Cyber Bot

VR-Projektdokumentation

Virtuelle Realität und Animation / Autorensysteme
Sommersemester 2023
Prof. Dr. Carsten Lecon
Stefan Wehrenberg

Gruppe 2
Jenny Jost
Lea Marie Kindermann
Jana Weber
Lukas Wolf

Inhalt

1. Team

83272 - Jenny Jost
Software Engineering

82789 - Lea Marie Kindermann
Software Engineering

83122 - Jana Weber
User Experience

83526 - Lukas Wolf
User Experience

2. Aufgabenstellung

In Gruppenarbeit soll eine VR-Anwendung (Spiel) mit der Unity-Engine erstellt werden. Alle Assets sind erlaubt (mit Quellenangaben!). Es muss dokumentiert werden, welche Assets benutzt werden (Abgrenzung zu eigenen Assets).
Es soll mindestens eine Spielmechanik implementiert werden, die Feedback in Form von UI-Elementen (Punktestand, Treffermarkierung, Controller-Vibration, ...) bietet. Es muss ein grundlegendes Menü gestaltet werden, das mindestens das Verlassen der Anwendung ermöglicht.
Bei der Gestaltung der Anwendung dürfen Objekte aus dem Blender-Projekt einbezogen werden. Weitere Objekte müssen nicht zwangsläufig im höchsten Detaillierungsgrad vorliegen; es reicht, wenn die Form/ Funktion erkennbar ist.
Das Thema der Anwendung kann frei gewählt werden, muss aber mit dem Dozenten abgesprochen werden (Termin siehe unten).
Dieses Projekt darf im Team bearbeitet werden (idealweise 2-4 Personen - interdisziplinär). Pro Gruppe sollte nur eine Abgabe erfolgen (mit Angabe der Gruppenmitglieder).
Die Abgabe erfolgt über Git (notfalls über BwSync&Share) und besteht aus:

- Kurze „Spielanleitung“
- Dokumentation
- Unity-Projekt mit allen Dateien
- Kurzes Demo-Video ihres Spiels (z.B. mit OBS aufgenommen)
- Im Labor lauffähiger Windows/Android-Build der Anwendung

3. Projektidee

Unsere kreative Projektidee entstand aus dem Wunsch, einen virtuellen Shooter zu entwickeln. Mit Begeisterung beschlossen wir, den mit Hilfe von Blender erstellen Roboter erneut zu verwenden und in das Spiel zu integrieren. Ziel ist es, den Roboter durch gezielte Schüsse auszuschalten. Dabei wurde uns schnell bewusst, dass der Roboter, den wir zuvor in einer vorherigen Aufgabe entworfen hatten, perfekt in dieses Konzept passte und somit eine ideale Ergänzung darstellt.

Bei der Wahl der Umgebung hatten wir zunächst die Idee, eine Lagerhalle als Schauplatz zu nutzen. Doch je weiter wir in die Konzeptphase eintauchten, desto faszinierender erschien uns die Vorstellung, den Spieler in den Weltraum zu versetzen - genauer gesagt, auf ein verlassenes Raumschiff. Diese

Entscheidung gibt uns die Möglichkeit, eine Spielwelt zu erschaffen, in der der Spieler das Gefühl hat, im Gang eines verlassenen, rostigen Raumschiffs zu stehen.

Unsere Projektidee ist ein packendes Spielerlebnis, bei dem der Spieler die herausfordernde Aufgabe hat, den Roboter auf dem verlassenen Raumschiff zu besiegen. Der virtuelle Kampf gegen die immer neu erscheinenden Roboter schafft eine spannende und immersive Spielwelt, die die Spieler in ihren Bann zieht.

4. Projektumsetzung

4.1 Software

- Blender
- Unity
- Clideo (https://clideo.com/de/cut-audio)

Das Spiel wurde von uns mit der Unity-Engine erstellt.
Die Robotermodelle, sowie deren Animationen wurden bereits in der vorherigen Aufgabe mithilfe der Software Blender erstellt und nun in Unity implementiert.
Bei der Klanggestaltung griffen wir auf Musik aus ... Bibliothek zurück, diese haben wir mithilfe von ... zugeschnitten und in unser Projekt integriert.

