**Maturaarbeit**

**Titelblatt**

Visualisierung des Rubik’s Cube in Java

Gymnasium Neufeld

Abteilung Mathematik und Naturwissenschaften

Erstellt von: Rafael Müller

Betreut durch: Tobias Bäumlin

**Inhaltsverzeichnis**

**Abstract**

(Zielsetzung, Vorgehen, Weg zum Ziel)

**Vorwort**

(Grund für die Erstellung der Arbeit, persönliche Sicht auf die Arbeit, Adressat)

In dieser Maturaarbeit habe ich es geschafft zwei meiner Begeisterungen zu vereinen. Das eine ist der Rubik’s Cube, das andere ist die Informatik.

Der Würfel hat vor einigen Jahren an Weihnachten meine Begeisterung geweckt. Damals bekam mein jüngerer Bruder einen solchen geschenkt. Seine Begeisterung hielt sich in Grenzen. Als der Würfel verdreht war und ihm die Lösung versagt blieb, legte er den Würfel zur Seite.

Ich versuchte mich natürlich auch mit dem Würfel und drehte eifrig herum. Doch meine ersten Versuche scheiterten ebenfalls kläglich. Später versuchte ich im Internet nach Lösungsmethoden; und tatsächlich fand ich einige Tutorials. Beim Schreiben meiner Maturaarbeit kam mir diese Zeit immer wieder in den Sinn, weil ich damals schon ähnlich vorging, wie jetzt bei der Maturaarbeit.

Ich recherchierte auf unterschiedlichen Seiten bis ich den Würfel zum ersten Mal lösen konnte. Mit der Zeit lernte ich die Züge auswendig. Als ich den Würfel auf Zeit zu lösen versuchte, recherchierte ich nach weiteren Kombinationen, die es mir ermöglichten, den Würfel noch schneller zu lösen.

In meiner ersten Studienwoche im Gymnasium konnte ich andere meiner Klasse für den Würfel begeistern. Ich lernte ihnen mein Wissen. Inzwischen hat sich das „Würfelfieber“ wieder etwas abgeschwächt, doch viele erinnern sich immer noch an die Lösungswege.

Die Informatik begeisterte mich bereits früher. Ich hatte schon früh eine Leidenschaft für das Technische. In der sechsten Klasse wurde dies weiter gefördert, als unsere Klasse die einzige war, die eigene Computer im Schulzimmer hatte. Natürlich begann alles im kleinen Rahmen. Am Gymnasium wählte ich das Ergänzungsfach Informatik. Den Anfang fand ich sehr mühselig. Es ist ähnlich, wie wenn man eine neue Sprache lernt: Zu Beginn versteht man kaum etwas und erst nach einer Menge Aufwand kann man sich über kleine Erfolge freuen. Mit der Wahl des Ergänzungsfachs in Informatik machte ich einen grossen Schritt in diese Richtung. Dort lernte ich weitere Interessante Fakten zur Java-Programmierung.

Diese Arbeit soll dich dazu verleiten in die abstrakte Welt einzutauchen und dir zeigen, dass weder der Rubik’s Cube noch die Informatik abschreckende Themen sind. Im Gegenteil erscheint es mir sehr spannend, das Unbekannte zu entdecken und neues daraus zu lernen.

*„Lernen ist wie Rudern gegen den Strom. Sobald man aufhört, treibt man zurück“*

(Loazi, chinesischer Denker des 6. Jahrhunderts v. Chr.)

**Einleitung**

(Problemschilderung/Fragestellung, Zielsetzung, Ausgrenzungen, Lösungsweg)

Es ist ein Mysterium. Ob gross oder klein, jeder versucht sich damit. Es hat eine grosse Faszination, weil die sechs Farben immer andere Muster bilden und wenn man den Dreh einmal heraus hat, will man das Knobelspiel immer und immer wieder lösen. Es handelt sich um den Rubik’s Cube. Ein Spiel, das eine fast unbegrenzte Möglichkeit an Kombinationen hat und gleichzeitig immer ein Ganzes ist.

In meiner Maturaarbeit will ich das dreidimensionale Original auf einem Bildschirm darstelle. Dabei sollen die gleichen Manipulationen vorgenommen werden können, wie beim Original.

