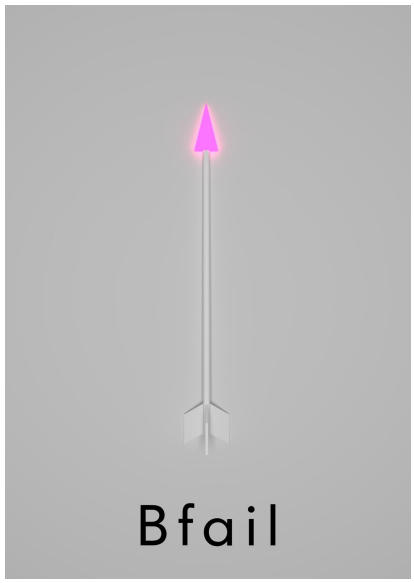
Bfail



Aaron Zingerle (1527398)
Daniel Ratzinger (1427247)

Die aktuelle Implementierung umfasst die Basics von Model-Loading, Physik-Implementierung und Rendering.

“Features”

- rudimentäre Pfeil-Mechanik
- bewegliche Kamera
- Lighting mit einer Lichtquelle
- Blinn-Phong-Shader mit Texture Mapping

“Gameplay”

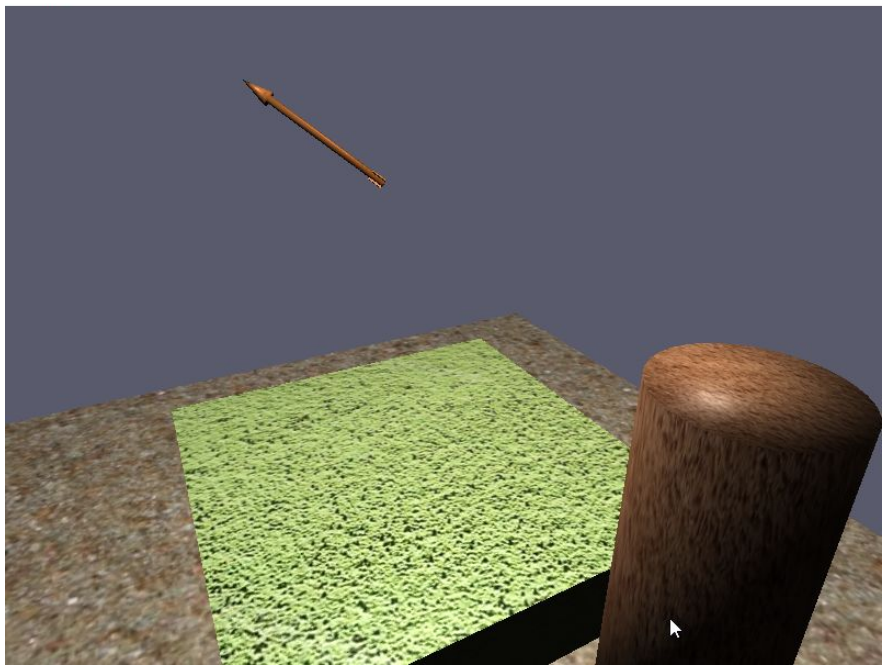
In der Abgabeversion kann der Spieler die Kamera durch die Szene steuern und von der Kameraposition aus einen Pfeil “schießen” (werfen). Einzige Physikalische Interaktion des Pfeils ist aber im Moment die Schwerkraft und eine Groundplane.

Controls

Die Kamera ist mit typischer WASD-Steuerung und per Maus frei beweglich.

Tastenbelegung

W	*Kamera vorwärts*
S	*Kamera rückwärts*
A	*Kamera links (strafe left)*
D	*Kamera rechts (strafe right)*
Maus	*Kamera-Sicht*
Space	*Pfeil schießen*
F	*FPS-output toggle*
R	*Szene zurücksetzen*
Q	*Physikberechnung anhalten / fortsetzen*



Implementierung

Für die Abgabe wurden zwei Models in Blender erstellt und texturiert (Pfeil und Environment-Model). Die dabei aktuell verwendeten Texturen sollen nicht der Ästhetik dienen, sondern die Modellstrukturen besser veranschaulichen.

Wir verwenden ***Assimp*** um die Models zu laden (die momentan ausschließlich aus .obj-Files bestehen) . Dabei benutzen wir zwei Klassen: die Model und die Mesh Klasse. Der Model-klasse geben wir das File zum Modell, die dann daraus die einzelnen Nodes des Modells ausliest und diese als Mesh-Objekte speichert. Die Model-Klasse haben wir noch in eine GameObject Klasse gepackt, die zusätzliche Informationen zum Modell enthält und von SceneObject ableitet. Sie speichert Informationen wie: die zugehörige model/transform-matrix, zugehörige Collision-Shape.

Weiters verwenden wir aktuell einen Blinn-Phong-Shader der in GLSL implementiert ist, um die Objekte mit einer statischen Lichtquelle zu beleuchten. Hierbei verwenden wir zusätzlich zwei Klassen: die Camera und Shader Klasse. Die Camera Klasse definiert einfach eine Kamera, deren Position und Blickrichtung und enthält zusätzlich noch Methoden um die Kamera zu transformieren/ im Raum zu verschieben. Die Shader Klasse dient zum kompilieren und zum Handhaben der Shader.

Für die Physikberechnung und Kollisionsabfrage benutzen wir ***Bullet***. Die für die einzelnen Objekten der Szene verwendeten Collider-Shapes, erstellen wir derzeit noch im Code. (wollen sie aber später nach Möglichkeit automatisch generieren, bzw. Über ein Bullet File einlesen)

Known Issues

- Pfeilbewegung etwas random
- Groundplane stimmt nicht mit dem Environment-Model überein
- Schattierung am Pfeil wird bei Drehung nicht aktualisiert (Lichtquelle hardcoded)

Ressourcen

- learnopengl.com
- stackoverflow <3
- tuwel