eine Zielposition auf der Objektkomponente mit der größten Oberseite zu finden. Kann dies nicht erreicht werden, so instantiiert der Raumagent eine Positionierung von Objekt 1 auf der höchsten Komponente des Objektes 2. Wenn auch dies nicht gelingt, wird versucht, eine erlaubte Zielposition bez. des Modells der Objekt-Bounding-Boxen zu finden. Somit wird erreicht, daß der neuimplementierte Agent für diese Transformation zumindest die Ergebnisse des von (Otto, 1994) entwickelten Raumagenten erbringt.

Eine weitere Besonderheit bez. des Transformationsverhaltens stellt die neuimplementierte Funktion `size` dar, die für die Größenveränderung der Objekte verantwortlich ist. Um ein möglichst universelles Verhalten sowohl bei der Objektvergrößerung als auch bei der Objektverkleinerung zu erreichen, werden vor der Objektskalierung evtl. auf dem Objekt aufliegende Objekte entfernt und nach Instantiierung einer erlaubten und durch einen Kollisionstest verifizierten Skalierung durch künstlich generierte Verschiebeanweisungen neu auf das veränderte Objekt bewegt. Erst wenn auch diese nachgeschalteten Teiltransformationen erfolgreich sind, gilt die Gesamttransformation als korrekt. Diesem Verfahren wurde aufgrund seines besseren Transformationsverhaltens bez. des Auffindens möglicher Lösungen gegenüber einer einfachen Verschiebung evtl. aufliegender Objekte in y-Richtung der Vorzug gegeben.

Die restlichen Transformationsfunktionen verhalten sich bis auf den Kollisionstest, der nun beschrieben werden soll, weitestgehend wie in (Otto, 1994) beschrieben, um die Kontinuität des Gesamtsystemverhaltens nicht zu stören.

Die eigentliche Berechnung der Kollisionstests

Die Durchführung der Kollisionstests wird durch die Funktion `überlapp` in Abhängigkeit des aktuellen Objektmodells realisiert. Eine Illustration der im folgenden beschriebenen Verfahren wird in Abbildung 6 gegeben.
Für das Objektmodell *Objekt-Bounding-Box müssen* jeweils die Objekt-Bounding-Boxen aller bewegten Objekte gegen die Objekt-Bounding-Boxen aller unbewegten Objekte getestet werden. Dieses Verhalten wird durch zwei Schleifen realisiert, die für die gegebene Objektkonstellation die einzelnen Objektpaare testet, bis eine Kollision gefunden wurde bzw. alle Objekt-Bounding-Boxen erfolgreich auf Nichtkollision getestet wurden.

Wird eine Kollision detektiert, so wird die Abarbeitung der Objektlisten unterbrochen, und der aufrufenden Funktion wird als Kollisionskennzeichen das störende Objekt zurückgegeben. Diese muß dann die Objektposition modifizieren und einen neuen Kollisionstest starten.
Bevor jedoch der Kollisionstest auf den Objekt-Bounding-Boxen berechnet wird, testet die Funktion `nicht_betroffen`, ob überhaupt eine Kollision für die Objekt-Bounding-Boxen vorliegen kann. Dazu wird ein Kollisionstest mit objektumgebenden Sphären durchgeführt, deren Durchmesser durch die Länge der Raumdiagonalen der betreffenden Objekt-Bounding-Box gegeben ist.
Nur wenn die Sphären sich durchdringen, ist eine Kollision der eingeschlossenen Objekt-Bounding-Boxen überhaupt möglich. Kollidieren die Sphären, so wird mit der Funktion `kollision_zylinder` ein zweiter Vortest durchgeführt.
Nur wenn sich auch die objekteinschließenden Zylinder, welche die Objekte besser als die Objektsphären beschreiben, durchdringen, wird der Kollisionstest für die Objekt-Bounding-Boxen durchgeführt. Durch dieses Vorgehen können bereits viele Objekte als Kollisionspartner ausgeschlossen werden, bevor der aufwendigere Bounding-Box-Test sich mit ihnen befassen muß.
Der eigentliche Bounding-Box-Test wird in der Funktion `kollision_eckpunkt` realisiert. In dieser Funktion werden für jede Seitenfläche der Bounding-Box des Objektes 1 alle Eckpunkte der Bounding-Box des Objektes 2 auf Trennung getestet. Dies geschieht durch Aufruf der Funktion `trenn_ebene`, die den in (Merkler, 1996) beschriebenen Trennebenentest realisiert.
Wird eine Ebene als Trennebene erkannt, so können die beiden Bounding-Boxen nicht kollidieren, und das nächste Bounding-Box-Paar kann getestet werden.
Wenn für ein Bounding-Box-Paar bei dem beschriebenen Test keine Trennebene gefunden werden kann, so werden in einem zweiten Schritt die Seitenflächen von Objekt 2 gegen die Eckpunkte von Objekt 1 getestet. Wird in diesem Rechenschritt eine Trennebene gefunden, so können die beiden Bounding-Boxen nicht kollidieren.
Wird in keinem der beiden Schritte eine Trennebene ermittelt, so liegt bez. des Objektmodells der Objekt-Bounding-Boxen eine Kollision vor, und der Transformationsvektor muß modifiziert werden. Kann für die beiden Objekte keine Kollision festgestellt werden, so muß mit der Funktion `is_on` getestet werden, ob für die bewegten Objekte eine Unterlage existiert. Dazu wird die Liste möglicher Kollisionsobjekte nach einem unterstützenden Objekt durchsucht. Im anderen Fall ist die Anordnung nicht erlaubt und muß modifiziert werden.