Musikquellen:
Destroy Sound: https://pixabay.com/de/sound-effects/break-robot-61522/
Spawn Sound: https://pixabay.com/de/sound-effects/little-robot-sound-84657/
Countdown Sound: https://pixabay.com/de/sound-effects/female-robotic-countdown-5-to-1-47653/
Shoot Sound: https://pixabay.com/de/sound-effects/sci-fi-gun-shot-x6-14447/
GameOver Sound: https://pixabay.com/de/sound-effects/gameover-86548/
Hintergundmusik: https://www.youtube.com/watch?v=iyvOaW6AVoY

4.2 Assets

- Umgebung/ Korridore: SciFi Modular Hallway aus Turbosquid (Shutterstock)
- Kisten: https://assetstore.unity.com/packages/3d/props/sci-fi-crate-70278
- Sci-fi Gui wie Buttons: https://assetstore.unity.com/packages/2d/gui/sci-fi-gui-skin-15606
- Roboterarm: https://assetstore.unity.com/packages/3d/environments/sci-fi/free-sci-fi-office-pack-195067
- Tv: https://assetstore.unity.com/packages/3d/environments/sci-fi/free-sci-fi-office-pack-195067
- Regale:https://assetstore.unity.com/packages/3d/environments/sci-fi/free-sci-fi-office-pack-195067
- Karton: https://assetstore.unity.com/packages/3d/environments/sci-fi/free-sci-fi-office-pack-195067
- Pc:https://assetstore.unity.com/packages/3d/environments/sci-fi/free-sci-fi-office-pack-195067
- Server:https://assetstore.unity.com/packages/3d/environments/sci-fi/free-sci-fi-office-pack-195067
- Pistole: https://assetstore.unity.com/packages/3d/props/guns/sci-fi-shortgun-114355
- Sternenhimmel: https://assetstore.unity.com/packages/3d/environments/sci-fi/real-stars-skybox-lite-116333

4.3 Scripte

MenuController:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class MenuController : MonoBehaviour
{
    public void StartBtn()
    {
        SceneManager.LoadScene("cyberBot");

        Debug.Log(" wurde gedr◆ckt");
```

```
    }

    public void BeendenBtn()
    {
        Application.Quit();
    }
}
```

Das Skript MenuController ist für die Funktionen im Menü zuständig. Die Funktion "StartBtn()" wird aufgerufen, sobald der Start-Button betätigt wird. Dadurch wird die Szene geladen, die das Level beinhaltet. Die Funktion "BeendenBtn()" beendet die Anwendung. Das Skript ist in der Szene "StartMenu" zu finden und wurde dem Objekt "MenuCanvas" zugewiesen.

StartGame:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class StartGame : MonoBehaviour
{
    public SpawnBots startGame;
    public TimerScript timer;
    public ScoreManager scoreManager;
    public Collider gameCollider;
    public float disableDuration = 125f; // 2 Minuten in Sekunden

    private void OnCollisionEnter(Collision collision)
    {
        // Überprüfe, ob der Collider mit dem gewünschten Tag
getroffen wurde
        if (collision.collider.CompareTag("bullet"))
        {
            Debug.Log("RUFT AUF");

            // Überprüfe, ob startGame und timer nicht null sind,
bevor die Methoden aufgerufen werden
            if (startGame != null && timer != null)
            {
                // Score zurück setzen
                scoreManager.ResetScore();
                timer.ResetTimer();
```

```
                Debug.Log("AKTUELLER SCORE: " +
scoreManager.score);
                // Timer restet


                startGame.StartSpawning();
                Invoke("StartTimerDelayed", 5f);

                // Collider für die definierte Zeit deaktivieren
                StartCoroutine(DisableColliderForDuration());
            }
        }
    }

    private void StartTimerDelayed()
    {
        timer.StartTimer();
    }

    private IEnumerator DisableColliderForDuration()
    {
        // Collider deaktivieren
        gameCollider.enabled = false;

        // Warte für die definierte Dauer
        yield return new WaitForSeconds(disableDuration);

        // Collider wieder aktivieren
        gameCollider.enabled = true;
    }
}
```

