Graphicus

Programm zur Darstellung von dreidimensionalen Graphen

Paul Stahr, Yakup Ates

9. November 2015

Inhaltsverzeichnis

1 Verwendete Software **6**
1.1 OpenGl . . . 6
1.2 Java . . . 6
1.3 LWJGL . . . 6

2 Installation/Systemanforderungen **7**
2.1 Windows . . . 7
2.2 Linux . . . 7
2.2.1 LWJGL Installation . . . 7
2.3 Aktivierung . . . 8

3 Das erste Projekt **9**

4 User Interface **12**
4.1 Hauptfenster . . . 12
4.1.1 Menüleiste . . . 12
4.1.2 Toolmenü . . . 13
4.1.3 Anzeigefenster der Graphen . . . 14
4.1.4 Liste der Graphen . . . 14
4.2 Graphen . . . 15
4.2.1 Graphentypen . . . 15
4.2.2 Transformationsmatrix . . . 16
4.3 Zeichen . . . 16
4.4 Optionen . . . 16
4.4.1 Bild . . . 16
4.4.2 Licht/Umgebung . . . 17
4.4.3 Erweitert . . . 18
4.5 Projektinformationen . . . 18
4.6 Update . . . 18
4.7 Matrix Erstellen . . . 18
4.8 Log . . . 19
4.9 Info . . . 19

5 Datentypen **20**
5.1 Integer . . . 20
5.2 Fließkommazahl . . . 20
5.3 Zeichen . . . 20
5.4 String . . . 20
5.5 Liste . . . 20

6 Wichtige Zeichen **21**
6.1 Verknüpfungen . . . 21
6.2 Mengen . . . 21

6.3 Zahlen . . . 22

7 Funktionen und Rechenoperationen 23
7.1 Grundrechenarten . . . 23
7.1.1 Addition . . . 23
7.1.2 Subtraktion . . . 23
7.1.3 Multiplikation . . . 23
7.1.4 Matrizenmultiplikation . . . 23
7.1.5 Kreuzprodukt . . . 23
7.1.6 Skalarprodukt . . . 23
7.1.7 Division . . . 23
7.1.8 Modulo . . . 23
7.2 Konvertierungen . . . 24
7.2.1 Integer . . . 24
7.2.2 Float . . . 24
7.2.3 Char . . . 24
7.2.4 Zeichenkette . . . 24
7.2.5 Compile . . . 24
7.2.6 Char zu Zeichenkette . . . 24
7.2.7 Format . . . 25
7.3 Zufall . . . 25
7.3.1 Zufallszahl . . . 25
7.3.2 Liste mit Zufallszahlen . . . 25
7.3.3 Matrix mit Zufallszahlen . . . 25
7.4 Binär . . . 25
7.4.1 Nicht . . . 25
7.4.2 Und . . . 25
7.4.3 Oder . . . 25
7.4.4 Gleich . . . 26
7.4.5 Ungleich . . . 26
7.4.6 Kleiner . . . 26
7.4.7 Größer . . . 26
7.4.8 Kleiner gleich . . . 26
7.4.9 Größer gleich . . . 26
7.4.10 Element aus . . . 26
7.4.11 Nicht Element aus . . . 26
7.4.12 Teilmenge aus . . . 26
7.4.13 Nicht Teilmenge aus . . . 26
7.5 Vereinigung . . . 26
7.6 System . . . 27
7.6.1 Garbage Collector . . . 27
7.6.2 Speichern . . . 27
7.6.3 Lesen . . . 27
7.6.4 Schreiben . . . 27

7.7 Runden . . . 27
7.8 Maximum . . . 28
7.8.1 Maximum von zwei Zahlen . . . 28
7.8.2 Maximum aus einer Liste . . . 28
7.9 Minimum . . . 28
7.9.1 Minimum von zwei Zahlen . . . 28
7.9.2 Minimum aus einer Liste . . . 28
7.10 Absolut . . . 28
7.11 Kardinalität . . . 28
7.12 Größter gemeinsamer Teiler . . . 29
7.13 Kleinstes gemeinsames Vielfaches . . . 29
7.14 Kombinatorik . . . 29
7.14.1 Anzahl der Permutationen . . . 29
7.14.2 Anzahl der Kombinationen . . . 30
7.15 If Abrage . . . 31
7.16 Abfrage von Benutzereingabe . . . 31
7.17 Warten . . . 31
7.18 Wurzel . . . 31
7.18.1 Quadrat Wurzel . . . 31
7.18.2 N-te Wurzel . . . 32
7.18.3 Würfel . . . 33
7.19 Potenz . . . 33
7.20 Trigonometrisch . . . 34
7.20.1 Sinus . . . 34
7.20.2 Cosinus . . . 35
7.20.3 Tangens . . . 35
7.20.4 Hyperbolischer Sinus . . . 35
7.20.5 Hyperbolischer Cosinus . . . 36
7.20.6 Hyperbolischer Tangens . . . 36
7.21 Variablen . . . 36
7.21.1 Definiere . . . 36
7.21.2 Zuweisung . . . 36
7.21.3 Lösche . . . 36
7.22 Löse . . . 37

8 Interessante/Formschöne Graphen 38
8.1 Zwei Wellen . . . 38
8.2 Kugel . . . 38
8.3 Limette . . . 38
8.4 Donut . . . 39
8.5 Schnecke . . . 39
8.6 Kleinsche Flasche . . . 39

9 Dateien **40**
9.1 Log 40
9.2 Optionen 40
9.3 Cubemaps 40
9.4 Graph 40
9.5 Off 40
9.6 Jar 40

10 Häufige Probleme **41**
10.1 Programm startet nicht 41
10.2 Kein 3D-Fenster 41

1 Verwendete Software

1.1 OpenGl

OpenGl (Open Graphics Library) ist eine Betriebssystem- und Programmiersprachenunabhängige Schnittstelle, zur Entwicklung von 2D- und 3D- Computergrafik. Die erste Version wurde am 1. Juli 1992 veröffentlicht. Zur Beschleunigung der Berechnung wird die Grafikkarte verwendet. Heutzutage gibt es viele Spiele, aber auch wissenschaftliche Programme die OpenGl zum Rendern ihrer Inhalte benutzen.

1.2 Java

Die erste Version der Programmiersprache Java ist im Jahr 1995 erschienen. Java ist eine Plattformunabhängige, objektorientierte Programmiersprache. Mit einer passenden Laufzeitumgebung sind Programme in Java ohne Änderung des Quellcodes auf jedem Betriebssystem lauffähig.

1.3 LWJGL

Trotz der Programmiersprachenunabhängigkeit von OpenGl ist es dennoch nötig eine Schnittstelle zwischen Java und OpenGl zu haben. LWJGL ist eine API die dies übernimmt. Es ist möglichst klein gehalten, und überträgt die Befehle, die zur Steuerung von OpenGl nötig sind mit möglichst geringer Aufrufzeit in Java, sodass sie dort dann aufrufbar sind. Die offizielle Website ist `http://lwjgl.org`.

2 Installation/Systemanforderungen

Das Programm benötigt von sich aus keine Installation, es kann in jedes beliebige Verzeichnis oder auf jeden USB-Stick gezogen werden. Dennoch müssen bestimmte Voraussetzungen erfüllt sein:

1. Das System muss OpenGl fähig sein. Andernfalls kann nur ein sehr geringer Teil des Programms genutzt werden.
2. Das System sollte über 512 MB Ram, 1Ghz CPU und 128 V-Ram verfügen.
3. Die Jave-Runtine-Environment muss mit mindestens Version 6 installiert sein. Für Windows gibt es allerdings eine Version, die ohne die java-jre auskommt.
4. Es sollte Schreibezugriff auf das Programm geben, da sonst keine Log-Dateien geschrieben werden können.

2.1 Windows

Am sinnvollsten ist es die Standardversion des Programms unter Windows zu benutzen. Falls Sie jedoch keine JVM auf ihrem System haben, dann können sie die Windows Version benutzen, die komplett ohne weitere Software auskommt.

2.2 Linux

LWJGL muss auf dem System installiert sein. Das Programm benötigt mindestens Version 1.7.2.

2.2.1 LWJGL Installation

Die Installation von LWJGL unterteilt sich in wenige Schritte. Sie müssen die Grundbefehle des Terminals benutzen können und eine Verbindung zum Internet haben.

1. Laden Sie *LWJGL* runter (z.B. unter `http://lwjgl.org`).
2. Entpacken Sie die archivierte Datei und öffne Sie den Entpackten Ordner.
3. Wechseln sie in den Ordner ./lwjgl-[VERSIONSNUMMER]/native/linux.
4. Kopieren Sie die Dateien die sich im Ordner "linux" befinden, in ihren /usr/lib Ordner. (Adminstratorrechte erforderlich)
5. Sollte sich LWJGL noch in einem anderen lib-Ordner befinden löschen sie dieses. Folgende Dateien sind Teil von lwjgl:

   „libjinput-linux64.so“„liblwjgl64.so“„libopenal64.so“
   „libjinput-linux.so“ „liblwjgl.so“ „libopenal.so“

6. Aktualisiere Sie die Liste der installierten Bibliotheken mit "sudo ldconfig".

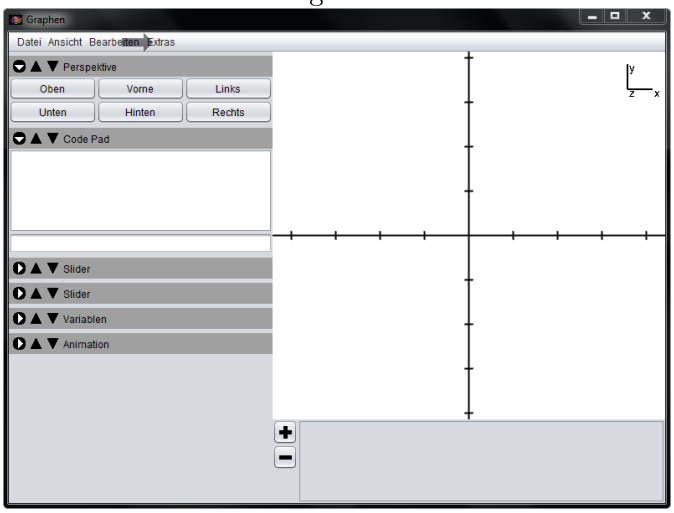
Falls keine Fehler aufgetreten sind, haben Sie nun *LWJGL* erfolgreich installiert.

