Dodgeball Simulator

Doris Rhomberg (01649642)
Jan Kompatscher (01527584)

Implementierung

Von der **Main** aus werden alle weiteren Klassen aufgerufen. Dort werden auch Teile von PhysX implementiert und der Raum, in denen sich die Figuren bewegen, ist hard-coded.
Im **Window** wird das Fenster erstellt.
In der Klasse **Settings** werden alle Interaktionen implementiert. Die intuitive Steuerung beinhaltet Gehen, Laufen, Herumschauen und Werfen des Balles. Der Ball wird in Richtung eines Punktes etwas über der Bildschirmmitte geworfen, damit die Gegner nicht vom Spieler selbst verdeckt werden, wenn auf sie gezielt wird.
In den Klassen **Model** und **Mesh** werden die Modelle mittels Assimp geladen.
Die Klasse **Ball** erstellt einen Ball.
Unsere **Camera** ist eine 3rd Person Kamera. Sie rotiert um den Charakter, der sich jedoch nicht mitdreht, und kann nicht höher als 50 Grad und niedriger als -40 Grad bewegen.
CollisionCallback meldet sich, wenn eine Kollision passiert und ruft dann Enemy/PlayerCharacter.hit() auf.
Enemy implementiert die Gegner, die sich random bewegen. **PlayerCharacter** implementiert den eigenen Spieler.
Die Klasse **Physics** hat bislang noch keine Funktion, da wir das alles in der Main realisieren und das aber in Zukunft in diese Klasse auslagern möchten. Bislang verwenden wir die Physik für den Ball und die collision detection (zwischen Ball und Charactern).
Shader erzeugt Shader.
Text_Renderer erstellt und rendert Text mit der FreeType Library.

Features of the Game

- Vom Start Game Screen wird mit "ENTER" das Spiel gestartet.
- Bewegen mit WASD, laufen durch gleichzeitiges Drücken von shift.
- Der Ball geht automatisch an den Charakter über, in dessen Feld er zur Ruhe kommt.
- Werfen des Balles durch Drücken der linken Maustaste.
- Zurücksetzen des Spiels durch Drücken von „R“.
- Aktivieren/deaktivieren des HUDs durch Drücken von „H“.
- Anzeige der Leben des Spielers mit Hilfe von Herzen im linken oberen Eck.
- Anzeige ob der Spieler den Ball hat durch Ball-zeichen im rechten oberen Eck.
- Anzeige des Spielstandes in der oberen Mitte des Bildschirms.
- Spielende, wenn einer der Teams 3 Punkte erzielt hat.
- GameOver und GameLost Screen.

Texturen

Die Texturen sind an die mit Assimp eingebundenen Modelle, die mit Blender erstellt wurden, angefügt. Die Textur wird dann im Fragment Shader durch entsprechende Kolorierung der entsprechenden Fragmente auf die Modelle aufgetragen.

Modell Spieler: https://free3d.com/3d-model/little-boy-36169.html

Texturen Turnhalle: https://www.cg.tuwien.ac.at/courses/Textures/
Textur Ball: https://www.cgtrader.com/free-3d-models/sports/toy/soviet-toy-ball

Zusätzliche Bibliotheken

Assimp: https://github.com/assimp/assimp/releases/tag/v4.1.0/
Freetype: https://sourceforge.net/projects/freetype/files/freetype2/
GLEW: http://glew.sourceforge.net/
GLFW: https://www.glfw.org/download.html
GLM: https://glm.g-truc.net/0.9.9/index.html
PHYSX: https://developer.nvidia.com/physx-sdk