Da ich mich nicht mit Matrizen auskenne, habe ich mich entschieden eine Bibliothek zu verwenden, die mir die Rechenaufgaben abnimmt. Ich habe mich entschieden die Java 3D Bibliothek von Oracle zu verwenden.

**Geschichte des Rubik’s Cube**

Die Geschichte des Rubik’s Cube beginnt bei einem Dozenten an der Fakultät für Innenarchitektur. Er unterrichtete an der Universität für angewandte Künste in Budapest. Von Natur interessierten ihn Geometrie und dreidimensionale Formen. Seinen Studenten brachte er viele Modelle aus Holz, Pappe, Papier oder Plastik mit, um ihnen seine Ideen zu erklären.

Später stellte er sich selbst die Aufgabe ein dreidimensionales Objekt zu erfinden. Er wollte ein Knobelspiel erfinden, das viele Kombinationen zulässt und immer ein Ganzes ist. Bei seinen ersten Überlegungen schien ihm dies unmöglich, dass sein Modell zusammenhält und gleichzeitig in drei Achsen verdreht werden kann. Seine ersten Versuche einen Würfel mit Gummibändern zusammenzuhalten scheiterten. Nach längeren Überlegenen kam er auf die Idee, dass die innere Struktur rund sein musste, damit sich die einzelnen Bestandteile des Würfels in alle drei Richtungen verdrehen liessen. Er entschied sich die sechs verschiedenen Flächen durch Farben voneinander zu unterscheiden.

Im Jahre 1974 kam die erste Patentanmeldung für seinen Prototyp.

Der ungarische Spielzeughersteller „Politechnika“ nahm sich der Aufgabe an, den Würfel als Massenprodukt herzustellen. Wegen der Komplexität im inneren des Würfels und der schlechten inneren wirtschaftlichen Lage stellte sich dies als schwierig heraus. Erst 1979 konnten die ersten Würfel in Spielzeugläden gekauft werden.

Zwei im Westen lebende Ungaren, erreichten, dass der Würfel auch auf der anderen Seite des Eisernen Vorhang zum Massenprodukt wurde. Es handelte sich dabei um Dr. Laczi Tibor und Tom Kremer.

Dr. Tibor entdeckte den Würfel auf einer Geschäftsreise. Er erkannte sofort das Potenzial des Würfels. Er nahm den Würfel mit und erhoffte sich auf der Nürnberger Spielzeugmesse 1979 einen interessierten Spielzeughersteller zu finden. Obwohl sein Erfolg ausblieb, traf er Tom Kremer, der ihn von da an unterstützte. Die beiden Männer waren es, die den Würfel auch auf der anderen Seite des Eisernen Vorhangs Bekanntheit verschafften.

Dr. Laczi ging zurück nach Ungarn, um die Produktion anzutreiben und Kremer machte eine Welt-Tour, um Spielzeughersteller für sein Produkt zu überzeugen. Bei den grossen Firmen löste der Würfel kein grosses Interesse aus, da die Produktion zu schwierig und zu teurer erschien.

Kremer gelang es nach vielen Fehlversuchen mit der Firma „Ideal Toy Corporation“ eine Bestellung von einer Million Exemplaren auszuhandeln.

1980 hatte der Würfel auf mehreren westlichen Spielzeugmessen Erfolg. Bei der Lieferung der Würfel gab es Probleme, da die westlichen Qualitätsansprüche sehr hoch waren.

(Quelle: <http://www.rubiks-zauberwuerfel.de/home.html?geschichte.html>)

The cube celebrates its 25th anniversary on 26 July 2007. A special, limited edition Rubik’s Cube is produced to mark the occasion. World Rubik's Cube Championships are held in Budapest from 5-7 October, marking the 25thanniversary of the competition. Ernő Rubik was there to award the prizes in person.

Today Ernő Rubik is now enjoying his retirement but the Rubik’s Studio continues to design games and work with young designers in Hungary and beyond.