2.3 Aktivierung

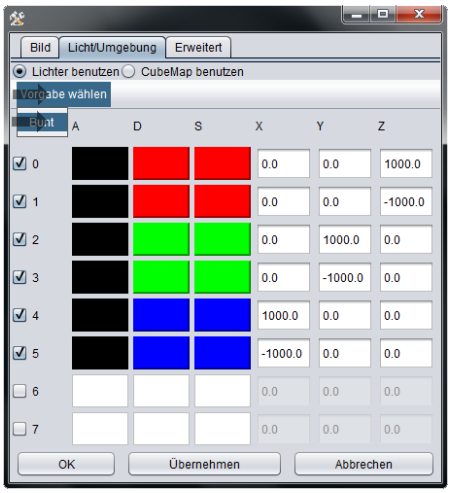
Nach erfolgreicher Installation öffnet sich beim Starten ein Fenster in dem sie gebeten werden, das Produkt zu aktivieren. Ohne Aktivierung sind speicher, lade und exportfunktionen nicht nutzbar. Die Aktivierung ist möglich indem sie eine E-mail an paul.stahr@gmx.de mit der angezeigten Id senden. Ihnen wird dann in den nächsten Tagen ein Code zugeschickt, den sie in dieses Fenster eintragen können. Der Code Funktioniert nur für das Entsprechende System. Selbstverständlich müssen sie die Software bei Updates nicht erneut Aktivieren, es sei denn dies wird explizit im Changelog erwähnt.

3 Das erste Projekt

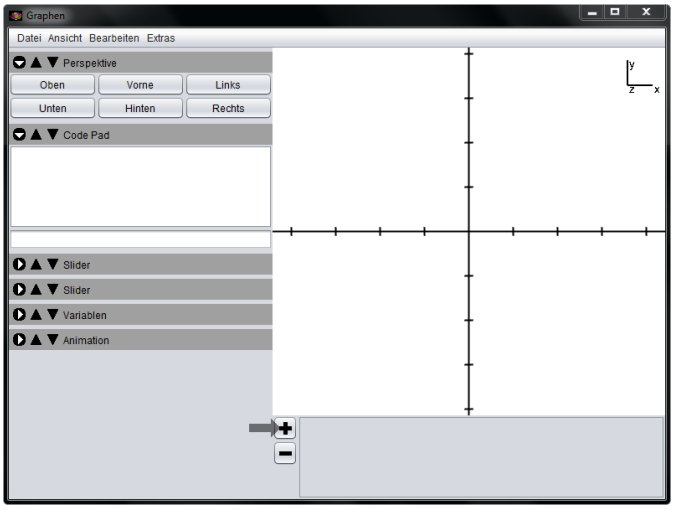
Dies ist eine Anleitung um einen einfachen Graphen zu zeichnen.
Nach dem Start des Programmes sollte das Fenster in etwa so aussehen:



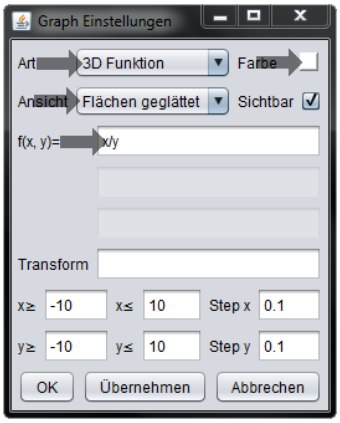
1. Gehen sie als erstes auf Extras und dort auf Optionen.



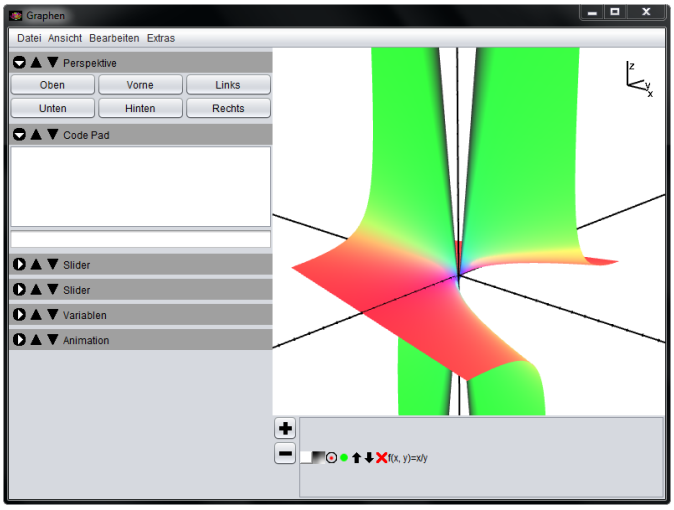
1. Wählen sie die Registrierkarte Licht/Umgebung.
2. Gehen sie auf Vorgabe Wählen und dort auf Bunt.
3. Wenn alles der Abbildung entspricht, dann können sie auf OK drücken.



1. Um einen Graphen hinzuzufügen klicken sie auf den Plusbutton.



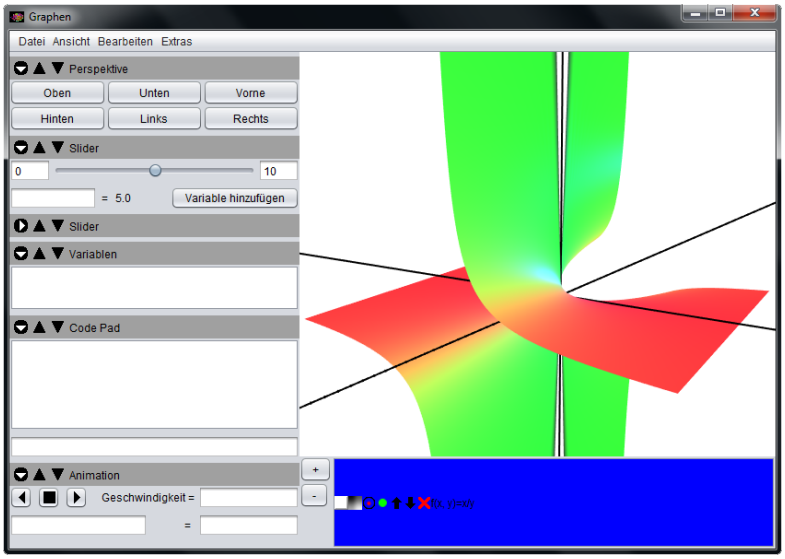
1. Wählen sie als Typ 3D-Funktion
2. Die Ansicht stellen sie auf Flächen geglättet
3. Die Farbe stellen sie auf weiß
4. Als Funktion geben sie „x/y“ein.
5. Klicken sie auf OK



Nun können sie mittels Rechts-, Linksklick und dem Mausrad die Perspektive im 3D-Fenster ändern. Wenn die Ansicht bei ihnen wie in der Abbildung aussieht, haben sie das Tutorial erfolgreich beendet.

4 User Interface

4.1 Hauptfenster



Dies ist das Hauptfenster des Programms. Es unterteilt sich in vier Bereiche:
1.Menüleiste
2.Toolmenü
3.Anzeigefenster für Graphen
4.Funktionen der aktuellen Graphen

4.1.1 Menüleiste

Die Menüleiste ist oben zu sehen, diese unterteilt sich in 3 teile, 'Datei', 'Ansicht', 'Bearbeiten' und 'Extras'. Die Option 'Datei' unterteilt sich in in 7 weitere Optionen.

1. Neu: Stellt alles wieder auf "default", somit werden aktuelle Graphen, Berechnungen etc. gelöscht. Das Programm ist bereit für ein neues Projekt.

2. Öffnen: Ermöglicht dem Benutzer gespeicherte Projekte wieder aufzurufen.

3. Zuletzt geöffnet: Stellt die letzten 9 geöffneten Projekte dar.

4. Speichern: Falls gerade mit einem bereits gespeichertem Projekt gearbeitet wird, kann hier die Datei mit den aktuellen Einstellungen überschrieben werden, ohne eine neue Datei angeben zu müssen.

5. Speichern ... :Speichert das aktuelle Projekt an einem vom Benutzer ausgewählten Ort.

6. Exportieren: Exportiert das Projekt, oder Teile davon in ein anderes Dateiformat.

7. Beenden: Beendet das Programm.

In 'Ansicht' kann der Benutzer unter 3 Optionen wählen.

1. Tools anzeigen: Ermöglicht das Ein-/Ausblenden der Leiste der linken Seite, der Benutzer bekommt somit ein größeres Fenster für den Graphen.

2. Graphenmenü anzeigen: Ermöglicht das Ein-/Ausblenden der Leiste der unteren Seite, wo die Funktionen aufgelistet werden. Der Benutzer bekommt so ein größeres Fenster für den Graphen.

3. Log anzeigen: Zeigt die Meldungen auf der Konsole, die für den Benutzer hilfreich sein könnten, um mögliche Fehler zu beheben.

In 'Bearbeiten' kann der Benutzer unter 5 Optionen wählen.

