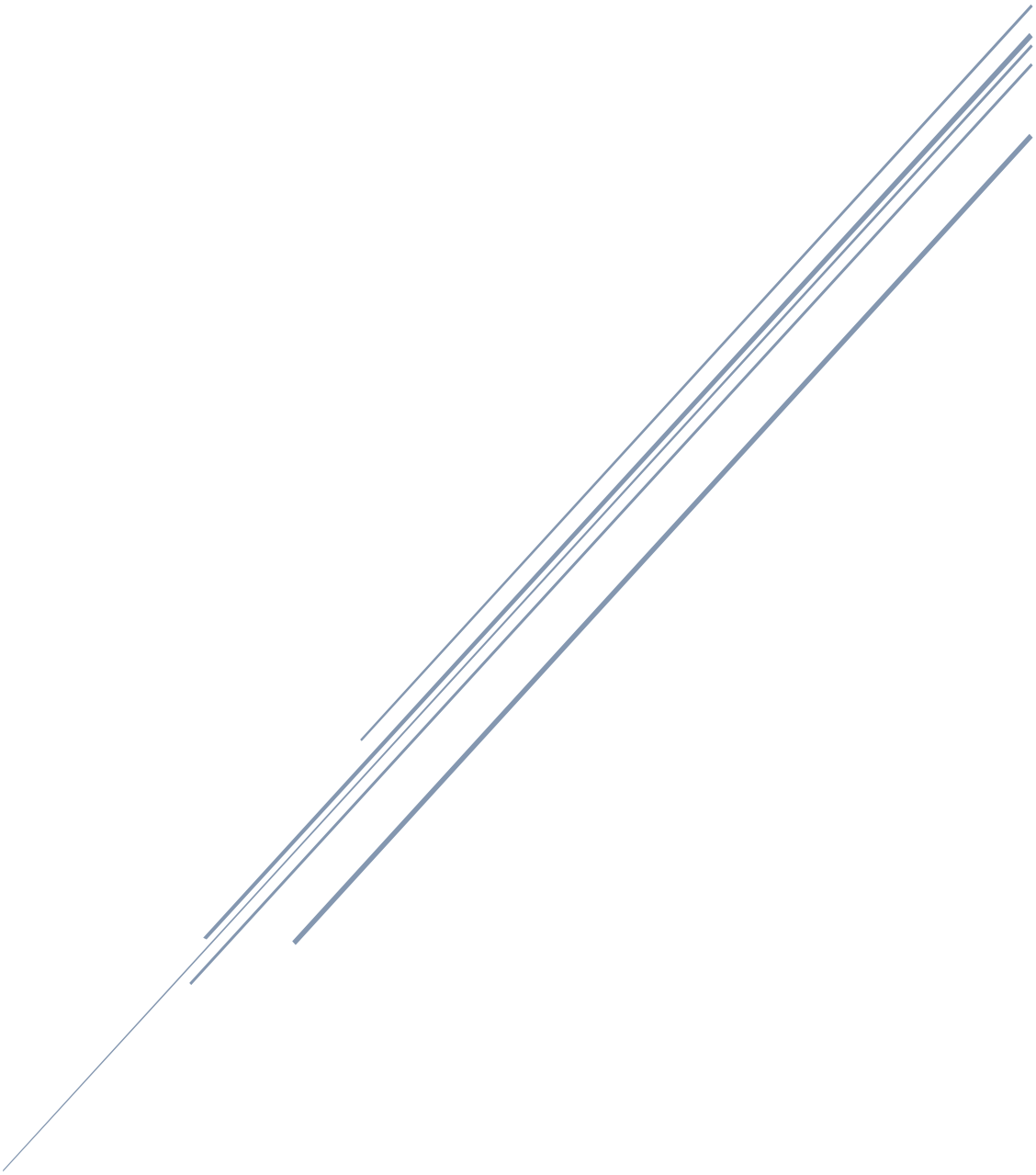
A07-SOLARPY

Maran Christian



TGM
5BHITM

Inhalt

Aufgabenstellung

Wir wollen nun unser Wissen aus Medientechnik und SEW nützen um eine etwas kreativere Applikation zu erstellen.

Eine wichtige Library zur Erstellung von Games mit 3D-Grafik ist Pygame. Die 3D-Unterstützung wird mittels PyOpenGL erreicht.

