

Inhaltsverzeichnis

1	Einleitung	3
1.1	Motivation	3
1.2	Aufbau der Arbeit	3
2	Grundlagen	5
2.1	E-Learning	5
2.1.1	Definition	5
2.1.1.1	Medien	5
2.1.1.2	Interaktion	6
2.1.1.3	Prüfungen	6
2.1.2	Varianten des E-Learning	7
2.1.3	Einsatz von E-Learning	7
2.1.3.1	Allgemein	7
2.1.3.2	An Hochschulen	7
2.1.3.3	An der Universität Koblenz	8
2.2	Virtueller Campus Rheinland-Pfalz	8
2.2.1	Definition	8
2.2.2	Möglichkeiten der Kursgestaltung	8
3	Konzeption	9
3.1	Intention des Kurses	9
3.2	Aufbau des Kurses	10
3.3	Zielgruppenanalyse	11
3.4	Zulassung zu Prüfungen	12
3.5	Aufgabenerstellung	13
3.5.1	Generierte Aufgaben	13
3.5.2	Von Dozenten erstellte Aufgaben	13
4	Praktische Umsetzung	15
4.1	Namenskonvention	15
4.1.1	Motivation	15

4.2	Koordinatentransformationen	16
4.2.1	Motivation	16
4.2.2	Applet : Transformationsmatritzen	17
4.2.3	Implementation	19
4.2.3.1	Das Main-Paket	20
4.2.3.2	Das Gui-Paket	21
4.2.3.3	Das Exercis-Paket	25
4.3	Szenegraphen	26
4.3.1	Motivation	26
4.3.2	Applet : Szenegraph	27
4.3.3	Implementation	28
4.3.3.1	Die Mainklasse	28
4.3.3.2	Das Gui-Paket	28
4.3.3.3	Das sGraph-Paket	29
4.4	Zusammenfassung der Arbeit	32
4.5	Ausblick	32

Kapitel 1

Einleitung

1.1 Motivation

In der aktuellen Hochschullandschaft findet man hauptsächlich die Unterrichtsform des Frontalunterrichts. Das bedeutet der Lehrende hält während der Vorlesungszeit einen monologartigen Vortrag über den Unterrichtsstoff. Hierbei nehmen die Studenten nur passiv am Geschehen teil bzw. protokollieren das vom Professor Vorgetragene um es später zu Hause eigenständig nachzuarbeiten. E-Learning hingegen lässt den Lernenden aktiv Einfluss darauf nehmen, wie und in welchem Tempo er Lehrinhalte bearbeitet.

1.2 Aufbau der Arbeit

In dieser Studienarbeit soll ein E-Learning-Kurs für den Themenbereich der Computergrafik erstellt werden. Die Arbeit besteht aus drei Teilen. Neben dieser schriftlichen Ausarbeitung habe ich einen E-Learning-Kurs mit Hilfe des Learning Management Systems WebCT¹ erstellt und hierfür zwei Applets erstellt, die Übungsaufgaben zu den Themen Koordinatentransformationen und Szenegraphen erstellen. Um Zugriff auf den Kurs zu erhalten muss man sich mit seiner Kennung des virtuellen Campus Rheinland-Pfalz auf folgender Seite anmelden:

<https://vcrp5.vcrp.de/webct/logon/469933644001>

Diese Ausarbeitung ist in fünf Hauptkapitel unterteilt. Das erste Kapitel gibt eine grundlegende Einleitung in das Thema und beschreibt den Aufbau dieser Arbeit. Das darauf folgende Kapitel „Grundlagen“ definiert grundlegende Begriffe des E-Learning und bietet einen Überblick über dessen Nutzung in

¹Nach der Übernahme der Firma Blackboard Inc. ist das System in „Blackboard Learning System“ umbenannt worden

der aktuellen Hochschullandschaft im Allgemeinen und der Universität Koblenz im Speziellen. Das dritte Kapitel beschreibt die Konzeption des von mir erstellten E-Learning-Kurses. Das Kapitel „Praktische Umsetzung“ beschreibt den inhaltlichen Aufbau des Kurses und erläutert die Implementation der beiden Applets. Das letzte Kapitel enthält eine Zusammenfassung der Arbeit und zeigt mögliche weitere Arbeiten in diesem Bereich auf.

Kapitel 2

Grundlagen

2.1 E-Learning

2.1.1 Definition

Der Begriff E-Learning ist momentan noch nicht sehr eindeutig definiert worden. Man könnte ihn als ein sog. „Buzz-Wort“ bezeichnen, das jeder gerne benutzt und damit wirbt, allerdings noch niemand genau weiß was man darunter zu verstehen hat. Es sind zahlreiche Synonyme für E-Learning entstanden die sich nur geringfügig in ihrer Bedeutung unterscheiden. Zu den am häufigsten genutzten Begriffen in diesem Bereich zählen:

- Computer Based Training (CBT)
- Web Based Training (WBT)
- Blended Learning
- u.Ä.

In erster Linie versammelt sich unter dem Begriff E-Learning eine Menge von diversen, digitalen Lernanwendungen. Mit Hilfe von verschiedenen Medien wird der Lehrinhalt vermittelt. Hierbei kann der Lernende selbst das Tempo bestimmen und nicht verstandene Themen beliebig oft wiederholen. In einigen Anwendungen kann auch die Reihenfolge der bearbeiteten Themen selbst bestimmt werden.

Da E-Learning wie bereits erwähnt momentan ein Modewort ist, wird häufig auch das Bereitstellen von Lernmaterial (Skripten, Aufgaben, etc.) oder der bloße Einsatz von Computern während des Unterrichts als E-Learning bezeichnet.

2.1.1.1 Medien

Bei den eingesetzten digitalen Medien unterscheidet man zwischen statischen und dynamischen Medien.

Unter statischen Medien versteht man Medien die sich während der Anzeige nicht verändern. Hierzu zählen z.B. Texte und Bilder. Sie haben den Vorteil, dass der Lernende sie in seinem persönlichen Tempo bearbeiten kann. Wird ein Satz nicht beim ersten Lesen verstanden kann er einfach nochmal gelesen werden. Des Weiteren können statische Medien leicht ausgedruckt werden um beispielsweise unterwegs bearbeitet oder auswendig gelernt zu werden.

Zu den dynamischen Medien zählt man Videos, Animationen, Audioaufnahmen und interaktive Elemente. Durch sie können Inhalte interessanter und ansprechender gestaltet werden. Außerdem können sie als Motivationselement nach einem längeren Text oder anderen, von vielen Benutzern als anstrengend empfundenen Elementen, eingesetzt werden.

2.1.1.2 Interaktion

Einer der größten Vorteile von elektronischen Lernangeboten sind die Möglichkeiten der Interaktion zwischen dem Benutzer und dem System. Hierdurch kann sich das System genau an die Lernbedürfnisse der Benutzer anpassen.

Da der Benutzer selbst entscheidet wann er die nächste Seite angezeigt bekommen möchte, bestimmt er selbst das Lerntempo. Dadurch, dass E-Learning-Kurse meist allein bearbeitet werden steht der Benutzer nicht unter dem Druck sich an das Tempo der Lerngruppe anpassen zu müssen. Hierdurch entsteht bei stärkeren Schülern keine Langeweile durch Unterforderung und schwächere Schüler werden nicht durch den zusätzlichen Druck noch mehr belastet, wodurch sie zusätzlich demotiviert werden.

Die Interaktion erlaubt außerdem komplexe Sachverhalte leichter zu verdeutlichen. Durch Animationen, die der Benutzer selber steuern oder beeinflussen kann, können diese Schritt für Schritt erklärt werden.

2.1.1.3 Prüfungen

Die Interaktionsmöglichkeiten von E-Learning-Kursen erlauben es auch zu überprüfen ob der vermittelte Stoff vom Lernenden verstanden worden ist. Durch kurze Tests mit Inhalts- oder Verständnisfragen kann man schnell ermitteln, ob bei einem Thema noch Nachholbedarf besteht, oder man mit dem nächsten Kapitel beginnen kann. Erlernte Lösungswege bzw. Methoden können anhand von Übungsaufgaben eingeübt werden. Hierbei können die Ergebnisse oft sofort vom System überprüft werden.

Bei einigen Systemen ist die korrekte Bearbeitung dieser Prüfungen am Ende eines inhaltlichen Blocks Bedingung zur Bearbeitung des thematisch nächsten Blocks. Diese hängt auch davon ab, ob der Lehrstoff aufeinander aufbauend strukturiert ist oder es sich um viele einander nicht überschneidende Themenbereiche handelt.

Werden die Kurse online angeboten, besteht des Weiteren die Möglichkeit Prüfungen entweder als Zulassung zu schriftlichen Klausuren (meist im universitären Bereich) oder

als Bedingung zur Erlangung eines Nachweises (Zeugnis, Zertifikat o. ä.).

2.1.2 Varianten des E-Learning

Es gibt sehr unterschiedliche E-Learning-Konzepte. Die Skala reicht von reinen E-Learning-Kursen, bei denen der Lernende wie im klassischen Selbststudium allein den Stoff bearbeitet, über integriertes Lernen (sog. „Blended Learning“) bis zu Ansätzen bei denen elektronische Aufbereitung des Lehrinhalts nur als Ergänzung zum regulären Präsenzunterricht sind.