1. Kopieren: Noch nicht Implementiert

2. Ausschneiden: Noch nicht Implementiert

3. Einfügen: Noch nicht Implementiert

4. Projekteinstellungen: Ermöglicht das eintragen von Name, Autor und eine Beschreibung zum aktuellen Projekt. Siehe 4.5.

In 'Extras' kann der Benutzer unter 6 Optionen wählen.

1. Hilfe: Öffnet diese über das Programm, um Hilfe für den Benutzer zu gewährleisten.

2. Zeichentabelle: Stellt 40 nützliche Sonderzeichen dar, die man kopieren und im Programm verwenden kann. Siehe 4.3.

3. Auf updates prüfen: Prüft das Programm auf neuste Versionen, und stellt eine Liste der Verbesserungen von Version zu Version dar.Siehe 4.6.

4. Einstellungen: Hier können Einstellungen für das Programm getroffen werden.

5. LWJGL-Lizenz: Hier wird die LWJGL-Lizenz dargestellt(Englisch).

6. Über ... : Hier sind die Bezeichnung des Programms, die Version und die Autoren des Programms aufgeführt. Siehe 4.9.

4.1.2 Toolmenü

Das Toolmenü ist auf der linken Seite des Programms und unterteilt sich in viele Bereiche.

1. Perspektive: Der Betrachtungswinkel zum Graphen kann hier gewählt werden.

2. Slider: Hier können Variablen deklariert und anhand eines Sliders auf einen Wert gesetzt werden.

3. Variablen: Hier werden die deklarierten Variablen aufgelistet.

4. Code Pad: Hier können Berechnungen gemacht werden, es ist gleichzeitig eine Konsole und ein Taschenrechner. Um die Globalen Variablen in die Rechnungen mit einzubezeihen muss zusätzlich zu Enter noch Shift gedrückt werden.

5. Animation: Eine Variable wird hier pro Sekunde um einen gegebenen Wert erhöht, um Animationen mit Graphen zu ermöglichen.

4.1.3 Anzeigefenster der Graphen

Hier werden die Graphen angezeigt, außerdem wenn erwünscht Informationen über Framerate und Speicherauslastung. Die Steuerung der Perspektive ist mit Maus und Tastatur möglich:
Maus:

Tasten	Funktion
links	dreht alles um die Kamera
rechts	dreht alles um den Nullpunkt
Mausrad	Bewegung nach vorne/hinten

Tastatur

Tasten	Funktion
Num links	Bewegung nach links
Num rechts	Bewegung nach rechts
Num oben	Bewegung nach oben
Num unten	Bewegung nach unten
Num plus	Bewegung nach vorne
Num minus	Bewegung nach hinten
Pfeil links	Drehung nach links
Pfeil rechts	Drehung nach rechts
Pfeil oben	Drehung nach unten
Pfeil unten	Drehung nach oben

4.1.4 Liste der Graphen

Hier wird der Graph mit dem vom Benutzer gewählten Einstellungen dargestellt, vorausgesetzt, dass das Graphenmenü unter 'Ansicht' nicht Ausgeblendet wurde. Es sind pro Graph 7 Symbole auf der linken Seite zu finden. Die Symbole werden von Links nach Rechts erklärt:

1. Beim ersten Symbol kann die Farbe für die Funktion die als Graph dargestellt werden soll, ausgewählt werden.

2. Beim zweiten Symbol von links gibt es 4 Möglichkeiten den Graphen darzustellen:

   Punkte.

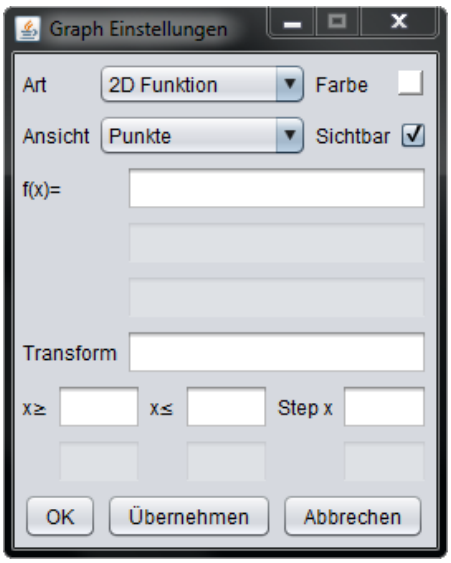
   Linien.

   Fläche.(Nur bei 3D möglich)

Fläche geglättet. (Nur bei 3D möglich)

3. Ausgewählte Funktion Einblenden: Ausblenden:
4. Funktion wird berechnet: Funktion ist berechnet:
5. Funktion nach oben verschieben: Funktion nach unten verschieben:
6. Ausgewählte Funktion löschen:

4.2 Graphen



Hier können die Einstellungen für einen Graphen gemacht werden.

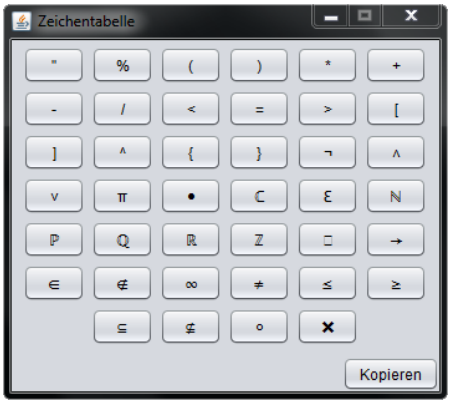
4.2.1 Graphentypen

2D Funktion	Ordnet jedem x einen y zu.
2D Parametrisch	Ordnet jedem t einen x und y zu
2D Polar	Bildet die Funktion in einem Kreis ab
2D Plot	Macht zwei Listen als Plot sichtbar
3D Linie	Ordnet jedem t ein x, y und z zu
3D Funktion	Ordnet jedem x und y ein z zu
3D Parametrisch	Ordnet jedem u und v ein x, y und z zu
3D Polar	noch nicht implementiert
3D Plot	Macht drei Listen als Plot sichtbar
3D Kartesisch	Funktionen der Form x*a+y*b+z*c=d

4.2.2 Transformationsmatrix

Zur Transformation kann hier eine $4 \cdot 4$ Matrix eingegeben werden. Die Funktion wird dann anhand dieser Matrix im homogenen Koordinatensystem transformiert.

4.3 Zeichen

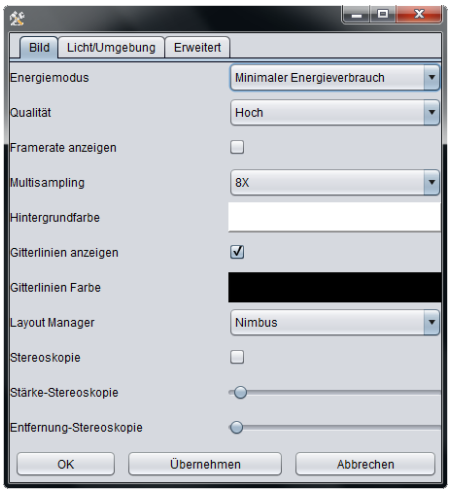


Hier findet man alle Zeichen, die für das Programm wichtig sind. Man kann sie bei bedarf mit dem Button „Kopieren“in die Zwischenablage legen.

4.4 Optionen

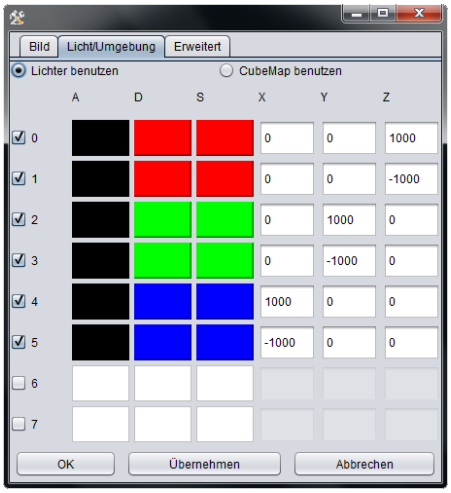
In den Optionen sind alle Einstellungen zu finden, die für das Programm wichtig sind. Alles was hier eingestellt werden kann wird in der Options-datei gespeichert und ist somit auf dem gesamten System gültig.

4.4.1 Bild



In diesem Fenster ist es möglich allgemeine Einstellungen zum Bild zu machen.

4.4.2 Licht/Umgebung



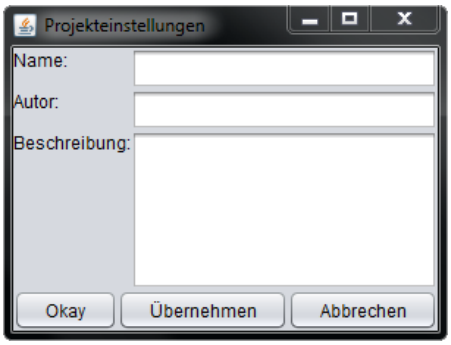
Hier können die Einstellungen Für das Licht, oder eine CubeMap die von den Graphen gespiegelt wird, gemacht werden. Da die Berechnung von
echtem Licht viel zu aufwändig wäre vereinfacht man das Modell und teilt das Licht in drei Gruppen ein.

Lichtart	Abkürzung	Effekt	Positionsabhängig
Ambient	A	Grundlicht	Nein
Diffuse	D	Matte Reflektion	Lichtquelle zu Objekt
Specular	S	Glanzeffekte	Lichtquelle zu Objekt und Kamera

Cubemaps sind 6 Bilder, die wie ein Würfel um die Szene angeordnet werden. Diese Bilder können dann reflektiert werden. Dies ist zwar nicht mit dem sogenanten Raytracing zu vergleichen, da Graphen nicht sich selbst spiegeln können, es sorgt aber dennoch für sehr schöne Effekte. Das Verzeichnis der Cubemap kann frei gewählt werden. Eine Beispielmap befindet sich im Programmordner unter „/data/cubemaps/1/“.