Die Kombination ermöglicht eine einfache und schnelle Entwicklung.

Während pygame sich um Fensteraufbau, Kollisionen und Events kümmert, sind grafische Objekte mittel OpenGL möglich.

Die Aufgabenstellung:

Erstellen Sie eine einfache Animation unseres Sonnensystems:

In einem Team (2) sind folgende Anforderungen zu erfüllen.

- Ein zentraler Stern
- Zumindest 2 Planeten, die sich um die eigene Achse und in elliptischen Bahnen um den Zentralstern drehen
- Ein Planet hat zumindest einen Mond, der sich zusätzlich um seinen Planeten bewegt
- Kreativität ist gefragt: Weitere Planeten, Asteroiden, Galaxien,...
- Zumindest ein Planet wird mit einer Textur belegt (Erde, Mars,... sind im Netz verfügbar)

Events:

- Mittels Maus kann die Kameraposition angepasst werden: Zumindest eine Überkopf-Sicht und parallel der Planentenbahnen
- Da es sich um eine Animation handelt, kann diese auch gestoppt werden. Mittels Tasten kann die Geschwindigkeit gedrosselt und beschleunigt werden.
- Mittels Mausklick kann eine Punktlichtquelle und die Textierung ein- und ausgeschaltet werden.
- Schatten: Auch Monde und Planeten werfen Schatten.

Hinweise:

- Ein Objekt kann einfach mittels glutSolidSphere() erstellt werden.
- Die Planten werden mittels Modelkommandos bewegt: glRotate(), glTranslate()
- Die Kameraposition wird mittels gluLookAt() gesetzt
- Bedenken Sie bei der Perspektive, dass entfernte Objekte kleiner - nahe entsprechende größer darzustellen sind.
  Wichtig ist dabei auch eine möglichst glaubhafte Darstellung. gluPerspective(), glFrustum()
- Für das Einbetten einer Textur wird die Library Pillow benötigt! Die Community unterstützt Sie bei der Verwendung.

Zeitaufwandsaufzeichnung

Arbeitspakete	Geschätzte Zeit	Tatsächliche Zeit	Status
1 Libraries-Recherche	00h 30min	00h 35min	Done
1.1 Evaluierung der Libraries	00h 30min	00h 35min	Done
2 Anlernen der Libraries	04h 00min	03h 30min	Done
2.1 Pygame	01h 30min	01h 00min	Done
2.1.1 Darstellung	00h 30min	00h 30min	Done
2.1. 2 Steuerung	01h 00min	00h 30min	Done
2.2 OpenGL	02h 30min	02h 00min	Done
2.2.1 Erzeugen von Objekten	00h 30min	00h 45min	Done
2.2.2 Translation von Objekten	00h 30min	00h 30min	Done
2.2.3 Rotation von Objekten	01h 00min	00h 30min	Done
2.2.4 Buffering	00h 30min	00h 15min	Done
2.3 Pillow	00h 30min	00h 30min	Done
2.3.1 Texturierung von Objekten	00h 30min	00h 30min	Done
3 Backend-Design	02h 00min	01h 30min	Done
3.1 UML erstellen	02h 00min	01h 30min	Done
4 GUI-Design	01h 30min	01h 00min	Done
4.1 GUI-Prototypen erstellen	01h 30min	01h 00min	Done
5 3D-Objekte erstellen	02h 00min	01h 00min	Done
5.1 Fixstern erstellen	00h 30min	00h 15min	Done
5.2 Planet 1 erstellen	00h 30min	00h 15min	Done
5.3 Planet 2 erstellen	00h 30min	00h 15min	Done
5.4 Mond erstellen	00h 30min	00h 15min	Done
6 Texturen zuweisen	01h 00min	01h 00min	Done
6.1 Fixstern Textur zuweisen	00h 15min	00h 30min	Done
6.2 Planet 1 Textur zuweisen	00h 15min	00h 10min	Done
6.3 Planet 2 Textur zuweisen	00h 15min	00h 10min	Done
6.4 Mond Textur zuweisen	00h 15min	00h 10min	Done
7 Animationssteurung	05h 00min	04h 40min	Done
7.1 Rotation um ein Objekt	01h 30min	01h 30min	Done
7.2 Rotation um die eigene Achse	00h 45min	00h 30min	Done
7.3 Rotation von mehreren Objekten	02h 00min	01h 30min	Done
7.4 Geschwindigkeit einstellen	00h 15min	00h 10min	Done
7.5 Rotation starten	01h 00min	00h 30min	Done
8 Kamera	02h 00min	00h 45min	Done
8.1 Kameraview erzeugen	01h 00min	00h 30min	Done
8.2 Postionsänderung mittels Tasten	01h 00min	00h 15min	Done
9 Lichtquelle	04h 00min	02h 35min	Done
9.1 Lichtquelle erzeugen	02h 00min	01h 30min	Done
9.2 Lichtquelle positionieren	01h 00min	00h 45min	Done
9.3 Lichtquelle an-/ausschalten	01h 00min	00h 20min	Done
10 Dokumentation	04h 30min	02h 00min	Done
10.1 Code dokumentieren	01h 30min	01h 00min	Done
10.2 Sphinx Doku erzeugen(rst-File etc)	01h 00min		
10.3 Protokoll	02h 00min	02h 00min	Done
Summe	26h 30min	18h 35min	