2.1.3 Einsatz von E-Learning

2.1.3.1 Allgemein

E-Learning wird sehr intensiv im Bereich von Fern- bzw. Teleschulen eingesetzt. Aber auch in normalen Bildungseinrichtungen werden immer mehr E-Learning-Konzepte integriert. Besonders im Bereich der beruflichen Weiterbildung werden viele E-Learning-Kurse entwickelt und eingesetzt.

2.1.3.2 An Hochschulen

E-Learning scheint in der deutschen Hochschullandschaft in den letzten Jahren immer mehr an Bedeutung zu erlangen. Dass Lernmaterialien wie Vorlesungsskripte, Präsentationsfolien und Aufgabenzettel den Studenten online zur Verfügung gestellt werden ist mittlerweile selbst in informatikfernen Studienrichtungen selbstverständlich geworden.

Ebenfalls immer häufiger werden Abgaben von Übungsaufgaben über das Internet. Hierzu werden meist sogenannte Learning Management Systeme (LMS)¹ eingesetzt. Der Markt für LMS ist sehr unübersichtlich, da es sehr viele Systeme in diesem Bereich gibt.

In Studiengängen mit Informatikbezug werden viele E-Learning-Angebote als Ergebnisse von Projekt- oder Abschlussarbeiten, wie z.B. dieser Studienarbeit, erstellt². Allerdings werden diese dann meist nur als zusätzliches Lernangebot auf den Vorlesungsseiten verlinkt, ohne in den universitären Lernprozess integriert zu werden.

Von Lehrkräften initiierte E-Learning-Angebote sind meist noch experimentell angelegt. Der Schwerpunkt des Interesses liegt hierbei meist auf der Akzeptanz der Lernenden.

¹Synonyme für LMS: Lernplattform, Lernmanagementsystem

²Liste von E-Learning-Angeboten siehe Anhang

2.1.3.3 An der Universität Koblenz

An der Universität Koblenz versucht das interdisziplinäre „Institut für Wissensmedien“ (IWM)³ Lehrbeauftragte der Universität bei der Realisierung von E-Learning-Angeboten zu unterstützen. Es hilft unter anderem bei der Durchführung von Onlineklausuren, der Entwicklung von E-Learning Kursen oder bei der Aufzeichnung von Veranstaltungen.

Die Universität Koblenz-Landau nutzt das LMS WebCT, welches im nächsten Kapitel näher beschrieben wird. Des Weiteren besteht die Möglichkeit ein ILIAS-System zu nutzen.

2.2 Virtueller Campus Rheinland-Pfalz

2.2.1 Definition

Der „Virtuelle Campus Rheinland-Pfalz“ (VCRP) ist eine vom Land Rheinland-Pfalz eingerichtete gemeinsame wissenschaftliche Einrichtung der Hochschulen des Bundeslandes Rheinland-Pfalz. Ihre Aufgabe ist es E-Learning-Angebote der Hochschulen, auch hochschulübergreifend, zu unterstützen bzw. zu koordinieren. Unter Anderem verwaltet und administriert sie hierzu die Installation der WebCT-Lernplattform, die von den Hochschulen in Rheinland-Pfalz genutzt wird.

2.2.2 Möglichkeiten der Kursgestaltung

Die vom VCRP angebotene Lernplattform WebCT bietet alle zur Zeit gängigen Optionen eines LMS an. Bei der Erstellung eines Kurses kann man Tools wie Tests, Kursforum, Emailfunktionen und viele Weitere einfach aktivieren.

Eine der von WebCT angebotenen Möglichkeiten zur Kursverwaltung ist die Unterteilung von Kursabschnitten in Lernmodule. Hierbei kann man angeben welche Ziele ein Modul verfolgt, welche Bedingungen erfüllt sein müssen, damit es für einen Lernenden freigeschaltet wird und welche Reihenfolge der Module vom Dozenten empfohlen wird bzw. obligatorisch ist. Der eigentliche Kursinhalt wird in Form von HTML-Seiten gespeichert. Diese können entweder in einem beliebigen Programm erstellt und anschließend in das LMS hochgeladen oder direkt in WebCT geschrieben werden. Für die zweite Option bietet die Plattform einen HTML-Editor mit minimalem Funktionsumfang, wie etwa Schriftgröße und Farbe zu verändern. Des Weiteren können Medien wie Filme, Flashanimationen, Bilder, Audiodateien und ähnliche Formate hochgeladen und eingebunden werden.

³<http://www.uni-koblenz-landau.de/koblenz/iwm/iwm>

Kapitel 3

Konzeption

3.1 Intention des Kurses

Bei der Konzeption dieses Kurses stellte sich sehr schnell die Frage für welchen Zweck er genutzt werden soll. Hierbei boten sich drei unterschiedliche Varianten an:

1. Nachbearbeitung des Stoffes (Wiederholung nach den Vorlesungen)
2. Vorbereitung auf die Vorlesungen
3. eigenständiges Lernen des Stoffes (unabhängig vom Lehrbetrieb)

Die erste Variante geht davon aus, dass der Kurs dazu dient, nach dem Hören einer Vorlesung den Unterrichtsstoff zu Hause zu wiederholen oder weiter zu vertiefen. Dies bietet die Möglichkeit komplizierte Sachverhalte noch einmal in Ruhe selbst überdenken zu können. Beispiele, die in Vorlesungen auf Grund von Zeitmangel meistens nur kurz erläutert werden, können von den Studenten noch einmal Schritt für Schritt nachvollzogen werden. Sollte ein Student an einer Vorlesung nicht teilnehmen können, bietet der Kurs außerdem die Möglichkeit das Versäumte selbständig nachzuholen. Hier zeigt sich allerdings auch ein Problem dieser Variante und des E-Learnings im Allgemeinen: da auch der gesamte Lehrstoff zu Hause bearbeitet und gelernt werden kann, stellt sich die Frage ob Präsenzveranstaltungen überhaupt noch nötig sind. Dies genauer zu erörtern würde jedoch den Rahmen dieser Arbeit sprengen. Daher verweise ich zu diesem Thema auf die Bücher von Klaus Fellbaum und Nicolae Nistor ([Fel04, NN03])

Die zweite Variante geht davon aus, dass die Studenten den Vorlesungsstoff einer Vorlesung bereits vor der jeweiligen Vorlesung bearbeitet haben. Dies hätte den großen Vorteil, dass die Vorlesung intensiver zum Nachfragen von nicht verstandenden Sachverhalten genutzt werden könnte. Der inhaltliche Verlauf der Vorlesung würde hierdurch mehr von den Studenten und deren Lernbedürfnissen beeinflusst werden. Es bestünde allerdings die Gefahr, dass bei diesem Ansatz die Vorlesung ganz den Charakter einer Übungsstunde annehmen würde. Da die Studenten jedoch lediglich mit entscheiden

können welche Themen intensiver von dem Dozenten erklärt werden sollen und nicht selbst Aufgaben zu lösen haben, wären zusätzliche Übungsstunden zur Besprechung der Übungsaufgaben weiterhin sinnvoll. Dieser theoretisch sehr viel versprechende Ansatz ist allerdings in der Praxis wohl nur sehr schwer umzusetzen. Zum Einem müsste der Aufbau der Vorlesung an diese Gegebenheiten angepasst werden, was einen erheblichen Mehraufwand für den Dozenten bedeuten würde. Zum Anderen ist es sehr fraglich ob es gelingen würde die Studenten zu motivieren sich vor einer Vorlesung den entsprechenden Stoff bereits anzueignen, wenn man ihnen gleichzeitig, die Möglichkeit gibt sich während der Vorlesung den Stoff auf Nachfrage nochmals erklären zu lassen.

Die dritte Variante geht davon aus, dass der Kurs genutzt wird um sich eigenständig die Inhalte zu erarbeiten, ohne eine parallel dazu laufende Veranstaltung an der Universität. In dieser Situation kann man davon ausgehen, dass sich der Lernende aus eigenem Interesse mit dem Thema beschäftigt, also intrinsisch motiviert ist (s. 3.3), was einen anderen konzeptionellen Aufbau des Kurses erfordern würde. Die Übungsaufgaben müssten außerdem automatisch oder vom lernenden selbst korrigiert werden. Da keine weiteren Personen involviert wären, würde es keine zeitlichen Beschränkungen in Bezug auf den Beginn und die Dauer des Kurses geben. Für die in dieser Studienarbeit gewählten Themen der Computergrafik 2 ist diese Variante nicht empfehlenswert. Das Themengebiet ist sehr speziell auf diesen Bereich beschränkt und setzt Vorkenntnisse voraus, die bei einem Kurs zum eigenständigen Lernen nicht erwartet werden können.

Ich habe mich für diese Arbeit für die erste Variante entschieden, da sie am einfachsten in den bestehenden Vorlesungsbetrieb integriert werden kann. Die anderen beiden Varianten halte ich auf Grund der bereits erwähnten Nachteile für den hier erwünschten Zweck als nicht empfehlenswert.

Des Weiteren soll der Kurs durch ein Glossar als ein Nachschlagewerk genutzt werden können. Dies könnte auch bei der Vorbereitung auf Klausuren oder bei anderen Projekten hilfreich sein. Hierfür werden die Lektionen mit Schlüsselwörtern versehen, die angeben welche Themen in der jeweiligen Lektion erörtert werden.

3.2 Aufbau des Kurses

Der Kurs soll sich inhaltlich nach den Themen der Vorlesung richten. Die einzelnen Themenbereiche werden in Lernmodule eingeteilt. Die Lernmodule bauen inhaltlich meist nicht aufeinander auf. Bei Ausnahmen wird in der Beschreibung sowie der Einleitung des Lernmoduls darauf hingewiesen welche Vorkenntnisse benötigt werden. Dies ermöglicht das Bearbeiten in einer beliebigen Reihenfolge bzw. das Bearbeiten von Themen, die noch nicht verstanden worden sind.