Das Skript "StartGame" wird ausgelöst, wenn auf den Start-Button in der Szene "cyberBot" geschossen wird. Es setzt den Timer und den Score zurück und ruft die Funktion zum Spawnen der Bots auf. Außerdem wird der Timer gestartet. Nachdem das Spiel gestartet wird, wird der Startknopf für die Dauer des Spiels ausgeschaltet. Das Skript findet man unter "Spielstart/Menu->TV 32 inch 2->Bildschirm Canvas->Start".

EndGame:

using System.Collections;

```
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class EndGame : MonoBehaviour
{
    private void OnCollisionEnter(Collision collision)
    {
        // Überprüfe, ob der Collider mit dem gewünschten Tag
getroffen wurde
        if (collision.collider.CompareTag("bullet"))
        {
            // gehe zurück in die Hauptmenüszene
            SceneManager.LoadScene("StartMenu");
        }
    }
}
```

Das Skript "EndGame" wird ausgelöst, wenn in der Szene "cyberBot" auf den Button "Menu" geschossen wird. Dadurch wird die Szene gewechselt und der Spieler befindet sich im Menü. Das Skript findet man unter "Spielstart/Menu->TV 32 inch 2->Bildschirm Canvas->Button1 (1)".

TimerScript:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using TMPro;

public class TimerScript : MonoBehaviour
{
    public float totalTime = 120f; // Gesamte Zeit in Sekunden
    private float currentTime; // Aktuelle Zeit in Sekunden
    [SerializeField] TextMeshProUGUI timerText;

    private bool timerStarted = false;

    private void Start()
```

```
    {
        currentTime = totalTime;
    }

    private void Update()
    {
        // Aktualisiere den Timer und zeige ihn im Textfeld an,
nur wenn der Timer gestartet wurde
        if (timerStarted)
        {
            currentTime -= Time.deltaTime;

            // Überprüfe, ob die Zeit abgelaufen ist
            if (currentTime <= 0f)
            {
                currentTime = 0f;

            }

            // Konvertiere die verbleibende Zeit in Minuten und
Sekunden
            int minutes = Mathf.FloorToInt(currentTime / 60f);
            int seconds = Mathf.FloorToInt(currentTime % 60f);

            // Aktualisiere den Text des Textfelds
            timerText.text = string.Format("{0:00}:{1:00}",
minutes, seconds);
        }
    }

    public void StartTimer()
    {
        // Starte den Timer
        timerStarted = true;
    }

    public void ResetTimer()
    {
        currentTime = totalTime;
        timerStarted = false;
    }
}
```

Im TimerScript wird zu Beginn die Zeit auf 2 Minuten gesetzt, und in jedem neuen Frame wird der Timer aktualisiert, sodass der Spieler sehen kann, wie lange eine Runde noch dauert. Außerdem enthält das Skript die Funktionen StartTimer() und ResetTimer(), die im Skript "StartGame" aufgerufen werden. Das Skript findet man unter "Spielstart/Menu->TV 32 inch 2->Bildschirm Canvas->Button1 (1)->TimerObject".

ScoreManager:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ScoreManager : MonoBehaviour
{
    public static ScoreManager Instance;

    public int score = 0;

    private void Awake()
    {
        if (Instance == null)
            Instance = this;
    }

    public void AddScore(int points)
    {
        score += points;


        Debug.Log("Punktezahl" + score);
    }

    public void ResetScore()
    {
        score = 0;
    }
}
```

Das Skript "ScoreManager" ist für die Verwaltung der Punktzahl im Spiel verantwortlich. Mit der Funktion "AddScore()" werden Punkte zum bisherigen Punktestand hinzugefügt. Hier befindet sich auch die Funktion "ResetScore()", die im Skript "StartGame" aufgerufen wird. Der ScoreManager befindet sich im ScoreManager-Objekt.

