FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

**Predmet:** Računalna Animacija
**Ak. god.** 2023/2024
**Nastavnik:** prof. dr. sc. Željka Mihajlović

**Student:** Jelena Dorić
**Matični broj:** 036503928

Zagreb, 15.01.2024.

# INTERAKTIVNA TERENSKA GRAFIKA

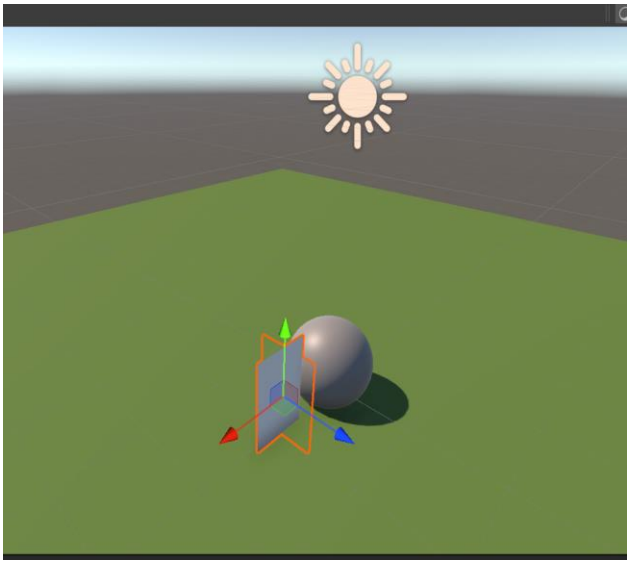
# Tehnička dokumentacija

# 1. Uvod

Ovaj dokument predstavlja tehničku dokumentaciju za projekt „Interaktivna Terenska Grafika“. Projekt se fokusira na implementaciju interaktivnog travnatog terena unutar Unity okruženja koristeći alat Unity Shader Graph.
U računalnoj animaciji, shader je  programski kod koji definira vizualne karakteristike površina i svjetlosne efekte u 3D modeliranju. On je odgovoran za računanje kako svjetlost integrira s materijalima, utječući na boju, teksturu, sjaj i druge vizualne aspekte. Shader-i su ključni za stvaranje dojma dubine, realnosti i dinamike u virutalnim scenama.
Unity Shader Graph je napredni alat unutar Unity Engine-a koji pruža mogućnost razvoja prilagođenim shader-a kroz vizualno programiranje. Korištenjem sistema čvorova, korisnici mogu kreirati kompleksne interakcije i efekte.
U ovom projektu, Unity Shader Graph se koristi za kreiranje realistične simulacije trave, koja se prilagođava i reagira na interakcije loptice i utjecaj vjetra. Detaljan opis implementacije i tehničkih aspekata projekta, uključujući poces kreiranja interaktivnog travnatog terena i ponašanja loptice, obrađen je u sljedećem poglavlju dokumentacije.