In der Einleitung eines Lernmoduls wird kurz beschrieben welche Inhalte behandelt werden und, falls notwendig, welche Vorkenntnisse vorausgesetzt werden. Jedes Lernmodul enthält vier bis fünf Teile:

1. Erklärung des Sachverhalts
2. Veranschaulichung durch Ausprobieren (mittels kleiner Applets, optionaler Teil)
3. Fakten die man wissen muss
4. Übungsaufgaben
5. Abschließender Wissenstest

Im ersten Teil wird der Sachverhalt durch Texte, Bilder und weitere Medien erklärt. Das Problem und deren Lösung werden theoretisch erläutert und anhand von Beispielen veranschaulicht.

Der zweite Teil besteht aus kleinen Applets, in denen der Lernende die neuen Sachverhalte und die darin enthaltenen mathematischen Systeme selbst ausprobieren kann. Hierfür wird das vorher beschriebene System grafisch dargestellt und die Variablenwerte angezeigt. Durch Veränderung von Variablen oder Verschieben von Punkten können die Veränderungen auf das Gesamtsystem sofort erkannt werden. Diese spielerische Herangehensweise erlaubt es dem Lernenden die neuen Erkenntnisse selbst zu überprüfen.

Der dritte Teil gibt den Sachverhalt noch einmal in sehr kurzer Form wieder. Er dient auch als Einstieg für das Nachschlagen im Glossar. Neben der Kurzbeschreibung werden alle für das Verständnis notwendige Fakten und Formeln angegeben. Dies können sowohl Berechnungsformeln (z.B. für Beleuchtungsmodelle), Methodensyntaxen von OpenGL als auch einfache Begriffe sein. Dem Lernenden wird empfohlen diese Fakten auswendig zu lernen und anschließend noch einmal den ersten Teil zu lesen.

Der vierte Teil stellt Aufgaben zum Einüben der neuen Methoden bereit. Ähnlich den Übungszetteln und den Aufgaben in Klausuren soll der Lernende hier gestellte Probleme selbständig lösen. Anschließend korrigiert das System - soweit möglich - die Lösung des Lernenden. Bei nicht durch das System lösbaren Aufgaben wird eine Musterlösung angezeigt, die mit der eigenen Lösung verglichen werden kann.

Im letzten Teil muss der Lernende einen Wissenstest bestehen, um das Lernmodul abzuschließen. Hierbei werden sowohl Aufgaben gestellt als auch Fakten abgefragt. Der Test soll dazu dienen, dass der Lernende überprüfen kann ob er den Sachverhalt vollständig erfasst hat. Die Bewertung des Tests geht in die Gesamtstatistik ein.

3.3 Zielgruppenanalyse

Da es nach Prof. Dr. Michael Kerres ([Ker01], S.135) nicht möglich ist, ein E-Learning-Angebot soweit zu individualisieren, dass es von Benutzern mit unterschiedlichsten Vorkenntnissen und Lernpräferenzen genutzt werden kann, ist es wichtig die Zielgruppe eines Kurses genau zu definieren. Das bedeutet man sollte sich von vornherein bewusst

sein, dass der Kurs nur für eine bestimmte Gruppe von Lernenden sinnvoll genutzt werden kann. Kerres weist weiter darauf hin, dass im Gegensatz zur Planung von konventionellem Präsenzunterricht, eine Anpassungen an die Lernenden nur schwer oder meist gar nicht durchgeführt werden kann.

Die Zielgruppenanalyse für den Kurs dieser Studienarbeit bringt aufgrund der genauen Kenntnisse der Lernenden und der Rahmenbedingungen sehr genaue Ergebnisse. Die soziodemografischen Merkmale können ziemlich genau definiert werden: Da es sich um Studenten der Informatik handelt, ist die Mehrheit männlich. Das Durchschnittsalter beträgt 20 bis 30 Jahre. Ebenfalls kann man davon ausgehen, dass alle Benutzer des Kurses sicher im Umgang mit Computern und neuen Medien sind.

Da sich der Inhalt des Kurses an der Vorlesung „Computergrafik 2“ orientiert, sind grundlegende Themen der Computergrafik, die in der Vorgängervorlesung behandelt worden sind, als Vorwissen bei den Lernenden vorauszusetzen. Außerdem sollten Grundlagen der Matrizenarithmetik sowie grundlegende Informatikkenntnisse vorhanden sein, um den Kurs erfolversprechend anwenden zu können.

Man kann davon ausgehen, dass die Mehrheit der Benutzer extrinsisch motiviert ist. Das bedeutet, dass die Lernenden nicht hauptsächlich aus Eigeninteresse (intrinsische Motivation) sondern aufgrund von äußeren Folgen des Lernen motiviert sind. Im Fall dieses Kurses sind die Folgen das Erlangen von Leistungsnachweisen (in Form von Klausurzulassungen oder Leistungsscheinen). Bei extrinsisch motivierten Lernern wird das Lernen oft mit Anstrengung verbunden, daher ist es wichtig sie immer wieder zu motivieren. Aufgrund des eindeutigen Lernziels dieses Kurses sind die Lernaktivitäten eher vorgegeben und es werden oft Rückmeldungen über den Lernerfolg erbracht.

Kerres schlägt für Zielgruppen mit Überwiegend extrinsischer Motivation folgende Anregungen vor ([Ker01],S.140):

- beim Einstieg motivierende Maßnahmen vorsehen
- Aufmerksamkeit auf Ziele des Lernens lenken
- Lehrstoff in definierte, überschaubare Einheiten einteilen
- Pausen vorsehen
- Konsistenz der Präsentation durchführen
- Lernfortschritt rückmelden

3.4 Zulassung zu Prüfungen

Zu den Vorlesungen der Computergrafik an der Universität Koblenz werden wöchentlich Aufgabenblätter ausgegeben. Diese müssen von den Studenten bearbeitet und in schrift-

licher Form abgegeben werden. Um an der Klausur am Ende einer Veranstaltung teilnehmen zu dürfen, müssen die Studenten mindestens 50 Prozent der erreichbaren Punkte aller Aufgabenblätter erbringen. Die Abgabe der bearbeiteten Aufgaben erfolgt hierbei sehr umständlich. Alle Ergebnisse müssen in schriftlicher Form in einen Briefkasten an der Uni eingeworfen werden. Diese Praxis wird von vielen Studenten als umständlich und, gerade für einen Informatikstudiengang, veraltet angesehen.

Der für diese Arbeit erstellte E-Learning-Kurs nutzt die Funktionalität von WebCT um eine Verteilung und Abgabe der Übungsaufgaben online durchzuführen. Dies hat für die Lehrenden den Vorteil, dass ein Großteil des Verwaltungsaufwandes (wöchentliches Bereitstellen der Aufgaben, Einteilung der Lerngruppen, Verwaltung der Ergebnisse...) von dem LMS übernommen werden. Für die Studenten hat es den Vorteil, dass sie nicht für die Abgabe zur Uni kommen müssen und dass sie gemeinsam von zu Hause aus an einer Aufgabe arbeiten können.

Bei dieser Praxis könnten generierte Aufgaben zum Einsatz kommen. (s. 3.5)

3.5 Aufgabenerstellung

Ein oft unterschätztes Problem ist die Erstellung von Übungsaufgaben. Der Schwierigkeitsgrad muss genau an die Bedürfnisse der Zielgruppe angepasst werden. Ist er zu niedrig sind die Lernenden schnell gelangweilt, ist er zu hoch wirkt dies demotivierend und ist nicht sehr förderlich für den Lernprozess.

3.5.1 Generierte Aufgaben

Bei bestimmten Aufgabenarten ist es möglich Aufgaben von einem Programm generieren zu lassen. Bestimmte Variablen der Aufgabe werden hierbei durch einen Zufallsgenerator erzeugt. Der Dozent muss im Idealfall nur die Grenzen für diese Variablenwerte festlegen. Dieses Verfahren funktioniert für viele mathematische Aufgabentypen, bei denen der Schwerpunkt auf dem Lösungsverfahren liegt.

Generierte Aufgaben haben den Vorteil, dass sie für den Dozenten weniger Aufwand darstellen. Außerdem erhält man durch die zufälligen Variablenwerte eine enorme Anzahl an unterschiedlichen Aufgaben. Generierte Aufgaben lassen sich allerdings nur für wenige Aufgabentypen realisieren.

3.5.2 Von Dozenten erstellte Aufgaben

Die meisten Übungsaufgaben müssen von den Dozenten selbst erstellt werden. Dies ist eine sehr zeitaufwändige Arbeit, da man viele mögliche Aufgaben selbst lösen muss, um herauszufinden ob die Aufgabe dem Niveau der Lernenden entspricht. Des Weiteren müssen mögliche Fallen vermieden oder bewusst eingebaut werden. Bei Aufgaben mit mathematischem Charakter ist es wünschenswert, dass die Ergebnisse nicht all zu viele

Nachkommastellen besitzen. Auf diese Weise erhalten die Lernenden einen Hinweis auf die Richtigkeit ihrer Lösung.