ScoreDisplay:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;

public class ScoreDisplay : MonoBehaviour



{
    [SerializeField] TextMeshProUGUI scoreText;

    public ScoreManager scoreManager;
    // Start is called before the first frame update
    void Start()
    {
        scoreText = GetComponent<TextMeshProUGUI>();

    }

    // Update is called once per frame
    void Update()
    {
        scoreText.text = "Score: " + scoreManager.score;
    }
}
```

Das Skript "ScoreDisplay" wird verwendet, um den Punktestand auf dem Bildschirm anzuzeigen. Der Text wird in jedem Frame aktualisiert. Der Punktestand wird aus dem Skript "ScoreManager" gelesen. Das Skript befindet sich unter "Spielstart/Menu->TV 32 inch 2->Bildschirm Canvas->Button1 (1)->ScoreText".

SpawnBots:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class SpawnBots : MonoBehaviour
{
    public GameObject bot;

     // Zeitintervall zwischen den Spawns für feste bots (in
Sekunden)
    public float spawnInterval = 5.0f;

    public float startCountdown = 5.0f;


    // für feste Bots
     public Transform[] spawnPoints;

     public Transform[] spawnPointsMove;

     public Transform[] deletePoints;

    //Punkt wo doe Bots hinschauen
     public Transform lookAt;



     // Zeitversatz zum Entfernen des Objekts (in Sekunden)
     public float despawnDelay = 30f;

    //Soundeffekte
    public AudioSource countdownSource;
    public AudioSource spawnSound;
    public AudioSource gameOver;


    //Animator
    Animator botAnimator;

    // startet eine Runde
      public void StartSpawning()
    {
```

```
        // Damit das spiel nah 2min aufhört
        Invoke("DisableSpawn", 120f);

        // Spielt countdown
       countdownSource.Play();
        InvokeRepeating("SpawnBotsStand",
startCountdown,spawnInterval);


        InvokeRepeating("SpawnAndMoveBots", 60f ,10f);


    }

    private void SpawnBotsStand()
        {


        // Zufällig einen Spawnpunkt auswählen
        int randomIndex = Random.Range(0, spawnPoints.Length);
        Transform selectedSpawnPoint = spawnPoints[randomIndex];


        // Position des ausgewählten Spawnpunkts verwenden
        Vector3 spawnPosition = selectedSpawnPoint.position;

         GameObject instantiatedBot = Instantiate(bot,
spawnPosition, Quaternion.LookRotation(lookAt.position +
spawnPosition));
         spawnSound.Play();
         botAnimator = instantiatedBot.GetComponent<Animator>();


         if( instantiatedBot !=null){

            StartCoroutine(DeleteBotAfterTime(instantiatedBot,
despawnDelay));


        }
```

```
    }


    private void SpawnAndMoveBots()
{

    // Zufällig einen Spawnpunkt auswählen
    int randomIndex = Random.Range(0, spawnPointsMove.Length);
    Transform selectedSpawnPoint = spawnPointsMove[randomIndex];

    // Position des ausgewählten Spawnpunkts verwenden
    Vector3 spawnPosition = selectedSpawnPoint.position;

    // Bot spawnen
    GameObject instantiatedBot = Instantiate(bot, spawnPosition,
Quaternion.LookRotation(lookAt.position + spawnPosition));

    botAnimator = instantiatedBot.GetComponent<Animator>();


    // Bot zu einem bestimmten Punkt bewegen
    Vector3 targetPosition = deletePoints[randomIndex].position ;
    float movementSpeed = 5f; // Beispielgeschwindigkeit, mit der
sich der Bot bewegt

    botAnimator.SetTrigger("moving");
    StartCoroutine(MoveBot(instantiatedBot, targetPosition,
movementSpeed));


}

private IEnumerator MoveBot(GameObject botToMove, Vector3
targetPosition, float speed)
{
    while (botToMove.transform.position != targetPosition &&
botToMove !=null)
    {
            if (botToMove != null)
            {
                // Bot zum Ziel bewegen
```

```
                botToMove.transform.position =
Vector3.MoveTowards(botToMove.transform.position, targetPosition,
speed * Time.deltaTime);
            }
        yield return null;
    }

    // Bot zerstören
    if(botToMove != null){
       Destroy(botToMove);
    }
}

 private void DisableSpawn()
    {

        CancelInvoke("SpawnBotsStand");
        CancelInvoke("SpawnAndMoveBots");
        // gameOver sound
        gameOver.Play();
    }


    private IEnumerator DeleteBotAfterTime(GameObject bot,  float
delay){

        yield return new WaitForSeconds(20f);
        if(bot != null){
        Destroy(bot);
        }
    }

}
```

