Marlene Dorfinger, Muhammedmehdi Kanyildiz

Visit our solar system …

Inhalt

[1. Aufgabenstellung 2](#_Toc413669676)

[2. Zeitaufzeichnung 3](#_Toc413669677)

[3. Designüberlegung 4](#_Toc413669678)

[3.1 UML 4](#_Toc413669679)

[3.3 Splashscreen Prototyp 5](#_Toc413669680)

[3.4 Verwendete Libraries 5](#_Toc413669681)

[4. Funktionalität 5](#_Toc413669682)

[4.1 Pygame installieren 5](#_Toc413669683)

[4.2 PyOpenGL und Pillow installieren 5](#_Toc413669684)

[4.3 Texturen mit Pillow 5](#_Toc413669685)

[4.4 Sonnensysteminfos 6](#_Toc413669686)

[4.5 Prototyp 6](#_Toc413669687)

[4.6 Startbildschirm 6](#_Toc413669688)

[5. Fortschritt 7](#_Toc413669689)

[6. Fehleranalysen 7](#_Toc413669690)

[Quellen 7](#_Toc413669691)

# 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.

 Tutorials:

* Pygame: https://www.youtube.com/watch?v=K5F-aGDIYaM

Viel Erfolg!

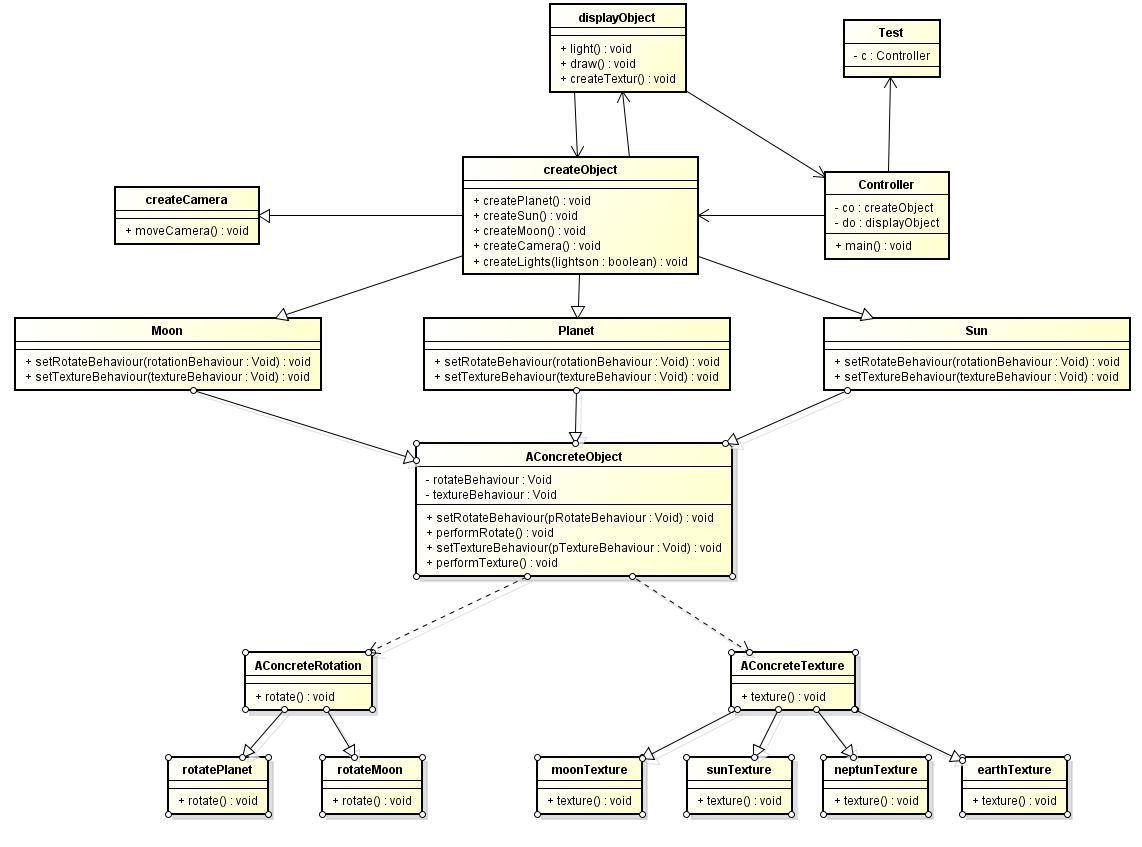
