Spiel: Astro Blaster

Gruppe: Nostalgic Duo

Team: Pfammatter Joel, Wüthrich Markus

Ein Bild, das Text, Natur, Nachthimmel, Outdoorobjekt enthält.

Automatisch generierte Beschreibung

Astro Blaster ist ein Arcade Game welches 1981 released wurde. Es ist ein Shooter in welchem der Spieler ein Raumschiff steuert und mehrere Wellen von Gegnern zerstört. Wir haben dieses Game nun in 3D neu umgesetzt.

Link zum Unity Projekt im Gitlab:

https://gitlab.enterpriselab.ch/ImmersiveRealities/StudentProjects/gamedev-hs21/semesterprojekt/g06-nostalgic-duo/astro\_blaster

Link zum WebGL Build auf Gitlab:

https://gitlab.enterpriselab.ch/ImmersiveRealities/StudentProjects/gamedev-hs21/semesterprojekt/g06-nostalgic-duo/astro\_blaster\_build

WASD oder Pfeiltasten um das Raumschiff zu bewegen und Leertaste zum Schiessen.

# Umsetzung

## Funktionalität

* Bewegung über WASD und Pfeiltasten
* Schiessen über Leertaste
* Gegner sind zerstörbar
* Gegner bewegen sich
* Basic UI (health, heat, fuel, score)
* Waffe überhitzt, wenn man zu schnell schiesst.
* Fuel wird mit der Zeit verringert
* Score anhand der getöteten Gegner und dem übriggebliebenen Fuel
* Sound für Juiciness
* Modelle sind selbst mit Blender erstellt worden

## Best Practice

|  |  |
| --- | --- |
| Element | Wo zu finden? |
| Single Responsibility Principle | Enemy prefab, Player prefab, Health Behaviour, Singleton Reference Provider Behaviour, Score Manipulate Behaviour, Random Action Run Behaviour, Pooled Object Behaviour, Patrol Point Move Behaviour, Destroy On Collision Behaviour, other behaviours |
| Managers | UIManager, ShootManager, ScoreManager, FuelManager, PlayerMovement, other behaviours |
| Events | Hittable Behaviour, Random Action Run Behaviour, Pooled Projectile Shoot Behaviour, other behaviours |
| Object Pooling | Enemies, Enemy Bullets, Player Bullets |
| Interfaces | TargetController.cs, EnemyDestroyerBehaviour.cs and other behaviours accessing global components |

# Nutzungstests

## Test 1

Ein Bild, das Text, Screenshot enthält.

Automatisch generierte Beschreibung

|  |  |
| --- | --- |
| Datum | 18.11.2021 |
| Teilnehmende | Informatik Student |
| Stand Spiel | Alpha Version |
| Ziel | Wie fühlt sich das Spiel an? |
| Dauer | 5 Minuten |
| Methode | Think Aloud |

|  |  |  |
| --- | --- | --- |
| Identifiziertes Problem | Lösung | Umgesetzt? |
| Die Steuerung des Raumschiffs ist zu langsam | Den Speed der Bewegungssteuerung erhöhen | Ja |
| Die Gegner bewegen sich zu langsam | Den Speed der Gegner erhöhen | Ja |
| Gegner ausserhalb des Schiessbereichs | Gegner klarer aus dem Schiessbereich entfernen wenn sie unerreichbar sein sollten | Ja |
| Der Spieler versteckt zu viel vom Spiel | Spieler verkleinern | Ja |

## Test 2

|  |  |
| --- | --- |
| Datum | 19.11.2021 |
| Teilnehmende | Informatik Studenten |
| Stand Spiel | Alpha Version |
| Ziel | Wie fühlt sich das Spiel an? |
| Dauer | 5 Minuten |
| Methode | Think Aloud |

|  |  |  |
| --- | --- | --- |
| Identifiziertes Problem | Lösung | Umgesetzt? |
| Gegner ausserhalb des Schiessbereichs | Gegner klarer aus dem Schiessbereich entfernen, wenn sie unerreichbar sein sollten | Ja |
| Schiff "neigt" sich bei Bewegung in die Begegungsrichtung damit sich der Schiessbereich vergrössert | Nein |
| Der Spieler versteckt zu viel vom Spiel | Spieler verkleinern | Ja |
| 4-dimensionale Bewegung "clunky", 8-dimensionale wurde erwartet | 8-dimensionales movement erlauben | Nein |
| Schwierig zu sehen wo die Schüsse landen werden | Laser pointer für den Spieler erstellen | Nein |
| Dem Spieler mehr Schüsse erlauben damit der Spieler mit den Schüssen "zielen" kann | Ja |
| Gegner wehren sich nicht | Gegner schiessen zurück | Ja |
| Überhitzungsstatus schwierig zu sehen | Überhitzungsstatus am Spieler darstellen mit z.B. rauch oder rot glühendem Schiff | Nein |
| Overheat geht zu langsam runter | Geschwindigkeit erhöhen mit dem der Overheat counter runtergeht (frequency) | Ja |
| Spieler und Gegner sehr schwierig zu unterscheiden | Spieler und Gegner erhalten eigene 3D-Modelle | Ja |