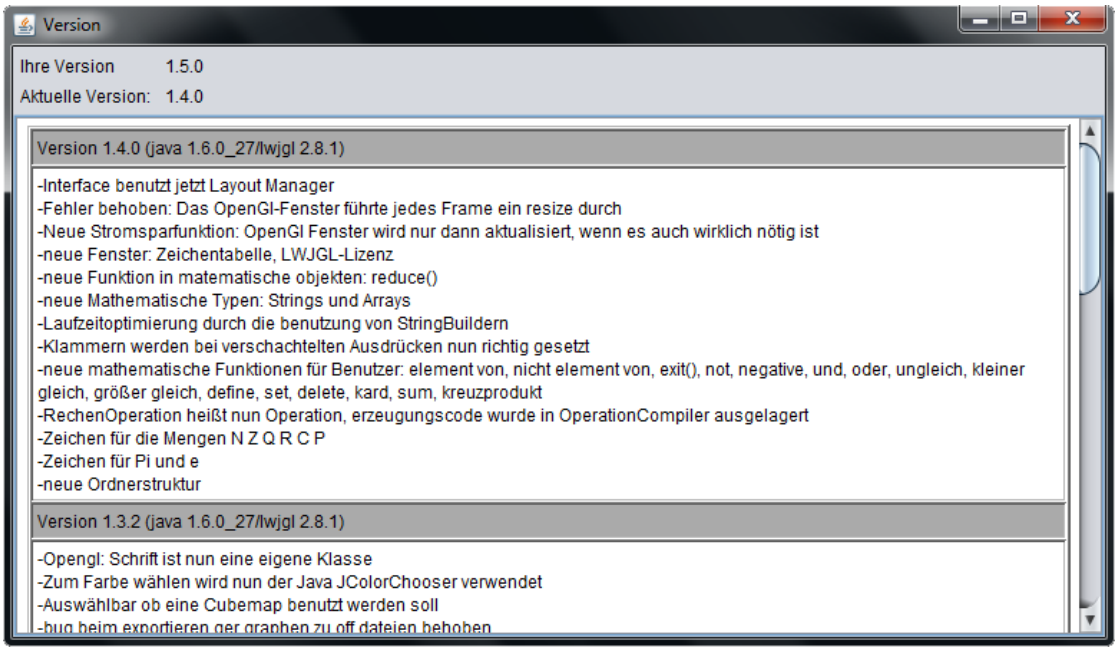
4.4.3 Erweitert

4.5 Projektinformationen



Hier können allgemeine Projektinformationen eingestellt werden.

4.6 Update

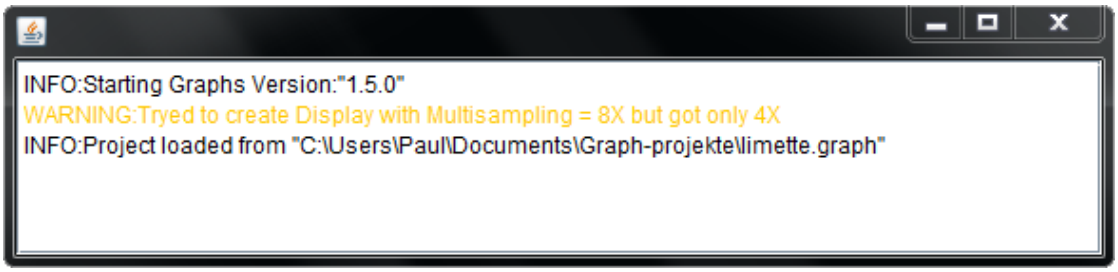


Hier wird die eigene, sowie die aktuelle Version des Programms angezeigt. Zusätzlich kann man einsehen, was sich in den Versionen geändert hat. Damit das Changelog und die aktuelle Version angezeigt werden können, ist eine Verbindung zum Internet nötig.

4.7 Matrix Erstellen

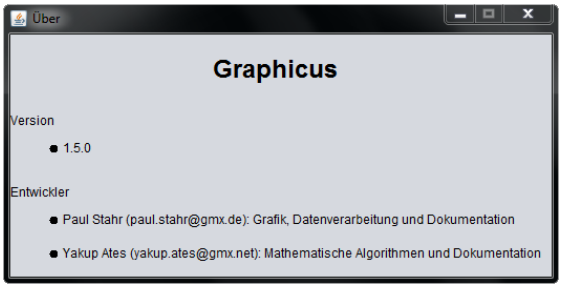
In diesem Fenster wird eine Erleichterung zum Erstellen von $4 \cdot 4$ Matrizen kommen. Es ist jedoch noch nicht fertig implementiert.

4.8 Log



Im Log kann man eventuelle Fehler des Programms feststellen, oder einfach Informationen über das was getan wurde erhalten. Hier erscheinen allerdings nur Meldungen für den Benutzer. Deutlich detailliertere Informationen erhält man in den Log-Dateien. Siehe 9.1.

4.9 Info



Im Informationsfenster kann man die Autoren und die Version des Programms sehen.

5 Datentypen

5.1 Integer

Ein Integer ist eine Ganzzahl. Der Wertebereich den ein Integer umfassen kann liegt bei -2^{63} bis $2^{63} - 1$. Sollte dieser Bereich bei einer Berechnung überschritten werden wird die Berechnung mit Fließkommazahlen durchgeführt. Um einen Wert zu einer Ganzzahl zu konvertieren kann die Int-Funktion genutzt werden.

5.2 Fließkommazahl

Bei einer Fließkommazahl ist die Positions des Kommas nicht fest, es kann fließen. Das ermöglicht es sehr große aber auch sehr kleine Zahlen zu verarbeiten. Der Wertebereich liegt bei $-1.79769313486231570^{308}$ bis $1.79769313486231570^{308}$. Beim überschreiten dieses Bereiches wird die Zahl durch ein ∞ oder NaN ersetzt. Bei Rechnungen mit einer Ganzzahl und einer Fließkommazahl wird die Ganzzahl vor der Rechnung automatisch in eine Fließkommazahl umgewandelt. Die Fließkommazahl hat gegenüber der Ganzzahl den Nachteil, dass sie immer nur etwa 15 Stellen-Genauigkeit besitzt. Um eine Zahl zu einer Fließkommazahl zu konvertieren, kann die Float-Funktion genutzt werden.

5.3 Zeichen

Ein Zeichen ist dadurch gekennzeichnet, dass sich an jeder Seite ein ' befindet. In der Informatik ist jedem Zeichen ein Wert zugeordnet. Bei diesem Programm gelten die Zuordnungen aus dem unicode-Zeichensatz. Um eine Zahl zu dem zugehörigen Zeichen zu konvertieren kann die char-Funktion verwendet werden.

5.4 String

Ein String ist eine Kette von Zeichen. Sie wird dadurch gekennzeichnet, dass sich an Anfang und Ende des Strings Anführungszeichen befinden. Es können alle Zeichen hineingeschrieben werden, außer Anführungszichen. Um eine Rechnung zu einem String zu konvertieren, kann die String-Funktion genutzt werden.

5.5 Liste

Eine Liste kann mehrere Objekte enthalten. Sie ist durch geschweifte Klammern gekennzeichnet. Eine verschachtelte Liste mit weiteren Listen in der Liste kann zusätzlich als Matrix interpretiert werden. Bei Rechnungen mit Listen wirken sich die meisten Operationen so aus, dass die Zahl bzw. jedes Element der ersten Liste mit jedem Element der zweiten Liste verknüpft wird, und daraus eine neue Liste entsteht. Die Liste ist das einzige Objekt, dass nicht immun gegen Veränderungen ist. Es können einzelne Werte ausgetauscht werden.

6 Wichtige Zeichen

6.1 Verknüpfungen

Zeichen	Name	Rückgabetyp
"	Anführungszeichen	
%	Modulo (7.1.8)	Zahl
*	Multiplikation (7.1.3)	Zahl
+	Addition/Vorzeichen (7.1.1)	Zahl
-	Subtraktion/Vorzeichen (7.1.2)	Zahl
/	Division (7.1.7)	Zahl
$<$	Kleiner (7.4.6)	Wahrheitswert
$=$	Gleich (7.4.4)	Wahrheitswert
$>$	Größer (7.4.7)	Wahrheitswert
^	Expon	Zahl
$\neg$	Nicht (7.4.1)	Wahrheitswert/Ganzzahl
$\curlywedge$	Und (7.4.2)	Wahrheitswert/Ganzzahl
$\curlyvee$	Oder (7.4.3)	Wahrheitswert/Ganzzahl
$\bullet$	Multiplikation für Matrizen (7.1.4)	Matrix
$\rightarrow$	Zuweisung	Rechnung
$\in$	Element aus (7.4.10)	Wahrheitswert
$\notin$	Nicht Element aus (7.4.11)	Wahrheitswert
$\neq$	Ungleich (7.4.5)	Wahrheitswert
$\leq$	Kleiner gleich (7.4.8)	Wahrheitswert
$\geq$	Größer gleich (7.4.9)	Wahrheitswert
$\subseteq$	Teilmenge aus (7.4.12)	Wahrheitswert
$\nsubseteq$	Nicht Teilmenge aus (7.4.13)	Wahrheitswert
$\circ$	Vereinigung	String
$\times$	Kreuzprodukt	Matrix

6.2 Mengen

Zeichen	Name
$\mathbb{C}$	Komplexe Zahlen
$\mathbb{N}$	Natürliche Zahlen
$\mathbb{P}$	Primzahlen
$\mathbb{Q}$	Rationale Zahlen
$\mathbb{R}$	Reelle Zahlen
$\mathbb{Z}$	Ganze Zahlen

6.3 Zahlen

Zeichen	Name
∞	Unendlich
π	Konstante Kreiszahl Pi
e	Eulersche Konstante
i	Imaginäre Zahl

7 Funktionen und Rechenoperationen

7.1 Grundrechenarten

Grundsätzlich gilt bei allen Rechenarten, dass wenn zwei Ganzzahlen verknüpft werden, das Ergebinis wenn möglich ebenfalls eine Ganzzahl ist. Sollte die Rechnung jedoch eine Kommazahl oder eine Zahl die den Wertebereich der Ganzzahlen über oder unterschreitet ergeben, so wird eine Fließkommazahl zurückgegeben. Außerdem werden Berechnungen beliebig umgestellt um schon möglichst viel zu Berechnen. Beispielsweise ergibt $2*(b/3) = 0.6666666666666666 * b$, wenn die Variablen erst später eingegeben werden.