(Quelle: <http://www.rubiks.com/world/history.php>)

**Theoretischer Teil**

**Aufbau**

**Der Scene Graph**

Der Scene Graph ist das zentrale Grundelement von Java 3D. Es handelt sich um eine hierarchische Baumstruktur. Alle Objekte, die verwendet werden, werden in die Baumstruktur eingesetzt. Das Universe bildet die Wurzel des Scene Graph. Es bildet den dreidimensionalen Raum, in welchem sämtliche Objekte eingesetzt werden.  
Darunter folgt das Local. Es stellt den Ursprung im Raum dar. Mit ihm wird ein Koordinatensystem geschaffen. Unterhalb des Local verzweigt sich der Baum in zwei unterschiedliche Branch Graph -Objekte. Die beiden Subgrafen haben unterschiedliche Aufgaben. Der Content Branch Graph beinhaltet sämtlichen Inhalt einer Applikation. Das heisst Geometiren, Erscheinungen, Bewegungen und Beleuchtungen. Der View Branch Graph beinhaltet die Ansicht. Das heisst die Kameraposition und die Kameraausrichtung.

Da das Universe, Locale und der View Branch Graph in jedem Programm gleich aufgebaut werden, gibt es ein Hilfsobjekt, das Simple Universe. Es erstellt diese Objekte und fügt die Standartwerde ein. Ich habe das Simple Universe in meinem Programm verwendet und mich deshalb nicht tiefgehend mit dem View Branch Graph auseinandergesetzt.

Im Content Branch Graph gibt es weiter Node und Node Component Objekte. Die meisten gehören zu den Node’s.

Ein Node ist ein Knotenpunkt im Scene Graph. Wenn er am Ende eines Astes steht und somit keine Kind-Objekte hat, nennt man ihn Leaf Node. Hat es unter dem Node Kind-Objekte, so gehört er zu den Group Node Typen.

In meinem Projekt waren bei den Group Node’s die Transform Group und die Branch Group von Bedeutung. Es gibt jedoch noch andere (Bild der Klassenhirarchie von Java3D).

Ein Branch Graf wird zum Leben erweckt, wenn er in ein Locale eingesetzt wird. Alle Kind-Objekte des Branch Graf werden ebenfalls lebendig. Um Parametewr von lebenden Objekten zu verändern müssen Capabilities gesetzt werden.

Bevor ein Branch Graph in ein Local eingesetz wird, sollte er compiled werden. Dadurch wird er effiezeinter.

BranchGrafen sind die einzigen, die Kinder eines Locals sein dürfen.

Canvas3D Class

The Canvas3D class is derived from the Canvas class of the Abstract Windowing Toolkit (AWT). At

least one Canvas3D object must be referenced in the viewing branch graph of the scene graph3.

Transform3D ermöglichen Translationen und Rotation von Objekten in Java 3D. Sie werden nur mit Transformations Gruppen verwendet. Die Transform3D ist kein SceneGrapph Objekt. Er spezifiert eine Transformation der TransformGroup.

Durch das Multiplizieren von zwei Transform3D können mehrere Drehungen zusammengefügt werden. Dadurch kann ein Objekt in einem Schritt um mehrere unterschiedliche Achsen gedreht werden.

TransformGroup beinhaltet Transformationen, die sie von Transform3d erhält. Es ist die einzige Klasse, die die Orientierung, Ort und die Grösse von Objekten verändern kann.

**Arc**

Zwischen den Datenelementen im Scene Graph gibt es Beziehungen. Diese heissen Arc. Man unterscheidet auch hier zwischen zwei unterschiedlichen Beziehungen.

Ein Eltern-Kind Beziehung („Parent-Child link“) kann bei einem Group Node auftreten. Dabei kann es mehrere Kind-Objekte geben jedoch immer genau ein Eltern-Objekt.

Eine Referenz („referenz“) verbindet einen Node Component –Objekt mit einem Szenengraf-Node.

Node Componenten beschreiben die Gestaltungs- und das Aussehens-Attribute.

Ein Node bildet im Szenengraf eine Wurzel. Von ihr führt zu jedem Node Leaf genau ein Weg. Deshalb sind auch Kreise in einem Szenengraf nicht erlaubt. Der Weg wird als Szenengraf-Weg bezeichnet.

**Praktischer Teil**

**Stein**

Mit dem Wort „Stein“ sind die einzelnen kleinen Bestandteile des Würfels gemeint.

**Wände**

Unter diesem Begriff verstehe ich eine Seite mit neuen Steinen. Zu beachten ist, dass die meisten Steine in mehreren Wänden vorkommen.

