**Dokumentation Golfinity VR**

Virtuelle Realität und Animation SS 23

Patrick Flögel 85824

Felix Kurz 86041

Markus Kohlstrunk 86765

Lukas Deli 86764

**Inhaltsverzeichnis**

Spielidee

Verwendete Assets

Skripte

Zusätzliche Abgabe: Blender Datei

Spielidee:

Als Spielidee haben wir uns Minigolf ausgesucht. Unser Spiel orientiert sich dabei stark an realitätsnahen Maßen und Regeln. Man spielt einen Rundkurs, welcher 18 Bahnen enthält und man eben mit möglichst wenig Schlägen durchkommen soll.

Dadurch, dass das Spiel eben nur ein Spiel ist, kann man sich einige Bonusfeatures zu Nutze machen. Hiermit ist gemeint, dass sobald der Ball im Gras landet er an den Anfang der Bahn zurückgesetzt wird.

Des Weiteren haben wir uns überlegt, dass jede Bahn ein kleines Holzschild besitzt, welches Schläge und Highscore anzeigt. Zuletzt gibt es noch ein großes Holzschild, welches die Gesamtschläge und den Gesamthighscore anzeigt. Sobald man bei einer Bahn eingelocht hat, werden der Ball und man selbst zur nächsten Bahn teleportiert.

Verwendete Assets:

Für unser Projekt haben wir nur wenige Assets verwendet. Alle Assets, die hier nicht aufgelistet werden, gehen aus Eigenproduktion hervor.

Gras Textur:

<https://www.istockphoto.com/de/foto/nahtlose-rasen-gras-wiederholen-rasen-textur-gm1159879162-317320586>

Wald Backdrop Textur:

Enthalten im Standardumfang des Computerspiels OMSI II

Baum 3D-Objekt:

<https://sketchfab.com/3d-models/low-poly-tree-70f0e767fc2f449fa6fef9c2308b395f>

Ambiente Sound Effekt:

<https://getsoundly.com>

Alle weiteren Sound Effekte:

<https://www.zapsplat.com>

Skripte:

1. Bahn.cs

Ein Bild, das Text, Screenshot, Software enthält.

Automatisch generierte Beschreibung

Dieses Skript ist als Component zu jeder Bahn hinzugefügt. Es stellt für das Script Schlagzähler.cs die Assoziation zu dem zur Bahn gehörenden Schild zur Verfügung, sowie die Positionen, an die der Ball und der Spieler teleportiert werden können. Das Schild wird automatisch aus den Kind-Objekten des Bahn-Objekts gesucht. Die Positionen ergeben sich dem Ursprungspunkt der Bahn, welcher immer an der Vorderkante der Bahn liegt.

1. Bahnschild.cs

Ein Bild, das Text, Screenshot enthält.

Automatisch generierte Beschreibung

Dieses Skript ist als Component zu jedem kleinen Schild hinzugefügt, das am Rand einer Bahn steht. Es verwaltet den auf dem Schild stehenden Text sowie die Anzahl der Schläge auf der Bahn und den Bahnhighscore. Die Text-Objekte werden automatisch aus den Kind-Objekten des Schild-Objekts gesucht. Über das „PlayerPrefs“-Objekt werden die Highscores auf der Festplatte gespeichert, damit sie auch nach dem Neustart des Programms noch zur Verfügung stehen.  
(Fortsetzung des Scripts auf Folgeseite)

Ein Bild, das Text, Screenshot, Software enthält.

Automatisch generierte Beschreibung

1. Schlagzaehler.cs

Ein Bild, das Text, Screenshot enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text, Screenshot, Software enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text, Screenshot enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text, Screenshot enthält.

Automatisch generierte Beschreibung

Dieses Skript ist auf dem Ball als Component hinzugefügt. Es regelt alles, was mit dem Zählen der Schläge und setzen der Highscores zu tun hat, sowie das Verhalten des Balls im Falle einer Kollision mit dem Schläger oder dem Gras bzw. das Verhalten des Balls, wenn dieser in das Loch fällt. Alle Bahnen werden in dem Array „bahnen“ referenziert. Die Reihenfolge der Objekte bestimmt die Reihenfolge, in welcher die Bahnen bespielt werden.