7.1.1 Addition

Mit dem Zeichen „+“können zwei Zahlen addiert werden.

7.1.2 Subtraktion

Mit dem Zeichen „-“können zwei Zahlen subtrahiert werden.

7.1.3 Multiplikation

Mit dem Zeichen „*“können zwei Zahlen miteinander multipliziert werden.

7.1.4 Matrizenmultiplikation

Da bei einer normalen Multiplikation Matrizen als Listen gewertet werden, gibt es ein Zeichen, dass extra dafür ist Matrizen zu Multiplizieren. Dieses Zeichen ist: „•“.

7.1.5 Kreuzprodukt

Das Kreuzprodukt lässt sich auf zwei Vektoren anwenden. Eine Matrix ist dann ein Vektor, wenn sie nur eine Spalte besitzt. Das Ergebnis ist ebenfalls ein Vektor.

7.1.6 Skalarprodukt

Das Skalarprodukt zweier Vektoren lässt sich mit dem Zeichen „$*$“berechnen.

7.1.7 Division

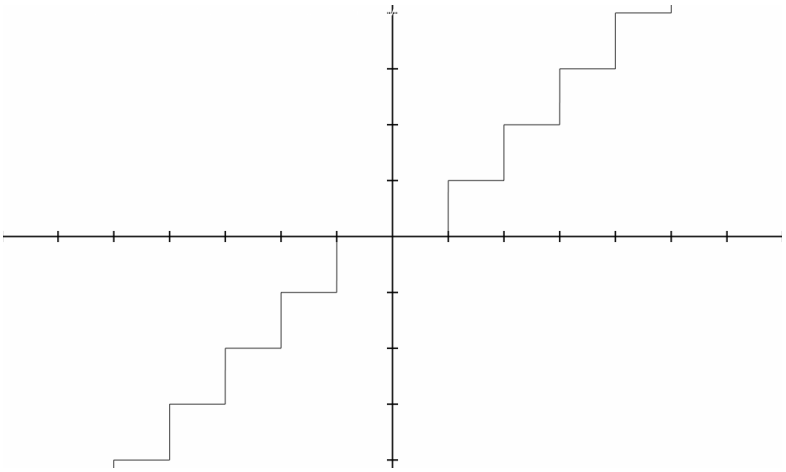
Mit dem Zeichen „/“wird eine Zahl durch eine andere geteilt.

7.1.8 Modulo

Modulo ist der Rest der bei einer Division auftritt, wenn das Ergebnis eine Ganzzahl sein soll. Die Besonderheit hier ist, dass wenn eine der beiden Zahlen negativ ist, das damit auch das Ergebnis ein negatives ist.

7.2 Konvertierungen

7.2.1 Integer



Die Funktion int() kann genutzt werden um aus einer Fließkommazahl, einem Zeichen oder String eine Ganzzahl zu erzeugen. Die Nachkommastellen werden bei einem Fließkommawert einfach abgeschnitten. Will man das nicht haben, kann man die Funktionen round(a) (siehe 7.4.9) und int(a) kombinieren. Beispielsweise ist int(round(1.6)) = 2 während int(1.6) = 1 ist.

7.2.2 Float

Die Funktion float(a) kann genutzt aus werden um aus einer Ganzzahl, Zeichen oder Zeichenkette eine Fließkommazahl zu erzeugen.

7.2.3 Char

Die Funktion char(a) erzeugt aus einer Zahl das ihm in Unicode zugeordnete Zeichen. Der Aufruf char(65) ergibt beispielsweise 'A'.

7.2.4 Zeichenkette

Die Funktion string(a) erzeugt aus einer Rechnung einen String. Die Rechnung selbst wird abei nicht verändert.

7.2.5 Compile

Die Funktion compile(a) kann genutzt werden, um aus einem String wieder eine Rechnung zu erhalten.

7.2.6 Char zu Zeichenkette

Hat man eine Liste aus Zeichen und will diese zu einem String konvertieren, kann man die Funktion chartostring(a) nutzen.

7.2.7 Format

Mit der Funktion format(a,b) ist es möglich eine Rechnung zu einer Zeichenkette für andere Programme zu konvertieren. Die Variable b ist die Rechnung und a definiert durch eine Zeichenkette das gewünschte Programm. Es sind folgende Ausgaben möglich:

- "calgraph":Standartausgabe des Programmes
- "latex":Ausgabe für latex-Formeln
- "open_office":Ausgabe für Open-Office-Formeleditor

7.3 Zufall

7.3.1 Zufallszahl

Zufallszahlen können mit der Funktion rand() erzeugt werden, wie eine Zahl zwischen 0 und 1 zurück gibt.

7.3.2 Liste mit Zufallszahlen

Eine Liste mit Zufallszahlen kann durch die Funktion randlist(x) erzeugt werden.

7.3.3 Matrix mit Zufallszahlen

Eine Matrix mit Zufallszahlen kann durch die Funktion randmat(x,y) erzeugt werden.

7.4 Binär

7.4.1 Nicht

Kehrt den Wahrheitswert um.

7.4.2 Und

Diese Funktion kann auf Wahrheitswerte sowie Ganzzahlen angewendet werden. Das Ergebnis der Operation kann an folgender Wahrheitstabelle abgelesen werden.

	true	false
true	true	false
false	false	false

7.4.3 Oder

Diese Funktion kann auf Wahrheitswerte sowie Ganzzahlen angewendet werden. Das Ergebnis der Operation kann an folgender Wahrheitstabelle abgelesen werden.

	true	false
true	true	true
false	true	false

7.4.4 Gleich

Prüft ob zwei Zahlen oder Wahrheitswerte gleich sind und gibt true oder false zurück.

7.4.5 Ungleich

Prüft ob zwei Zahlen oder Wahrheitswerte ungleich sind und gibt true oder false zurück.

7.4.6 Kleiner

Die Verknüpfung $<$ prüft ob eine Zahl kleiner ist als eine andere.

7.4.7 Größer

Die Verknüpfung $=$ prüft ob eine Zahl größer ist als eine andere.

7.4.8 Kleiner gleich

Die Verknüpfung $\leq$ prüft ob eine Zahl kleiner oder gleich einer anderen ist.

7.4.9 Größer gleich

Die Verknüpfung $\geq$ prüft ob eine Zahl größer oder gleich einer anderen ist.

7.4.10 Element aus

Die Verknüpfung $\in$ prüft ob etwas ein Element aus einer Menge oder Liste ist.

7.4.11 Nicht Element aus

Die Verknüpfung $\notin$ prüft ob etwas nicht ein Element aus einer Menge oder Liste ist.

7.4.12 Teilmenge aus

Die Verknüpfung $\subseteq$ prüft ob eine Menge die Teilmenge einer anderen ist.

7.4.13 Nicht Teilmenge aus

Die Verknüpfung $\nsubseteq$ prüft ob eine Menge nicht die Teilmenge einer anderen ist.

7.5 Vereinigung

Die Vereinigung kann genutzt werden um aus zwei Strings einen zu machen. Der Fachbegriff für diese Operation lautet Konkatenation.

7.6 System

7.6.1 Garbage Collector

Durch die Funktion gc kann der „Garbage Collector“von Java aufgerufen werden. Dadurch werden nicht mehr benötigte Ressourcen wieder freigegeben. Es gibt jedoch keine Garantie, dass der Garbage Collector wirklich aufgerufen wird. Gibt immer true zurück.

7.6.2 Speichern

Die Funktion save() Speichert das Projekt auf das aktuell geöffnete. Gibt true zurück wenn es erfolgreich war und false wenn nicht.

7.6.3 Lesen

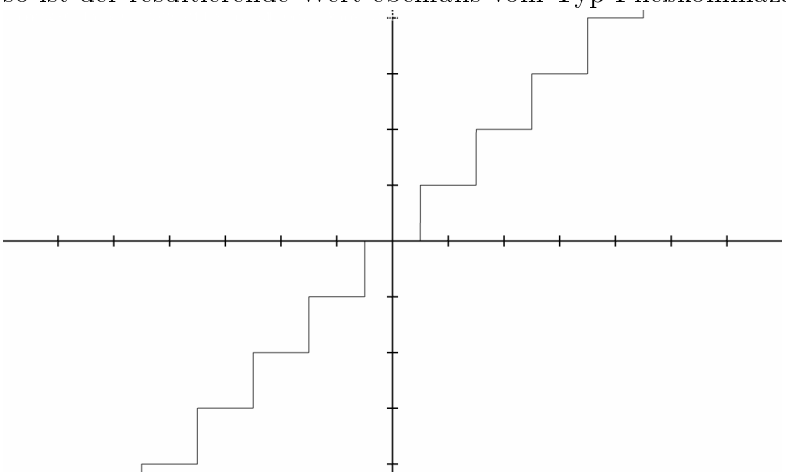
Mit der Funktion read(a) können Bilder oder Textdateien in das Programm gelesen werden. Bilder werden dadurch zu einer zweidimensionalen Liste, Textdateien zu einer Zeichenkette. Die Variable muss eine Zeichenkette mit dem dateipfad entsprechen.