# Zeitaufzeichnung

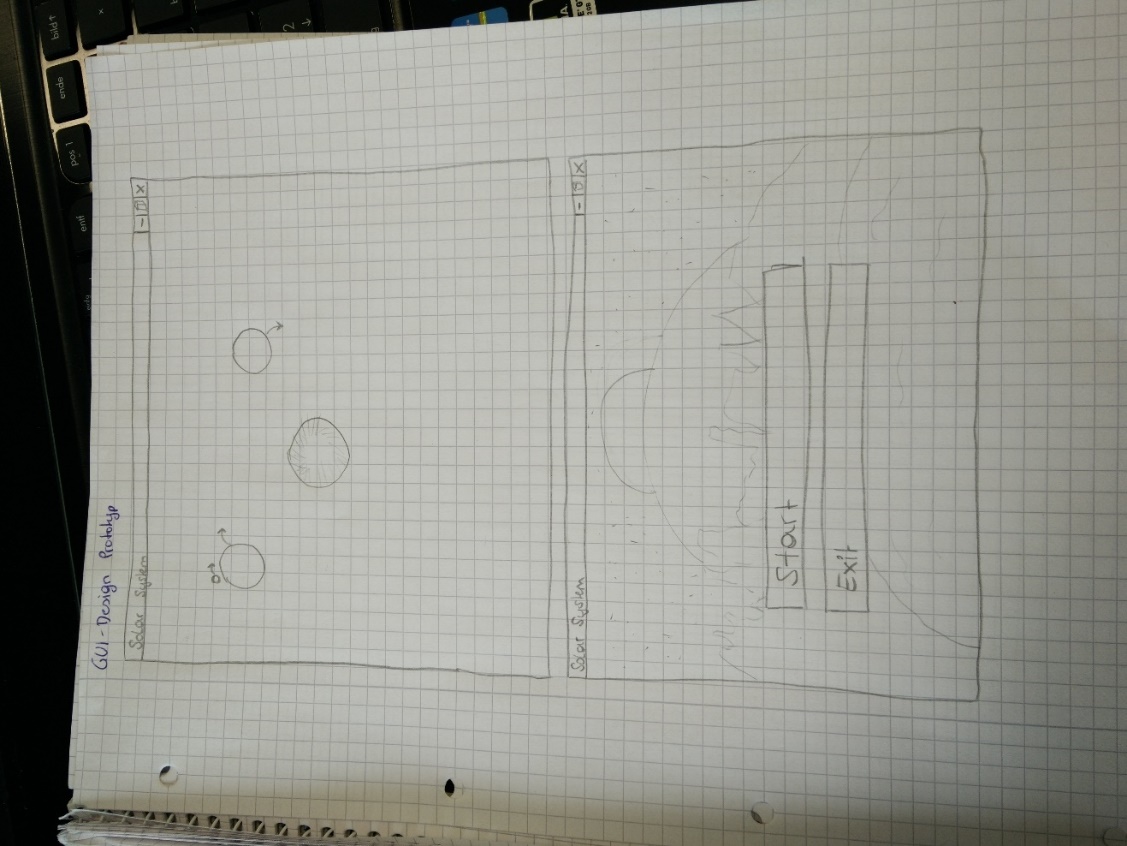
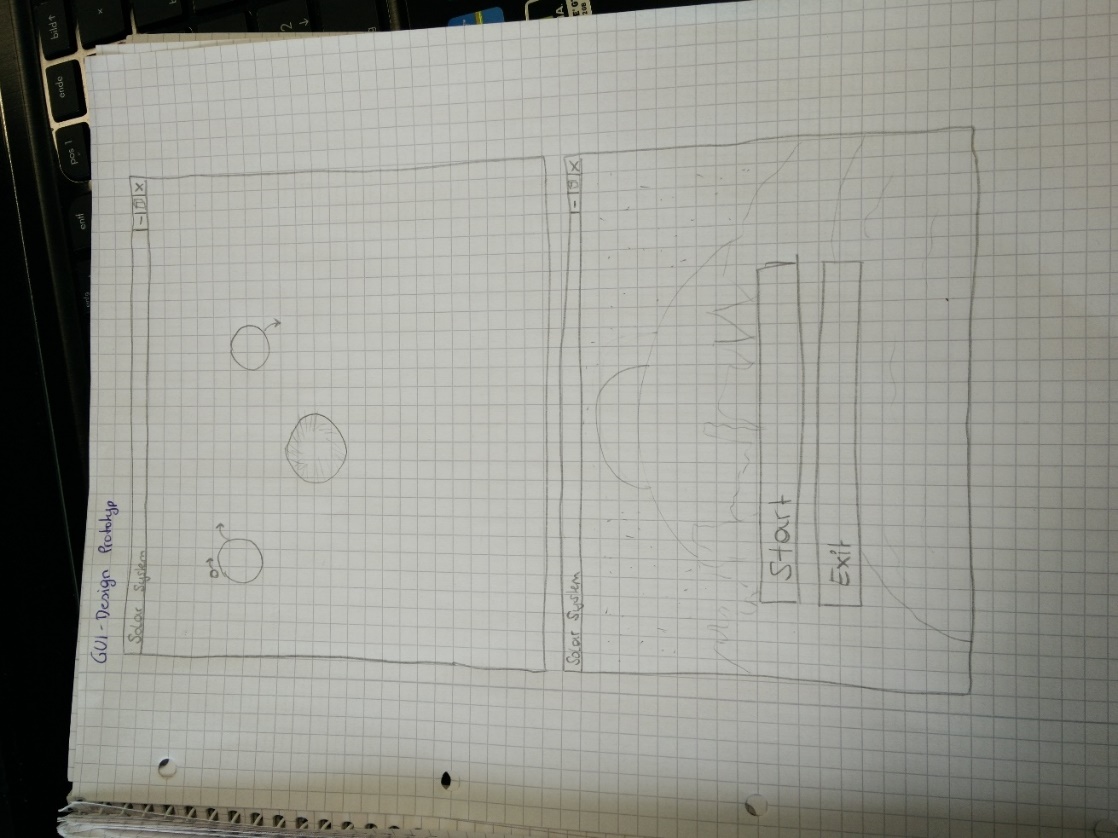
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Aufgabe | Priorität | Erwartete Zeit | Tatsächliche Zeit | Zuständigkeit | Status |
| OpenGL lernen | Hoch | 2:00 h | 3:00h | Dorfinger, Kanyildiz | F |
| Libraries suchen | Hoch | 1:00 h | 0:30 h | Dorfinger, Kanyildiz | F |
| Informieren über Sonnensystem | Mittel | 1:00 h | 0:30h | Dorfinger,  Kanyildiz | F |
| Planeten implementieren (mind. 2) | Hoch | 1:00 h | 1:00h | Kanyildiz | F |
| Zentralstern implementieren | Hoch | 1:00 h | 0:30h | Kanyildiz | F |
| Mond(e) implementieren | Hoch | 1:00 h |  | Kanyildiz | F |
| Lichtquellen erstellen | Hoch | 1:00 h | 1:30h | Dorfinger | F |
| Texturen erstellen oder aus Internet suchen | Mittel | 2:00 h | 1:00 h | Dorfinger | F |
| Texturen laden | Hoch | 0:30 h | 0:30 h | Dorfinger | F |