1. ClubHead.cs

Ein Bild, das Text, Screenshot, Software enthält.

Automatisch generierte Beschreibung

Dieses Skirpt ist als Component auf dem Kopf des Goldschlägers hinzugefügt. Es berechnet die Bewegungsrichtung und -geschwindigkeit des Golfschlägers, was bei der Kollision mit dem Golfball von dem Skrip „Schlagzaehler.cs“ verwendet wird.

1. SceneTransitionManager.cs

Ein Bild, das Text, Screenshot enthält.

Automatisch generierte Beschreibung

Dieses Skript ist als Component dem Player-Objekt hinzugefügt. Es organisiert den Szenenwechsel, wenn der Spieler von der Lobby zum Rundkurs oder wieder zurück teleportiert wird.

1. FadeScreen.cs

Ein Bild, das Text, Screenshot, Software enthält.

Automatisch generierte Beschreibung

Dieses Skript ist als Component auf dem Kamera-Objekt hinzugefügt. Es sollte einen Übergang zu/von Schwarz bei der Teleportation von der Lobby zum Rundkurs oder wieder zurück realisieren. Davon wurde im Rahmen des Testens später allerdings abgesehen und die Übergangsdauer deshalb auf 0 gesetzt.

Zusätzliche Abgabe: Blender Datei:

Vorwort:

Als zusätzliche Abgabe haben wir noch eine Blender Datei hinterlegt, welche das Erstellen von Golfbahnen erleichtert. Die Erstellung war recht aufwändig, weshalb es schade wäre die Datei nicht mitabzugeben.

Die Blender Datei enthält eine Collection namens „Assets“, welche verschiedene Segmente einer Golfbahn enthält. Eine Bahn kann dann durch das Zusammenfügen der Segmente erstellt werden und man kann natürlich auch noch eigene zusätzliche Elemente hinzu modellieren. Im Folgenden werden die einzelnen Segmente mit ihren veränderlichen Eigenschaften beschrieben.

Wichtige Information:

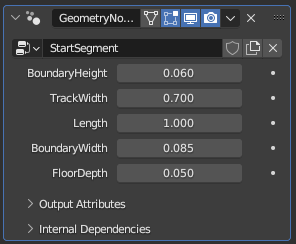
Bei jedem Segment kann man die „BoundaryHeight“ (Bandenhöhe), „BoundaryWidth“ (Bandenbreite), „TrackWidth“ (Streckenbreite), „Length“ (Länge) und die „FloorDepth“ (Bodentiefe) beliebig verändern bzw. die Größe / Skalierung einstellen.

Im Folgenden werden deswegen nur die anderen veränderbaren Werte beschriben.

Start Segment:

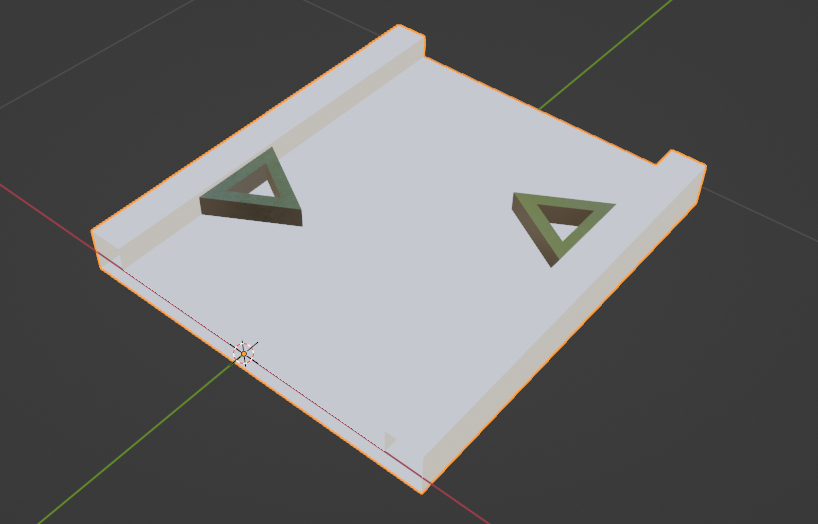
Das „StartSegment“ bildet den Anfang einer Bahn oder ist als Eckstück vorhanden und es sind lediglich die zuvor genannten Werte veränderlich.