7.6.4 Schreiben

Mit der Funktion write(a, b) kann eine Zeichenkette in eine Datei geschrieben werden. Dabei muss a eine Zeichenkette sein, die den Speicherort definiert und b muss die Zeichenkette sein, die geschrieben werden soll.

7.7 Runden

Die Funktion round(a) kann eine Zahl runden. Wird ihr ein Fließkommawert übergeben, so ist der resultierende Wert ebenfalls vom Typ Fließkommazahl sein.



7.8 Maximum

7.8.1 Maximum von zwei Zahlen

Die Funktion max(a,b) gibt den Wert zurück, der größer ist.

7.8.2 Maximum aus einer Liste

Die Funktion max(a) muss mit einer Liste aufgerufen werden und gibt daraus den Größten Wert zurück.

7.9 Minimum

7.9.1 Minimum von zwei Zahlen

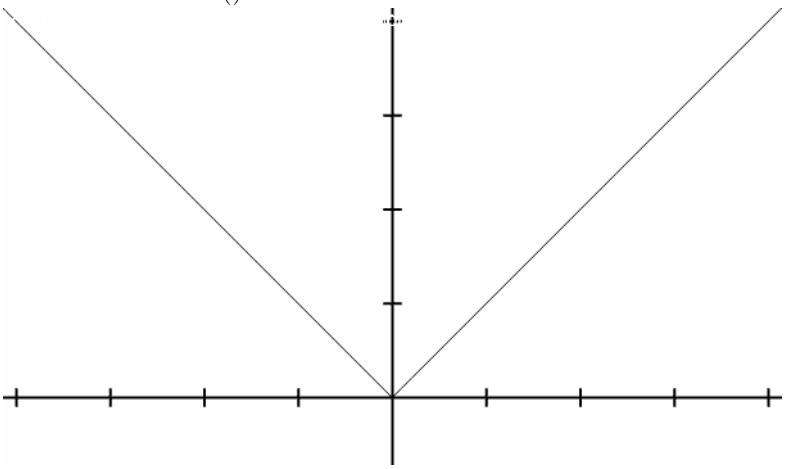
Die Funktion min(a,b) gibt den Wert zurück, der kleiner ist.

7.9.2 Minimum aus einer Liste

Die Funktion min(a) muss mit einer Liste aufgerufen werden und gibt daraus den kleinsten Wert zurück.

7.10 Absolut

Die Funktion abs() macht aus einer Zahl eine Positive.



7.11 Kardinalität

Mit der Funktion kard(a) kann die Kardinalität, oder auch Anzahl der Elemente einer Liste ermittelt werden.

7.12 Größter gemeinsamer Teiler

Die Funktion ggt(a,b) berechnet den größten gemeinsamen Teiler aus zwei Zahlen. Das Ergebnis ist auch bei negativen Eingabewerten positiv. Die Methode sieht folgendermaßen aus:

```
public static final long ggt (long a, long b){
    if (a < 0)
        a = -a;
    if (b < 0)
        b = -b;
    while ((a%=b)!=0)
        if ((b%=a)==0)
            return a;
    return b;
}
```

7.13 Kleinstes gemeinsames Vielfaches

Die Funktion kgv(a,b) berechnet das kleinste gemeinsame Vielfache zweier Zahlen. Sollte das Ergebnis nicht in den Bereich der Ganzzahlen passen, so wird -1 zurückgegeben. Außerdem ist das Ergebnis immer positiv, auch wenn die Eingabe negativ war. Die Methode sieht folgendermaßen aus:

```
public static final long kgv (long a, long b){
    if (a < 0)
        a = -a;
     if (b < 0)
        b = -b;
    final long c = a / ggt(a,b);
    final long kgv = c * b;
    return kgv / b == c ? kgv : -1;
}
```

7.14 Kombinatorik

7.14.1 Anzahl der Permutationen

Die Funktion um die Anzahl der Permutationen zu berechnen lautet npr(a,b). Die Methode, die hinter der Berechnung steckt ist folgende:

```
public static final long npr(long n, long k){
    if (n < 0 || k < 0 || n < k)
        return -2;
    if (n < fakCacheLong.length)
        return fakCacheLong[(int)n]/fakCacheLong[(int)(n-k)];
    if (n<(k<<1))
        k = n-k;
    long erg = 1;
    for (long i=k;i<n;i++)
        if (erg != (erg *= i)/i)
            return -1;
    return erg;
}
```

7.14.2 Anzahl der Kombinationen

Der Binomialkoeffizient ist eine Mathematische Funktion, um kombinatorische Aufgaben zu lösen. Die Methode heißt ncr(n, k), wobei n und k als Ganzzahlen interpretiert werden. Bei zu großen Werten ist der Rückgabewert -1 und bei ungültigen Eingabeparametern -2. Folgende Mathematische Formel steckt hinter der Funktion.
$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!}$
Dennoch sieht unserer Methode etwas anders aus:

```
public static final long ncr (final long n, long k){
    if (k > n || k < 0 || n < 0)
        return -2;
    if (n < fakCacheLong.length)
        return fakCacheLong[(int)n]/(fakCacheLong[(int)k]*fakCacheLong[(int)(
            n-k)]);
    if (n<(k<<1))
        k = n-k;
    long erg = 1;
    final long nk = n-k;
    for (int i=1;i<=k;i++)
        if ((erg /= i) != (erg *= nk + i) / (nk + i))
            return -1;
    return erg;
}
```

Die Methode lässt sich in 4 Teile unterteilen:

1. Überprüfen ob falsche nicht lösbare Eingaben gemacht wurden.
2. Das Ergebnis wenn möglich auf schnelle Weise durch bereits berechnete Fakultäten berechnen.
3. ncr(n,n-k) statt ncr(n,k) berechnen, falls $n < k \cdot 2$, da das Ergebnis das gleiche ist, aber das neue k kleiner, also Effizienter.
4. Durch eine Schleife den ncr berechnen. Danach das Ergebnis ausgeben, oder wenn während der Berechnung ein Überlauf auftritt -1.

.

7.15 If Abrage

Mit der Funktion ifn(a,b,c) können Abfragen realisiert werden. Sollte die Berechnung in a true ergeben wird b zurückgegeben, bei false c.

7.16 Abfrage von Benutzereingabe

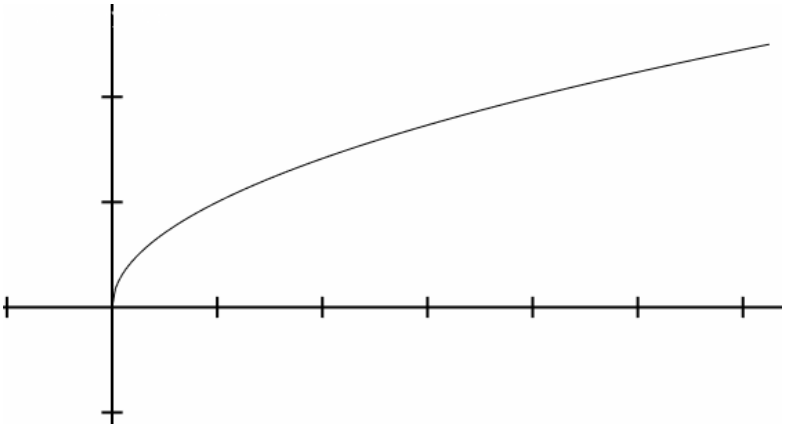
Mit der Funktion request(a) kann man den Benutzer dazu auffordern eine Eingabe zu machen. Der Parameter a muss eine Zeichenkette sein und der Rückgabewert ist ebenfalls immer eine Zeichenkette. Die Methode pausiert die Berechnung so lange, bis eine Eingabe gemacht wurde. Es ist auch möglich mit request(a,b) durch b ein timeout in Sekunden anzugeben, nach dem sich das Fenster automatisch schließt.

7.17 Warten

Mit der Funktion sleep(a), kann eine Zeit gewartet werden. Die Zeit wird dabei in Sekunden abgegeben.

7.18 Wurzel

7.18.1 Quadrat Wurzel



Hier werde ich den so genannten „sqrt“Algorithmus erläutern. Es handelt sich nämlich um die Berechnung der quadratischen Wurzel von dem gegebenen Parameter x.

Im Mathematischen Sinne kann man die Wurzel mit einem Näherungsverfahren, und zwar dem Newton-Verfahren.
Wie bereits erwähnt ist es ein Näherungsverfahren, welches genauer wird je öfter man es iteriert.
Sei x gegeben, wobei $x \in |x >= 0$, und y das daraus berechnete Ergebnis, wobei $y \in$.

