|  |
| --- |
|  |
| Visit Our Solar System |
| **SEW** |
| Schwarz Stephan, Gala Mateusz |
| 11/11/2015 |
|  |
| **C:\Users\Stephan\Desktop\TGM_Logo_solo.png**C:\Users\Stephan\Desktop\it_logo.png |

Inhalt

[SEW 0](#_Toc434996000)

[Aufgabenstellung 2](#_Toc434996001)

[Aufwandsabschätzung 5](#_Toc434996002)

### 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!

Lighting

def setupLighting():

*""" Initializing Lighting and Light0*

:return:

*"""*

zeros = (0.15, 0.15, 0.15, 0.3)

ones = (1.0, 1.0, 1.0, 0.3)

half = (0.5, 0.5, 0.5, 0.5)

glMaterialfv(GL\_FRONT\_AND\_BACK, GL\_AMBIENT, zeros)

glMaterialfv(GL\_FRONT\_AND\_BACK, GL\_SPECULAR, half)

glMaterialf(GL\_FRONT\_AND\_BACK, GL\_SHININESS, 15)

glLightfv(GL\_LIGHT0, GL\_AMBIENT, zeros)

glLightfv(GL\_LIGHT0, GL\_DIFFUSE, ones)

glLightfv(GL\_LIGHT0, GL\_SPECULAR, half)

glEnable(GL\_LIGHT0)

glEnable(GL\_LIGHTING)

glColorMaterial(GL\_FRONT\_AND\_BACK, GL\_DIFFUSE)

glTexGeni(GL\_S, GL\_TEXTURE\_GEN\_MODE, GL\_SPHERE\_MAP)

glTexGeni(GL\_T, GL\_TEXTURE\_GEN\_MODE, GL\_SPHERE\_MAP)

glEnable(GL\_TEXTURE\_GEN\_S)

glEnable(GL\_TEXTURE\_GEN\_T)

glEnable(GL\_COLOR\_MATERIAL)

glEnable(GL\_NORMALIZE)

glShadeModel(GL\_SMOOTH)

### Aufwandsabschätzung

|  |  |  |
| --- | --- | --- |
|  | Schwarz | Gala |
| Recherche |  |  |
| Umsetzung |  |  |
| Dokumentation |  |  |
| Stunden |  |  |