| Texturen auf Objekte legen | Hoch | 1:00 h |  | Dorfinger | F |
| Zentralstern dreht sich um sich selbst | Hoch | 1:00 h | 0:50h | Kanyildiz | F |
| Planeten drehen sich um eigene Achse und um Zentrealstern | Hoch | 1:00 h | 2:00h | Kanyildiz | F |
| Monde drehen sich um sich selbst, um Planeten und um den Zentralstern | Hoch | 1:00 h |  | Kanyildiz | F |
| Implementieren der Tastensteuerung (Animation und Geschwindigkeit) | Hoch | 1:00 h | 0:30h | Kanyildiz | I |
| Einschalten/Ausschalten von Textur und Lichtquelle | Mittel | 1:00 h | 0:50h | Dorfinger,  Kanyildiz | F |
| Schatten zu Lichtern | Mittel | 1:00 h | 1:00h | Kanyildiz | F |
| Perspektive ändern wen näher/weiter weg | Mittel | 1:00 h | 0:30h | Kanyildiz | F |
| 3D-Splashscreen | Mittel | 2:00 h | 2:00 h | Dorfinger | F |
| Gesamt |  | 20:30h | 14:60 h |  |  |

Legende Status:  
I … Implementierung  
T … Test  
K … Dokumentation  
F … Fertig