Die Gleichung :

$$\frac{(2-1)\cdot y^n + x}{2\cdot y^{n-1}}$$

entspricht dem Newton-Verfahren, diese haben wir jedoch etwas angepasst, um unnötige Rechenoperationen zu vermeiden. Es resultierte :

$$\frac{y\cdot y + x}{2\cdot y}$$

Außerdem haben wir ausgeschlossen, dass $x < 0$ ist, und wenn $x = 0$, dann ergibt es ohne eine Berechnung 0.
Sehen wir uns den Algorithmus mal an.

```
public static final double sqrt(final double n){
    if(n<0)
        return Double.NaN;
    if(n==0)
        return 0;
    double erg=n;
    for(int i=1;i!=11;i++)
        erg=(erg*erg+n)/(2*erg);
    return erg;
}
```

An dem Algorithmus wird die oben genannte Gleichung sehr schön und eindeutig deutlich. Wir iterieren die Gleichung 10-mal, da uns bereits die Genauigkeit des Ergebnisses nach 10 Schritten genügte. Da der Algorithmus nicht besonders schwer ist, werde ich ihn nicht detailliert durchgehen. Was möglicherweise jedoch nicht klar ist, ist das „Double.NaN“, NaN steht für Not a Number, aus dem Algorithmus geht also hervor das wenn die Wurzel von x berechnet werden soll, x jedoch kleiner als 0 ist, so ist es nicht definiert und man bekommt NaN ausgegeben.

7.18.2 N-te Wurzel

Auch haben wir einen Algorithmus zum Berechnen der n-ten Wurzel von x. Die oben genannte Gleichung ist hier unverändert und das Verfahren dementsprechend identisch:

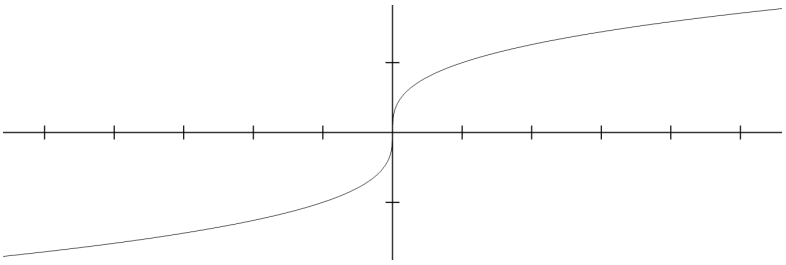
$$\frac{(2-1)\cdot y^n + x}{2\cdot y^{(n-1)}}$$

Da wir die Gleichung jedoch nicht verändert haben, ist der Algorithmus etwas anders.

```
public static double rt(double y, int n){
    if(n<=0)
        return Double.NaN;
    if(y==0)
        return 0;
    double erg=y;
    for(int i=1;i!=200;i++)
        erg=((n-1)*pow(erg,n)+y)/(n*pow(erg,n-1));
    return erg;
}
```

Auch hier kommt die genannte Gleichung sehr gut zum Vorschein. Der Parameter n ist hier die Potenz der Funktion und y die zu berechnende Zahl. Die hier genutzte Methode pow(x, exponent) ist ebenfalls von uns geschrieben und berechnet eine Potenz von x mit dem Exponenten exponent. Ich komme später auf diese Methode zurück, um sie etwas weiter zu erläutern. Die Iteration haben wir provisorisch auf 199 gesetzt, um eine genaue grenze zu ermitteln müssten wir eine ausführliche und Zeitaufwendige Statistik führen, belassen wir es vorerst auf 199.
Falls nicht klar ist, was mit der Iteration 199 gemeint ist: Das heißt, das die oben aufgeführte Funktion 199-mal wiederholt wird, um dementsprechend sehr genaue Ergebnisse zu ermitteln, da wie bereits erwähnt „je "ofter man iteriert, desto genauer wird das Ergebnis".

7.18.3 Würfel



Die 3-te Wurzel auch Würfel-Wurzel genannt kann durch die Funktion cbrt() gezogen werden.

7.19 Potenz

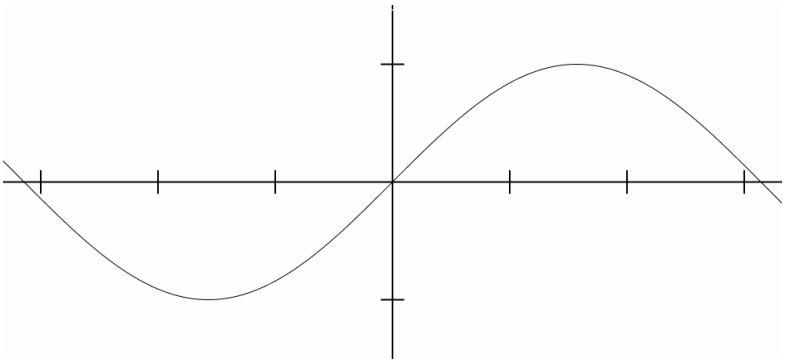
Die pow(x, exponent) Methode haben wir bereits bei der Berechnung der n-ten Wurzel von x gesehen, was aber ist die pow Methode und was macht sie? Die pow(x, exponent) Methode berechnet die Potenzen einer Zahl x. Beispielsweise wenn wir als x 2 eingeben und als Exponent 2, so wird er uns die 4 ausgeben, da $2^2 = 4$. Um unser Beispiel weiter zu analysieren: $2^2 == 4$, der Exponent beschreibt also die Anzahl, der Zahl x, welche mit sich selbst multipliziert wird, wenn wir also 2 haben, rechnen wir , die Zahl 2 erscheint zwei mal, da der Exponent 2 ist. Sehen wir uns den Algorithmus an.

```
public static Number pow (long x, long exponent){
  if (exponent<0)
    return Math.pow(x, exponent);
  if(exponent==0)
    return 1;
  BigInteger erg = BigInteger.valueOf(x);
  while (exponent != 1){
    erg = (exponent&1)==0 ? erg.multiply(erg) : erg.multiply(erg).multiply(
        BigInteger.valueOf(x));
    exponent >>= 1;
  }
  return erg;
}
```

Um Syntax Fragen direkt zu klären: $a >>= 1$ heißt, $a = a/2$, und man sieht das erg den Typ BigInterger hat, BigInterger kann sehr große Zahlen beinhalten, jedoch muss man Methoden für Rechenoperationen benutzen, in dieser Methode sieht man die Multiplikation also multiply(erg). Also wird hier entweder erg mit sich selbst multipliziert, oder mit x. Exponent&1 heißt so viel wie Exponent modulo 2.

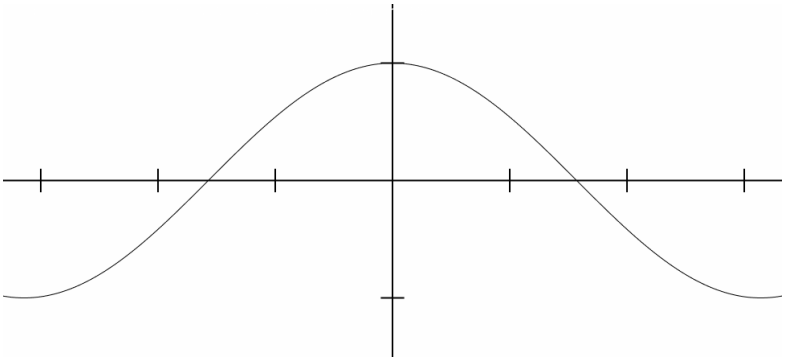
7.20 Trigonometrisch

7.20.1 Sinus



Die Sinusfunktion kann durch sin(a) berechnet werden. Die Berechnung erfolgt im Bogenmaß.

7.20.2 Cosinus

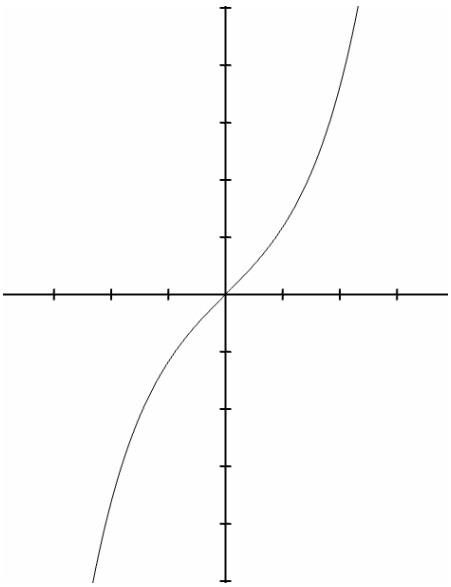


Die Cosinusfunktion kann durch cos(a) berechnet werden. Die Berechnung erfolgt im Bogenmaß.

7.20.3 Tangens

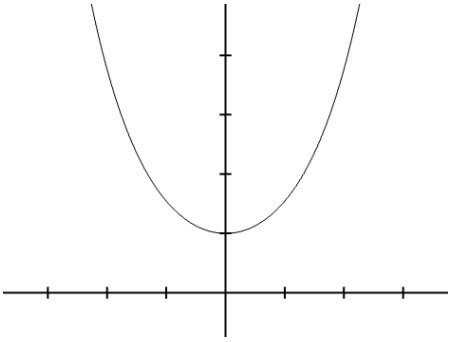
Die Tangensfunktion kann durch tan(a) berechnet werden. Die Berechnung erfolgt im Bogenmaß.

7.20.4 Hyperbolischer Sinus



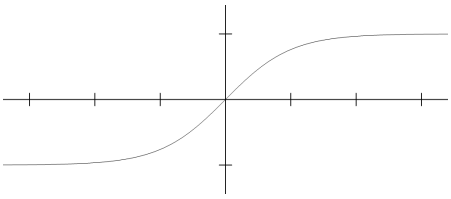
Der Hyperbolische Sinus kann mit der Funktion sinh(a) berechnet werden.

7.20.5 Hyperbolischer Cosinus



Der Hyperbolische Cosinus kann mit cos(a) berechnet werden.

7.20.6 Hyperbolischer Tangens



Der Hyperbolische Tangens kann mit der Funktion tanh(a) berechnet werden.

7.21 Variablen

7.21.1 Definiere

Mit der Funktion define(a) kann eine Variable mit dem Namen a definiert werden. Eine neue Variable enthält nach der Definition einen Nullzeiger.