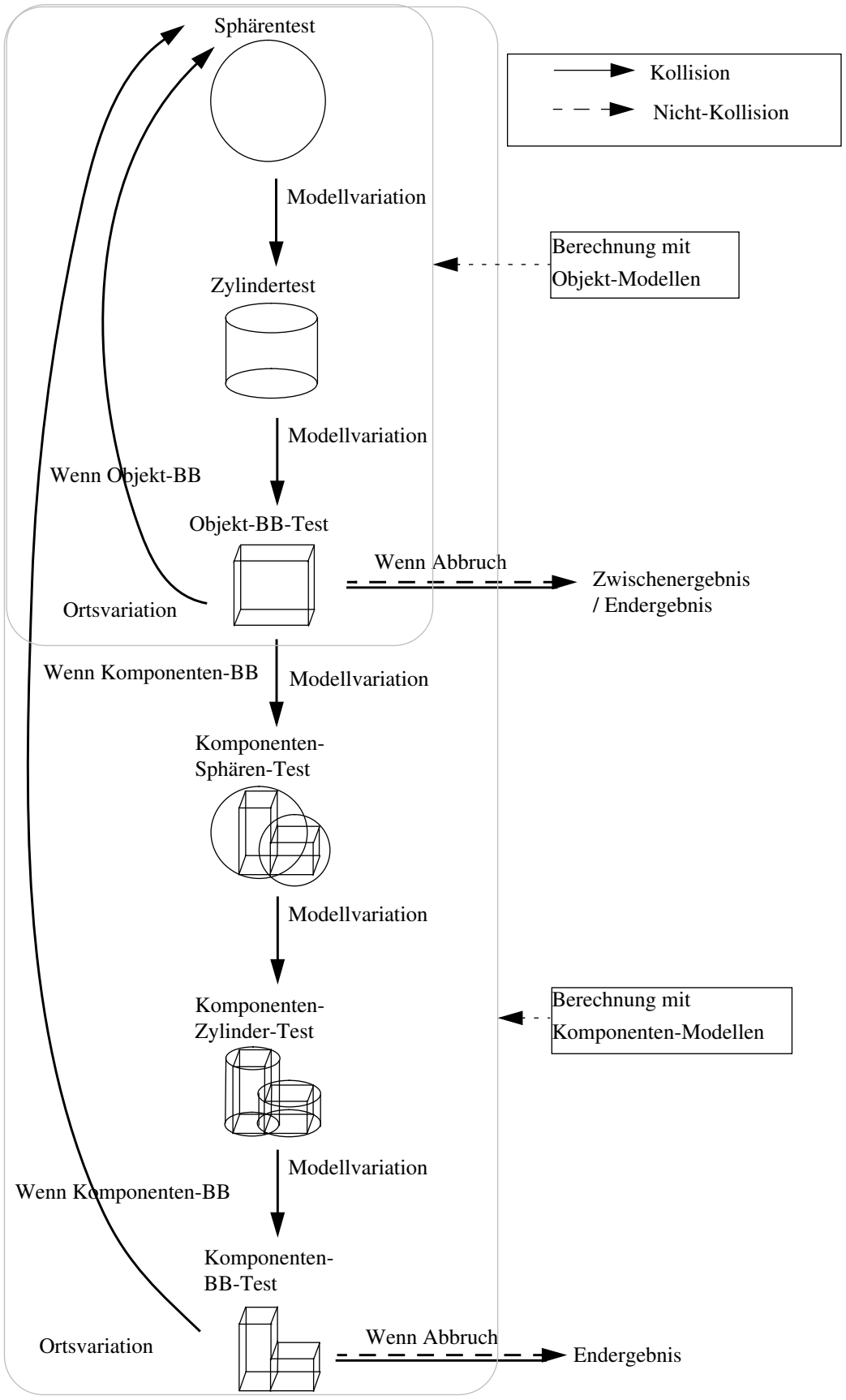


Abbildung 6: Ablauf der hierarchischen Kollisionsberechnung für das VIENA-System

Der Berechnungsverlauf für das Modell der *Komponenten-Bounding-Boxen* ist ähnlich dem eben vorgestellten. In diesem Fall müssen alle Komponenten der bewegten Objekte gegen alle Komponenten der statischen Objekte getestet werden.
Dieses Verhalten wird durch vier geschachtelte Schleifen realisiert, die jeweils alle Objekte und deren Komponenten in den entsprechenden Objektlisten durchlaufen. Der Raumagent führt einen Test bez. der Objekt-Modelle (Objekt-Sphäre, Objekt-Zylinder, Objekt-Bounding-Box) durch, bevor der Test auf den Komponenten-Modellen berechnet wird. Wird keine Kollision für die Objekt-Modelle festgestellt, so können die betrachteten Objekte auch bez. der Komponenten-Modelle nicht kollidieren.
Nur wenn für die Objekt-Modelle eine Kollision deteketiert wird, kann auch für die Komponenten-Modelle eine Kollision stattfinden. Dies wird im nächsten Arbeitsschritt überprüft.

Bevor der Kollisionstest für die Komponenten-Bounding-Boxen des betrachteten Objektpaares durchgeführt wird, werden wieder Kollisionstests bez. des Sphären-Modells und des Zylinder-Modells durchgeführt, um möglichst viele Komponenten für eine Kollision des Komponenten-Bounding-Box-Modells auszuschließen.
Dabei umschließen die Sphären und Zylinder jeweils vollständig eine einzelne Komponente. Nur wenn bez. des Komponenten-Sphären-Modells und des Komponenten-Zylinder-Modells eine Kollision detektiert wird, wird für das entsprechende Komponentenpaar der Kollisionstest für die Komponenten-Bounding-Boxen vorgenommen. Dies geschieht wiederum durch die Funktionen `kollision_eckpunkt` und `trenn_ebene`, deren Arbeitsweise oben beschrieben wurde.
Wird für eines der Komponentenpaare eine Kollision festgestellt, bewirkte der Transformationsversuch einen nicht erlaubten Zustand, der modifiziert werden muß. Wird keine Kollision detektiert, so muß für die bewegten Objekte ein unter-stützendes Objekt gesucht werden, um die Transformation vollständig zu verifizieren. Dies geschieht mit der Funktion `is_on`.
Nur wenn keine Kollision detektiert wurde und ein unterstützendes Objekt gefunden wurde, bewirkt der Transformationsversuch einen erlaubten Szenenzustand und kann dem Buchhalter als Transformationsergebnis mitgeteilt werden. Dies wird durch die Funktion `ergebnis_zum_buchhalter` realisiert.

Als Transformationsergebnis werden dem Buchhalter die berechneten Vektoren für Translation, Rotation und Skalierung für alle bewegten Objekte mitgeteilt. Diese liegen nach dem Auffinden einer erlaubten Zielposition für alle bewegten Objekte vor und werden direkt aus der Objektdatenstruktur in die Form einer VIENA-Kommunikationsliste umgewandelt.

Wurde für die geforderte Raumtransformation keine Lösung gefunden, so wird eine Fehlermeldung generiert und ggf. dem Auftraggeber mitgeteilt.
Nach Durchführung der Transformation gibt der Raumagent den dynamisch angelegten Speicher für die temporären Objektstrukturen frei und wartet auf den nächsten Auftrag.