Kapitel 4

Praktische Umsetzung

4.1 Namenskonvention

4.1.1 Motivation

Bei der Beschäftigung mit mehreren Koordinatensystemen und deren Transformation kann es einem sehr schnell passieren, dass man die Variablennamen vertauscht. Für das Verständnis von Transformationen von einem Koordinatensystem in ein anderes ist es sehr wichtig genau zu wissen welche Achse in welchem System gemeint ist. Das kurze Lernmodul „Namenskonventionen“ legt deswegen eine für diesen Kurs und die darin enthaltenen Applets eine Konvention zur Bezeichnung der Werte fest.

Der erste Buchstabe eines Namens wird groß geschrieben und bezeichnet um welche Achse es sich handelt:

Bezeichner	Achse
X	1. Achse
Y	2. Achse
Z	3. Achse

Tabelle 4.1: Weltkoordinatensystem

Bezeichner	Achse
U	1. Achse
V	2. Achse
W	3. Achse

Tabelle 4.2: Kamerakoordinatensystem

Bezeichner	in Koordinaten des
W	Weltkoordinatensystem
C	Kamerakoordinatensystem

Tabelle 4.4: Vektorbezeichnung

Der zweite Buchstabe wird klein geschrieben und bezeichnet um welche Komponente und in welchem Koordinatensystem diese Komponente angegeben ist:

Bezeichner	Komponente	in Koordinaten des
x	1. Komponente	Weltkoordinatensystem
y	2. Komponente	Weltkoordinatensystem
z	3. Komponente	Weltkoordinatensystem
u	1. Komponente	Kamerakoordinatensystem
v	2. Komponente	Kamerakoordinatensystem
w	3. Komponente	Kamerakoordinatensystem

Tabelle 4.3: Komponentenbezeichnung

Die beiden Bezeichner werden durch einen Unterstrich („_“) getrennt.

Bei dem Applet zur Koordinatentransformation müssen für die Angabe von Skalarprodukten keine einzelnen Komponenten sondern ganze Vektoren bezeichnet werden. Dies ist realisiert worden durch die Möglichkeit den zweiten Buchstaben groß zu schreiben um hierdurch anzugeben in welchem Koordinatensystem der Vektor bezeichnet werden soll:

4.2 Koordinatentransformationen

4.2.1 Motivation

Nicht nur in der Computergrafik und der Bildverarbeitung, sondern auch in vielen anderen wissenschaftlichen Feldern, sieht man sich oft mit dem Problem konfrontiert Koordinaten aus einem Koordinatensystem in ein anderes transformieren zu müssen. In der Bildverarbeitung nutzt die Epipolargemoetrie Koordinatentransformationen um Koordinaten aus zwei Kamerakoordinatensystemen in das Weltkoordinatensystem zu transformieren und somit aus zwei zweidimensionalen Bildern dreidimensionale Modelle zu erstellen.

In der Computergrafik benötigt man Koordinatentransformationen um 3D-Modelle, die in ihrem eigenen Koordinatensystem erstellt werden, in das Weltkoordinatensystem der gesamten Szene zu lokalisieren. Bei der Berechnung des Bildes, welches auf dem Bildschirm ausgegeben wird, muss man aus dem Weltkoordinatensystem in das Kame-

rakoordinatensystem transformieren um das von diesem Standpunkt aus korrekte Bild zu erhalten. Die im nächsten Kapitel behandelten Szenegraphen nutzen Koordinatentransformationen um Elemente einer Szene in unterschiedlichen Koordinatensystemen zu speichern. Eine weitere Anwendung von Koordinatentransformationen findet man im Bereich der virtuellen oder augmented Reality. Hier werden Koordinaten von Trackern, die im Weltkoordinatensystem bekannt sind, genutzt, um Punkte aus dem Kamerakoordinatensystem in das Weltkoordinatensystem zu transformieren.

In dem für diese Arbeit erstellten E-Learning-Kurs werden Koordinatentransformationen anhand von Transformationen aus Welt- nach Kamerakoordinaten und umgekehrt erklärt und eingeübt. Im zweiten Lernmodul werden sie für die Erklärung von Szenegraphen eingesetzt.

4.2.2 Applet : Transformationsmatritzen

Das Applet zur Koordinatentransformation soll das Berechnen von Transformationen von einem System in ein anderes trainieren. Die beiden Systeme werden als Welt- und Kamerakoordinatensystem bezeichnet. Hierdurch werden Verwechslungen der beiden Systeme vermindert. Bei einer Bezeichnung wie beispielsweise System c1 und c2 wäre dies nicht gewährleistet. Dem Benutzer werden die Werte der Achsen der beiden Koordinatensysteme, die Distanz zwischen beiden in Form eines Vektors A und ein beliebiger Punkt p angezeigt. Es gibt vier verschiedene Arten von Aufgabentypen, die sich darin unterscheiden, ob die angegebenen Werte in Koordinaten des Welt- oder Kamerakoordinatensystems angegeben sind. Je nach Aufgabentyp benötigt man eine andere Transformationsmatrix um Koordinaten zwischen den Systemen zu transformieren:

Aufgabentyp	Systeme in	A in	P in	Transformiert in	Transformationsmatrix
Typ 1	W	W	W	C	M1
Typ 2	C	C	W	C	M2
Typ 3	C	C	C	W	M3
Typ 4	W	W	C	W	M4

Tabelle 4.5: Aufgabentypen

$$M1: \begin{pmatrix} u_x & u_y & u_z & -A_w \circ \vec{u}_w \\ v_x & v_y & v_z & -A_w \circ \vec{v}_w \\ w_x & w_y & w_z & -A_w \circ \vec{w}_w \\ 0 & 0 & 0 & 1 \end{pmatrix} M2: \begin{pmatrix} x_u & x_v & x_w & A_u \\ y_u & y_v & y_w & A_v \\ z_u & z_v & z_w & A_w \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Durch klicken der Werte rechts, werden diese in das selektierte Feld eingefügt.

Zurücksetzen Lösen nächste Aufgabe

P_C=

<input type="text" value="0.0"/>	Ergebnis					
<input type="text" value="0.0"/>						
<input type="text" value="0.0"/>						
<input type="text" value="0.0"/>						

Weltkoordinatensystem

in Weltkoordinaten

X_x	1.0	Y_x	0.0	Z_x	0.0
X_y	0.0	Y_y	1.0	Z_y	0.0
X_z	0.0	Y_z	0.0	Z_z	1.0

in Kamerakoordinaten

X_u	0.996	Y_u	0.0070	Z_u	-0.089
X_v	0.0010	Y_v	0.996	Z_v	0.091
X_w	0.089	Y_w	-0.091	Z_w	0.992

Kamerakoordinatensystem

in Weltkoordinaten

U_x	0.996	V_x	0.0010	W_x	0.089
U_y	0.0070	V_y	0.996	W_y	-0.091
U_z	-0.089	V_z	0.091	W_z	0.992

in Kamerakoordinaten

U_u	1.0	V_u	0.0	W_u	0.0
U_v	0.0	V_v	1.0	W_v	0.0
U_w	0.0	V_w	0.0	W_w	1.0

A

in Weltkoordinaten			in Kamerakoordinaten		
A_x	2.785	A_u	2.518	A_v	2.738
A_y	2.466	A_w	3.051	A_z	3.051
A_z	3.05				

p

in Weltkoordinaten			in Kamerakoordinaten		
p_x	1.263	p_u	-1.416	p_v	-0.301
p_y	2.269	p_w	-1.74	p_z	
p_z	1.92				

Abbildung 4.1: Applet Transformationen

$$\text{M3: } \begin{pmatrix} x_u & x_v & x_w & -A_c \circ \vec{x}_c \\ y_u & y_v & y_w & -A_c \circ \vec{y}_c \\ z_u & z_v & y_w & -A_c \circ \vec{z}_c \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{M4: } \begin{pmatrix} u_x & u_y & u_z & A_x \\ v_x & v_y & v_z & A_y \\ w_x & w_y & w_z & A_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Die Aufgabe des Benutzer besteht darin, die richtige Transformationsmatrix aus den gegebenen Werten anzugeben. Hierfür kann er auf die Werte auf der rechten Seite klicken um sie in das aktivierte Eingabefeld der Lösungsmatrix zu übertragen. Alternativ ist es ebenfalls möglich in den Eingabefeldern die Bezeichner, die neben den Werten angezeigt werden, anzugeben. Hierbei ist auch die Verwendung eines Skalarprodukts und der Negation möglich.

Nach der Angabe der Transformationsmatrix müssen die Werte des zu transformierenden Punktes angegeben werden. Nach Klicken des Lösen-Knopfes wird von dem Applet verglichen, ob die Eingaben der Lösung entsprechen und der Ergebnispunkt der eingegebenen Transformation wird angezeigt. Ist das Ergebnis korrekt, wird dies durch eine grüne Umrandung des transformierten Punktes angezeigt. Ein falsches Ergebnis wird rot umrandet.

Der Zurücksetzen-Knopf bewirkt, dass alle Eingabefelder auf Null gesetzt werden. Der Nächste-Aufgabe-Knopf bewirkt, dass eine neue zufällige Aufgabe erzeugt wird.

4.2.3 Implementation

Das Applet ist in drei Pakete unterteilt. Das main Paket enthält die Klasse Main, die eine Erweiterung der Klasse JApplet ist. Das Paket gui enthält die graphischen Elemente des Applet, mit Ausnahme der OpenGL-Anzeige, welche direkt in die Mainklasse integriert ist. Das exercises-Paket enthält die verschiedenen Aufgabentypen und die Programmlogik zur Berechnung der Transformationen.