Im Skript "SpawnBot" befinden sich die Funktionen, um zwei verschiedene Arten von Bots zu spawnen. Die Funktion "StartSpawning()" wird vom Skript "StartGame" aufgerufen. Diese Funktion ruft die Spawndurchführungsfunktionen in einem bestimmten Rhythmus auf. Zusätzlich wird hier der Countdown-Sound abgespielt und die Funktion zum Beenden des Spawnens aufgerufen. Sowohl für die fahrenden als auch für die stehenden Bots werden die Spawnpunkte zufällig gewählt. Beide Bots spawnen mit einem Geräusch. Die stehenden Bots verschwinden nach 30 Sekunden, wenn sie nicht

getroffen werden. Fahrende Bots verschwinden, wenn sie einen bestimmten Punkt erreichen, falls sie nicht vorher getroffen wurden. Die Bots verfügen über eine zusätzliche Fahr-Animation. Die Bewegung wird mit der Funktion "MoveBot()" durchgeführt. Die Funktion "DisableSpawn()" sorgt dafür, dass nach 2 Minuten keine weiteren Bots spawnen und der Gameover-Sound abgespielt wird. Das Skript befindet sich unter "Spielstart/Menu->TV 32 inch 2->Bildschirm Canvas".

HeadHitbox:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class HeadHitbox : MonoBehaviour
{
    public AudioSource fallSound;
    public int scoreForHit = 5;

    private void OnCollisionEnter(Collision collision)
    {
        // Überprüfe, ob die Kollision mit einem Projektil
stattfindet
        if (collision.gameObject.CompareTag("bullet"))
        {
            fallSound.Play();
            // Vergebe die Punktzahl für das Treffen der
Head-Hitbox
            ScoreManager.Instance.AddScore(scoreForHit);

            // Zerstöre das Projektil
            Destroy(collision.gameObject);
        }
    }
}
```

Das Skript "HeadHitbox" ist dafür verantwortlich, Kopftreffer zu registrieren. Wenn ein Treffer erfolgt, wird der Treffersound abgespielt und 5 Punkte zum Score hinzugefügt. Die Kugel wird beim Aufprall zerstört. Das Skript befindet sich im Modell "Richtig", das in den Assets unter "Assets->Model" zu finden ist. Innerhalb des Modells befindet es sich im Objekt "head-hitbox"..

FootHitbox:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class FootHitbox : MonoBehaviour
{
    public AudioSource fallSound;
    public int scoreForHit = 1;

    private void OnCollisionEnter(Collision collision)
    {
        // Überprüfe, ob die Kollision mit einem Projektil
stattfindet
        if (collision.gameObject.CompareTag("bullet"))
        {
            fallSound.Play();
            // Vergebe die Punktzahl für das Treffen der
Foot-Hitbox
            ScoreManager.Instance.AddScore(scoreForHit);

            // Zerstöre das Projektil
            Destroy(collision.gameObject);
        }
    }
}
```

Das Skript "FootHitbox" ist dafür verantwortlich, Fußtreffer zu registrieren und verhält sich ähnlich wie das "HeadHitbox"-Skript. In diesem Fall wird jedoch nur 1 Punkt zum Score hinzugefügt. Das Skript befindet sich im Modell "Richtig", das in den Assets unter "Assets->Model" zu finden ist. Innerhalb des Modells befindet es sich im Objekt "foot-hitbox".