Designüberlegung

Notwendige Libraries

- PyQt
- PyGame
- PyOpenGL
- Pyglet Überlegungen

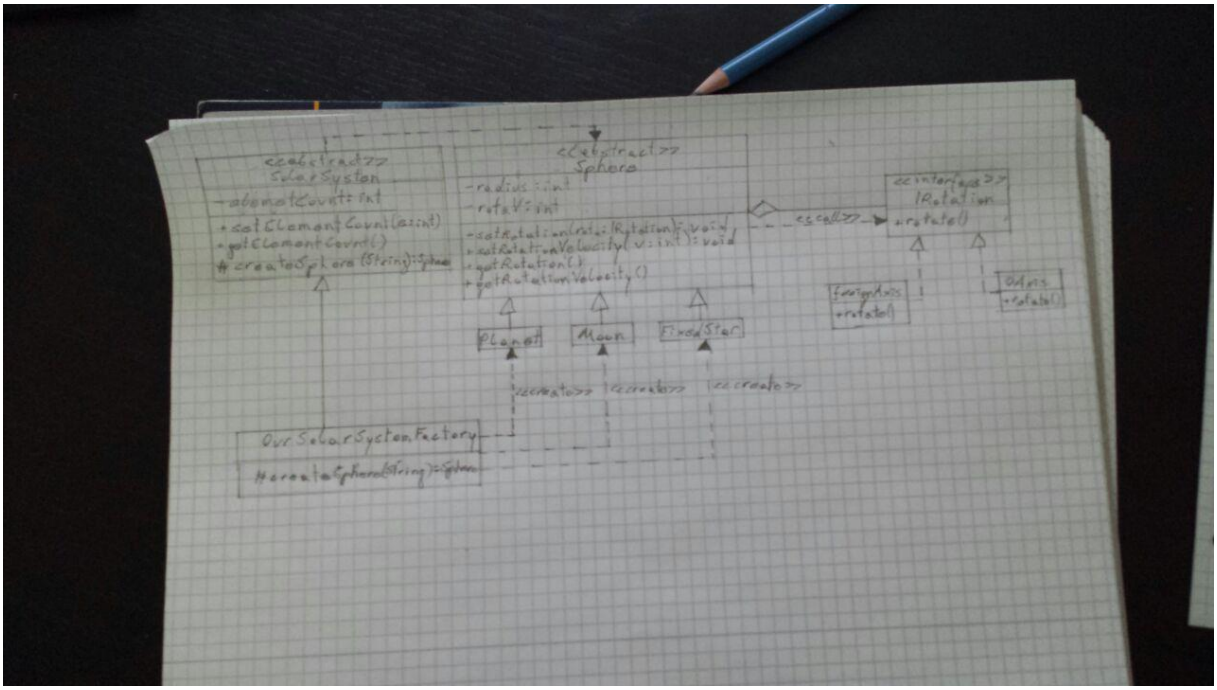
Splashscreen wird mittels Pygame eingebunden.

Der OpenGL-Teil wird mit PyOpenGL implementiert und dann mittels PyGame eingebunden.