**Würfel**

Unter diesem Begriff verstehe ich die Gesamtheit des Rubik’s Cube

**Vorgehen**

Das Vorgehen habe ich in vier Teile unterteilt. Der erste Teil beinhaltet die Planung meiner Arbeit. Der zweite beschreibt die Informationsbeschaffung. Der dritte Teil die Informationsverarbeitung. Der vierte die Erstellung des Produkts

**Planung**

Bei früheren Arbeiten hatte ich immer wieder Zeitprobleme. Aus diesem Grund habe ich mich entschieden für diese Arbeit einen genauen Zeitplan zu erstellen, indem ich wichtige Etappenpunkte einzeichnete. Im Vorfeld erwies sich diese Arbeit als sehr vage, weil ich nicht genau wusste, auf was für Probleme ich in den einzelnen Teilschritten stossen würde.

Bei der Planung setzte ich zuerst die grossen Meilensteine. Der Abgabetermin war einer dieser Punkte. Danach überlegte ich mir, wo der grösste Zeitaufwand bestand. Diese Zeitpunkte verlegt ich so gut es ging in meine Ferien, da ich zu dieser Zeit am wenigsten unvorhergesehene Aufgaben haben würde. Im nächsten Schritt überlegte ich mir, wann ich mit meiner Betreuungsperson Rücksprachen halten sollte, um allfällige Unklarheiten zu bereinigen.

(Einfügen des Zeitplans)

**Informationsbeschaffung**

Meine erste Informationssuche begann auf dem Internet. Dort hatte ich keinen Erfolg. Da zu dieser Zeit alle unserer Klasse auf der Suche nach Informationen für ihre Arbeiten waren, halfen uns mehrere Lehrpersonen, indem sie mit uns in verschiedenen Bibliotheken führten und uns weitere Tipps zur Informationsbeschaffung gaben. Ich hatte in den Bibliotheken leider keinen Erfolg, da mein Thema sehr spezifisch ist. Vielleicht hatte ich auch nicht die richtigen Fragen gestellt. Ich wüsste zu dieser Zeit in keiner Weise, wie ich meine Arbeit umsetzten wollte.

Durch einen glücklichen Zufall lernte ich einen Studenten der ETH-Zürich kennen, der sich selbst mit dem Rubik’s Cube beschäftigte. Von ihm konnte ich das ersten Bücher auslehnen, welche sich mit dem Problem des Raumes in der Informatik beschäftigten und allgemein mit Java. Jedoch waren diese Bücher sehr theoretisch. Meine Betreuungsperson gab mir danach den guten Rat, ich solle die Java 3D Bibliothek verwenden. Für diese Bibliothek fand ich im Interneteinige Tutorial, welche ich als Hauptquelle bis zum Ende der Arbeit verwende.

**Informationsverarbeitung**

**Der Rubik’s Cube**

Zu Beginn dieses Punktes habe ich mich mit intensiv mit dem ersten Teilgebiet meiner Arbeit beschäftigt: Dem Rubik’s Cube. Dabei galt meine grösste Aufmerksamkeit dem Aufbau des Würfels.

Als erste wichtige Erkenntnis, gilt, dass sich der Würfel auf neun Arten verdrehen lässt. (Ob die Rotation Vorwärts oder Rückwärts ausgeführt wird, habe ich nicht berücksichtigt, weil durch die Wiederholung einer Drehung das gleiche Ergebnis erzielt werden kann, wie mit einer Rückwärtsbewegung.) Die Rotationen lassen sich in drei Gruppen einteilen: Die eine Gruppe beinhaltet die Drehung von den drei horizontalen Ebenen um die y-Achse. Die zweite Gruppe beinhaltet drei Drehungen um die x-Achse und die letzte Gruppe enthält die Drehung von drei Ebenen um die z-Achse.

(Bild der verschieden Drehmöglichkeiten)

Wenn man den Würfel als Leihe ansieht, könnte man vermuten, dass der Würfel aus siebenundzwanzig kleinen und vom Aufbau her identischen Steinen besteht (3 \* 3 \* 3). Diese Annahme zu wiederlegen war ein weiterer wichtiger Schritt in die Richtung meines Ziels. Der Würfel besteht aus drei Steintypen. Diese Unterteilung wird aufgrund ihrer Eigenschaften gemacht. Die Anzahl der farbigen Flächen spielt dabei eine entscheidende Rolle.