Das Applet nutzt das OpenGL-Binding Lightweight Java Game Library (LWJGL)¹. LWJGL ist eine Api, die es ermöglicht, OpenGL in Javaprogrammen zu nutzen. Hierfür wurde der Befehlssatz von OpenGL an Objekte von Java gebunden. Der Java-Programmierer braucht sich nicht um das eigentlich Binding kümmern, sondern kann durch Zugriff auf statische Objekte (wie z.B. G11, Display...) OpenGL Befehle ausführen. LWJGL bietet außerdem ein Utility-Paket an, das Klassen zur Vektor- und Matrizenverarbeitung bereitstellt. Diese habe ich in dem Applet genutzt um die Koordinatensysteme zu speichern und die Transformationen zu berechnen. Ein weiter Vorteil von LWJGL, den ich für diese Arbeit nutzen konnte, ist der sogenannte Appletloader. Dieser unterstützt das Erstellen von Applet, die LWJGL nutzen. Bei der Ausführung solcher Applet, muss der Browser des Anwenders die für LWJGL notwendigen Bibliotheken auf den eigenen Rechner herunterladen. Das Herunterladen von unbekanntenen Bibliotheken ist jedoch bei den meisten

¹<http://www.lwjgl.org>

Browsern verboten oder wird nur für signierte Bibliotheken erlaubt. Der Appletloader nutzt hierfür eine signierte Version der nativen Bibliotheken.

4.2.3.1 Das Main-Paket

Main
- exercise : Exercise - finished : boolean - xRot : float = 0 - yRot : float = 0 - zRot : float = 0 - explanation : JLabel + openGLCanvas : AWTGLCanvas - valuesPanel : ValuesPanel - inputPanel : InputPanel - centerPanel : JPanel - setZeroButton : JButton - resolveButton : JButton - nextButton : JButton
+ init () + start () - render () - drawAxis (axis : Vector3f, origin : Vector3f) - run () + initGL () - rotate () - keyboardInputs () + newRandomExcercise () - cleanup () + changeExerciseType (newType : int) - createButtons () : JPanel + getExercise () : Exercise + getValuesPanel () : ValuesPanel + getInputPanel () : InputPanel

Abbildung 4.2: Transformationen: Mainklasse

Die Mainklasse ist die zentrale Startklasse des Applets. Als diese enthält sie die Methoden `init()` und `start()`, welche beim Aufrufen des Applet ausgeführt werden. Die als erstes ausgeführte `init()`-Methode initialisiert ein `AWTGLCanvas`-Objekt, das die Zeichenfläche für die OpenGL-Befehle repräsentiert. Des Weiteren werden eine Aufgabe vom Typ `Excercise` (s. 4.2.3.3) und die graphischen Bedienelemente erzeugt. Die `start()`-Methode, welche den eigentlichen Programmablauf enthält, ruft zuerst die Methode `initGL()` auf. Diese Methode initialisiert die OpenGL-Engine. Dies bedeutet, dass ein statisches Displayobjekt durch Aufruf der entsprechenden OpenGL-Befehle durch Einstellen der Auflösung, der Projektionsart und des Viewportes vorbereitet wird. Anschließend wird durch Übergabe des Canvas-Objekts dem Displayelement eine Zeichenfläche zur Verfügung gestellt. Anschließend wird die `run()`-Methode aufgerufen, die in einer Endlosschleife den eigentlichen Programmablauf durchführt. Bis zur Beendigung des Programmes durch den Benutzer werden Tastatureingaben bearbeitet und die beiden Koordinatensysteme der aktuellen Aufgabe gezeichnet.

4.2.3.2 Das Gui-Paket

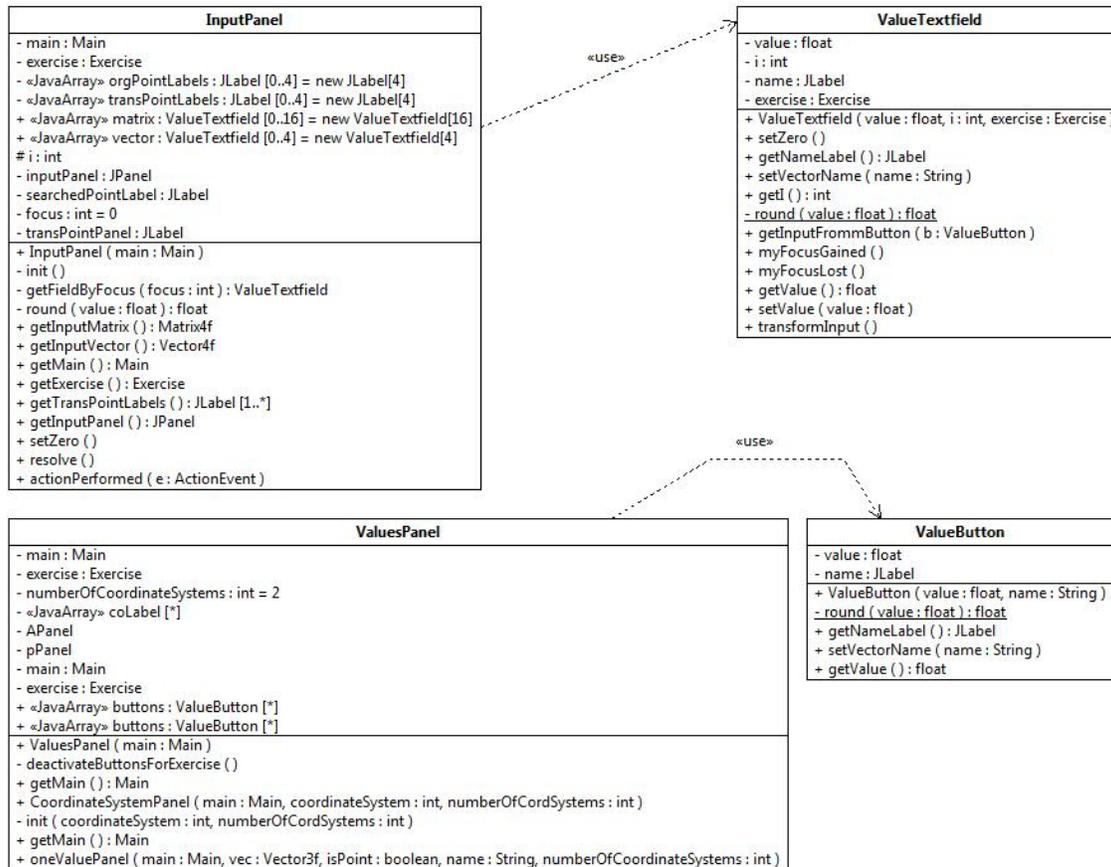


Abbildung 4.3: Transformationen: Guipaket

Das Gui-Paket enthält die zwei wichtigen Klassen ValuesPanel und InputPanel.

Das ValuesPanel stellt die beiden Koordinatensysteme durch Angabe der Achsen dar. Von jeder Achse werden die drei Komponenten als Knöpfe angezeigt. Die Knöpfe sind mit den jeweiligen Werten der Achsen beschriftet. Neben den Knöpfen wird der entsprechende Name des Wertes angegeben. Beide Koordinatensysteme werden jeweils in Koordinaten beider Systeme angezeigt. Unter den Koordinatensystemen werden der Vektor A und der Punkt P in jeweils beiden Koordinatensystemen angegeben.

Durch klicken der Werte rechts, werden diese in das selektierte Feld eingefügt

Zurücksetzen Lösen nächste Aufgabe

P_C=

0.0	0.0	0.0	0.0	0.0	0.0	Ergebnis
0.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	

Weltkoordinatensystem

in Weltkoordinaten

X_x	1.0	Y_x	0.0	Z_x	0.0
X_y	0.0	Y_y	1.0	Z_y	0.0
X_z	0.0	Y_z	0.0	Z_z	1.0

in Kamerakoordinaten

X_u	0.996	Y_u	0.0070	Z_u	-0.089
X_v	0.0010	Y_v	0.996	Z_v	0.091
X_w	0.089	Y_w	-0.091	Z_w	0.992

Kamerakoordinatensystem

in Weltkoordinaten

U_x	0.996	V_x	0.0010	W_x	0.089
U_y	0.0070	V_y	0.996	W_y	-0.091
U_z	-0.089	V_z	0.091	W_z	0.992

in Kamerakoordinaten

U_u	1.0	V_u	0.0	W_u	0.0
U_v	0.0	V_v	1.0	W_v	0.0
U_w	0.0	V_w	0.0	W_w	1.0

A

in Weltkoordinaten			in Kamerakoordinaten		
A_x	2.785	A_u	2.518	A_v	2.738
A_y	2.466	A_w	3.051	A_z	3.051
A_z	3.05				

p

in Weltkoordinaten			in Kamerakoordinaten		
p_x	1.263	p_u	-1.416	p_v	-0.301
p_y	2.269	p_w	-1.74	p_z	
p_z	1.92				

Abbildung 4.4: ValuesPanel

Durch klicken der Werte rechts, werden diese in das selektierte Feld eingefügt.