Rotationsgeschwindigkeit soll mit den Pfeiltasten verändert werden können → weitere Konfiguration womöglich mittels ausklappbaren Optionsinterface

Etwaige GUI-Komponenten mit PyQt umsetzen → Anpassungsmöglichkeit mittels Stylesheet

Codedesign/UML Classdiagram

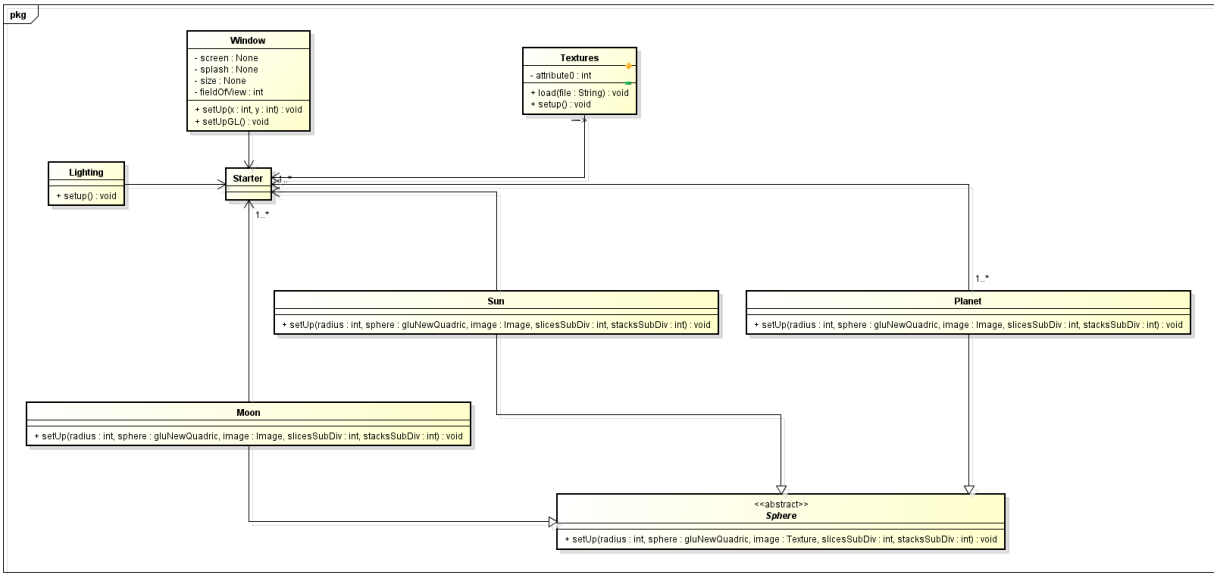


Vorübergehend ist das UML nur per Hand gezeichnete Grafik vorhanden, da ich schon seit Wochen darauf warte, dass mein Astah Students Account wieder freigeschalten wird.

MVC für die Grundstruktur des Projektes um eine gute Trennung der Komponenten zu gewährleisten.

Des Weiteren werden die verschiedenen Rotationen mit dem Strategy-Pattern gelöst. Um das ganze etwas leichter wiederverwendbar zu machen, sprich für andere Galaxien wird die Erzeugung über das Factory Pattern umgesetzt.

