A07-Solarpy

Maran Christian

TGM

5BHITM

Inhalt

[Aufgabenstellung 2](#_Toc413050960)

[Zeitaufwandsaufzeichnung 3](#_Toc413050961)

[Designüberlegung 3](#_Toc413050962)

[Notwendige Libraries 3](#_Toc413050963)

[Überlegungen 3](#_Toc413050964)

[Codedesign 3](#_Toc413050965)

[GUI-Design 4](#_Toc413050966)

[V1: 4](#_Toc413050967)

# 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 |
| Libraries-Recherche | 00h 30min | 00h 35min | done |
| Anlernen der Libraries | 04h 00min | 02h 30min | In progress |
| Backend-Design | 02h 00min | 00h 45min | In progress |
| GUI-Design | 01h 30min | 00h 40min | In progress |
| 3D-Objekte erstellen | 03h 00min | 00h 15min/Objekt | In progress |
| Texturen zuweisen | 01h 00min |  |  |
| Lichtquelle erzeugen | 02h 00min |  |  |
| Kameraview erzeugen | 02h 00min |  |  |
| Rotation der Objekte | 02h 30min |  | In progress |
| Rotation starten | 01h 00min |  |  |
| Lichtquelle an-/ausschalten | 01h 00min |  |  |
| Code dokumentieren | 01h 30min |  |  |
| Sphinx Doku erzeugen(rst-File etc) | 01h 00min |  |  |
| Protokoll | 02h 00min | 01h 00min | In progress |
| Summe | 24h 30min | 05h 45min |  |

# Designüberlegung

## Notwendige Libraries

* PyQt
* PyGame
* PyOpenGL
* Pillow/Pyglet -> wird sich dann bei der Recherche der beiden Libraries zeigen, welche schlussendlich verwendet wird

## 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

MVC für die Grundstruktur des Projektes um eine gute Trennung der Komponenten zu gewährleisten. Strategy-Pattern für die Drehung?

## GUI-Design

### V1