Zurücksetzen Lösen nächste Aufgabe

Weltkoordinatensystem

in Weltkoordinaten

X_x	1.0	Y_x	0.0	Z_x	0.0
X_y	0.0	Y_y	1.0	Z_y	0.0
X_z	0.0	Y_z	0.0	Z_z	1.0

in Kamerakoordinaten

X_u	0.996	Y_u	0.0070	Z_u	-0.089
X_v	0.0010	Y_v	0.996	Z_v	0.091
X_w	0.089	Y_w	-0.091	Z_w	0.992

Kamerakoordinatensystem

in Weltkoordinaten

U_x	0.996	V_x	0.0010	W_x	0.089
U_y	0.0070	V_y	0.996	W_y	-0.091
U_z	-0.089	V_z	0.091	W_z	0.992

in Kamerakoordinaten

U_u	1.0	V_u	0.0	W_u	0.0
U_v	0.0	V_v	1.0	W_v	0.0
U_w	0.0	V_w	0.0	W_w	1.0

A

in Weltkoordinaten			in Kamerakoordinaten		
A_x	2.785	A_y	2.466	A_u	2.518
A_v	2.466	A_z	3.05	A_v	2.738
A_z	3.05			A_w	3.051

P

in Weltkoordinaten			in Kamerakoordinaten		
P_x	1.263	P_y	1.92	P_u	-1.416
P_v	2.269			P_v	-0.301
P_z	1.92			P_w	-1.24

P_C=

0.0	0.0	0.0	0.0	0.0	Ergebnis
0.0	0.0	0.0	0.0	*	=
0.0	0.0	0.0	0.0		
0.0	0.0	0.0	0.0		

Abbildung 4.5: InputPanel

Das InputPanel stellt die eigentliche Aufgabe dar, die der Benutzer lösen soll. Es enthält eine Tabelle aus Eingabefeldern, die die Transformationmatrix repräsentieren. Neben der Matrix befinden sich weitere vier Eingabefelder, die den zu transformierenden Punkt repräsentieren. Der Benutzer kann nun auf zwei verschiedene Arten Werte in die Eingabefelder eingeben. Drückt er auf einen der Knöpfe des ValuePanels wird dessen Wert in das aktuell fokussierte Eingabefeld übertragen. Der Fokus wird durch Fettschrift in dem Eingabefeld angezeigt. Dieser Fokus entspricht nicht dem Fokus des Programmes, da dieser nach dem Drücken eines Knopfes auf den gedrückten Knopf wechselt. Die zweite Möglichkeit der Eingabe besteht darin in dem Eingabefeld den Namen eines Wertes anzugeben. Im obigen Bild würde zum Beispiel die Eingabe „U_x“ in den Wert „0.996“ umgewandelt werden. Hierbei ist auch die Nutzung des Minuszeichens erlaubt. Aus der Eingabe „-U_X“ würde also der Wert „-0.996“ werden. Der Grund diese Art der Eingabe zu implementieren war jedoch die dritte Möglichkeit der Texteingabe. In der letzten Spalte der Transformationsmatrizen M1 und M2 muss ein Skalarprodukt angegeben werden, was in den Knöpfen des ValuesPanel nicht vorhanden ist. Das Skalarprodukt ebenfalls als Knopf darzustellen hätte dazu geführt, dass dem Benutzer bereits ein Großteil der Lösung vorgegeben gewesen wäre. Deswegen habe ich bei der Texteingabe ein Skalarprodukt implementiert, durch welches die benötigten Werte eingegeben werden können. Die Eingabe „A_C*X_C“ liefert das Skalarprodukt des Vektors A und der X-Achse (beide in Kamerakoordinaten).

4.2.3.3 Das Exercis-Paket

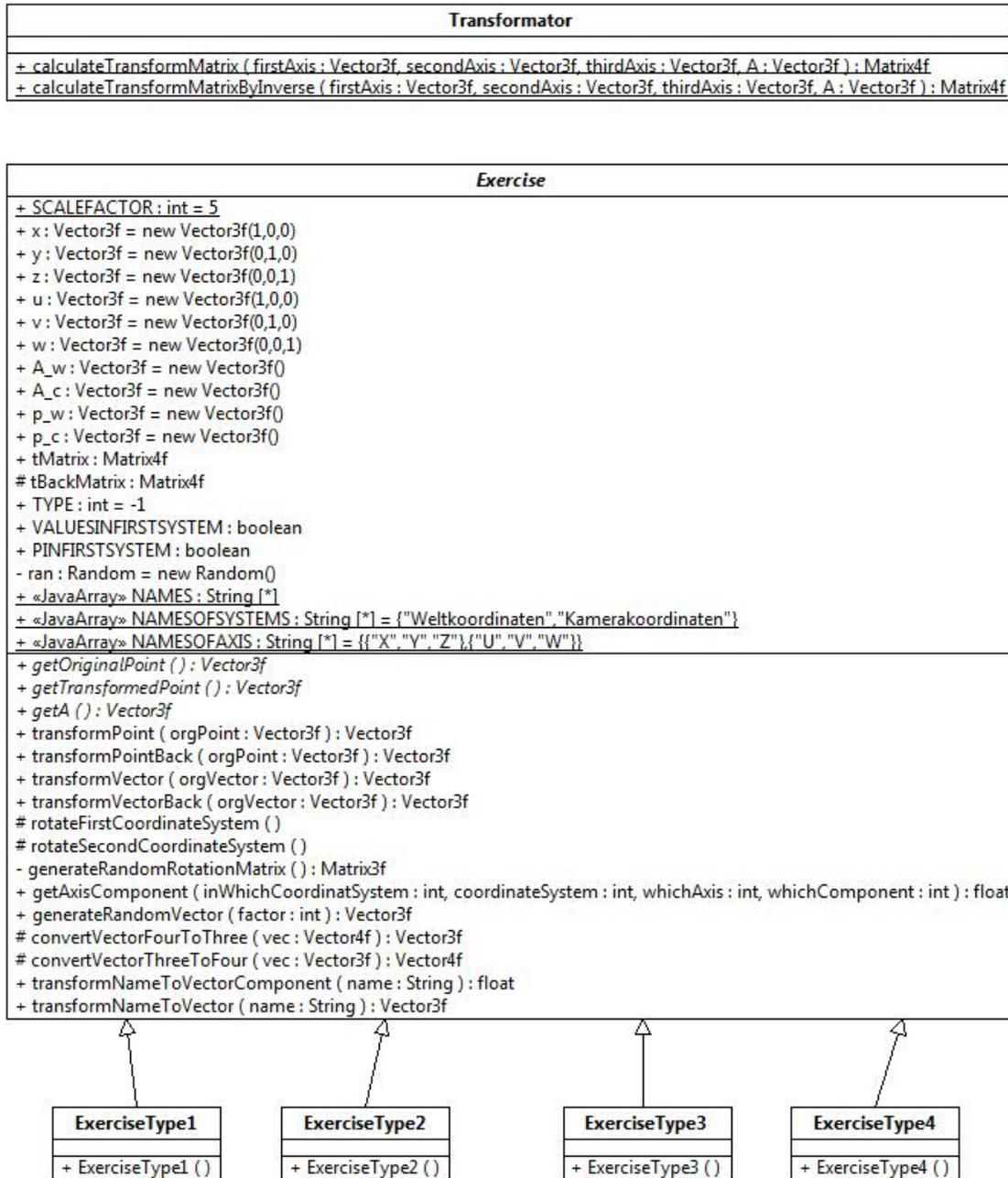


Abbildung 4.6: Transformationen: Exercisepaket

Das Paket Exercise enthält die Aufgaben und eine statische Klasse Transformator, welche die Transformationsmatrizen berechnet. Eine Aufgabe enthält für jedes Koordinatensystem drei Vektoren, die die Achsen der Systeme speichern. Abhängig vom Aufgabentyp repräsentieren diese Werte Koordinaten im ersten oder im zweiten Koordinatensystem. Der Vektor A und der Punkt P sind jeweils in Koordinaten des ersten und des zweiten Systems gespeichert. Des Weiteren speichert eine Aufgabe zwei vier mal vier Matrizen, die die Transformation in beide Richtungen darstellen. Die transformXXX() und transformXXXBack()-Methoden transformieren ihnen übergebene Punkte. Die Richtung der Transformation ist hierbei abhängig von dem Aufgabentyp. Die vier Aufgabentypen sind durch Ableitungen der abstrakten Klasse Exercise modelliert worden. Sie unterscheiden sich darin, welche Werte im Konstruktor der Klasse generiert werden. Die Koordinatensysteme werden anfangs beide „normiert“. Das heißt, die erste Achse ist (1,0,0), die zweite (0,1,0) und die dritte Achse (0,0,1). Bei den Aufgabentypen, die die Systeme in Koordinaten des ersten Systems speichern (Typ 1 u. 4) wird nun das zweite System um einen zufälligen Winkel um eine zufällige Achse gedreht. Bei den beiden anderen Aufgabentypen wird das erste System gedreht.

4.3 Szenegraphen

4.3.1 Motivation

Szenegraphen sind baumartige Datenstrukturen, die dreidimensionale Welten bzw. Szenen speichern und verwalten. Es gibt keine genaue theoretische Definition von Szenegraphen sondern nur eine große Anzahl von praktischen Implementationen². Der Grundgedanke eines Szenegraphen ist es eine Szene hierarchisch zu ordnen. Die Knoten eines Szenegraphen können Transformationen, also eine Translation und/oder eine Rotation, geometrische Informationen, also die eigentlichen 3D-Punkte der Objekte, oder andere Informationen wie Farbänderungen, Texturen oder ähnliches enthalten.