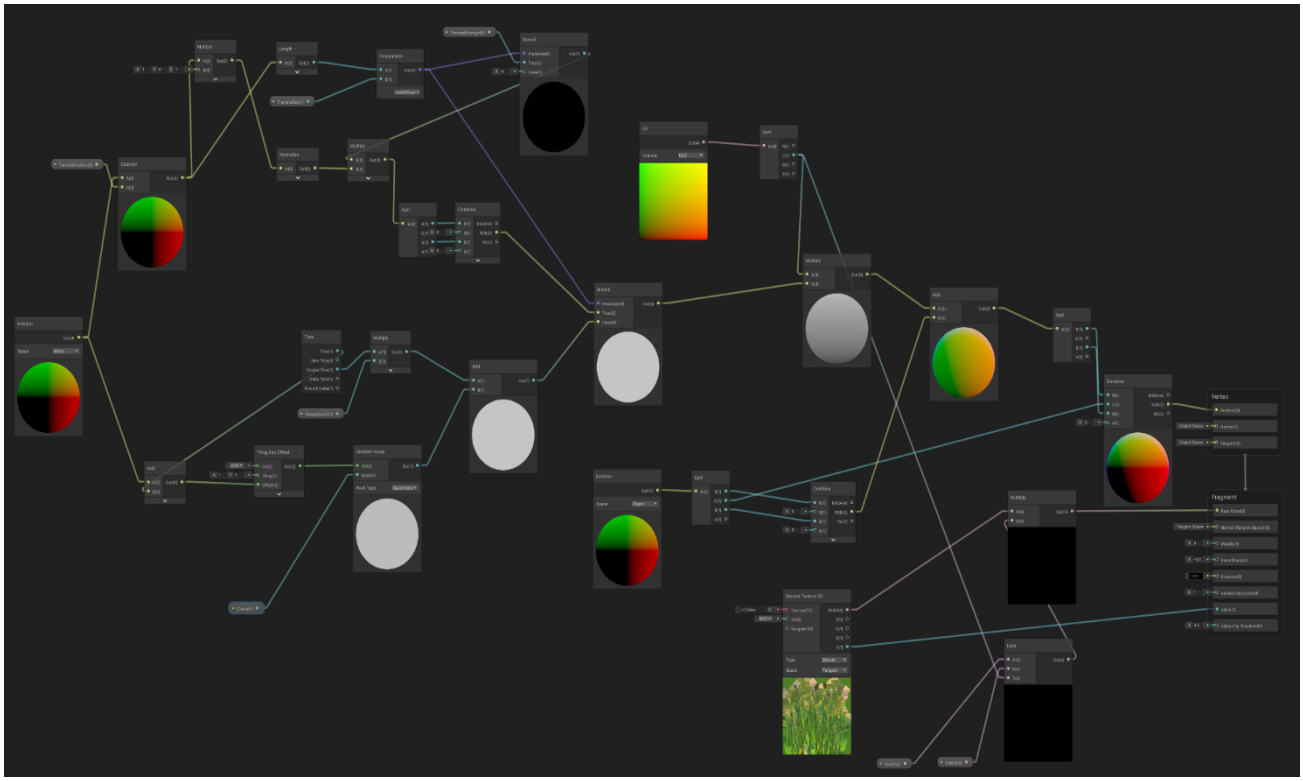
# *2. Opis projekta*

Slika 1 prikazuje početnu scenu u Unity okruženju koja sadrži osnovne komponente koje su temelj za izradu interaktivne terenske grafike. Scena uključuje objekt koji se sastoji od dvije ravnine, sferni objekt koji simbolizira lopticu te ravninu koja predstavlja teren za travu.
Tijekom razvoja projekta, prvo je kreiran materijal u Unity okruženju te je postavljen na objekt koji se sastoji od dvije ravnine. Upravo taj objekt će predstavljati jedan 3D modela trave.



**Slika 1: Početna scena projekta u Unity radnom okruženju**

Shader Graph objekt implementira Vertex Shader, koji je zadužen za manipulaciju pojedinih vrhova 3D modela. Ovaj shader postavljen je kao shader za materijal koji se nalazi na objektu trave.
Daljni razvoj uključuje kreiranje shader-a unutar Shader Graph-a koji prikazuje Slika 2.