7.21.2 Zuweisung

Die Zuweisung ist dazu da um Variablen Werte zuzuweisen. Das Zeichen für die Zuweisung ist $b \rightarrow a$, es kann aber auch die Funktion $set(a, b)$ genutzt werden. Dabei ist a die Variable der der Wert b zugewiesen werden Soll. Es ist auch möglich eine Listenstelle einen Wert zuzuweisen. Wenn man beispielsweise $5 \rightarrow a[2]$ schreibt und $a = \{1, 2, 3\}$ war, dann ist danach $a = \{1, 2, 5\}$. Wird eine Variable in der ein Liste gespeichert ist, einer anderen zugewiesen, so wird die Liste kopiert, was aus Performancegründen berücksichtigt werden sollte.

7.21.3 Lösche

Mit der Funktion delete(a) kann eine Variable wieder gelöscht werden.

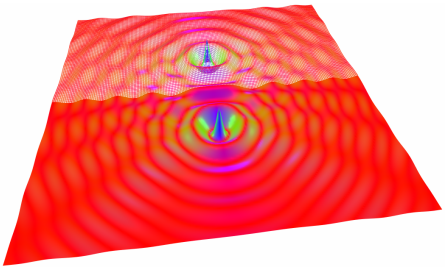
7.22 Löse

Die Funktion solve(a,b) kann zum Auflösen einfacher Gleichungen benutzt werden. Dabei ist a die Gleichung und b die Variable, nach der aufgelöst werden soll.

8 Interessante/Formschöne Graphen

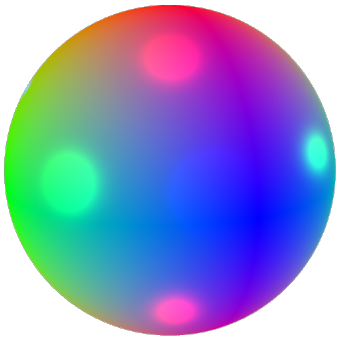
8.1 Zwei Wellen

Art	3D Funktion
f(x,y)	$\left(\frac{\sin\left(a\cdot\sqrt{(x-10)^2+y^2}+c\right)}{\sqrt{(x-10)^2+y^2+0.5}}+\frac{\sin\left(a\cdot\sqrt{(x+10)^2+y^2}+c\right)}{\sqrt{(x+10)^2+y^2+0.5}}\right)\cdot b$
min u	-30
max u	30
step u	0.3
min v	-30
max v	30
step v	0.3



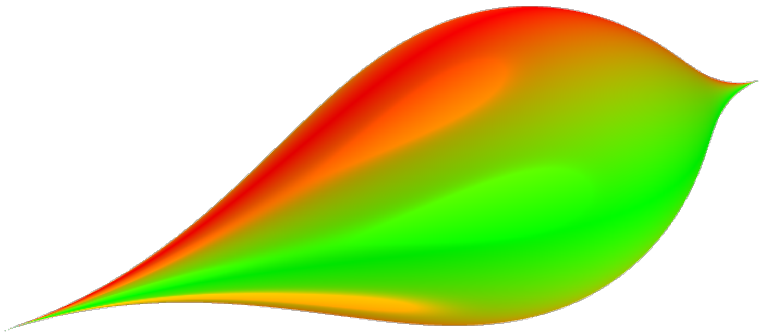
8.2 Kugel

Art	3D Parametrisch
x(u,v)	$\cos(u)\cdot\sqrt{1-v^2}$
y(u,v)	$\sin(u)\cdot\sqrt{1-v^2}$
z(u,v)	v
min u	$-\pi$
max u	π
step u	$\frac{\pi}{100}$
min v	-1
max v	1
step v	0.001



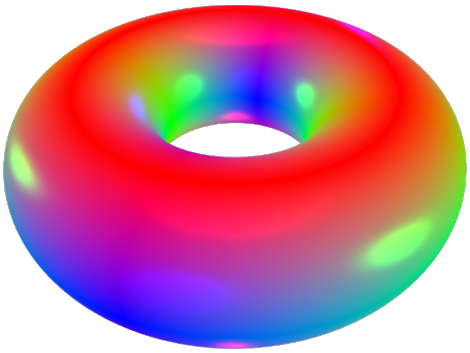
8.3 Limette

Art	3D Parametrisch
x(u,v)	v
y(u,v)	$\frac{\sin(u)\cdot(\cos(v)+1)}{2}$
z(u,v)	$\frac{\cos(u)\cdot(\cos(v)+1)}{2}$
min u	$-\pi$
max u	π
step u	$\frac{\pi}{200}$
min v	$-\pi$
max v	π
step v	$\frac{\pi}{200}$



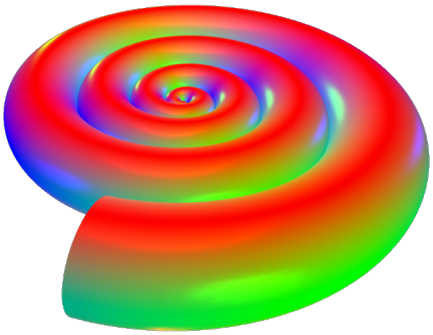
8.4 Donut

Art	3D Parametrisch
x(u,v)	$(\sin(u)+2)\cdot\sin(v)$
y(u,v)	$(\sin(u)+2)\cdot\cos(v)$
z(u,v)	$-cos(u)$
min u	$-\pi$
max u	Pi
step u	$\frac{\pi}{200}$
min v	$-\pi$
max v	π
step v	$\frac{\pi}{200}$



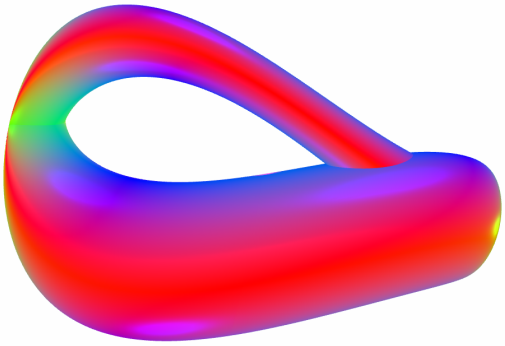
8.5 Schnecke

Art	3D Parametrisch
$x(u,v)$	$(\sin(u)+\frac{v}{2\cdot\pi})\cdot\sin(v)\cdot v$
$y(u,v)$	$(\sin(u)+\frac{v}{2\cdot\pi})\cdot\cos(v)\cdot v$
$z(u,v)$	$\cos(u)\cdot v$
Min u	$-\pi$
Max u	π
Step u	$\frac{\pi}{50}$
Min v	0
Max v	30
Step v	0.05



8.6 Kleinsche Flasche

Art	3D Parametrisch
$x(u,v)$	$3\cdot\cos(u)\cdot(1+\sin(u))-(2-\cos(u))\cdot\cos(v)$
$y(u,v)$	$8\cdot\sin(u)$
$z(u,v)$	$\sin(v)\cdot(2-\cos(u))$
Min u	$-\pi$
Max u	π
Step u	$\frac{\pi}{100}$
Min v	$-\pi$
Max v	π
Step v	$\frac{\pi}{100}$



9 Dateien

9.1 Log

Zusätzlich zu der Programm- und Konsolenausgabe gibt es noch zwei weitere Logdateien. Diese befinden sich im Programmordner, genauer gesagt dort im Ordner log. Dort gibt es die Dateien logfile-user.txt und logfile-debug.txt. Wie der Name schon sagt ist das Erste als Einblick für den Benutzer gedacht und das Zweite für den Entwickler. Bei Problemen mit dem Programm sollte daher auch immer das Entwickler-Log mitgeschickt werden.

9.2 Optionen

Die Programmoptionen befinden sich an der Stelle Anwendungsdaten/.graphs/options.txt. Der Pfad kann je nach System etwas abweichen. Sollte dieser Ordner, bzw. die Datei nicht existieren oder Fehler enthalten, so wird der Ordner automatisch erstellt, bzw. eine Datei mit den Standardeinstellungen erstellt.

9.3 Cubemaps

Das Programm kann mit jeder Cubemap umgehen, wenn sie in der Form vorliegt, dass sich 6 von 0 bis 5 durchnummerierte und passend gedrehte Bilder in einem Ordner befinden. Der Pfad der zu benutzenden Cubemap lässt sich im Optionenfenster wählen.

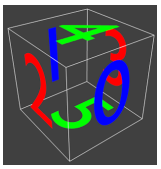


Abbildung 1: Cubemap

9.4 Graph

Das Standardformat für Projekte. Es ist ein Zip-Archiv, das die Daten in Form von Textdateien im Unicode-Format enthält.

9.5 Off

Ein Format, dass sich dafür eignet um die Graphen für andere Programme zu exportieren, die Sie dann rendern können. Dies kann sinnvoll sein, wenn einem die Qualität der Bilder nicht ausreicht, bzw Rendereinstellungen erwünscht sind, die vom Programm nicht unterstützt werden. Beispielsweise können mit dem freien Programm Blender sehr schöne Bilder erzeugt werden.

9.6 Jar

Die jar-Dateien sind die Programme, die von der jvm (Java Virtual Machine) interpretiert werden.

10 Häufige Probleme

10.1 Programm startet nicht

Ursache	Lösung
Java-Version ist zu alt	Installieren Sie Java mit Version >= 6.

10.2 Kein 3D-Fenster

Ursache	Lösung
Grafiktreiber unterstützt OpenGl nicht	Installieren Sie den neuesten Treiber des Herstellers und überprüfen Sie, ob OpenGl unterstützt wird.
LWJGL ist unter Linux nicht installiert oder zu alt	Installieren Sie mindestens Version 2.8.1 von LWJGL auf ihrem Linux System.