5. Test des neuen Raumagenten

In diesem Abschnitt soll die Arbeit des Raumagenten anhand einiger Beispielinstruktionen illustriert werden. Der neuimplementierte Raumagent wird bei der Realisation zweier Beispielinstruktionen mit dem bisherigen Raumagenten verglichen. Bei diesem Test verwendet der neue Raumagent das Objektmodell der Komponenten-Bounding-Boxen, während der andere Agent die zu verschiebenden Objekte mit einer gemeinsamen Mehr-Objekt-Bounding-Box umgibt. Für den Test starteten beide Agenten mit derselben Ausgangsszene. Beide Agenten führten die Transformationen bez. des intrinsischen Referenzsystems durch. Als Beispielanweisungen wurden die beiden Instruktionen "*put the palm on the chair.*" und "*move the chair right/forward.*" verwendet. Die unterschiedliche Richtungsangaben waren erforderlich, da der neuimplementierte Raumagent bei der Verschiebung von Objekten mit intrinsischen Merkmalen das objekteigene intrinsische Koordinatensystem zur Referenzauflösung verwendet, während der bisherige Raumagent in diesem Fall die Anweisung bez. des deiktischen Koordinatensystems ausführt.
Abbildung 7 zeigt die Raumbelegung des aktuellen Anwendungsszenarios. Die Rechtecke stellen die Objekt-Bounding-Boxen der Gegenstände in Aufsicht dar.
Für das Objekt chair1 wurden durch die beiden Kreise die Ausdehnung für den entsprechenden Objekt-Zylinder (kleiner Kreis) und die umhüllende Objekt-Sphäre (großer Kreis) angedeutet.

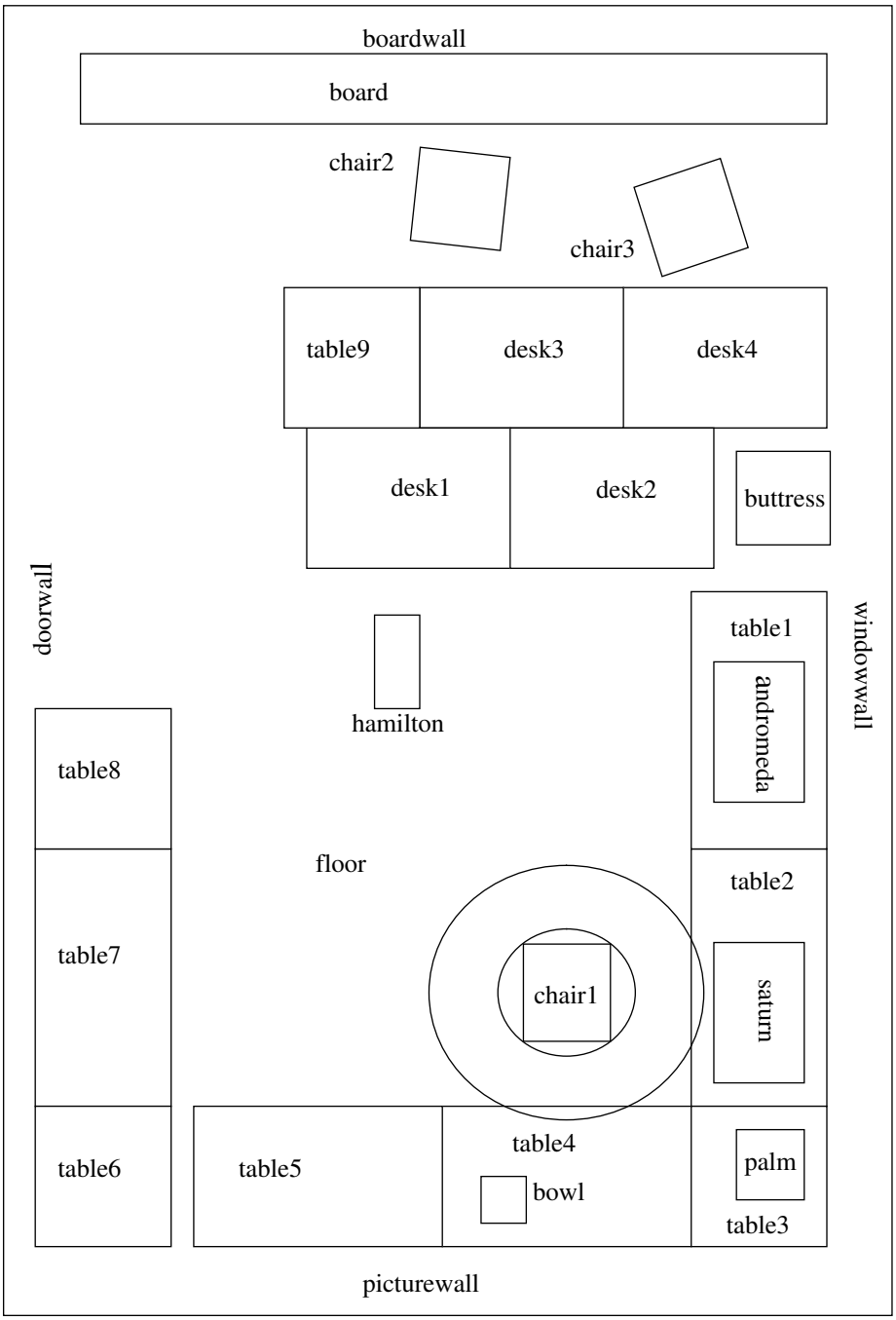


Abbildung 7: Raumbelegungsplan für das aktuelle Anwendungsszenario (Default-Startszene)

Bisheriger Raumagent

Die Beispieltransformationen für den von Karsten Otto entwickelten Agenten werden in Abbildung 8 illustriert.
Für die Anweisung "*put the palm on the chair.*" wird der in der Bildmitte sichtbare Stuhl als Zielobjekt bestimmt und gemäß des Objektmodells der Mehr-Objekt-Bounding-Boxen eine Zielposition auf der Oberseite der Stuhl-Bounding-Box ermittelt. Für den Betrachter scheint die Palme über der Sitzfläche des Stuhls zu schweben. Dieser Zustand ist jedoch gemäß dem Objektmodell der Mehr-Objekt-Bounding-Boxen erlaubt, da sich die Palme auf der Oberseite der Objekt-Bounding-Box befindet. In der nächsten Anweisung "*move the chair right.*" werden der Stuhl und die aufliegende Palme gemäß des Kamera-Koordinatensystems nach rechts verschoben. Für die Verschiebung wird für beide Objekte eine gemeinsame einhüllende Bounding-Box aus den Extremwerten beider Objekt-Bounding-Boxen generiert. Durch die Größe dieser Bounding-Box kann lediglich eine kurze Verschiebung realisiert werden, bevor eine Kollision mit der Schüssel, die sich auf dem rechten Tisch befindet, stattfindet. Die Verschiebeweite, die für das Suchen einer Zielposition verwendet wird, hängt von der Größe der gemeinsamen Bounding-Box ab. Die gefundene erlaubte Zielposition befindet sich relativ nah an der Startposition.

