

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 862

IZRADA VIZUALNIH UČINAKA

Matija Kunc

Zagreb, lipanj 2023.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 862

IZRADA VIZUALNIH UČINAKA

Matija Kunc

Zagreb, lipanj 2023.

ZAVRŠNI ZADATAK br. 862

Pristupnik: **Matija Kunc (0036534412)**
Studij: Elektrotehnika i informacijska tehnologija i Računarstvo
Modul: Računarstvo
Mentorica: prof. dr. sc. Željka Mihajlović

Zadatak: **Izrada vizualnih učinaka**

Opis zadatka:

Proučiti načine izrade i područja korištenja vizualnih učinaka (VFX). Proučiti i objasniti mogućnosti izrade vizualnih učinaka u grafičkom programskom pogonu Unreal Engine. Posebice obratiti pažnju na složenije primjere vizualnih učinaka. Razraditi postupke izrade vizualnih učinaka u grafičkom programskom pogonu Unreal Engine. Diskutirati utjecaj različitih parametara. Načiniti ocjenu rezultata i implementiranih algoritama. Izraditi odgovarajući programski proizvod. Koristiti grafičku programski pogon Unreal Engine. Rezultate rada načiniti dostupne putem Interneta. Radu priložiti algoritme, izvorne kodove i rezultate uz potrebna objašnjenja i dokumentaciju. Citirati korištenu literaturu i navesti dobivenu pomoć.

Rok za predaju rada: 9. lipnja 2023.

Sadržaj

1. Uvod.....	1
2. Korištene tehnologije i alati.....	2
2.1. Unreal Engine 5.....	2
2.2. Sustavi čestica	2
2.3. Niagara System	3
3. Izrada vizualnih učinaka.....	5
3.1. Izrada vatre.....	6
3.1.1. Vatra.....	6
3.1.2. Dim, iskre i toplinsko izobličenje	11
3.2. Tornado	14
3.2.1. Stup tornada.....	14
3.2.2. Dodatni detalji	19
3.3. Vodopad.....	21
3.3.1. Vodopad.....	21
3.3.2. Dodatni detalji	23
4. Moguća poboljšanja	25
5. Zaključak	26
Literatura	27
Sažetak	28
Summary	29
Privitak	30

1. Uvod

Vizualni učinci danas se često pojavljuju u filmskoj industriji, računalnim igrama i mnogim drugim medijima. Ključan su dio u prikazivanju svega što je tvorac neke računalne igre ili filma zamislio. Bili to svemirski brodovi, zmajevi, eksplozije, magija ili bilo što drugo što nam padne napamet, vizualnim učincima to je moguće prikazati (slika 1.1). Danas postoji mnoštvo tehnika izrade vizualnih učinaka: animacija, 3D modeliranje, rotoskopija, snimanje pokreta itd. Razvojem tehnologije vizualni učinci danas postaju sve bogatiji i sve realističniji pa postaje moguće izraditi takve vizualne učinke da je ljudskom oku teško razaznati stvarne od umjetnih efekata.



Slika 1.1: Primjer vizualnog učinka iz filma *Doctor Strange*

Cilj ovog rada je proučiti načine izrade vizualnih učinaka te izraditi vizualne učinke koristeći programski pogon Unreal Engine. Razradit ćemo i postupke u izradi primjera vizualnih učinaka.

2. Korištene tehnologije i alati

U ovom radu koristit ćemo se Unreal Engine 5 programskim pogonom. Unutar njega vizualne učinke izrađivat ćemo pomoću Niagara sustava. Niagara sustavi su Unreal Engine-ovi sustavi čestica kojima se često izrađuju vizualni učinci za videoigre.

2.1. Unreal Engine 5

Unreal Engine 5 je 3D grafički programski pogon za izradu videoigara. Često se koristi i u filmskoj industriji za izradu virtualnih setova. Sadrži alate za izradu videoigara od nule. Napravljen je da bude intuitivan za korištenje, omogućava veliku prenosivost projekata na različite platforme i besplatan je za korištenje što ga danas čini jednim od najpopularnijih *game engine*-a. Unreal Engine-ovi Niagara sustavi su glavni alat za izradu vizualnih učinaka.

2.2. Sustavi čestica

Sustavi čestica jedna su od tehnika izrade vizualnih efekata koja koristi velik broj sitnih objekata da prikaže učinke za koje nam poligoni (koji se inače koriste u 3D modeliranju) ne odgovaraju. Manipuliranjem čestica unutar sustava poput upravljanja silama koje na njih djeluju, upravljanja svojstvima kao što su njihova boja, oblik, veličina, itd. možemo dobiti vrlo zanimljive i vjerne prikaze nekih fenomena u prirodi. U ovom radu izrađeni su efekti vatre, vodopada i tornada. Primjer jednog sustava čestica vidimo na slici 2.1.

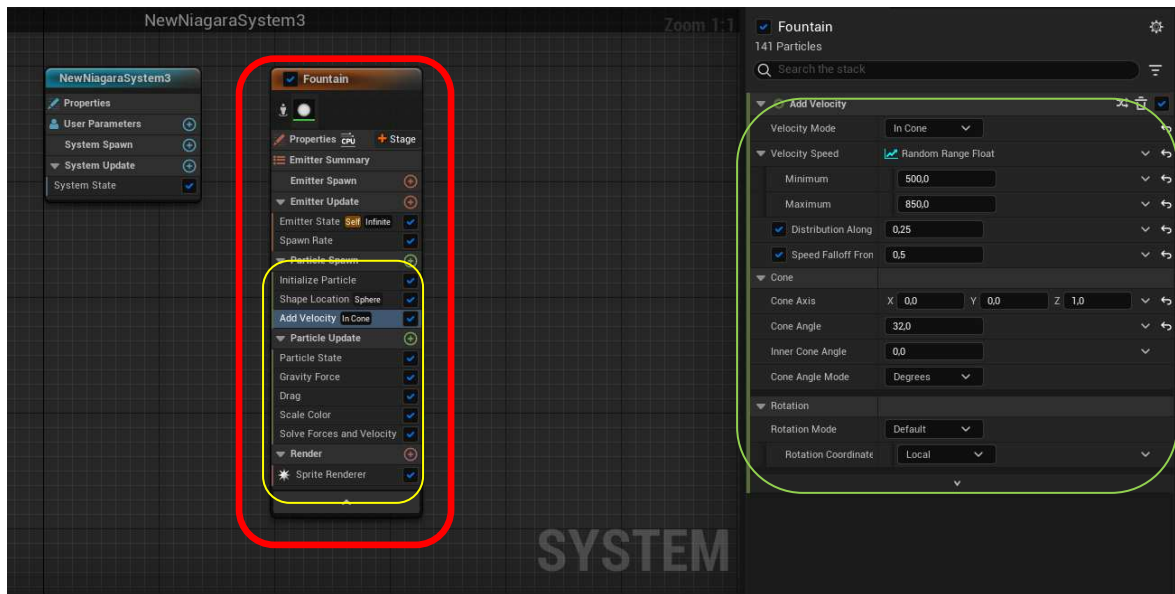


Slika 2.1: Sustav čestica

2.3. Niagara System

Niagara sustavi su Unreal Engine-ova realizacija sustava čestica. Omogućuju mnoštvo načina za manipulaciju česticama od velikih promjena do sitnih detalja. Niagara Sustavi sastoje se od 4 glavne komponente:

- Sustava
- Odašiljača (*Emitter*)
- Modula (*Modules*)
- Parametara



Slika 2.2: Uređivač Niagara sustava

Sustavi (cijela slika 2.2) su spremnici za više odašiljača (na slici 2.2 crveno) koji zajedno daju vizualni učinak. Odašiljači su izvori koji stvaraju šestice i njima upravljaju pomoću modula (slika 2.2 žuto). Moduli česticama mogu mijenjati svojstva i manipulirati njihovim kretanjem. Svaki modul se sastoji od parametara (slika 2.1 zeleno) kojima je moguće upravljati koliko i na koji način neki modul utječe na čestice. Gotovo svaku vrijednost parametara moguće je vrlo precizno zadati. Unreal Engine omogućuje da se svaka vrijednost zada pomoću raznih vrsta varijabli. To mogu biti obični *float*, *int* i slično, ali mogu biti i cijele funkcije kao sinusi i kosinusi, moguće je zadati i krivulje koje će opisivati kako se neka varijable mijenja, i moguće je povezati vrijednosti s unosima korisnika.

3. Izrada vizualnih učinaka

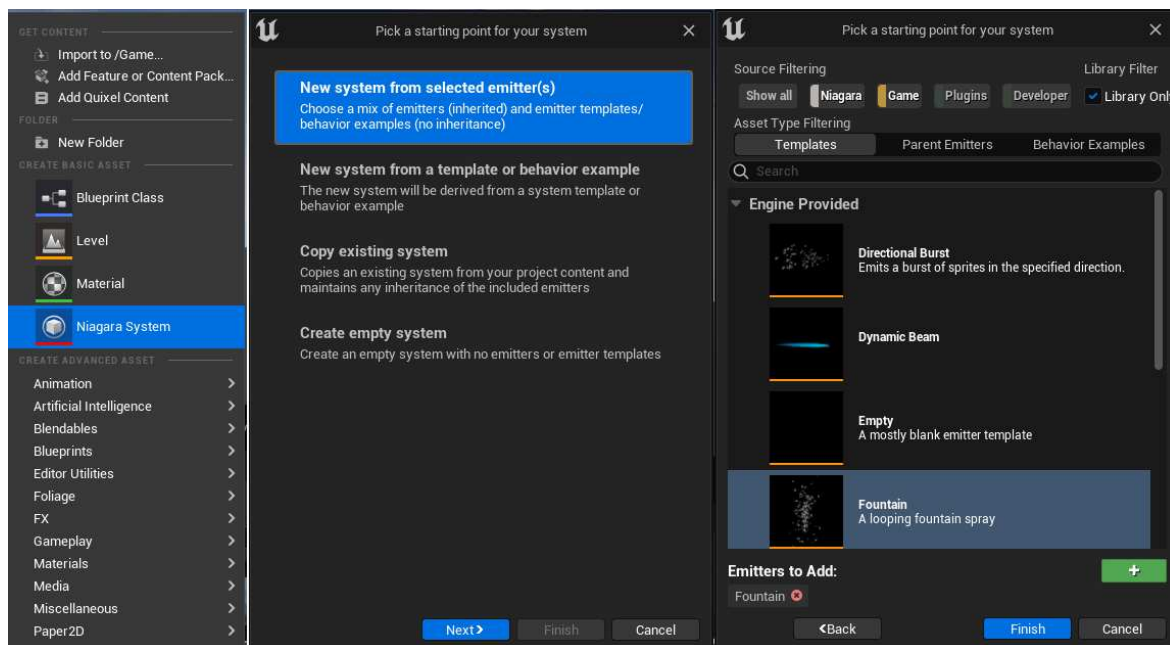
U ovom poglavlju bit će opisani postupci korišteni u izradi vizualnih učinaka napravljenih u Unreal Engine-u za potrebe ovog rada. No prije početka treba naglasiti da za izradu vizualnih učinaka nije korišteno ništa osim onoga što Unreal Engine nudi „po defaultu“ kada se pokrene novi projekt. Postoji mnoštvo materijala koji se mogu nabaviti s Unreal Engine Marketplace-a koje je netko već napravio i napravio dostupnima preko interneta za korištenje, ali oni neće biti korišteni kako bi se najbolje mogli upoznati s postupkom izrade vizualnih učinaka. Za svaki primjer istaknut će se aspekti koji ga čine drugačijim od drugih primjera. Također, vrijednosti parametara nemaju „točnu“ vrijednost već je potrebno eksperimentirati s različitim vrijednostima dok ne dobijemo izgled kakav želimo. Vrijednosti parametara bit će dane u opisu izrade, ali će služiti više kao okvirna vrijednost oko koje vizualni učinak izgleda bolje.

3.1. Izrada vatre

Prvi vizualni učinak koji je napravljen u ovom radu bio je vizualni učinak vatre. Vizualni učinak sastoji se od 4 dijela: same vatre, dima, sitnih letećih iskri i toplinskog izobličenja.

3.1.1. Vatra

Za početak izrade vatre napravljen je novi Niagara System. U *Content Browser*-u desnim klikom otvara se izbornik. Izabran je Niagara System. U novootvorenom izborniku odabrana je opcija *New system from selected emitter(s)*. Unreal Engine nam nudi nekoliko opcija ponašanja odašiljača koji će činiti naš sustav čestica. U našem slučaju vatre, najpogodniji je *Fountain* odašiljač. *Fountain* odašiljač izbacuje čestice prema gore koje zatim zbog utjecaja sile teže padaju pri čemu dobiva izgleda fontane. Sustav će izgledati kao na slici 2.1., a koraci stvaranja su prikazani slikom 3.1.



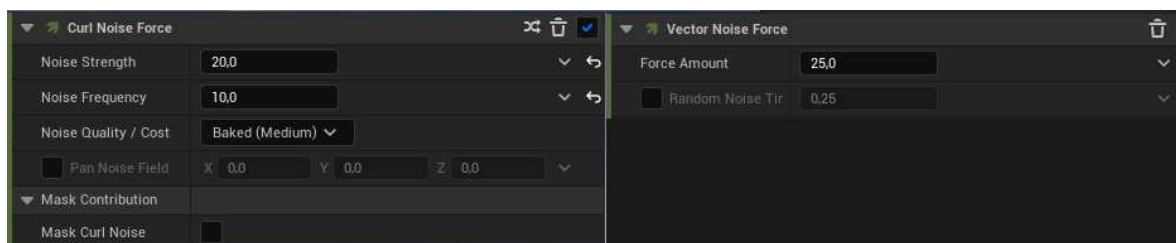
Slika 3.1: Stvaranje novog Niagara sustava

U ovom trenutku bilo je potrebno modelirati čestice da izgledaju kao vatra. Prvo je bilo potrebno ugasiti modul *Gravity Force* jer ne želimo da na naše čestice vatre utječe gravitacija. *Gravity Force* modul simulira utjecaj gravitacijske sile, kojoj je moguće mijenjati snagu i smjer mijenjajući vrijednosti x_y_z komponente njegovih parametara. Utjecaji ovakvih modula (npr. *Gravity Force* i *Drag*), koji simuliraju sile na čestice, zbrajaju se i računaju u modulu *Solve Forces and Velocity*. Nadalje, bilo je potrebno i ukloniti *Add Velocity* modul jer nam neće biti potrebno da čestice previše skaču (slika 3.2).