# Playability

|  |  |  |
| --- | --- | --- |
| Kriterium | Quelle | Beschreibung |
| Visibility of system Status | Vorlesung | Score, Überhitzung, Benzin, ... wird gut sichtbar unten auf dem Screen angezeigt. |
| Consistency and standards, Recognition rather than recall | Vorlesung | Das Raumschiff kann mit WASD und auch mit den Pfeiltasten bedient werden. (Gamer verwenden WASD aber nicht Gamer verwenden intuitiv die Pfeiltasten) |
| Aesthetic and minimalist design | Vorlesung | Simplistische Grafik und klare Unterscheidung von Spielrelevanten Elementen |

# Barrierefreiheit

|  |  |  |
| --- | --- | --- |
| Kriterium | Quelle | Beschreibung |
| Barrierefreiheit | Includification (Vorlesung) | Rot/Grün Farbschema für Spieler & Gegner Schüsse geändert zu einem freundlicheren Rot/Blau Farbschema für übliche Arten von Farbenblindheit |
| Barrierefreiheit | Basic Accessibility Guidlines[[1]](#footnote-1) | Motor & Cognitive simplicity of Controls; few buttons, alternative input |

# Juiciness

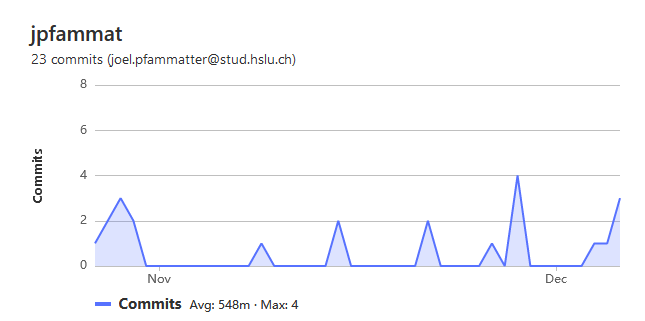
* Beim Schiessen wird ein arcade Blaster Sound abgespielt.
* Wenn die Gegner sterben, wird ein arcade Explosion Sound abgespielt.
* Wenn der Spieler getroffen wird, ein arcade Explosion Sound abgespielt.

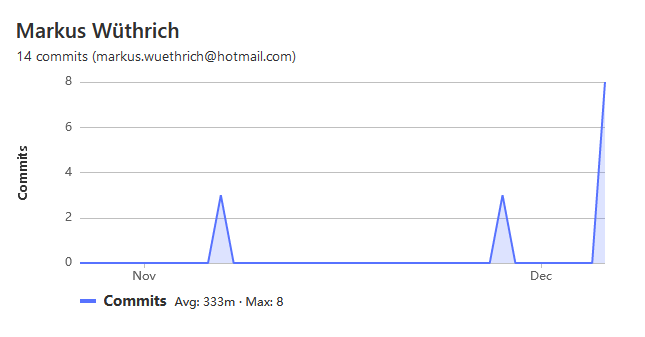
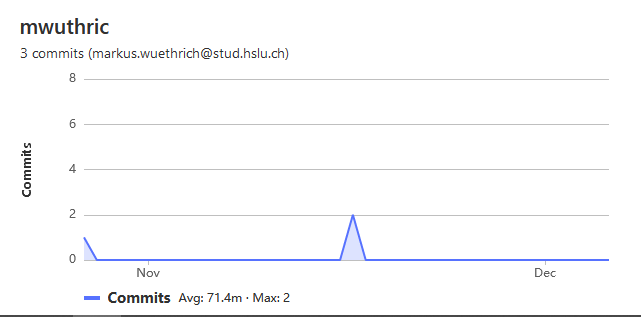
# Individueller Beitrag

## Übersicht Aktivitäten

|  |  |  |
| --- | --- | --- |
|  | Pfammatter Joel |  |
| Player Movement | X |  |
| Score Manager | X |  |
| Fuel function | X | X |
| Overheat function | X | X |
| Player shoot behaviour | X |  |
| Blender Models | X |  |
| UI | X | X |
| Enemy Movement |  | X |
| Enemy Shooting |  | X |
| Player Health |  | X |

## Gitlab Statistiken



* Markus Wüthrich: Die Commits sind z.T. sehr gross: Beispiele:
  + Commit 40deb972f896d1a5f7f69150b4e5b6ae49ae25a4,
  + Commit d4a2d1a8ff23f8b40373e0ad4607bf68b977fb6f
* Erklärung, ob nach Eindruck der Gruppe alle einen vergleichbaren Beitrag geleistet haben.