Neuimplementierter Raumagent

Die folgenden Beipieltransformationen werden in Abbildung 9 visualisiert.
Der neuimplementierte Raumagent verhält sich bei der Auswahl des Zielobjektes für die Anweisung *"put the palm on the chair."* analog zu dem bisherigen Raumagenten.
Für die Initialisierung einer möglichen Zielposition werden die Komponenten-Bounding-Boxen des Stuhles betrachtet. Dabei wird zuerst versucht, eine mögliche Transformation auf die Bounding-Box der Objektkomponente mit der größten Oberseite zu realisieren. Für die Beispielanweisung wird die Sitzfläche als die Komponente mit der größten Oberseite ermittelt. Durch die adäquate Auflösung der Objektmodelle wird eine erlaubte Zielposition auf der Oberseite der Sitzflächen-Bounding-Box gefunden. Für den Betrachter befindet sich die Palme direkt auf der Sitzfläche. Somit erscheint die Realisation der Verschiebeoperation in hohem Maße realistisch. Für die Anweisung *"move the chair forward."* wird das intrinsische Koordinatensystem des Stuhls als Referenzkoordinatensystem für die Ermittlung der Transformationsrichtung zugrunde gelegt. Die Palme wird als auf dem Stuhl aufliegendes Objekt erkannt, und mit ihm zusammen transformiert.

Bisheriger Raumagent (Mehr-Objekt-Bounding-Box-Modell)

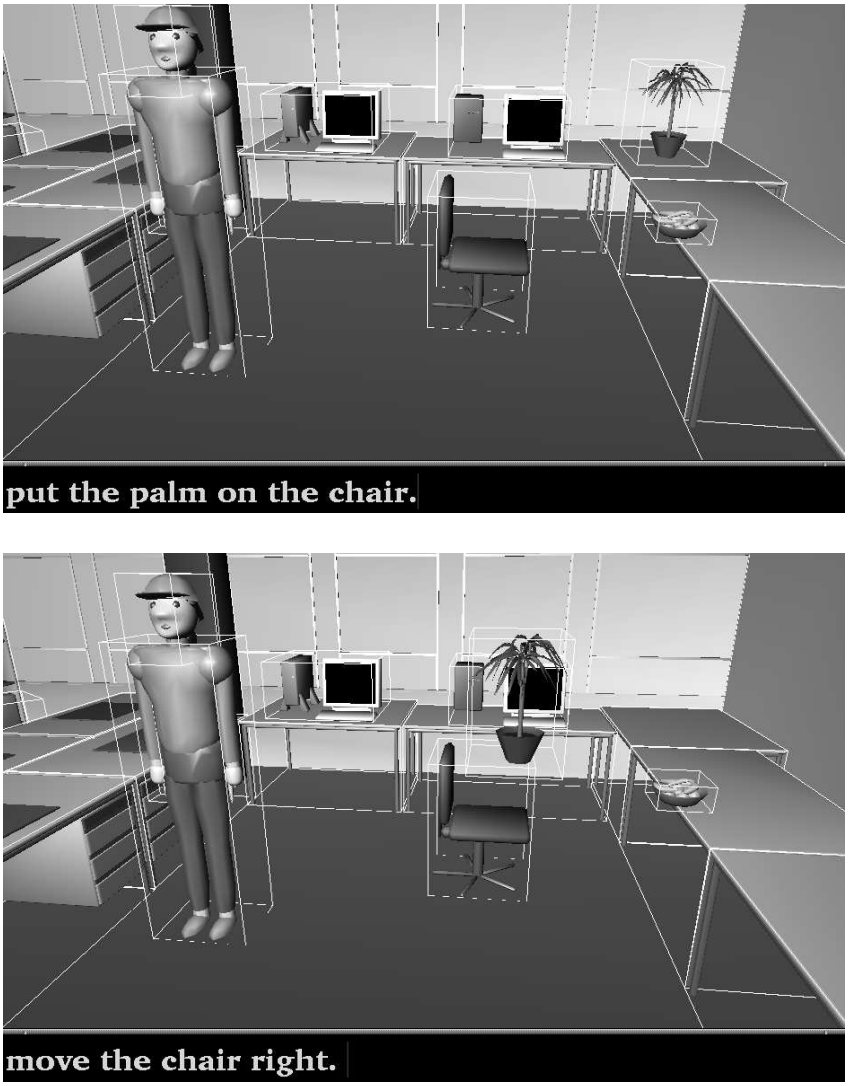
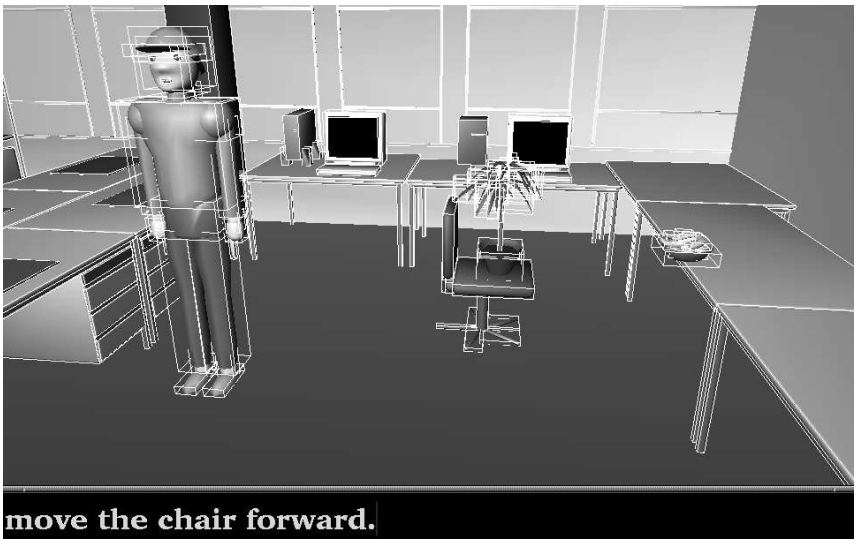


Abbildung 8: Transformationsrealisation durch den bisherigen Raumagenten
mit dem Objektmodell der Mehr-Objekt-Bounding-Boxen

Neuimplementierter Raumagent (Komponenten-Bounding-Box-Modell)