Slika 3.2: Uklonjeni *Gravity Force* i *Add Velocity*

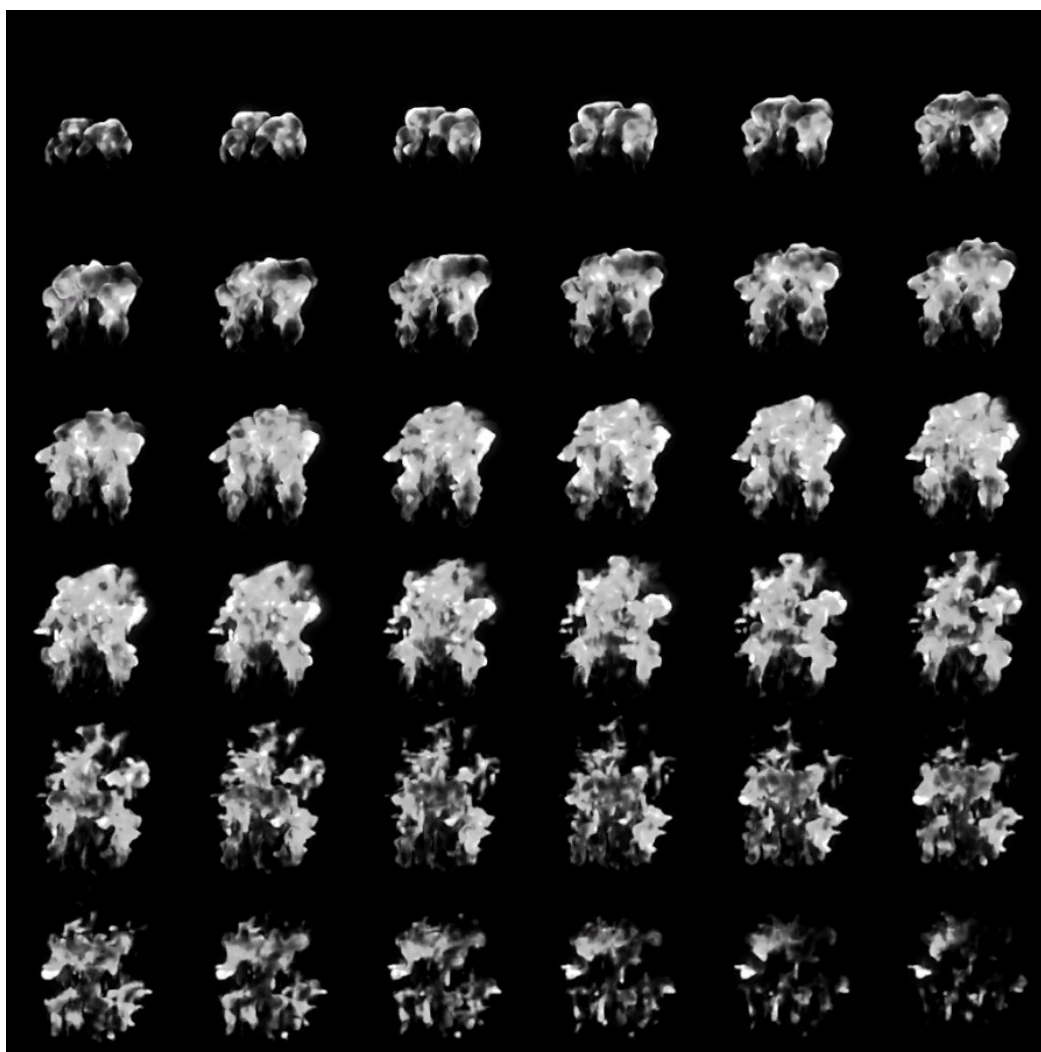
Sada je bilo potrebno dodati dvije sile šuma *Curl Noise Force* i *Vector Noise Force* (slika 3.3) kako bi dobili na nasumičnosti kretanja čestica i spriječili konstantno stvaranje čestica na istim pozicijama. Vrijednosti parametara ovih modula su sljedeće (slika 3.3):



Slika 3.3: Vrijednosti sila šuma

Sljedeći korak je zamijeniti izgled čestice (*Sprite*) iz obične kugle u nešto sličniji česticama vatre. Srećom, Unreal Engine u svom *Starter Content* direktoriju sadrži neke osnovne materijale, između ostaloga i materijal za vatru, odnosno *SubUV*.

Jedan vrlo zanimljivi modul koji je korišten u izradi ovog vizualnog učinka je *SubUV Animation* modul. Čestice u sustavima čestica nam često ne odgovaraju ako je njihov izgled statičan, u našem primjeru vatre, čestice kao kuglice ne daju dovoljno realistične rezultate kao što bismo htjeli. Zato postoje subUV teksture čestica. SubUV teksture čestica su slike sačinjene od većeg broja manjih sličica (*frame*) koje prikazuju izgled čestice kroz vrijeme (slika 3.4).



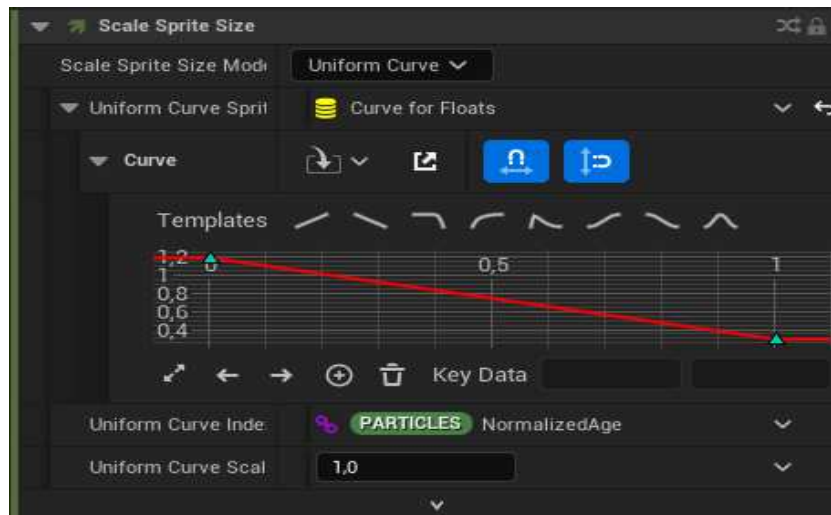
Slika 3.4: SubUV vatre

Naš modul *SubUV Animation* će interpolirati između svake slike i mijenjati izgled čestice kako se ona mijenja kroz teksturu. Da bismo to omogućili potrebno je i u *Sprite Renderer* modulu umjesto standardnog materijala čestice staviti materijal vatre pod nazivom *M_Fire_SubUV*. Pritom treba paziti da u parametrima ovog modula ispravno namjestimo *Sub Image Size* da odgovara dimenzijama matrice slika naše teksture. U našem slučaju 6x6. Dodatno, je bilo potrebno i ispravno unijeti podatke o početnom i krajnjem vremenskom okviru (engl. *frame*) u *SubUV Animation* modulu. Početni je namješten na 0, a krajnji na 36. Rezultat ovog cijelog postupka vidimo na slici (3.5).