Beispielsweise würde ein Szenegraph für eine Stadt für jeden Stadtteil einen Ast besitzen. Der erste Knoten eines solchen Astes würde eine Transformation enthalten, die zum jeweiligen Stadtteilzentrum transformieren würde. Der Stadtteil würde wieder aus mehreren Ästen bestehen, die Straßen repräsentieren. Die Straßen enthalten Häuser, diese Etagen und diese Zimmer. Auf diese Weise könnte man beliebig viele Details in dem Modell darstellen. Will man nun beispielsweise eine gesamte Straße drehen oder verschieben, genügt es bei einem Szenegraphen, diese Transformation vor den obersten Knoten dieser Straße einzufügen (in diesem Beispiel nach dem Stadtteilzentrums-Knoten). Alle Details die sich unterhalb der Transformation befinden werden nun automatisch ebenfalls transformiert.

²siehe hierzu Liste von Szenegraphimplementationen im Anhang

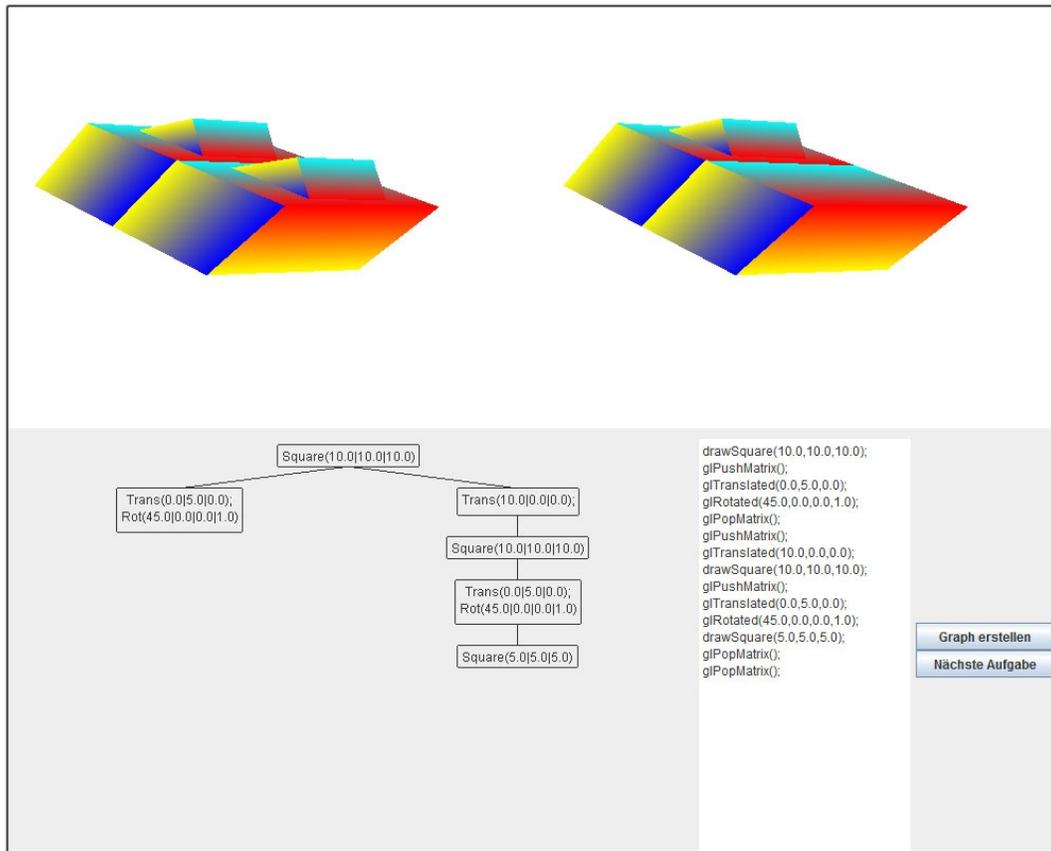


Abbildung 4.7: Szenegraph-Applet

4.3.2 Applet : Szenegraph

Das Applet Szenegraph soll den Aufbau und den Umgang mit Szenegraphen trainieren. Dem Benutzer wird eine Szene angezeigt. Unter der Anzeige wird der entsprechende Szenegraph dargestellt. Bei diesem fehlen jedoch einige Knoten. Rechts neben dem Graphen befindet sich der OpenGL-Code, der den angezeigten Graphen implementiert. Im rechten Teil der oberen Anzeige wird das Bild gezeichnet, welches der unvollständige Graph ergeben würde. Die Aufgabe besteht nun darin, den OpenGL-Code so zu verändern, dass der daraus entstehende Szenegraph dem Szenegraphen des Vorgabebildes entspricht.

Für die Erstellung von Aufgaben wird ein Szenegraph aus einer Sammlung von fertigen Vorlagen ausgewählt und von diesem einige Knoten entfernt. Ich habe bewusst keine Graphen zufällig generieren lassen, da es für den Benutzer sehr schwierig sein dürfte de-

ren Struktur nur auf Grund des Vorgabebildes zu verstehen. Bei den Vorlagen handelt es sich um einfache Darstellungen realer Objekte wie beispielsweise einem Auto oder einem Arm. Daher sollte die gemeinte Struktur einfach zu erkennen sein.

4.3.3 Implementation

Dieses Applet ist ebenfalls in drei Pakete unterteilt. Das Paket main enthält ebenfalls eine Mainklasse die von JApplet abgeleitet ist und das Paket gui enthält Klassen für die grafischen Bedienelemente. Das dritte Paket sGraph enthält eine einfache Implementation eines Szenegraphen, die an das Node-Core-Konzept von OpenSG angelehnt ist.

4.3.3.1 Die Mainklasse

Main
- resolveGraph : Node - actualGraph : Node - openGLCanvas : AWTGLCanvas - gPanel : SGraphPanel - cEdit : CodeEditor - finished : boolean - xRot : int = 0 - yRot : int = 0 - zRot : int = 0 - zDistance : int = 65 - buildTreeButton : JButton - nextButton : JButton
+ init () + start () - render () - initGL () - run () - rotate () - keyboardInputs () - cleanup () + createButtons () - createNewExercise () + stop () + getGraph () : Node + setGraph (n : Node)

Abbildung 4.8: Szenegraph: Mainklasse

Die Mainklasse ist prinzipiell aufgebaut wie die Mainklasse des Transformations-Applets. Die init()-Methode des Applet initialisiert die Komponenten und erstellt eine neue Aufgabe. Anschließend wird der OpenGL-Kontext initialisiert und das Programm zeichnet in einer Endlosschleife die beiden Bilder, die durch Eingaben des Benutzer beeinflusst werden.

4.3.3.2 Das Gui-Paket

Das Paket gui enthält die beiden Klassen SGraphPanel und CodeEditor. Das SGraphPanel ist abgeleitet von der Klasse JPanel und dient dazu, einen Szenegraphen zu visualisieren.

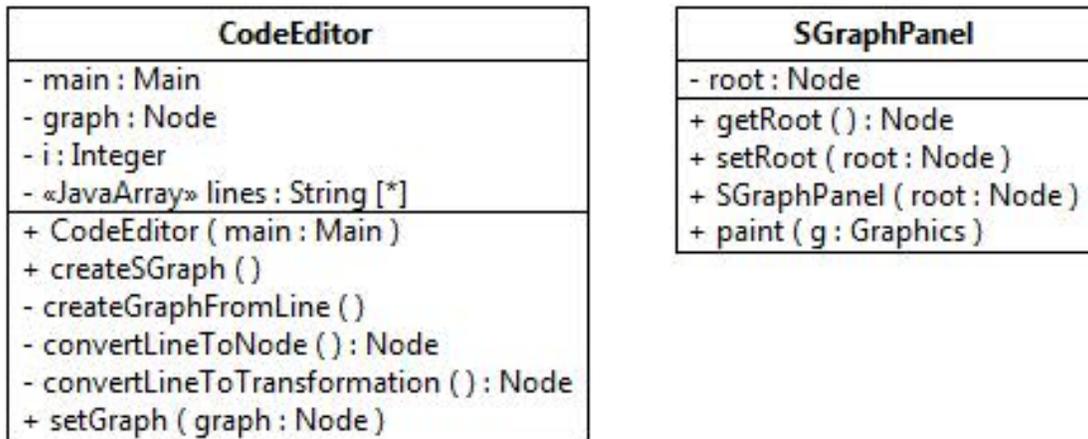


Abbildung 4.9: Szenegraph: Guipaket

Hierfür muss der Klasse ein Szenegraph übergeben werden. Soll nun das Panel gezeichnet werden ruft es in seiner Paintmethode die Paintmethode des Wurzelknotens des Szenegraphen auf. Diese zeichnet sich selbst und ruft anschließend rekursiv die Paintmethoden aller Unterknoten auf.

Die Klasse CodeEditor ist abgeleitet von der Klasse JTextPane. Sie zeigt denn OpenGL-Code des aktuellen Szenegraphens an. Der Benutzer kann den Code bearbeiten um so den gesuchten Graphen einzugeben. Die öffentliche Methode createSGraph() veranlasst den CodeEditor dazu den OpenGL-Code in einen Szenegraphen zu compilieren. Hierfür wird jede Zeile in einen Knoten übersetzt und als Kind oder Geschwister des letzten Knotens eingesetzt. Wird hierbei ein glPushMatrix()-Befehl bearbeitet, wird die convertLineToTransformation()-Methode aufgerufen. Diese erstellt einen Transformation-Core und fügt alle folgenden Transformationen in diesen Core ein, bis eine Zeile ohne Transformation erreicht wird. Kann eine Zeile nicht übersetzt werden wird dem Benutzer dies in einem Dialog mitgeteilt und für die entsprechende Zeile wird in den Graphen ein leerer Knoten eingefügt.