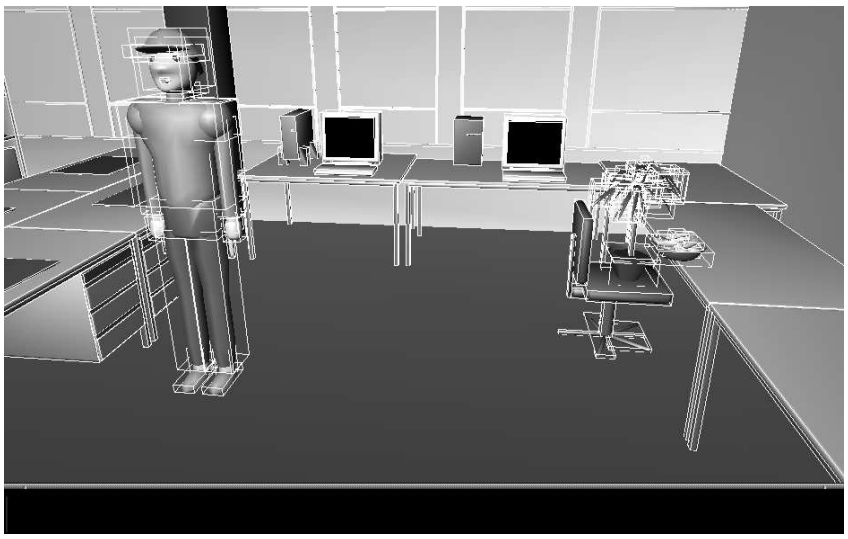


Abbildung 9: Transformationsrealisation durch den neuimplementierten Raumagenten mit dem Objektmodell der Komponenten-Bounding-Boxen

Für die Ermittlung einer erlaubten Verschiebeposition werden die Bounding-Boxen aller Komponenten der beiden zu verschiebenden Objekte betrachtet. Aufgrund der Auflösung der Struktur der beiden Objekte ist eine differenziertere Kollisionsdetektion möglich und der Stuhl kann soweit verschoben werden, daß der Blumentopf der Palme ein Stück weit unter die über die Tischkante herausragende Schüssel gestellt werden kann, bevor es zwischen dem Stamm der Palme und der Schüssel zu einer Kollision kommt.

Aufgrund der besseren Objektmodellierung ist mit dem neuen Raumagenten für die Beispielanweisungen eine bessere Transformationsrealisation möglich gewesen. Die Transformationsergebnisse erscheinen dabei physikalisch plausibler.

Laufzeitverhalten des Raumagenten mit verschiedenen Objektmodellen

Um quantitativ erfassen zu können, wie sich die verschiedenen Objektmodelle zur Transformationsrealisation auf die Zeit, die für die Durchführung einer Transformation benötigt wird, auswirken, wurde für eine Beispielinteraktion ein Laufzeittest durchgeführt. In (Merkler, 1996) werden Laufzeittests für die verschiedenen Objektmodelle ausführlich dargestellt. Nachstehend folgen einige Bemerkungen dazu.

Der Zeitvergleich für die verschiedenen Objektmodelle zeigt, daß Lesezeit und Rechenzeit im Vergleich zu der Gesamtzeit, die abhängig von der Auslastung des Computers ist, klein ausfallen. Daraus ergab sich, daß in einigen Fällen die Ergebnisse für die komplexen Objektmodelle früher vorlagen, als für das einfachste Modell der Mehr-Objekt-Bounding-Boxen.

Die benötigten Lesezeiten lagen bei den neuen Objektmodellen durchschnittlich bei 0.08 Sekunden, da jeweils die komplette Objektstruktur inkl. Komponentenstruktur aufgebaut werden muß, bei dem von Karsten Otto implementierten Mehr-Objekt-Bounding-Box-Modell bei 0.04 Sekunden.

Die Rechenzeiten lagen für das Mehr-Objekt-Bounding-Box-Modell bei 0.02 Sekunden, für das Objekt-Bounding-Box-Modell bei 0.02 Sekunden, für die Komponenten-Bounding-Boxen bei 0.05 Sekunden und für das Anytime-Modell, das sich in diesem Fall nicht auswirkt, ebenfalls bei 0.05 Sekunden. Die Rechenzeiten liegen sehr nahe zusammen, da durch die effizienten Vortests viele Bounding-Boxen auf einer frühen Berechnungssutfe als Kollisionspartner ausgeschlossen werden können.

Die Zeitmessung zeigt somit, daß die Raumagenten mit den neuen Objektmodellen geringfügig längere Rechenzeiten benötigen, aber bez. der räumlichen Auflösung bessere Ergebnisse generieren.

Die Zeit, die benötigt wird, um nach dem Start der neuen Raumagenten die Bounding-Box-Daten vom Anpasser zu erhalten, liegt im Bereich von $4 \bullet 10^{1}$ Sekunden. In diesen Initialisierungs-Zeitraum geht ein Teil der Zeit mit ein, die vom Anpasser für das Parsen der Szenen-datei und den Aufbau der internen Datenstrukturen benötigt wird.

Die benötigten Rechenzeiten für die Komponenten-Bounding-Boxen liegen aufgrund der effizienten Vortests und des lokalen Charakters einer Kollision durchschnittlich um den Faktor vier über denen für die Objekt-Bounding-Boxen, jedoch in der Größenordnung von $5 \bullet 10^{-2}$ - $5 \bullet 10^{-1}$ Sekunden für die derzeit 33 Objekte, die aus insgesamt 381 Komponenten zusammengesetzt sind. Diese Zeiten liegen deutlich unter der zur Kommunikation und zur Visualisierung der Szene benötigten Zeit, die im Bereich von etwa $5 \bullet 10^{0}$ Sekunden liegt, so daß die geringfügig längere Rechenzeit des feineren Objektmodells bei dem Test des Raumagenten bez. der Arbeit mit den verschiedenen Objekten subjektiv nicht wahrgenommen wird.

Vergleich Objekt-Bounding-Boxen / Komponenten-Bounding-Boxen

Die beiden folgenden Abbildungen zeigen denselben Szenenzustand des VIENA-Systems. Diese Objektanordnung wurde mit dem Objektmodell der Komponenten-Bounding-Boxen realisiert. Mit dem Objektmodell der Objekt-Bounding-Boxen würden für diese Anordnung Kollisionen für die Objektpaare Tisch-Stuhl, Stuhl-Computer und Schüssel-Computer detektiert werden und die Positionierung der Gegenstände in der abgebildeten Form wäre nicht möglich gewesen. Die Objekt-Bounding-Boxen wurden zur Illustration in Abbildung 10 über den eigentlichen Objekten abgebildet.