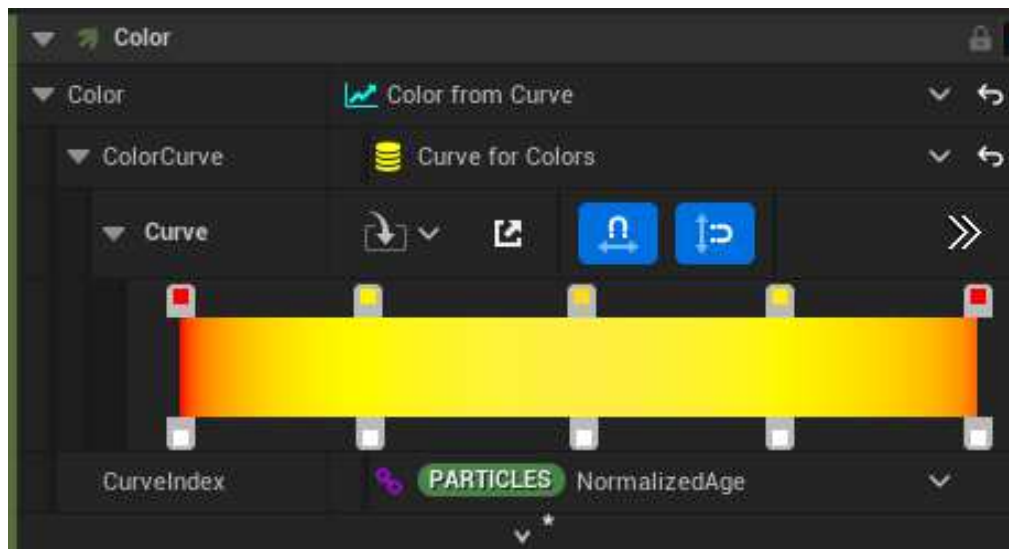


Slika 3.5: Dodani šum i SubUV animacija

U ovom trenutku dobiveno je nešto više nalik vatri. Sada je bilo potrebno manipulirati detaljima kao što su boja, veličina čestice, i slično, da dođemo korak bliže konačnom izgledu. Za veličinu čestice korišten je modul *Scale Sprite Size* koji će mijenjati veličinu čestice u ovisnosti o parametru kojem odlučimo. U radu je korišten dinamički parametar *NormalizedAge*, koji odgovara svojstvu starosti čestice. Unreal će dohvatiti vrijednost starosti svake čestice i njezinu veličinu mijenjati kako se ta vrijednost mijenja. Tu promjenu veličine ćemo zadati krivuljom (slika 3.6) čije vrijednosti odgovaraju vrijednostima kojima ćemo množiti parametar veličine, a vrijednosti *NormalizedAge* će onda biti parametar koji će se kretati po toj krivulji. Za boju je također korištena krivulja kojom se boja mijenja kroz žutu, narančastu i crvenu (slika 3.7).

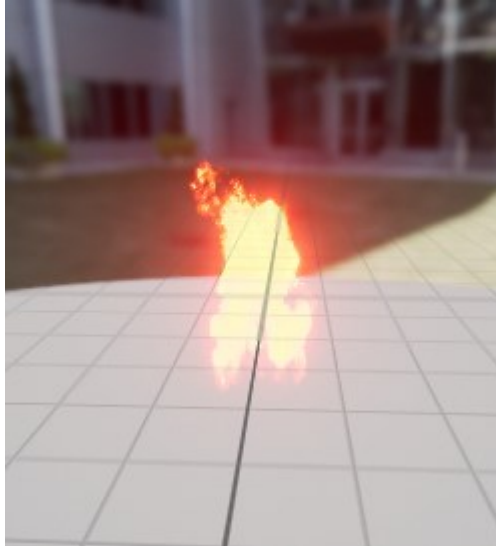


Slika 3.6: krivulja povezana s česticom pomoću *NormalizedAge*



Slika 3.7: Krivulja boje

Na kraju još su dodani moduli *Wind Force*, kako bi dobili dojam da na vatru u nekom smjeru utječe vjetar, i *Light Renderer* kako bi se u sceni vidjela svjetlost koju bi u vatra u stvarnosti trebala davati. *Wind Force* modul je zadan tako da kao parametre uzima smjer u kojem će puhati vjetar i njegova snaga. Ukupna sila koja djeluje na čestice i brzina kojom će se kretati se izračunava u zadnjem modulu *Solve Forces and Velocity*.



Slika 3.8: Konačni izgleda vatre (bez ostalih dijelova)

3.1.2. Dim, iskre i toplinsko izobličenje

Za dim je korišten opet odašiljač *Fountain*, sile su manipulirane tako da čestice dobiju izgled da izlaze iz vatre, a za SubUV je korištena tekstura dima koju Unreal Engine standardno nudi (slika 3.9).



Slika 3.9: Dim

Iskre su ostvarene isto pomoću *Fountain* odašiljača, u ovom slučaju nam zapravo odgovara izgled kuglice za oblik šestice pa posebni SubUV nije potreban. Za boju je korištena boja

ista kao i za vatru, s time da je prozirnost namještena da se sa starosti čestice povećava da se dobije ljepši prikaz laganog gašenja iskre dok leti (slika 3.10).



Slika 3.10: Iskre

Kako je bitno paziti na detalje ako želimo izraditi realističan vizualni učinak, dodali smo još i toplinsko izobličenje koje možemo vidjeti malo iznad vatre ili neke vruće površine. Toplinsko izobličenje je materijal napravljen od prozirne teksture koje izobličuje sliku kada kroz nju gledamo.