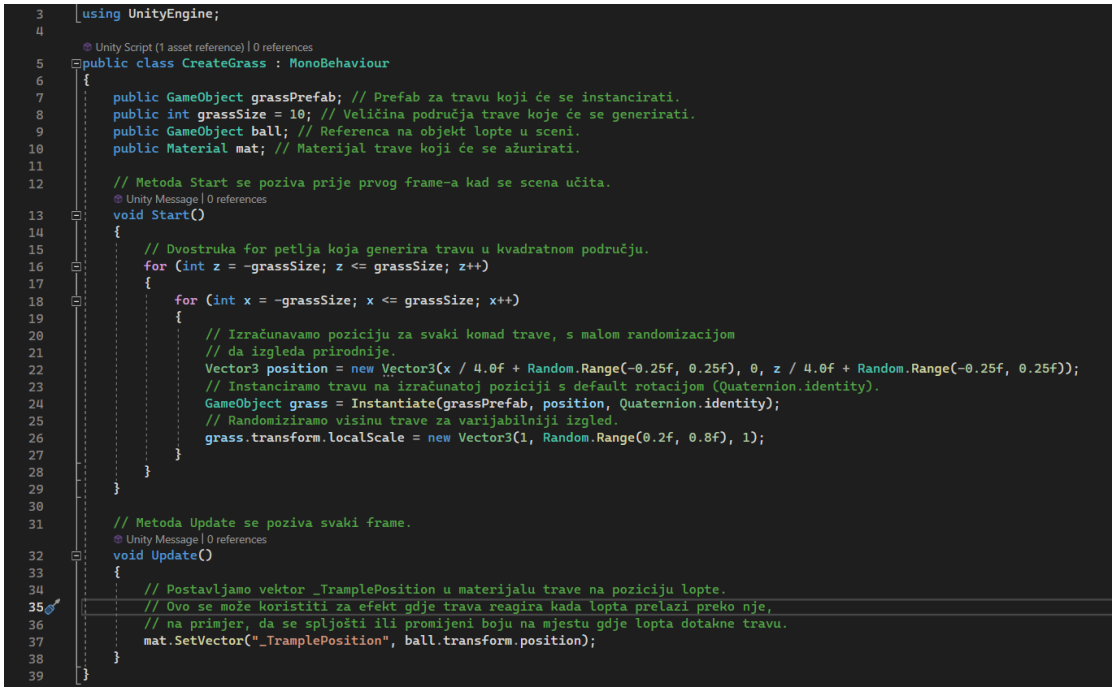


**Slika 2: Unity Shader Graph za materijal trave**

Kako bi se postigao efekt njihanja trave napravljen je float parametar SwayAmount, čime je definirana amplituda njihanja. Vrijednost čvora Time, odnosno Cosine Time, koja oscilira između -1 i 1, pomnožena je s vrijednosti parametra SwayAmount kako bi se postigla uniformna oscilacija trave, simulirajući kretanje uzrokovano vjetrom. Kako bi se ostvarilo prirodnije, neuniformno njihanje, uzete su koordinate vrha svake trave u odnosu na globalni referentni sustav scene. Izlazna vrijednost čvora Time, koja predstavlja rastuću vrijednost, zbraja se s globalnim koordinatama vrha. Time se postiže to da se njihanje ne događa sinkronizirano za svaku travku. Ova izlazna vrijednost je zatim postavljena kao Offset vrijednost u čvoru Tiling and Offset, koji manipulira UV koordinatama za teksturiranje 3D modela. Ovaj postupak uzrokuje da se tekstura pomakne preko površine objekta s vremenom, dajući dojam da se trava prirodno kreće pod utjecajem vjetra. Dodatno, u čvoru Gradient Noise, animacija šuma stvara se prema promjenama unesenim u offset ulaz, što rezultira varijabilnim, pirodnim uzorcima kretanja unutar trave.
Kako bi se postigao efekt interakcije između loptice i travnate površine, što uključuje savijanje i disperziju(gnječenje) trave na mjestima gdje loptica dodiruje teren, izveden je sljedeći postupak:
Prvi korak uključuje izračun relativne pozicije trave u odnosu na lopticu. To se postiže oduzimanjem vektora pozicije loptice od globalnih koordinata vrha svae travke. Ovime se dobiva vektor koji ukazuje na udaljenost i smjer od loptice do svakog pojedinog vrha materijala trave. Za određivanje područja koje će biti pod utjecajem loptice, koristi se parametar „TrampleSize“, koji predstavlja radijus unutar kojeg će trava biti savijena ili disperzirana od loptice. Pomoću čvora „Length“ računa se udaljenost između loptice i vrha materijala. Nakon toga se pomoću čvora za usporedbu(Comparison node), specifično

funkcije „manje od“, određuje nalazi li se svaki vrh unutar definiranog radija disperzije. Ako se vrh materijala nalazi unutar područja utjecaja, koristi se parametar „TrampleStrenght“ za određivanje inteziteta s kojim će se trava saviti. Normalizacijom vektora dobivenog u prvom koraku, određuje se smjer u kojem će se vrhovi trave saviti, simulirajući efekt pritiska loptice na travu. Ovaj vektor se zatim množi s vertikalnom(„V“) koordinatom UV mape kako bi se odredila količina savijanja duž vertikalne osi trave. Rezultat ovih izračuna se zatim primjenjuje na horizontalne x i z koordinate vrha trave unutar lokalnog koordinatnog sustava, što dovodi do promjene horizontalnog položaja vrhova trave, bez utjecaja na njihovu vertikalnu poziciju.