(Bild des offenen Cubes)

Der Mittelstein befindet sich in der Mitte einer Fläche. Er hat nur eine farbige Oberfläche. Es gibt insgesamt sechs Steine dieses Typs pro Würfel. Sie halten den Würfel zusammen. Daher können die Mittelsteine beim Drehen der Flächen in ihrer Position nicht verändert werden. Beim meinem Würfel heisst das konkret, dass der weisse Mittelstein immer gegenüber des gelben, der rote gegenüber des orangen, der blaue gegenüber des grünen Mittelsteines stehen wird, egal wie oft der Würfel gedreht wird.

Der Kantenstein ist bereits etwas spannender. Er befindet sich zwischen zwei Mittelsteinen und hat zwei farbige Oberflächen. Insgesamt gibt es zwölf Kantensteine. Dieser Steintyp ist wesentlich freier, da er seine Position im Würfel verändern kann.

Der Eckstein liegt, wie der Name schon sagt, in den Ecken des Würfels. Er hat drei farbige Oberflächen. Die Ecksteine können ihre Position untereinander vertauschen. Insgesamt gibt es acht Ecksteine.

Der aufmerksame Leser wird bemerkt haben, dass es insgesamt nur sechsundzwanzig Steine gibt (6 + 12 + 8). In der Mitte des Würfels gibt es keinen Stein. Dort ist die Mechanik des Würfels untergebracht. Es hat sich später herausgestellt, dass diese Erkenntnis nicht von Bedeutung ist.

Nach den beiden Erkenntnissen (Anzahl der Drehungen, Unterschiede der Steintypen) habe ich mir gedacht, dass ich drei Klassen bilden könnte, die alle von einer Elternklasse Stein ableiten. Die Kinderklassen sollten dabei die einzelnen Steintypen darstellen. Die Vaterklasse sollte mindestens zwei wichtige Methoden haben. Einerseits sollte eine Methode die Steine an eine Position im Raum setzen, andererseits sollte eine zweite Methode dem Steinen eine bestimmte Richtung zuweisen.

**Java Bibliothek**

Nun als ich den Würfel eingehend studiert hatte, konnte ich damit beginnen, mir einen Plan für die Programmierung zu überlegen. Als ich die Bibliothek endlich gefunden hatte, schrieb sich das Programm nicht selbst. Es kam eine zeitintensive Einarbeitungsphase. Während dieser Zeit las ich die verschiedenen Kapitel und versuchte die beschriebenen Beispiele abzuschreiben und zum Laufen zu bringen.

Was ich zum meiner Verblüffung feststellte, war, dass ich kaum Mathematik verwenden musste, da die Bibliothek alle Rechenarbeiten für mich übernahm. Wenn ich noch sämtliche mathematischen Probleme hätte lösen müssen, hätte mir mein Projekt den Zeitrahmen gesprengt.

Als ich anfing die Programme, die ich abschrieb langsam zu begreifen, versuchte ich sie immer ein wenig meinem Problem anzupassen. Dadurch konnte ich mich meinem Projekt langsam annähgern.

**Erstellung des Produkts**

Projekt Enddokumentation

Das Projekt besteht aus vier Klassen: der Stone-Klasse, der RubiksCube-Klasse, der GUI-Klasse und der Main-Klasse

Die Stone-Klasse hat die Hauptaufgabe einen einzelnen Stein zeichnen zu lassen. Der Konstruktor ruft zwei Methoden auf. Die Methode createGeometry(flaot x, flaot y, float z) ist für die Erstellung der Geometrie zuständig. Durch die drei übergebenen Parameter werden sechs Punkte ausgerechnet, die in drei Array mit zwei Elementen eingefügt werden. Diese Punkte werden danach in ein dreidimensionales alle acht Eckpunkte ausgerechnet und in einem Point3f-Objekt eingefügt. Diese Punkt werden danach verwendet, um die Polygone zu zeichnen.

**Ergebnisse/Diskussion**

**Literaturverzeichnis**

**Anhang**

**Selbstständigkeitserklärung**