Slika 3.11: Konačni izgled vatre u sceni

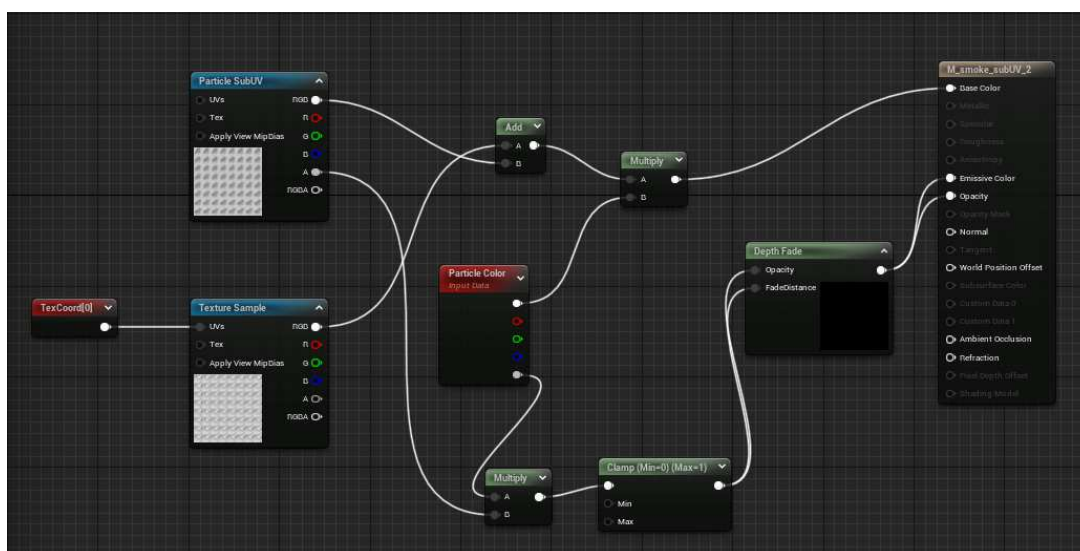
Slika 3.11 prikazuje konačni rezultat u jednostavnoj sceni koja je ručno izgrađena tako da se može lakše vidjeti svaki dio vatre koji je modeliran. Slika 3.12. prikazuje cijeli vizualni učinak sa svim odašiljačima i modulima.



Slika 3.12: Cijeli sustav vatre

3.2. Tornado

Drugi vizualni učinak izrađen za ovaj rad je tornado. Za izradu je bilo potrebno manipulirati silama koje će vrtjeti čestice oko stupa tornada. Cijeli vizualni učinak ponovno se sastoji od više dijelova koji će zajedno činiti cijeli tornado. Kako u Unreal Engine-u ne postoje materijali za izradu dijelova tornada kao što je bio slučaj za vatru, bilo je potrebno snaći se s onima što smo imali pa je za materijal za čestice korišten malo promijenjeni materijal za dim (odnosno SubUV dima) (slika 3.13).



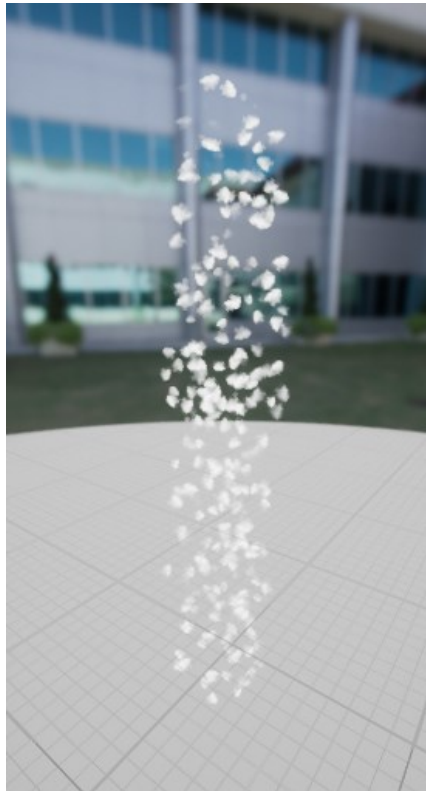
Slika 3.13: Materijal za dim

Materijal je zapravo malo modificirani materijal dima koji je svjetliji i podešen na *Unlit* model sjenčanja jer je u sceni bio puno tamniji nego u uređivaču i svaki drugi pokušaj promjene njegove boje u sceni bio je neuspješan.

3.2.1. Stup tornada

Prvo je bilo potrebno napraviti „stup“ tornada. Za to je korišten odašiljač „*Hanging Particles*“ koji čestice stvara statične u prostoru određenom sa *Shape Location* modulom. Standardno taj modul je namješten na sferu, međutim za tornado je više odgovarao cilindar. Stoga je odmah na početku kao tijelo unutar koje sve stvaraju čestice namješten cilindar. Dimenzije tog cilindra namještene su tako da je cilindar uzak po x i y komponenti,

a jako velik po z. Kako tornado treba biti velik u sceni, vrijednost visine cilindra je postavljena na 9000 jedinica, dok je radijus postavljen na 100 (slika 3.14).