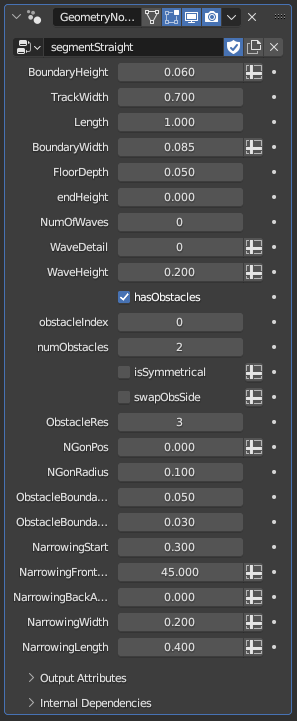




Straight Segment:

Das „StraightSegment“ ist ein gerades Verbindungsstück, welches viele Option mit sich bringt. Das Segment kann Hindernisse enthalten, wenn die Checkbox „HasObstacles“ ausgewählt ist, die „Endheight“ null ist und „NumOfWaves“ null ist.

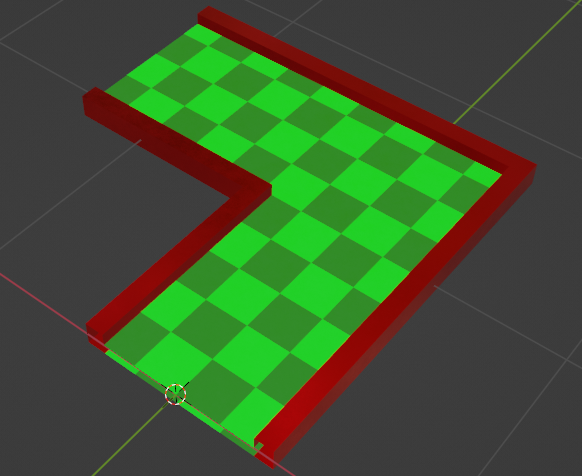


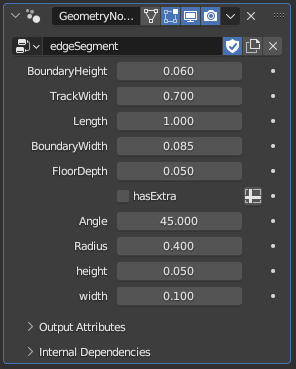


* „endHeight“ gibt den Höhenunterschied zwischen Start und Ende der Bahn an
* „NumOfWaves“ gibt die Anzahl der Wellen des Segments an
* „WaveDetail“ gibt an, wie detailliert die Rundung der Wellen ist
* „WaveHeight“ gibt die Höhe der Wellen an
* „HasObstacles“: Wenn die Checkbox „HasObstacles“ ausgewählt ist, die „Endheight“ null ist und „NumOfWaves“ null ist, kann das Segment Hindernisse enthalten
* „obstacleIndex“: Es gibt verschiedene Obstacles, welche über diesen Index ausgewählt werden können
* „numObstacles“ gibt die Anzhal der Obstacles an
* „isSymmetrical“ ordnet die Obstacles symmetrisch an, wenn ausgewählt
* „swapObsSide“ Spiegelung der Obstacle Anordnung
* „ObstacleRes“ Anzahl der Ecken des Obstacles (gleichseitiges Vieleck)
* „NGonPos“ ist das Offset entlang der Bahn (Verschiebung der Obstacles)
* „NGonRadius“ gibt den Radius des Obstacles an
* „ObstacleBoundaryHeight“ gibt die Höhe des Obstacles an
* „ObstacleBoundaryWidth“ gibt die Breite des Obstacles an
* „NarrowingStart“ gibt das Offset für Narrowing an
* „NarrowingFrontAngle“ ist der Winkel des vorderen Teils des Obstacles
* „NarrowingBackAngle“ ist der Winkel des hinteren Teils des Obstacles
* „NarrowingWidth“ gibt an wie weit das Obstacle in die Bahn ragt
* „NarrowingLength“ gibt die Länge des Obstacles an

Edge Segment:

Das „EdgeSegment“ ist ein 90° Winkel bzw. ein Eckstück der Bahn. Dieses kann noch ein extra Element im Eck haben.