Slika 3 prikazuje segment C# koda iz Unity projekta, koji implementira funkcionalnost za generiranje interaktivne travnate površine. Klasa „CreateGrass“, derivirana iz „MonoBehaviour“ klase, koristi se za inicijalizaciju i dinamičko ažuriranje trave u sceni. U metodi „Start“, dvostruka for petlja instancira prefab travke na izračunatim pozicijama unutar definiranog kvadratnog područja, dodajući slučajnu varijaciju u poziciji i veličini za realističan izgled. Metoda „Update“ kontinuirano ažurira materijal trave, postavljajući vektor pozicije „_TramplePosition“ na poziciju objekta lopte kako bi se omogućila interaktivnost trave s kretanjem lopte.



```
using UnityEngine;

Unity Script (1 asset reference) | 0 references
public class CreateGrass : MonoBehaviour
{
    public GameObject grassPrefab; // Prefab za travu koji će se instancirati.
    public int grassSize = 10; // Veličina područja trave koje će se generirati.
    public GameObject ball; // Referenca na objekt lopte u sceni.
    public Material mat; // Materijal trave koji će se ažurirati.

    // Metoda Start se poziva prije prvog frame-a kad se scena učita.
    Unity Message | 0 references
    void Start()
    {
        // Dvostruka for petlja koja generira travu u kvadratnom području.
        for (int z = -grassSize; z <= grassSize; z++)
        {
            for (int x = -grassSize; x <= grassSize; x++)
            {
                // Izračunavamo poziciju za svaki komad trave, s malom randomizacijom
                // da izgleda prirodnije.
                Vector3 position = new Vector3(x / 4.0f + Random.Range(-0.25f, 0.25f), 0, z / 4.0f + Random.Range(-0.25f, 0.25f));
                // Instanciramo travu na izračunatoj poziciji s default rotacijom (Quaternion.identity).
                GameObject grass = Instantiate(grassPrefab, position, Quaternion.identity);
                // Randomiziramo visinu trave za varijabilniji izgled.
                grass.transform.localScale = new Vector3(1, Random.Range(0.2f, 0.8f), 1);
            }
        }
    }

    // Metoda Update se poziva svaki frame.
    Unity Message | 0 references
    void Update()
    {
        // Postavljamo vektor _TramplePosition u materijalu trave na poziciju lopte.
        // Ovo se može koristiti za efekt gdje trava reagira kada lopta prelazi preko nje,
        // na primjer, da se spljošti ili promijeni boju na mjestu gdje lopta dotakne travu.
        mat.SetVector("_TramplePosition", ball.transform.position);
    }
}
```

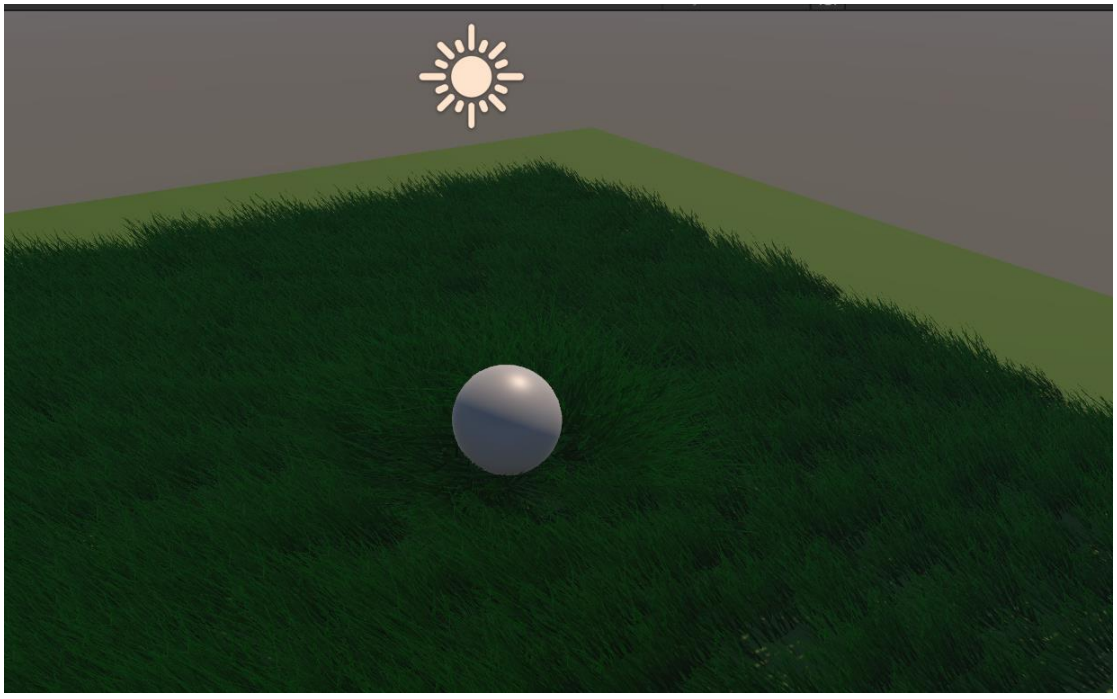
**Slika 3: Kod u datoteci naziva „CreateGrass.cpp“ za stvaranje travnate površine**