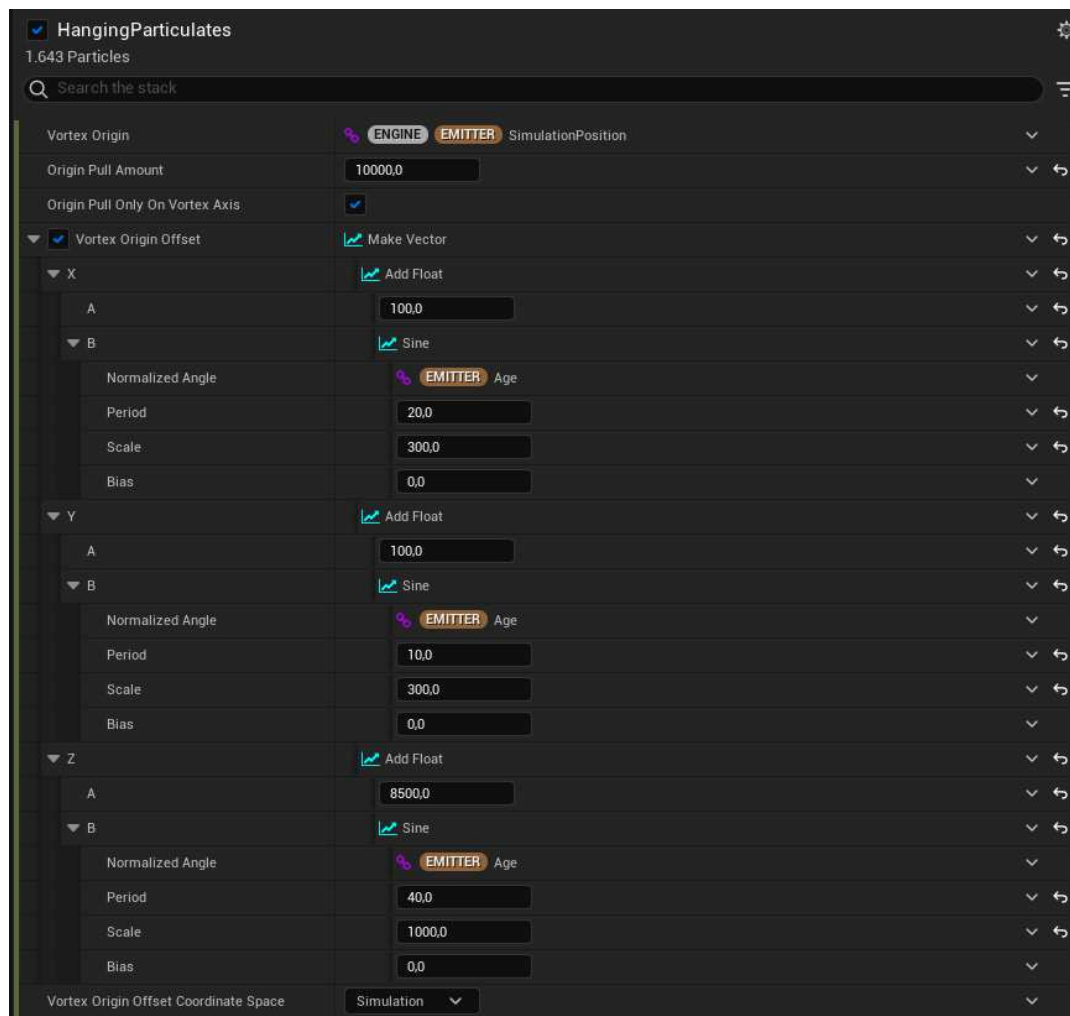


Slika 3.14: Cilindar s česticama

Kako se čestice ne bi toliko isticale i kako bi dali puninu tornadu, veličina čestica je povećana i povećana je stopa stvaranja čestica (*Spawn Rate*).

Kada je stup tornada bio gotov bilo je potrebno česticama u stupu dati silu koja će ih okretati oko njega. Ovo je ostvareno pomoću *Vortex Force* modula (slika 3.15). *Vortex Force* modul simulira utjecaj vrtložne sile koja djeluje na čestice i moguće ga je dosta precizno modelirati tako da daje stupu tornada da „pleše“ umjesto da je statičan. Prvi parametar koji je bio zanimljiv je *Force*. Ovaj parametar nam omogućava da zadamo točnu silu kojom želimo da utječe na čestice (u ovom slučaju namješten je na 6000) i orijentaciju te sile pomoću x, y, z koordinata. Sljedeći parametar koji je bilo potrebno urediti je *Origin* (izvor) sile. Tu je pod izvor dodano da je izvor povezan s pozicijom simulacije (odnosno da ovisi o sustavu odašiljača) što znači da će izvor pratiti odašiljač i uvijek biti u istom odnosu s njime. Tako nam izvor sile neće ostati na mjestu u sceni već će se kretati kako se pomiče i sustav. Zatim je namješten parametar *Origin Pull Amount* koji nam utječe na silu kojom će povlačiti čestice prema sebi.

Čestice sada imaju silu koja ih okreće, ali je bilo potrebno namjestiti odstupanje izvora sile od centra našega sustava (*Offset*). Bez dodavanja ovog parametra izvor sile će biti u centru našega sustava. Ako je sila u centru onda samo razbacuje čestice i gubi se izgleda stupa tornada. Stoga je promijenjen *Offset* da bi zadržali izgled stupa, ali i dobili proširenje stupa na gornjem i donjem dijelu. Ovo je ostvareno tako da je napravljen vektor koji sadrži x, y, z komponente koje označavaju točku u kojoj se izvor nalazi u odnosu na sustav. Ovdje je bilo moguće staviti konstantne vrijednosti svake komponente dok ne dobijemo proširenja stupa, ali onda bi stup bio statičan. Stoga je bio potreban drugačiji pristup.



Slika 3.15: Vortex Force modul (parametri)

Za svaku komponentu je dodan *Add Float* parametar koji uzima dvije vrijednost. Za prvu je dana konstanta, a za drugu je stvoren novi parametar koji je sinusoidna funkcija. Sinusoidna funkcija će imati parametre za skalu, period i pristranost. Njihovim uređivanjem zapravo modificiramo sinusoidu. Skalu je za x i y bilo potrebno staviti na

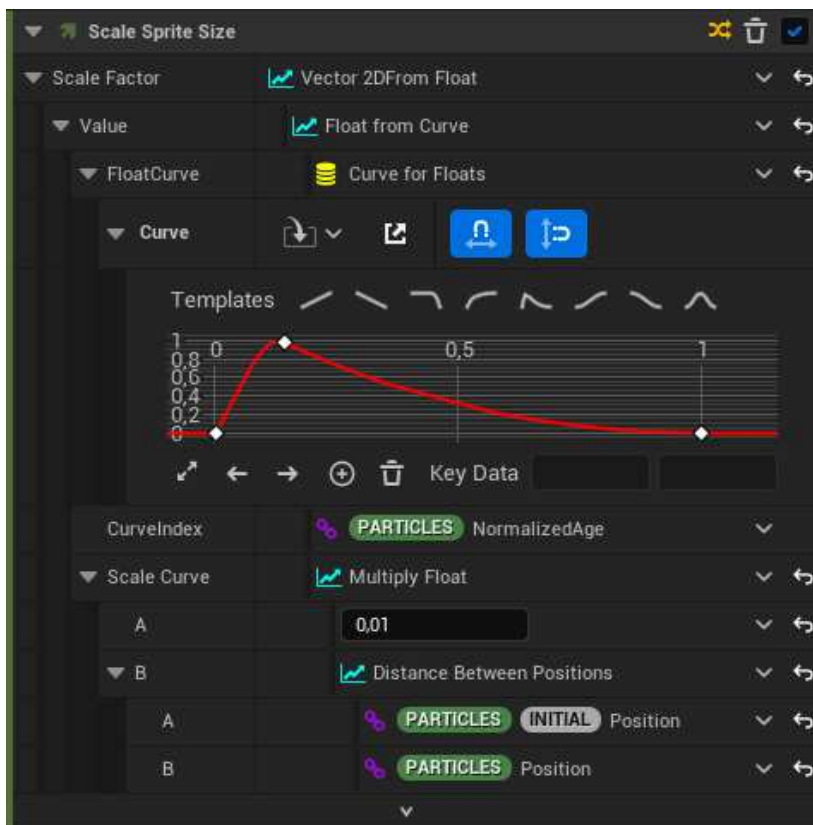
nešto manju vrijednost tako da nam tornado ne „pleše“ previše u tim smjerovima. Parametar *Normalized* nam zapravo omogućuje da zadamo u odnosu na koju vrijednost ćemo se kretati po toj sinusoidi. Vrijednost tog parametra je namještena na starost odašiljača. Ono što je sad dobiveno svime ovime jest da će se naš izvor sile kretati periodički u prostoru sustava u odnosu na starost odašiljača i povlačiti čestice sa sobom pa će se cijeli stup pomicati i davati dojam da tornado nije statičan već da se cijelo vrijeme mijenja. Također je dobiveno i gornje proširenje stupa jer smo manipulirali i z koordinatom te je postavili na visinu koja otprilike odgovara vrhu stupa. Na sličan način je dobiveno i donje proširenje. Korišten je još jedan modul *Vortex Force* koji je bio kopija prethodno napravljenog uz izmjenu z koordinate koja je u ovom slučaju puno niže od dna kako bi se ublažilo proširenje na dnu (slika 3.16).