Ja, beide Mitglieder haben einen Vergleichbaren Beitrag geleistet.

## Reflektion Pfammatter Joel

Problem: Es gab mehrere Probleme das Schiff zu steuern. Ich musste das Schiff mit WASD und den Pfeiltasten steuern können. Dass Schiff braucht auch einen Collider weil es im späteren Verlauf des Spiels von den Gegnern auch zerstört werden kann. Zudem musste ich dem Schiff mitteilen, wie weit es sich bewegen kann. Da das Schiff in der Luft fliegt und nicht auf dem Boden durfte es keine Gravität haben. Dies erschwerte wiederum die Steuerung mithilfe von Force da ich dafür Gravität gebraucht hätte. Um den Rand des Spielfeldes zu markieren, musste ich mehrere Optionen mit Kollisionen oder auch mit Triggern ausprobieren und nachlesen, wie diese sich genau verhalten.

Lösung: Damit es schlussendlich funktionierte gab ich dem Schiff einen Rigidbody ohne Gravity und einen Collider. Ich erstellte ein GameObject PlayerField welches vier Box Collider Kinder enthält. Das GameObject PlayerField enthällt nur einen Rigidbody ohne Gravity. Die vier Kinder enthalten jeweils einen Box Collider welcher mit Trigger ausgelöst wird.

Ein Bild, das Text, Tennis enthält.

Automatisch generierte Beschreibung

Abbildung 1: Playerfield mit 4 Boxcolliders

Im Script wird dann abgefragt, ob das Schiff mit einem dieser Collider getriggert ist. Wenn ja enthält das Schiff einen Lock so dass es zum Beispiel nicht weiter nach rechts gehen kann. Um das Schiff zu steuern wird bei einem Tastendruck die alte Position des Schiffs abgefragt und dann mittels der Funktion Vector3.Lerp() das Schiff um eine gewisse Anzahl Pixel nach x oder y verschoben.

## Reflektion Markus Wüthrich

Problem: Race Condition bei Referenzen die in der Start()-Methode initialisiert werden.

Lösung:

1. Referenzen in SingletonReferenceProviderBehaviour.cs auslagern so dass es einige Funktionalitäten übernimmt:

* Kreiert genau eine Singleton-Instanz die alle globalen Referenzen enthält, egal wie oft das Behaviour in der Szene vorkommt
* Lazy-Initialized die dependencies "at runtime" so dass die meisten erst in einem "Update()" Zyklus referenziert werden.

# Anhang

## 3rd Party Assets

|  |  |  |
| --- | --- | --- |
| Asset | Quelle | Lizenz |
| Skybox | https://assetstore.unity.com/packages/2d/textures-materials/classic-space-skybox-11596 | Free |
| Sounds | https://www.zapsplat.com/ | Free |
| Ship Texture | https://3dtextures.me/2019/08/28/techno-greeble-001/ | Free |
| Power Texture | http://bgfons.com/download/1421 | Free |
| Weapon Texture | https://www.wildtextures.com/free-textures/black-scratched-metal-plate/ | Free |

## Code Snippets

|  |  |
| --- | --- |
| Code | Quelle |
|  |  |

## Daten Nutzungstests

Notizen des Tests am 19.11.2021, mitschrieb des Testers Think-Aloud methodik

- Gegner ausserhalb des Bereichs

- Perspektive zu extrem

- Oft versteckt der spieler die Kugel

- Fühlt sich langsam an

- 8-directional movement vermisst

- schwierig mit perspektive zu sehen wo die schüsse landen werden

- gegner wehren sich nicht

- graphical display for overheat

- feedback when can0t shoot cause overheat

- Gegner in einer Reihe kann sehr einfach sein wenn man direkt die position hat

- Overheat geht zu langsam runter

Testrunde 2

- Orientierung in 3d-space, zielen schwierig

- Overheat leiste nicht offensichtlich -> warum kann ich nicht schiessen?

- Spieler und gegner schwierig zu unterscheiden

- Spiel fühlt sich langsam an

- Obere gegner kann man nicht treffen

-

1. http://gameaccessibilityguidelines.com/basic/ [↑](#footnote-ref-1)