Na slici 4 prikazan je kod napisan u jeziku C# koji definira klasu „BallMover“. Ova klasa upravlja kretanjem lopte u Unity-jevoj 3D sceni. Klasa postavlja osnovne parametre za brzinu kretanja i ciljnu poziciju te prati stanje kretanja lopte. U metodi „Start“, postavlja se referenca na glavnu kameru i inicijalna pozicija lopte. „Update“ metoda, obrađuje korisnički unos i upravlja kretanjem lopte prema poziciji odabranoj mišem. Raycast provjerava presjek kursora i collider-a, određujući ciljnu poziciju. Lopta se kontinuirano kreće dok ne postigne blizinu cilja, gdje se kretanje završava.

```
using UnityEngine;

Unity Script (1 asset reference) | 0 references
public class BallMover : MonoBehaviour
{
    public float moveSpeed = 5f; // Brzina kretanja lopte

    private Vector3 targetPosition; // Ciljna pozicija do koje se kreće lopta
    private bool isMoving = false; // Provjera je li lopta u pokretu

    private Camera mainCamera; // Referenca na glavnu kameru

    Unity Message | 0 references
    void Start()
    {
        mainCamera = Camera.main; // Postavljamo referencu na glavnu kameru
        targetPosition = transform.position; // Početna ciljna pozicija je trenutna pozicija lopte
    }

    // Update se zove jednom po frejmu
    Unity Message | 0 references
    void Update()
    {
        // Provjerava se je li korisnik kliknuo lijevi gumb miša
        if (Input.GetMouseButtonDown(0))
        {
            //Pretvara poziciju klika na ekranu u zraku koja se projicira iz pozicije kamere u smjeru gdje korisnik pokazuje mišem
            Ray ray = mainCamera.ScreenPointToRay(Input.mousePosition);
            RaycastHit hit;

            // Izvršava se raycast; provjerava siječe li se zraka s nekim objektom u sceni
            if (Physics.Raycast(ray, out hit))
            {
                // Sadržava 3D koordinate u sceni gdje se zraka prvi put siječe s kolajderom
                targetPosition = hit.point;
                isMoving = true; // Postavlja se da je lopta u pokretu
            }
        }

        // Ako je lopta u pokretu i nije preblizu ciljne pozicije, pomiče se prema ciljnoj poziciji
        if (isMoving && (targetPosition - transform.position).sqrMagnitude > 0.01f) //sqrMagnitude - kvadrat udaljenosti između dvije točkes
        {
            // Pomiče loptu prema ciljnoj poziciji
            transform.position = Vector3.MoveTowards(transform.position, new Vector3(targetPosition.x, transform.position.y, targetPosition.z), moveSpeed * Time.deltaTime);
        }
        else
        {
            isMoving = false; // Ako je lopta blizu cilja, postavlja se da više nije u pokretu
        }
    }
}
```

**Slika 4: Kod koji se nalazi u datoteci „BallMover.cpp“ pomoću kojeg se loptica kreće prema poziciji na koju je korisnik kliknuo mišem**

Na slici 5 prikazan je konačni rezultat projekta, gdje se vizualizira završena interakcija između lopte i travnate površine.



**Slika 5: Pokrenuta scena u Unity okruženju nakon završetka projekta**

# 3. Instalacija i pokretanje projekta

Za instalaciju Unity radnog okruženja, preuzmite i pokrenite Unity Hub, odaberite željenu verziju Unity Editor-a i pratite upute za instalaciju. Nakon instalacije, otvorite Unity Hub, idite na karticu „Projects“, kliknite „Add“ i navigirajte do direktorija gdje se nalazi projekt kako biste ga učitali. Kada je projekt učitan, otvorite ga u Unity Editor-u. Jednom kada je projekt otvoren, možete ga pokreuti pritiskom na „Play“ gumb unutar Unity Editor-a.