Slika 3.16: Stup tornada

Dodatno kako bi se još poboljšao oblik u modulu *Scale Sprite Size* (slika 3.17) je napravljeno da se mijenjaju veličine čestice. Promjena veličine je trebala biti manja što je čestica bliže stupu i rasti kako se udaljava. Ovo je ostvareno tako da je uzeta krivulja koja će mijenjati vrijednosti u odnosu na starost čestice, ali faktor kojim se skalira će ovisiti o povezanim parametrima *Initial Position* i *Position*. Ovi parametri, kao i neki drugi, vezani

su uz svaku česticu i moguće ih je dohvatiti ako želimo s njima nešto računati. U radu je korištena udaljenost između točke u kojoj je čestica nastala (*Initial Position*) i točke u kojoj se čestica trenutno nalazi (*Position*). Što je ta udaljenost veća, veći je faktor uvećanja čestice.



Slika 3.17: Parametri modula *Scale Sprite Size* tornada

3.2.2. Dodatni detalji

Iako je sâm tornado bio gotov, trebalo mu je još dodati neke detalje kako bi dobio na malo vjernijem prikazu. Zato su dodani novi odašiljači kojima su implementirane čestice prašine koje lete u blizini tornada i na donji dio stupa dodano je malo proširenje kako bi se dodatno popravio oblik. S ovim dodatcima vizualni učinak je završen i prikazan na slici 3.18, a cijeli sustav prikazan je na slici 3.19.



Slika 3.18: Izgled tornada kroz vrijeme u sceni



Slika 3.19: Sustav tornada

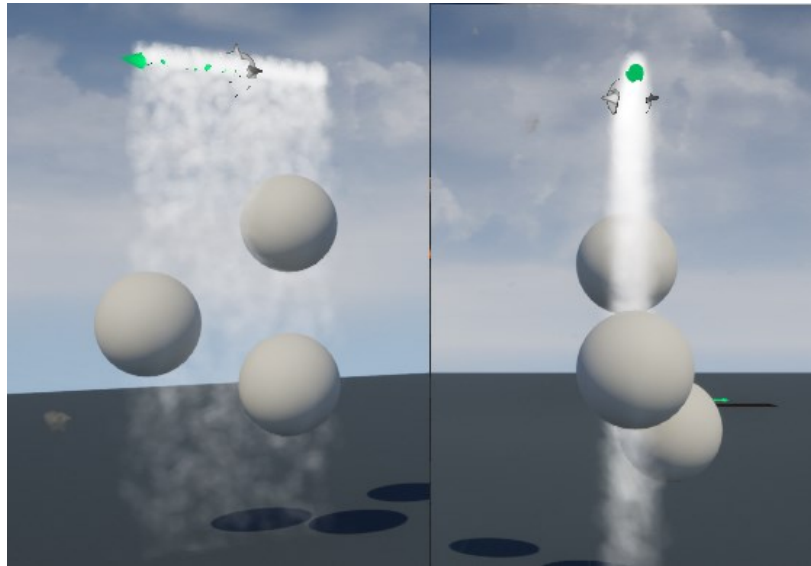
3.3. Vodopad

Posljednji vizualni učinak napravljen u ovom radu je vodopad. Ostvaren je od više sustava čestica. Razlog ovome je što vodopad ovisi o podlozi i o prostoru u kojem se nalazi, voda će se kretati kako je usmjerimo. Stoga ne želimo sve odašiljače fiksirati u odnosu na druge već će svaki biti poseban sustav za sebe i u sceni ćemo ih moći lako pozicionirati tako da prate razmještaj scene.

3.3.1. Vodopad

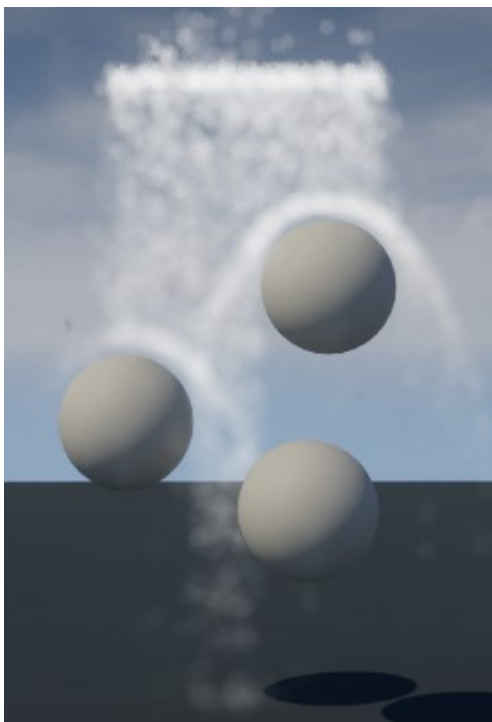
Prvi sustav je voda koja pada niz liticu. Ovaj sustav je relativno jednostavno napraviti. Korišten je *Fountain* odašiljač jer je najprikladniji. Za tijelo unutar kojega će se čestice stvarati nije odgovarala kugla koja je postavljena kao standardno. Bio je potreban oblik koji je ravan i izdužen u jednom smjeru. Stoga je oblik namješten na *Box/Plane*. Nakon čega je korišten mod *Plane*. Ovime je dobivena ravnina iz koje će čestice kretati. Tu ravninu je bilo potrebno još suziti po y koordinati (postavljeno na 0) i proširiti po x . Efektivno je time zapravo dobiven pravac. Nadalje, smanjena je brzina koju čestice dobivaju tako da ne skaču previše gore, ali ostavljena je na vrijednosti 100-200 kako ne bi vrh bio potpuno ravan. Korišten je modul *Gravity Force* kako bi se simulirao utjecaj sile teže i kako bi naše čestice padale. Veličina čestica je povećana da se dobije puni vodopad i smanji isticanje pojedinih čestica. Boja je isto namještena da više odgovara boji vodopada.

Glavni dio vodopada je vizualno napravljen. Međutim, postoji problem. Već je rečeno da je potrebno da naša voda prati teren i podlogu. Trenutno, voda prolazi kroz objekte u sceni i propada kroz scenu. Sustav je u ovoj fazi prikazan na slici (3.20).



Slika 3.20: Voda pada kroz objekte

Ovo se može popraviti korištenjem modula *Collision* [6]. Ovaj modul sadrži puno parametara kako bi se simulirala što bolje interakcija objekata u sceni u čestica vizualnog učinka. Neki od zanimljivijih parametara su *Collision Radius* koji nam omogućava točno definirati u odnosu na što će se izračunavati sudari (npr. u odnosu na dimenzije čestice ili na dimenzije koje mi sami odredimo). Sljedeći parametar koji je korišten je *Bounce*, odnosno odbijanje čestica. Zapravo, ovime odlučujemo o elastičnosti čestice pri udaranju o površinu. U slučaju vode odbijanje je namješteno na 0, s obzirom na to da voda ne bi trebala odskakati od površinu. Još jedan parametar koji je korišten je *Friction*, odnosno trenje, kojim modeliramo kako čestice klize niz površinu. Ovaj parametar je također podešen na 0 kako bi voda klizila bez smetnji niz objekte. Ono što je još bilo potrebno dodati su obujmice na objekte u sceni koji ih nisu imali. Sustav izrađen opisanim postupkom je na slici (3.21).