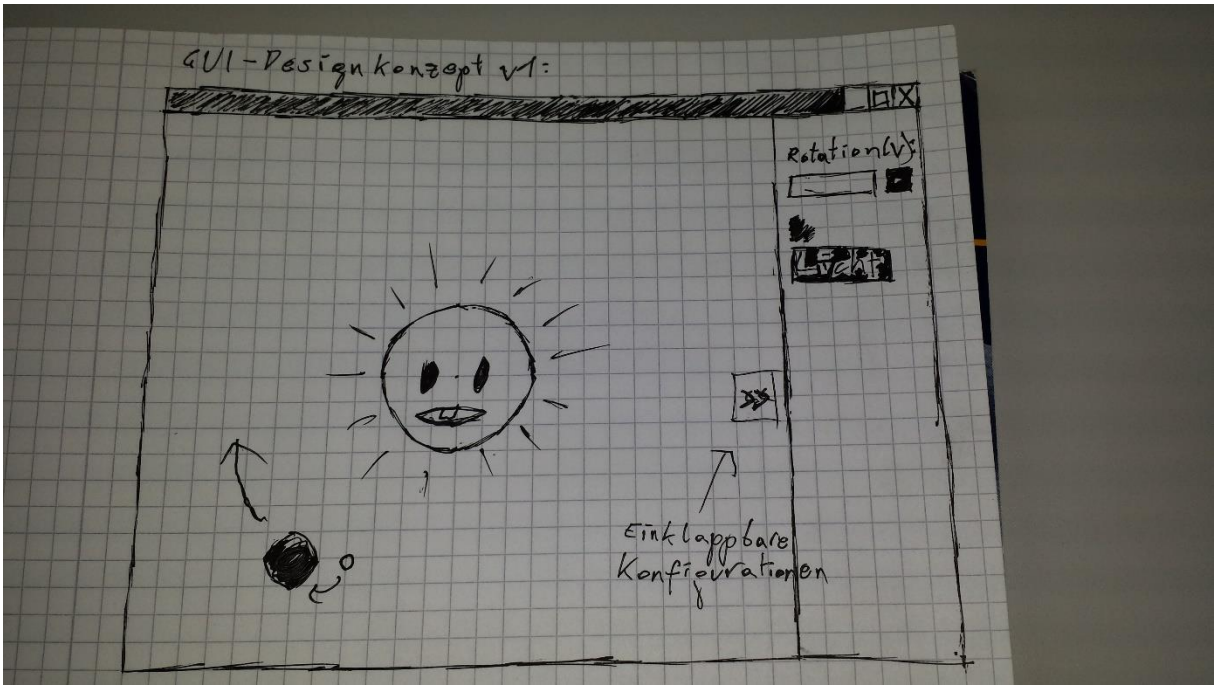
UML Abgabe



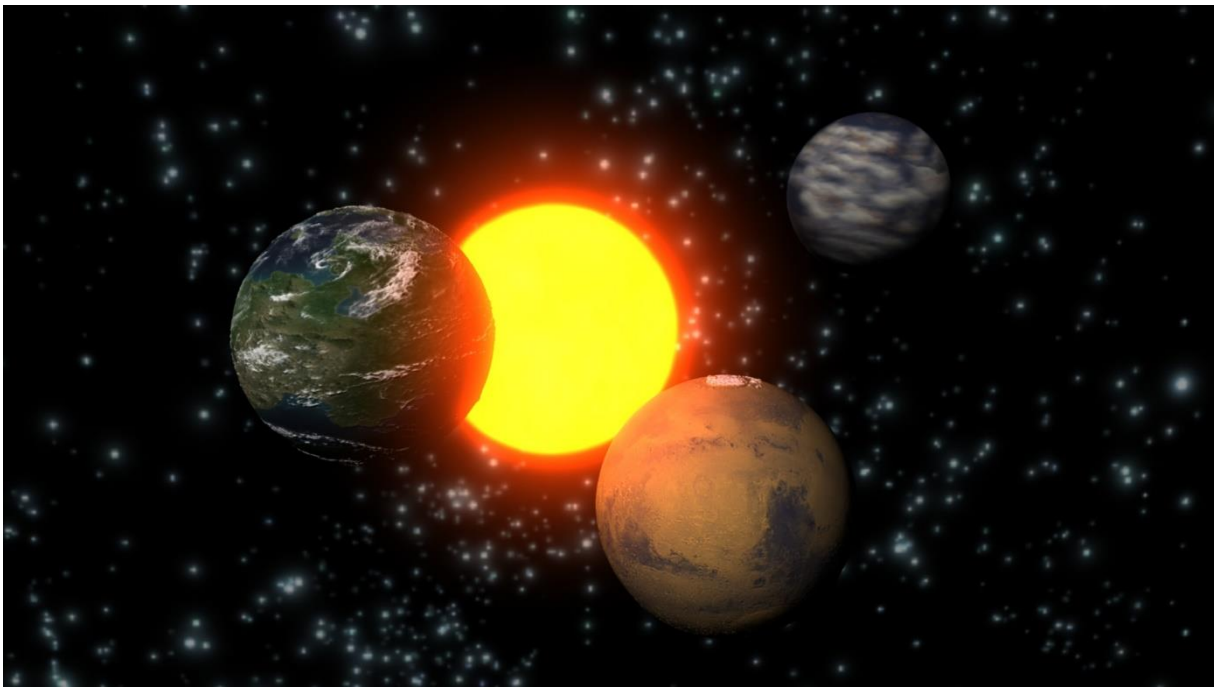
Leider hatte ich bei der Umsetzung des vorab geplanten Designs Probleme, welche ich bis zur Abgabe aufgrund fehlender Zeit leider nicht mehr ganz beheben konnte daher musste ich leider ohne Designpatterns auskommen, was allerdings die Wiederverwendbarkeit für andere Sonnensysteme keineswegs einschränkt es allerdings etwas umständlicher macht.