Abbildung 11 zeigt die Objekte mit den umgebenden Komponenten-Bounding-Boxen. Es ist zu erkennen, daß gemäß dieses Objektmodelles aufgrund der höheren räumlichen Auflösung für die Objektanordnung keine Kollision vorliegt.

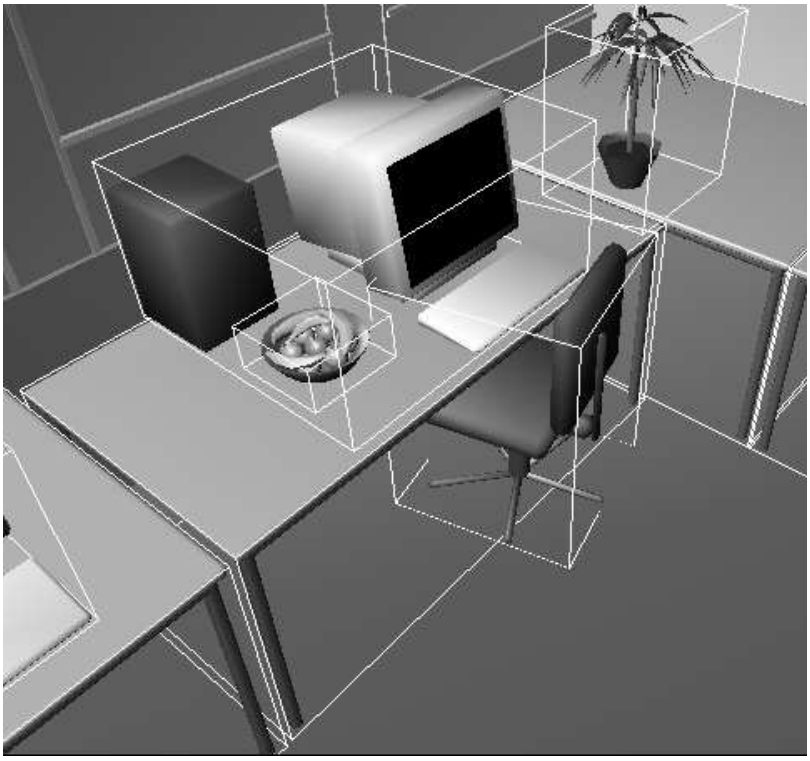


Abbildung 10: Objekte mit Objekt-Bounding-Boxen

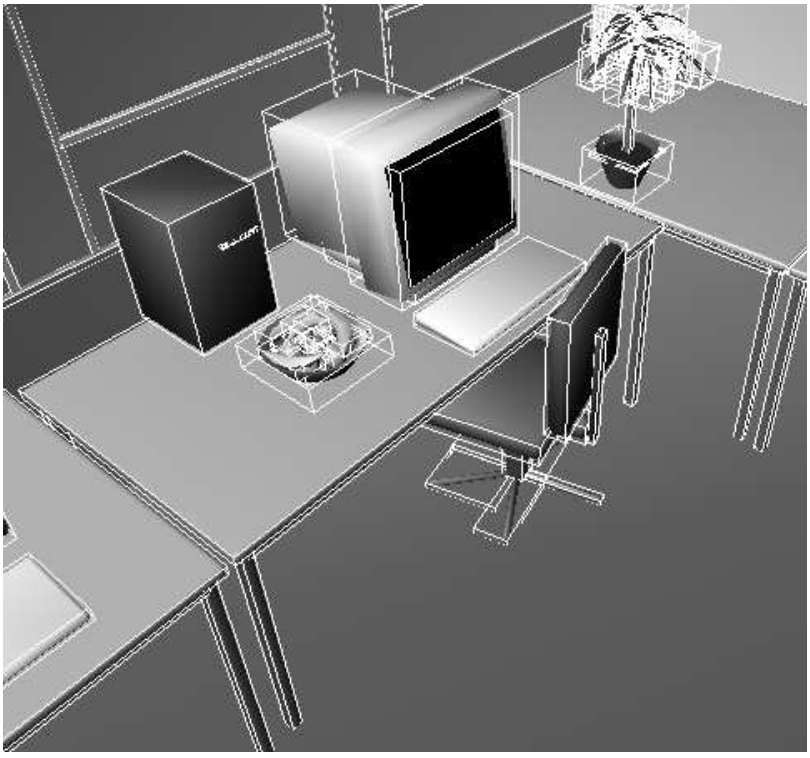


Abbildung 11: Objekte mit Komponenten-Bounding-Boxen

Illustration einiger Szenenzustände

Im folgenden sollen die Ergebnisse einiger Transformationen gezeigt werden, um die Möglichkeiten, die sich durch das neue Modell der Komponenten-Bounding-Boxen ergeben, zu illustrieren.

In Abbildung 12 befindet sich die Obstschüssel zusammen mit der Espressigo unter dem Tisch. Mit dem Objektmodell der Objekt-Bounding-Boxen wäre eine solche Anordnung nicht möglich gewesen, da in diesem Modell das Volumen unter der Tischplatte als zum Objekt Tisch zugehörig betrachtet wird und somit einen verbotenen Raumbereich darstellt.