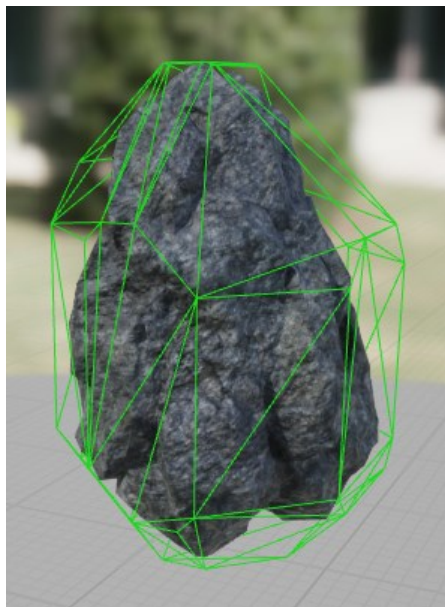
Slika 3.21: Namješten *Collision*

3.3.2. Dodatni detalji

Kako bi vizualni učinak dobio na realističnosti bilo je potrebno dodati neke detalje. To su: kapljice vode koje prskaju pri vrhu vodopada, suptilna maglica od sitnih kapljica vode, i pjena koja nastaje na površini vode u koju voda upada. Također je napravljena i mala scena modelirana pomoću raznih objekata i tekstura.

Kapljice su ostvarene na sličan način kao i glavni dio vodopada. Jedino su čestice znatno manje i brže. Materijal korišten je isti kao i za glavni dio. Maglica je izrađena s većim česticama vrlo velike prozirnosti, koje se kreću od dna vodopada pod kutom sporo prema gore. Pjena je sustav gusto naguranih čestica koje s vremenom postaju prozirnije i kreću se uzduž površine vode. Odašiljači svih ovih sustava su modelirani na isti način kao i za glavni dio tako da ih je u sceni lakše posložiti. Kao što je i prije rečeno svaki od ovih dijelova je poseban sustav za sebe, tako da ih je moguće prilagoditi sceni u kojoj se nalaze.

Scena je izrađena od osnovnih objekata i tekstura. Korišteni su objekti: kamena, kocke, plohe i raslinja. Nekima od ovih objekata bilo je potrebno dodati obujmice kako bi mogli detektirati sudare s česticama vode. Obujmicu za objekt kamena vidimo na slici 3.22.



Slika 3.22: Obujmica objekta

Cijeli vizualni učinak, sastavljen od svih dijelova i posložen u sceni prikazan je na slici 3.23.



Slika 3.23: Vodopad

4. Moguća poboljšanja

Uvijek postoji mjesta za poboljšanje pa tako i ovdje. Vizualni učinci iako izgledaju prihvatljivo, mogli bi izgledati puno realističnije.

Vizualni učinci izrađeni u ovom radu mogli bi se poboljšati korištenjem materijala dostupnih na *Unreal Engine Asset Store*-u ili izrađivanjem novih materijala od nule, kako bi se dobilo na realističnosti. Također, izrađeni vizualni učinci su napravljeni tako da ih nije moguće modificirati izvan *Niagara System* uređivača. Za učinke bi možda bilo korisno da ih je moguće modificirati direktno iz scene. Npr. mijenjati im boju, veličinu i slično. Ovo bi moglo biti ostvareno korištenjem parametara koji koriste unos korisnika. Za potrebe vizualnih učinaka u igrama moglo bi se dodati i interakcija s igračima putem skripti.

5. Zaključak

Vizualni učinci vrlo su popularno područje i njihova primjena vrlo je široka. Vizualnim učincima danas je moguće prikazati praktički sve što možemo zamisliti. U izradi se teži ostvariti učinke što je moguće realističnije kako bi se korisniku pružio bolji dojam njihove stvarnosti. Postoje mnogi alati prikladni za njihovu izradu, a u ovom radu korišten je grafički programski pogon Unreal Engine 5. Alat je vrlo moćan u izradi ne samo vizualnih učinaka, već i kompletnih video igara, filmskih setova i slično.

Glavna tema ovog rada je opis postupka izrade vizualnih učinaka. Vizualni učinci ostvareni su pomoću sustava čestica. Česticama je manipulirano tako da njihovim kretanjem i svojstvima dobijemo željeno ponašanje i u konačnici simuliramo neki dio našeg okruženja. Vizualni učinci izrađeni u ovom radu izgledaju prihvatljivo, ali mogli izgledati puno realističnije. Izrada realističnih i kvalitetnih vizualnih učinaka zahtjeva puno vremena i iskustva te poznavanje alata koji koristimo i svih njegovih mogućnosti.

Literatura

- [1] [Creating Visual Effects in Niagara for Unreal Engine | Unreal Engine 5.0 Documentation](#)
- [2] [Overview of Niagara Effects for Unreal Engine | Unreal Engine 5.0 Documentation](#)
- [3] [Key Concepts in Niagara Effects for Unreal Engine | Unreal Engine 5.0 Documentation](#)
- [4] [Quick Start for Niagara Effects in Unreal Engine | Unreal Engine 5.0 Documentation](#)
- [5] [What Is The Difference Between Animation And VFX \(buzzflick.com\)](#)
- [6] [Collisions in Niagara for Unreal Engine | Unreal Engine 5.0 Documentation](#)
- [7] [Niagara Fluids in Unreal Engine | Unreal Engine 5.0 Documentation](#)
- [8] [ArtStation - How to make a Fire Material in Unreal 5](#)
- [9] [Visual effects - Wikipedia](#)
- [10] [VFX - Everything You Need To Know - NFI](#)

Sažetak

Izrada vizualnih učinaka

U ovom završnom radu opisani su područja korištenja i izrade vizualnih učinaka. Izrađeni su primjeri vizualnih učinaka u grafičkom programskom pogonu Unreal Engine 5. Za izradu vizualnih učinaka korišteni su Niagara Sustavi čestica, koji su glavni alat za izradu vizualnih učinaka u pogonu Unreal Engine. Opisani su postupci izrade i objašnjeni koraci korišteni u tom procesu. Priloženi su rezultati i na temelju njih, diskutirana su moguća proširenja.

Ključne riječi: vizualni učinci, Sustavi čestica, Unreal Engine, Niagara Systems

Summary

Development of Visual Effects

This paper focuses on the areas of utilization and creation of visual effects. Examples of visual effects have been developed using the graphic engine Unreal Engine 5. The creation of these visual effects relies on the utilization of Niagara particle systems, which are the primary tool for visual effects creation in Unreal Engine. The procedures and steps used in this process are described and explained in detail. The results are provided, and based on them, potential improvements are discussed.

Keywords: visual effects, particle systems, Unreal Engine, Niagara Systems

Privitak

Rad je priložen i javno dostupan na gitlabu na sljedećoj poveznici:
https://gitlab.com/MatijaKunc/zavrsni_rad