GUI-Design

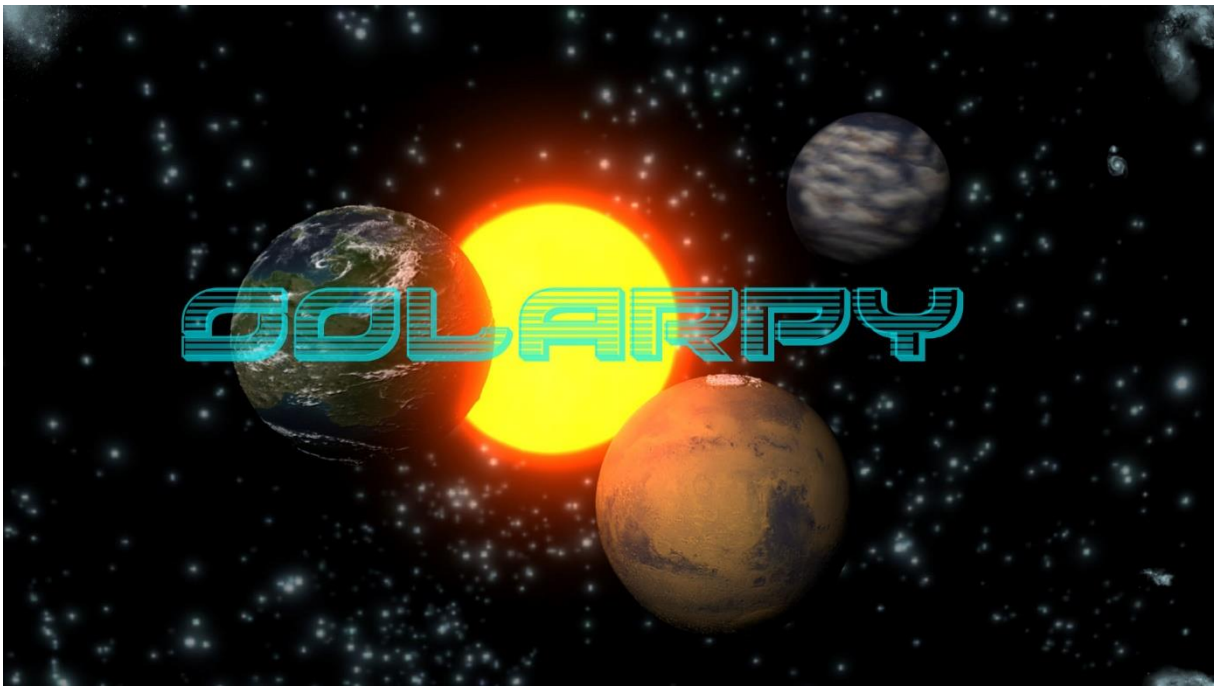
V1



Splashscreen



Aktuell wird an einem Splashscreen mit Animation gearbeitet, welche schon funktionieren würde, allerdings werden bei dieser Version die Lichter nicht vernünftig abgespeichert



Quellen

[1] thenewboston , Pygame (Python Game Development) Tutorial - 1 – Introduction, 10.11.2014, https://www.youtube.com/watch?v=K5F-aGDIYaM (zuletzt aufgerufen 23.02.2014)

[2] Rick Muller, Open a GLUT window and draw a sphere using Python/OpenGL (Python recipe), 27.10.2004, http://code.activestate.com/recipes/325391-open-a-glut-window-and-draw-a-sphere-using-pythono/ (zuletzt aufgerufen 28.02.2014)