# Designüberlegung

## 3.1 UML

  
  
3.2 GUI-Design Prototyp

## 3.3 Splashscreen Prototyp



## 3.4 Verwendete Libraries

* PyOpenGL  
  Verwendet für: Objekte, Lighting, Texturen
* PyGame  
  Verwendet für: Benutzersteuerung
* Tkinter  
  Verwendet für: Starbildschirm und messagebox
* Pillow  
  Verwendet für: Texturen

# Fortschritt

**Geschafft:**

* Die Sonne dreht sich um seine eigenen Achse
* Es drehen sich mehrere Planeten um die Sonne
* Die Belichtung funktioniert(Die Sonne ist vollkommen beleuchtet während nur die eine Hälfte der Planeten beleuchtet ist)
* Kamerasteuerung funktioniert zu 80%
* Die Texturierung
* Unterschiedliche Geschwindigkeit beim Rotieren
* Monde

**Ausstehend:**

* Steuerung optimieren

# Funktionalität

## 5.1 Pygame installieren

Man geht auf die Seite <http://www.lfd.uci.edu/~gohlke/pythonlibs/#pygame> und lädt sich die Version Pygame für Python 3.4 runter. Diese installiert man mit:

pip install --no-index --find-links=LocalPathToWheelFile PackageName

## 5.2 PyOpenGL und Pillow installieren

Mit pip install pyopengl installiert man sich die Library PyOpenGL.  
Mit pip install pillow installiert man sich die Library Pillow

## 5.3 Texturen mit Pillow

Mit der Library Pillow kann man Bilder öffnen und anzeigen lassen.   
Man muss die Library einbinden mit from PIL import Image.  
Mit im = Image.open("test.jpg") lädt man die Bilder.  
Mit im.show() kann man sich die Bilder anzeigen lassen.  
Mit print(im.format,im.size,im.mode) bekommt man Infos der Bilder.

Um ein Bild zu öffnen und es anzeigen zu lassen verwendet man folgenden Code:

from PIL import Image

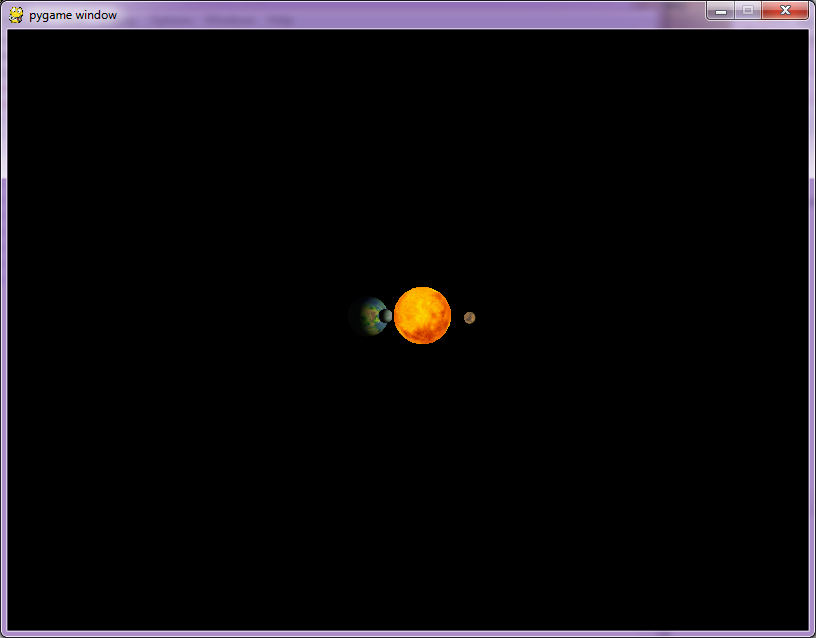
try:  
 im = Image.open("test.jpg") #image laden  
 print(im.format,im.size,im.mode) #infos ueber das bild  
 im.show() #image zeigen

except:  
 print ("Unable to load image")

## 5.4 Sonnensysteminfos

|  |  |  |
| --- | --- | --- |
| Name | Größe (Durchmesser) | Geschwindigkeit um Sonne (km/s) |
| Sonne | 1.390.000 km | - |
| Merkur | 4.900 km | 48 |
| Venus | 12.100 km | 35 |
| Erde | 12.800 km | 29.8 |
| Mars | 6.800 km | 24 |
| Jupiter | 143.000 km | 14 |
| Saturn | 120.500 km | 9.6 |
| Uranus | 51.100 km | 6.8 |
| Neptun | 49.500 km | 5.4 |

## 5.5 Prototyp



## 5.6 Splashscreen

Der Splashscreen wird mit tkinter implementiert.   
Foldende Seite hat mir geholfen den Splashscreen zu mahen.