4.3.3.3 Das sGraph-Paket

Das Paket sGraph enthält eine Implementation eines Szenegraphen. Die Architektur ist an das Node-Core-Konzept von OpenSG angelehnt. Eine Szenegraph besteht somit aus eine Anzahl von Knoten (Instanzen der Klasse Node). Jeder Node muss einen Kern (Core) besitzen. Kerne enthalten ein oder mehrere OpenGL-Befehle. Ein Core ist entweder ein leerer Core (EmptyCore), ein TransformationCore oder ein GeometryCore. Leere Cores sind die einzigen Cores ohne OpenGL-Befehle. Sie werden eingesetzt, wenn man mehrere

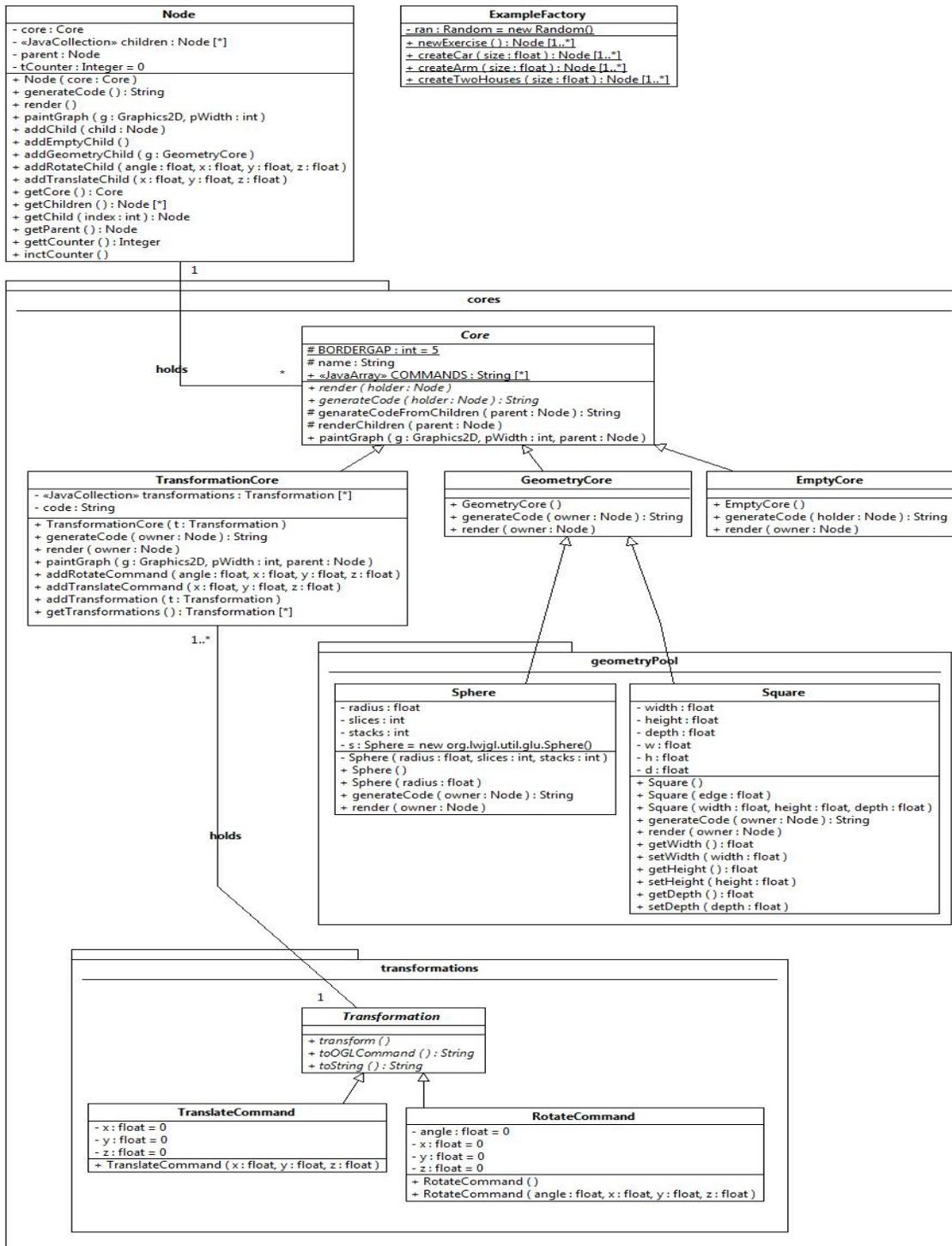


Abbildung 4.10: Szenenrgraph: sGraphpaket

Kinderknoten an einen Knoten anhängen möchte, ohne dass der Vaterknoten Code enthalten soll. TransformationCores beinhalten eine oder mehrere Transformationen. Transformationen sind entweder Translationen oder Rotationen in Form von Instanzen einer abgeleiteten Klasse von Transformation. GeometryCores enthalten Befehle zum Zeichnen von Objekten, z.B. Kugeln oder Quadern. Jeder Core ist durch die von der abstrakten Vaterklasse Core geerbten Methoden `render()` und `generateCode()` in der Lage sich und seine Kinder zu zeichnen und den OpenGL-Code, den er und seine Kinder repräsentieren, als String zurückzugeben. Beide Methoden bearbeiten erst den Core des aktuellen Knotens und rufen im Anschluss in der Methode `generateCodeFrom-/render-Children()` die jeweiligen Methoden ihrer Kinder auf.

Schluss

4.4 Zusammenfassung der Arbeit

E-Learning-Angebote bieten eine gute Erweiterung für den Hochschulbetrieb. Die Erstellung solcher Angebote erfordert jedoch viel Zeit und konzeptionelle Vorbereitung. Hierbei stellt sich die Frage ob der hohe Aufwand dem späteren Nutzen gerecht wird. Das Applet zur Koordinatentransformation beispielsweise benötigte schätzungsweise zwei Wochen Entwicklungszeit. Der Nutzen des Applet ist jedoch darauf beschränkt, vier Transformationsmatrizen und deren Anwendung auswendig zu lernen. Hierbei ist allerdings zu bedenken, dass der Einsatz des Applet, im Gegensatz zum einfachen Auswendiglernen der Matrizen, einige Vorteile mit sich bringt. Der erste Vorteil besteht darin, dass es sehr schwierig ist geeignete, überprüfbare Aufgaben zur Anwendung der Matrizen zu finden. Dies übernimmt das Applet für den Lernenden. Des Weiteren erleichtert die Anzeige der zu berechnenden Koordinatensystem und derer Punkte das Verständnis des Sachverhaltes. Aus diesen Gründen denke ich, dass sich der Mehraufwand zur Erstellung von E-Learning-Angeboten in den meisten Fällen lohnt.

4.5 Ausblick

Der im Rahmen dieser Arbeit erstellte Kurs bezieht sich thematisch nur auf einen kleinen Teil des Vorlesungsstoffes der Vorlesung Computergraphik II. Es wäre wünschenswert, Lernmodule und entsprechende Applets für den gesamten Lehrstoff zu erstellen, um somit den Kurs als vollständige Unterstützung zur Vorlesung anbieten zu können.

Anhang

LMS	URL	Universitäten die dieses LMS einsetzen
WebCT	http://www.webct.com	Hamburg, Koblenz
CommSy	http://www.commsy.net	div. Hochschulen Hamburgs
COLLIDE	http://www.collide.info	Duisburg, Essen
Moodle	http://moodle.hu-berlin.de/	Humboldt Universität Berlin
ILIAS	http://www.ilias.de/index-de.html	div. Universitäten weltweit
elecTUM	https://www.elearning.tum.de	Technische Universität München
OLAT	https://www.olat.uzh.ch/olat/dmz/	div. schweizer Unis, Uni München (LMU)

Tabelle 4.6: Learning Management Systeme

Name	URL
Open Inventor	http://oss.sgi.com/projects/inventor/
Performer	http://oss.sgi.com/projects/performer/
OpenSG	http://opensg.vrsourc.org/trac
Ogre	http://www.ogre3d.org/

Tabelle 4.7: OpenGL-Implementationen von Szenegraphen

Abbildungsverzeichnis

4.1	Applet Transformationen	18
4.2	Transformationen: Mainklasse	20
4.3	Transformationen: Guipaket	21
4.4	ValuesPanel	22
4.5	InputPanel	23
4.6	Transformationen: Exercisepaket	25
4.7	Szenegraph-Applet	27
4.8	Szenegraph: Mainklasse	28
4.9	Szenegraph: Guipaket	29
4.10	Szenegraph: sGraphpaket	30

Tabellenverzeichnis

4.1	Weltkoordinatensystem	15
4.2	Kamerakoordinatensystem	15
4.4	Vektorbezeichnung	16
4.3	Komponentenbezeichnung	16
4.5	Aufgabentypen	17
4.6	Learning Management Systeme	34
4.7	OpenGL-Implementationen von Szenegraphen	34

Literaturverzeichnis

- [Fel04] Klaus Fellbaum. *eLearning an der Hochschule*. Shaker Verlag, 2004.
- [Ker01] Prof. Dr. Michael Kerres. *Multimedial und telemediale Lernumgebungen*. Oldenbourg Verlag München Wien, 2001.
- [NN03] S. Wheeler M. Jalobeanu N. Nistor, S. English, editor. *Toward the virtual university - international online perspectives*. Inforamtion Age Publishing, 2003.