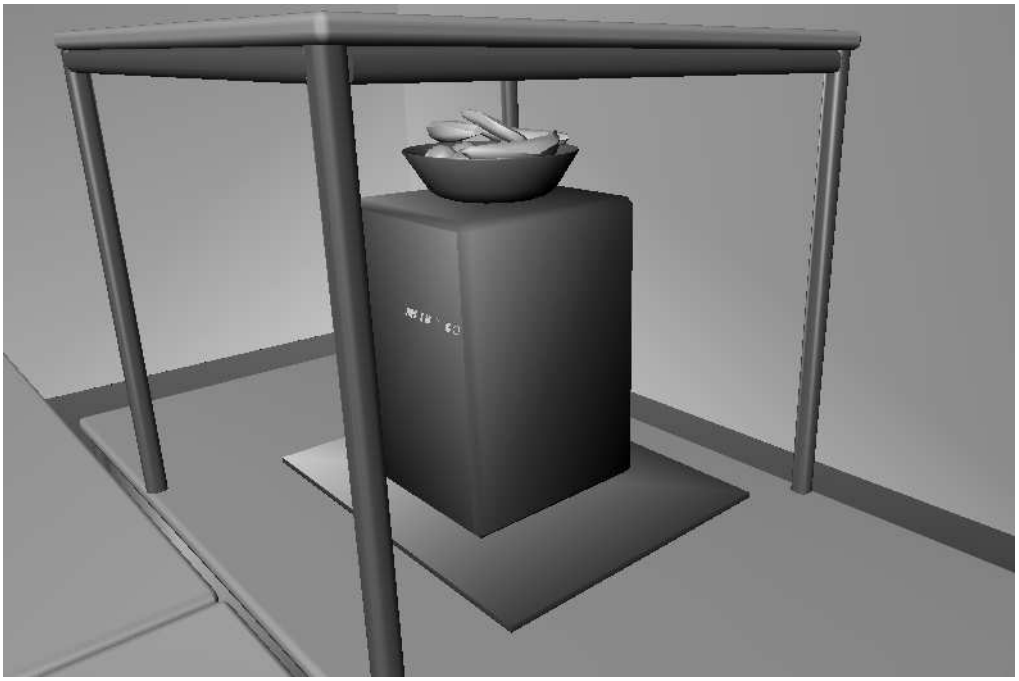


Abbildung 12: Obstschüssel mit Espressigo unter Tisch

Die folgende Abbildung 13 zeigt die Obstschüssel auf der Sitzfläche eines Stuhles, der ein kleines Stück unter einen Tisch geschoben wurde. Mit dem Modell der Objekt-Bounding-Boxen ist der Zugriff auf einzelne Objektkomponenten nicht möglich, so daß die Positionierung der Schüssel auf der Sitzfläche nicht realisierbar gewesen wäre.
Der Bereich unter dem Tisch würde gemäß dieses Modells wieder einen verbotenen Raumbereich darstellen, so daß die Verschiebung des Stuhles unter des Tisch ebenfalls nicht möglich gewesen wäre.

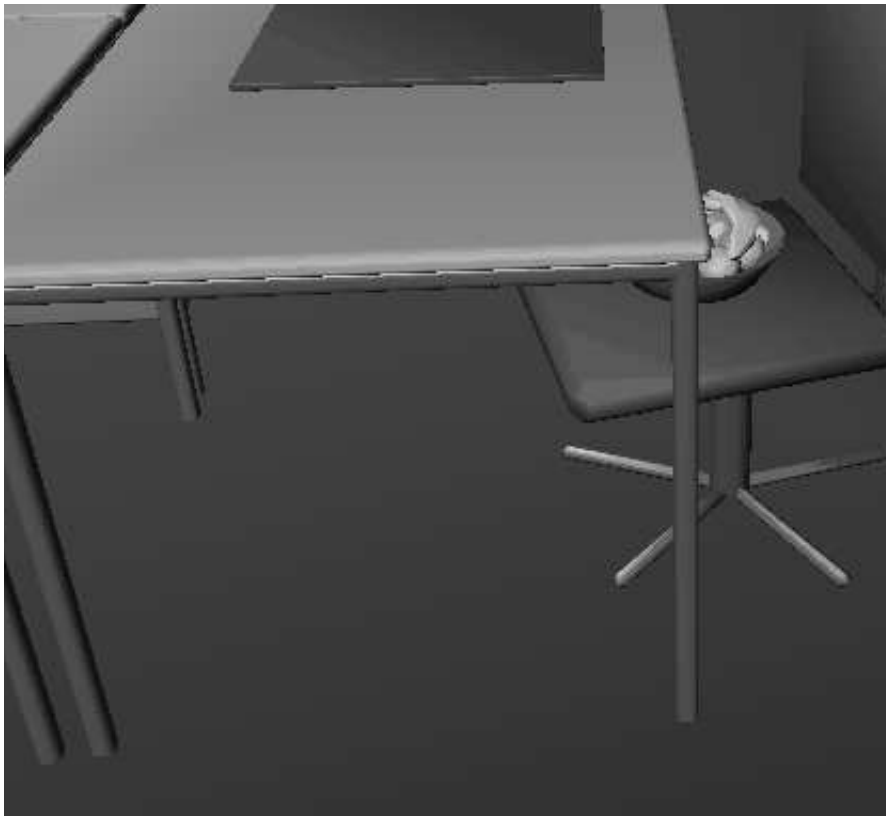


Abbildung 13: Schüssel auf Sitzfläche eines Stuhles unter einem Tisch

In den drei folgenden Abbildungen werden einige Ergebnisse der Skalierungsfunktion dargestellt. Es ist vorerst die Möglichkeit einer uniformen Vergrößerung bzw. Verkleinerung der Objekte um jeweils 25% der Ausgangsgröße realisiert worden.
Durch das Komponenten-Bounding-Box-Modell ist es prinzipiell möglich, einzelne Objektkomponenten direkt anzusprechen, wie in Abbildung 14 das linke Regalbrett. Dieses Verhalten wurde jedoch für das VIENA-System noch nicht realisiert und würde eine sinnvolle Erweiterung des Systems darstellen. Die in Abbildung 14 illustrierte Objektanordnung entstand für die Anweisung *put the chair on the board.* durch den direkten Zugriff des Agenten auf ein spezielles Regalbrett (`regalbrett3`) und soll in diesem Zusammenhang nur als Demonstration dienen.

Abbildung 14: Explorativ verkleinerte Gegenstände im Regal

Abbildung 15 und 16 illustrieren einige Objektgruppierungen mit skalierten Objekten. Auch diese Anordnungen sind nur aufgrund des verfeinerten Objektmodells möglich.

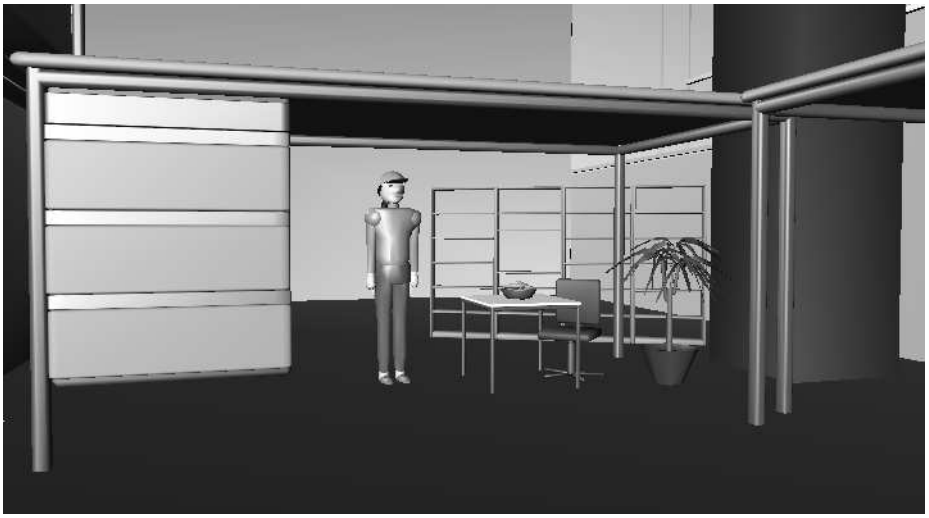


Abbildung 15: Mehrfach verkleinerte Objektgruppe unter Schreibtisch

Abbildung 16: Schreibtisch-Stapel

6. Resümee und Ausblick

Die implementierten Algorithmen zur Kollisionsdetektion erlauben eine adäquatere Manipulation der in der Szene involvierten virtuellen Objekte, wie aus den Beispielen des letzten Abschnitts deutlich wurde.
Durch das verfeinerte Objektmodell ist es ferner möglich, dem Benutzer neue Anweisungen für eine bessere Interaktion mit dem System zur Verfügung zu stellen. Mögliche Anweisungen könnten sein:

- Move Object 1 under Object 2.
- Put Object 1 in Object 2.