<http://code.activestate.com/recipes/576936-tkinter-splash-screen/>

## 5.7 Texturen auf Objekte legen

Zuerst muss man Texturen Einschalten, damit die Texturen angezeigt werden.

glEnable(GL\_TEXTURE\_2D)

dann muss man die Texturen auf ein Objekt legen mit

glBindTexture(GL\_TEXTURE\_2D, textures)

## 5.8 Texturen ein/aus schalten

Mit einer changeTexture-Methode wird die Textur ein bzw ausgeschalten. Diese Methode wird mit einem bool am laufen gehalten. Wenn der bool True ist, dann wird die Textur eingeschalten mit glEnable(GL\_TEXTURE\_2D) und die Texturen werden erneuert, außerdem wird mod auf False gesetzt. Wenn der bool falsch ist, wird die Textur ausgeschalet mit glDisable(GL\_TEXTURE\_2D), außerdem wird mod auf True gesetzt.

Nun muss nur noch ein Standartwert gesetzt werden und danach das ganze aufgerufen werden.

if event.key == pygame.K\_t:  
 self.\_\_texture.textureChange()

## 5.9 Messagebox

Die Massagebox ist dazu da, dem User die Tastensteuerung zu erklären. Ich habe im Internet recherchiert wie man diese Messagebox mit tkinter umsetzen kann, kam zu der Lösung „tkintermessagebox“ zu verwenden. Allerdings gibt es das in den neueren Pythonversionen nicht. Durch ein bisschen ausprobieren bin ich dann zu der Lösung gekommen „from tkinter import messagebox“ zu importen und dann die Box mit messagebox.showinfo(„Titel“, „Nachricht“) aufzurufen. Mit \n kann man in die nächste Zeile springen.

messagebox.showinfo("Steuerung", "W ... Ansicht ändern \n T ... Textur [..] ")

# Fehleranalysen

## 6.1 Texturen werden nicht angezeigt

Problem:   
Texturen laden zwar aber sie werden auf den Planeten nicht angezeigt.

Lösung:  
glEnable(GL\_TEXTURE\_2D)

## 6.2 messagebox

Problem:   
Die Messagebox öffnet sich mit der Nachricht, man muss sie aber schließen, damit sich das Programm öffnet. Messagebox verschwindet dann.

Provisorische Lösung:  
Die Steuerung wird auch in der Konsole ausgegeben.

## 6.3 Ruckeln der Applikation

Problem:   
Die Applikation hat geruckelt (Planeten, Drehung, Steuerung).

Lösung:   
Die Texturen kleiner von der Größe her machen.

# Quellen

1. Playlist zu Pygame (Python Game Development) von thenewboston, <https://www.youtube.com/watch?v=K5F-aGDIYaM&list=PL6gx4Cwl9DGAjkwJocj7vlc_mFU-4wXJq>   
   gesehen 23.02.2015
2. Unofficial Window Binaries für Python Extension Packages, <http://www.lfd.uci.edu/~gohlke/pythonlibs/#pygame>   
   gesehen 23.02.2015
3. How do I install Python libraries?, <http://stackoverflow.com/questions/21222114/how-do-i-install-python-libraries>   
   gesehen: 23.02.2015
4. Reference, <http://pillow.readthedocs.org/reference/index.html>   
   gesehen 01.03.2015
5. Umlaufgeschwindigkeiten von Planeten, <http://www.astrologie.de/forum/astrologie-allgemein-f1/umlaufgeschwindigkeit-der-planeten-t1933.html>   
   gesehen 02.03.2015
6. Größenvergleich der Planeten unseres Sonnensystems, <http://www.astronomie.de/astronomie-fuer-kinder/interessantes-fuer-lehrer-eltern/in-der-schule/groessenvergleich-der-planeten/>   
   gesehen 02.03.2015
7. Tkinter Splash Screen (Python Recipe), <http://code.activestate.com/recipes/576936-tkinter-splash-screen/>   
   gesehen 24.03.2015