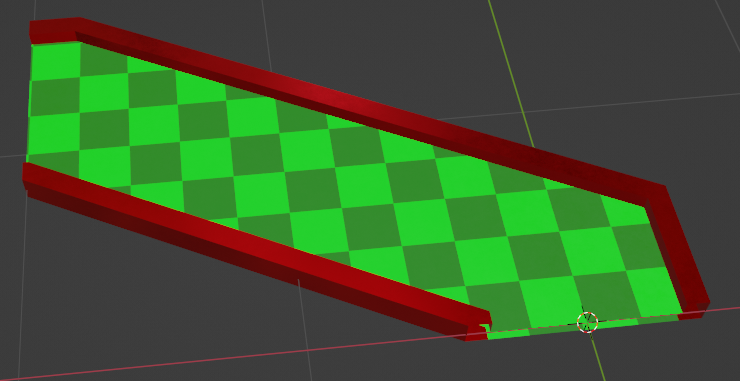


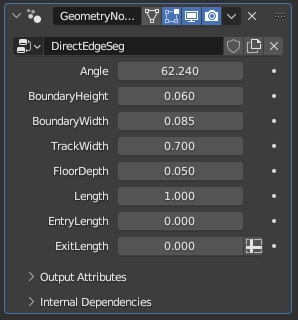


* „Angle“ ist der Winkel des extra Elements
* „Radius“ gibt die Tiefe der Einrückung an
* „height“ gibt die Höhe des extra Elements an
* „width“ gibt die Breite des extra Elements an

Direct Edge Segment:

Das „DirectEdgeSegment“ ist ein Eckstück, wobei man den Winkel verstellen kann. Des Weiteren können eine „EntryLength“ und „ExitLength“ eingestellt werden.

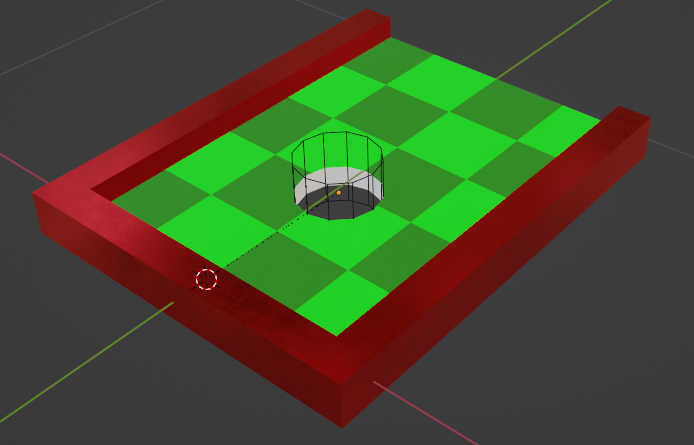


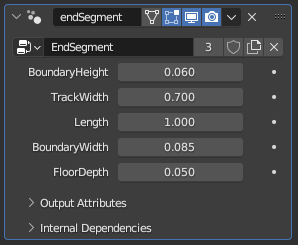
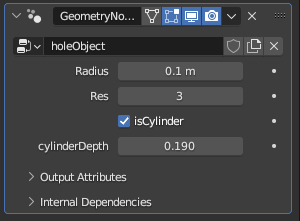


* „Angle“ gibt den Winkel an
* „Length“ gibt die Distanz der Mittelpunkte von Anfang und Ende des Segments an
* „EntryLength“ gibt an, wie lange die gerade Stecke vor dem Winkel sein soll
* „ExitLength“ gibt an, wie lange die gerade Strecke nach dem Winkel sein soll

End Segment:

Das „EndSegment“ bildet das typische Ende der Bahn und enthält somit auch das Loch. Das „EndSegment“ hat ein Kindobjekt das „HoleObject“. Aus dem „EndSegment“ wird mit Hilfe des „HoleObject“ das Loch ausgeschnitten. Hierbei ist Folgendes variabel:



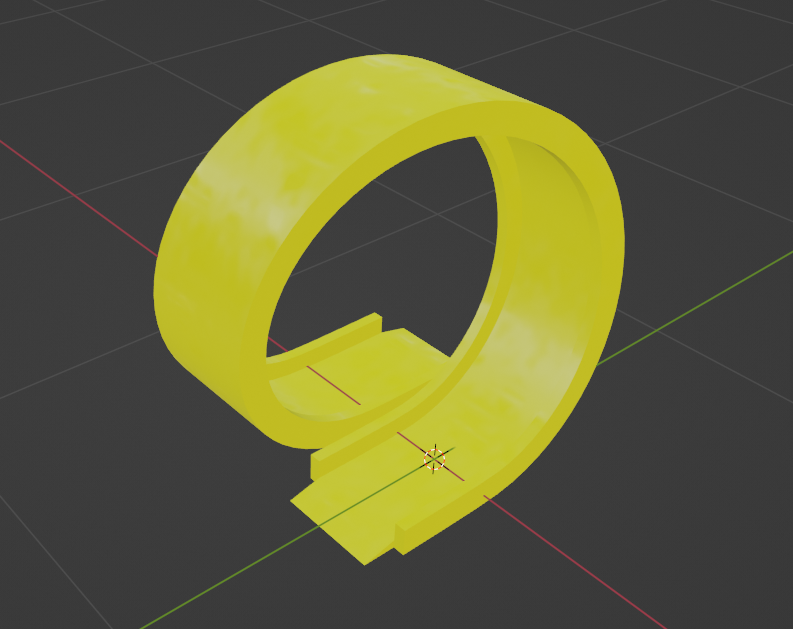
 

* „Radius“ gibt den Radius an
* „Res“ gibt den Grad der Rundung an
* „isCylinder“ gibt, wenn ausgewählt die Form eines Zylinders, wenn nicht, die Form einer Kugel
* „cylinderDepth“ gibt die Höhe des Zylinders an, falls zuvor ausgewählt

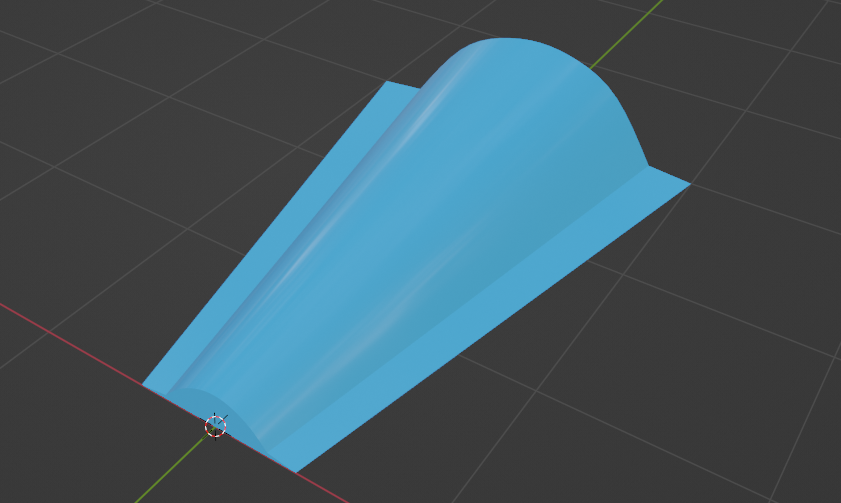
Obstacles:

Neben Der Collection „Assets“ existiert auch eine Collection names „Obstacles“, welche typische Hindernisse einer Minigolfbahn enthält.

Das erste Element ist ein „Looping“:



Das zweite Element sind „RoundObs“:



Erstellen einer Bahn:

Diese Segmente sollte man nun passend zusammenlegen, sodass die Kanten aufeinanderliegen.

Der GeometryNodes Modifier muss nun für jedes Segment applied werden. Danach muss man die Segmente joinen, sodass man genau ein Objekt erhält.

Als nächstes wechselt man in den Edit Mode, wählt alles aus und führt die Aktion „Merge by Distance“ aus.

Sollte man selbst gemodelte Objekte oder das „StraightSegment“ verwendet haben, so muss man diesen Elementen die Materials selbst zuweisen. Ansonsten sind die Materials bereits vorgefertigt.

Bei runden Elementen sollte man noch die Funktion „Shade Auto Smooth“ verwenden.

Am Ende sollte man das „HoleObject“ beim „EndSegment“ löschen.