Die vorgestellten Verfahren wurden speziell für das VIENA-Szenario entwickelt, sie lassen sich jedoch durch entsprechende Änderungen bei der Berechnung/Aktualisierung der Objektmodelle auch für allgemeinere Szenarien einsetzen. Die Interaktionsmöglichkeiten mit dem System haben sich hierdurch erheblich verbessert, wobei das geringfügige Anwachsen der benötigten Rechenzeit nicht bemerkbar ist.

Veränderung der Szene

Eine Vergrößerung der Objektzahl in dem Anwendungsszenario wird die benötigte Rechenzeit nur geringfügig erhöhen, da es sich bei einer Kollision in den meisten Fällen um ein lokales Ereignis handelt, so daß viele Objekte bereits durch die zeitgünstigen Vortests als potentielle Kollisionspartner ausgeschlossen werden können.
Bei dem Einbringen von Objekten komplexerer Struktur werden die Kollisionstests für die einzelnen Objektpaare aufwendiger, sofern die Komponentenstruktur berücksichtigt wird, und es bietet es sich an, ein Verfahren zu implementieren, welches Komponenten-Modelle, die kleiner als eine bestimmte festzulegende Mindestgröße sind, zu einem gemeinsamen Obermodell zusammenfaßt.
Dieses Vorgehen ist jedoch für die Objekte des derzeitige Anwendungsszenarios nicht angezeigt und wurde aus diesem Grund nicht weiterverfolgt.
Die weitere Verfeinerung der räumlichen Auflösung der Objektmodelle beispielsweise durch die Einführung einer Octree-Struktur scheint zumindest für das derzeitige Anwendungsszenario nicht sinnvoll, da die räumliche Auflösung durch das Objektmodell der Komponenten-Bounding-Boxen für eine befriedigende Transformations-Realisation ausreicht.

Aspekte der wissensbasierten Kollisionsvermeidung

Die Trennung der Kollisionsvermeidungsfunktionalität von den geometrischen Kollisionstests stellt eine weitere sinnvolle Modifikation der Raumagentur dar. Diese Überlegung ist deshalb sinnvoll, da zur physikalisch realistischen Durchführung von Transformationen physikalisches Wissen erforderlich ist. Dieses liegt momentan verteilt in der Funktionalität des Raumagenten. Für ein wissensbasiertes System ist es jedoch sinnvoll, dieses Wissen als Regelsystem formuliert zu verwenden, um die Transformationsberechnung transparent zu machen. Auch für die geschickte Berechnung der eigentlichen Kollisionstests kann räumliches Wissen eingesetzt werden. So ist es z.B. wahrscheinlicher, daß große Objekte miteinander kollidieren. Es ist also sinnvoll, die Objekte bzw. die Objektkomponenten bei den Kollisionstests der Größe nach abzuarbeiten.
Ein weiterer Punkt, der beachtet werden könnte, ist, daß Objekte, deren Richtungen maximaler Ausdehnung orthogonal ausgerichtet sind, mit größerer Wahrscheinlichkeit miteinander kollidieren, als wenn diese zueinander parallel sind.
Des weiteren ist die Wahrscheinlichkeit für extreme Außenflächen eines Objektes, in eine Kollision verwickelt zu werden, größer, als für Innenflächen. Betrachtet man einen

dynamischen Vorgang, so tritt eine Kollision zuerst an den Flächen auf, deren Flächennormalen einen negativen Vektorbeitrag in Transformationsrichtung besitzen (wenn die Flächennormalen einheitlich in das Objektinnere deuten). Diese Flächen könnten mit höherer Priorität untersucht werden.

Planung der Trajektorie

Die Arbeit des Raumagenten besteht im wesentlichen darin, für eine gegebene Objektposition und eine Anweisung, die eine qualitative Richtungsangabe enthält, eine erlaubte Zielposition zu finden. Bei der Suche einer solchen Position wird die Trajektorie, auf der sich das Objekt bei der Verschiebung in die Zielposition bewegt, nicht betrachtet. Für die Animation einer solchen Bewegung könnten die Anfangsposition und die vom Raumagenten ermittelte erlaubte Zielposition einem Trajektorienplaner, der beispielsweise auf einer Potentialfeldmethode aufsetzt, als Eingabe dienen. Die Aufgabe des Trajektorienplaners wäre es dann, für die Anfangsposition und die Endposition einen erlaubten Verbindungspfad zu suchen, auf dem das zu verschiebende Objekt bewegt werden kann. Dieses System könnte auf die vom Raumagenten generierten Objektmodelle zurückgreifen und würde die Animation des auf der Trajektorie bewegten Objektes ermöglichen. Sowohl bei der Planung der Zielposition wie auch bei dem Auffinden möglicher Trajektorien stehen durch die verfeinerte Kollisionsdetektion mehr Lösungsspielräume zur Verfügung.

Literaturverzeichnis

(Baraff, 1992). Baraff, D., Dynamic Simulation of Non-Penetrating Rigid Bodies, Ph.D. Thesis, Department of Computer Science, Cornell University, March 1992.

(Merkler, 1996). Merkler, M., Anytime-fähige Kollisionsvermeidung für eine interaktive graphische Entwurfsumgebung, (Diplomarbeit), Universität Bielefeld: Technische Fakultät, 1996.

(Otto, 1994). Otto, K., Ein Raumagent für Objekttransformationen in einer virtuellen Entwurfsumgebung (Diplomarbeit), Universität Bielefeld: Technische Fakultät, 1994.

(Wachsmuth & Cao, 1995). Wachsmuth, I., Cao, Y., Interactive Graphics Design with Situated Agents. In W. Strasser & F. Wahl (eds.): Graphics and Robotics (pp. 73-85), Springer.