BodyHitbox:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class BodyHitbox : MonoBehaviour
{
    public AudioSource fallSound;
    public int scoreForHit = 3;
```

```
    private void OnCollisionEnter(Collision collision)
    {
        // Überprüfe, ob die Kollision mit einem Projektil
stattfindet
        if (collision.gameObject.CompareTag("bullet"))
        {
            fallSound.Play();
            // Vergebe die Punktzahl für das Treffen der
Body-Hitbox
            ScoreManager.Instance.AddScore(scoreForHit);

            // Zerstöre das Projektil
            Destroy(collision.gameObject);
        }
    }
}
```

Das Skript "BodyHitbox" ist dafür verantwortlich, Körperhits zu registrieren und verhält sich ähnlich wie das "HeadHitbox"-Skript. In diesem Fall werden jedoch 3 Punkte zum Score hinzugefügt. Das Skript befindet sich im Modell "Richtig", das in den Assets unter "Assets->Model" zu finden ist. Innerhalb des Modells befindet es sich im Objekt "body-hitbox".

BulletCollisionHandler:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class BulletCollisionHandler : MonoBehaviour
{
    private Animator botAnimator;

    private void OnCollisionEnter(Collision collision)
    {
            // Abfrage ob eine Collision mit einer Hitbox
passiert ist
            if (collision.gameObject.CompareTag("headshot") ||
                collision.gameObject.CompareTag("bodyshot") ||
                collision.gameObject.CompareTag("footshot"))
            {
```

```
                // Zugriff auf den Roboter über das Elternobjekt
(Root-Objekt)
                GameObject robot =
collision.transform.root.gameObject;
                DestroyWithAnimation(robot);

            }

            // Zerstöre das Bullet-Objekt
            Destroy(gameObject);

    }

    private void DestroyWithAnimation(GameObject robot)
    {
        // führt die Animation aus
        botAnimator = robot.GetComponent<Animator>();
        botAnimator.SetTrigger("fall");

        // hole alle Hitboxen und deaktiviere jede
        Collider[] colliders =
robot.GetComponentsInChildren<Collider>();
        foreach (Collider collider in colliders)
        {
            collider.enabled = false;
        }

        // zerstöre den Roboter
        Destroy(robot, 1.2f);
    }
}
```

Im Skript "BulletCollisionHandler" wird bei einer Kollision mit dem Bot dieser zerstört. Zuerst wird eine Animation abgespielt. Zusätzlich wird der Collider vorübergehend deaktiviert, um zu verhindern, dass ein Bot mehrmals getroffen wird. Das Skript befindet sich im Modell "Bullet", das in den Assets unter "Assets->Model" zu finden ist.

FireBulletOnActivate:

```
using System.Collections;
using System.Collections.Generic;
```

```
using UnityEngine;
using UnityEngine.XR.Interaction.Toolkit;

public class FireBulletOnActivate : MonoBehaviour
{

    public GameObject bullet;
    public Transform spawnPoint;
    public float fireSpeed = 20;

      //Soundeffekt countdown
    public AudioSource shootSound;

    // Start is called before the first frame update
    void Start()
    {
        XRGrabInteractable grabbable =
GetComponent<XRGrabInteractable>();
        grabbable.activated.AddListener(FireBullet);
    }

    // Update is called once per frame
    void Update()
    {

    }

    public void FireBullet(ActivateEventArgs arg)
    {
        shootSound.Play();
        GameObject spawnBullet = Instantiate(bullet);
        spawnBullet.transform.position = spawnPoint.position;
        spawnBullet.GetComponent<Rigidbody>().velocity =
spawnPoint.forward * fireSpeed;
        Destroy(spawnBullet, 5);
    }

}
```

Im Skript "FireBulletOnActivate" wird ermöglicht, die Pistole aufzunehmen und damit zu schießen. Bei jedem Schuss wird ein Sound ausgelöst. Zudem werden die Kugeln nach 5 Sekunden zerstört, falls sie bis dahin kein Ziel treffen. Das Skript befindet sich unter "LowPoly Scifi_gun".

5. Abgabe

Alle Dateien sind unter dem D-Laufwerk im Ordner “AnimationUndVRGruppe2” gespeichert.