Filipp Kähärä

Tyyliteltyjen visuaalisten efektien toteutus Unreal Engine 5:ssa

Metropolia Ammattikorkeakoulu

Muotoilija (AMK)

Muotoilun tutkinto-ohjelma

Opinnäytetyö

29.04.2025

Tiivistelmä

Tekijä(t):	Filipp Kähärä
Otsikko:	Tyyliteltyjen visuaalisten efektien toteutus Unreal Engine 5:ssa
Sivumäärä:	93 + 4 liitettä
Aika:	29.4.2025
Tutkinto:	Muotoilija (AMK)
Tutkinto-ohjelma:	Muotoilun tutkinto-ohjelma
Pääaine:	3D-animointi ja -visualisointi
Ohjaaja(t):	Lehtori Kristian Simolin

Tämä opinnäytetyö käsittelee visuaalisten efektien määrittelyä selittämällä, mitä VFX on ja esittelemällä sen keskeisen teoreettisen taustan ja menetelmät. Opinnäytetyö kattaa perusmääritelmät, luokittelut sekä visuaalisten tehosteiden vaikutuksen pelisuunnitteluun.

Teknisellä puolella työ selittää, miten VFX luodaan Unreal Engine:ssä. Siinä käsitellään materiaaleja ja yleisesti käytettyjä tekniikoita, kuten tekstuurien manipulointia, proseduraalisten maskien luomista, gradienttikartoitusta sekä värien yhdistelymenetelmiä. Lisäksi mukana on lyhyt johdanto Niagara järjestelmään, jossa kuvataan sen toimintaa ja modulaarista rakennetta. Blueprintien integrointi materiaaleihin ja Niagaraan on myös käsitelty ja esitelty, miten Blueprintit yhdistävät visuaaliset tehosteet pelin logiikkaan.

Viimeinen luku on käytännön osuus, missä esitellään neljä esimerkkitehostetta, jotka on luotu Unreal Enginen materiaalieditorilla ja Niagaralla. Jokainen esimerkki hyödyntää yhä kehittyneempiä tekniikoita, mutta kokonaisuus on suunnattu edistyneille aloittelijoille ja harrastajille. Jokainen esimerkkitehoste on esitelty videoissa, joiden linkit löytyvät opinnäytetyön liiteosiosta.

Asiasanat:

Visuaaliset efektit, Reaaliaikainen VFX, Peli VFX, Unreal Engine, Varjostaja, Niagara

Abstract

Author(s):	Filipp Kähärä
Title:	Creation of Stylized VFX in Unreal Engine 5
Number of Pages:	93 + 4 appendices
Date:	29 April 2025
Degree:	Bachelor of Culture and Arts
Degree Programme:	Degree Programme in Design
Major:	3D-Animation and -Visualisation
Instructor(s):	Lecturer Kristian Simolin

This thesis defines visual effects in games by explaining what VFX is and outlining the core theory behind it. Thesis covers basic definitions, classifications, and the impact of visual effects on game design.

On the technical side, the thesis explains how VFX are created in Unreal Engine. It covers materials and commonly used techniques such as texture manipulation, procedural mask generation, gradient mapping, and color blending methods. A brief introduction for the Niagara system is provided, detailing its operation and modular structure. The integration of Blueprints with materials and Niagara is also discussed, showing how it links visual effects with game logic.

The practical section presents four example effects created using Unreal Engine's material editor and Niagara. Each example builds on more advanced techniques, yet the overall scope remains aimed at advanced beginner artists and enthusiasts. Each example effect will be provided with video demonstration, links to the videos can be found in the appendices section of this thesis.

Keywords:

VFX, Real-Time VFX, Game VFX, Unreal Engine, Shader, Niagara

Sisällys

1 Johdanto

Viihdeteollisuudessa käytettiin alun perin SFX tekniikoita (Erikoistehosteet), jotka toteutetaan fyysisesti kuvauspaikalla esimerkiksi mekaanisten laitteiden, lavasteiden tai pyrotekniikan avulla. Tietokonegrafiikan kehittyessä syntyivät VFX tekniikat (Visuaaliset tehosteet), joissa kuvia luodaan tai muokataan digitaalisesti jälkituotannossa. VFX mahdollistaa sellaisten elementtien lisäämisen, joita ei voitu kuvata käytännössä, kuten fantastisia olentoja, laajoja maisemia tai vaarallisia tilanteita.

Peleissä VFX toimii reaaliaikaisesti, reagoiden pelaajan toimintaan ja pelimaailman tapahtumiin. Ne parantavat pelikokemusta visuaalisesti ja toiminnallisesti, esimerkiksi näyttämällä räjähdyksiä, taikavoimia tai sääilmiöitä. Visuaaliset tehosteet ovat keskeinen osa pelien kehitystä, ja ne ovat yhtä tärkeitä kuin muutkin pelituotannon osa-alueet, kuten animaattorit tai ympäristöartistit. Visuaaliset tehosteet eivät ainoastaan paranna pelin visuaalista ilmettä, vaan ne myös vahvistavat pelaajan ja pelimaailman välistä vuorovaikutusta. Ne voivat esimerkiksi näyttää, miten pelaajan toimet vaikuttavat ympäristöön tai miten ympäristö reagoi pelaajan toimintaan.

Opinnäytetyön käytännöllisen luvussa käsitellään erityisesti toimintaefektit, kuten räjähdys, savu ja tulta, jättäen pois käyttöliittymä ja hahmoefektit. Teknisessä osuudessa keskitytään materiaalien ja partikkelijärjestelmien, kuten Niagara ja Unreal Enginen materiaalieditori. Tämä opinnäytetyö keskittyy tyyliteltyihin VFX tehosteisiin, mutta tunnistaa realististen efektien ymmärtämisen merkityksen kontekstin kannalta. Tuotantomenetelmä osiossa kerrotaan teoreettisista käsitteistä, kuten UV manipulointi, kohinan kerrostaminen ja materiaalifunktiot, sovelletaan suoraan käytännön esimerkeissä. Esimerkiksi "Häikäisevä valo" esimerkkityö hyödyntää säteittäistä UV kartoitusta ja tekstuurin vieritystä osoittaakseen, kuinka teoria muuntuu dynaamisiksi, reaaliaikaisiksi efekteiksi.

Tämä työ on tarkoitettu opiskelijoille tai harrastajille, jotka haluavat oppia varjostimen luomisen ja materiaalisuunnittelun perusteet, tai ammattilaisille, jotka haluavat laajentaa osaamista. Tavoitteena on kannustaa aloittelijoita, 3D taiteilijoita, harrastajia tai ketä tahansa, joka on kiinnostunut aloittamaan VFX:n luomisen Unreal Engine 5 pelimoottorissa. Jakamalla henkilökohtaiset kokemukseni Unreal Enginestä esitellään tekniikat, joita käytän efektien luomiseen, ja näytän, miten nämä tekniikat toteutetaan kuvien avulla, joissa esitellään nodekokoonpanot ja partikkelisysteemien asetukset.

2 VFX Teoria

2.1 VFX:n määritelmä

"Visuaaliset efektit" on suora käännös englannin kielen termistä "Visual Effects" (VFX). Tämä termi on hyväksytty ammattikielinen käsite myös suomen kielessä, ja sitä käytetään edelleen tässä opinnäytetyössä.

Visuaaliset efektit (VFX) ovat digitaalisia elementtejä, jotka lisätään kohtaukseen jälkikäsittelyn aikana tai reaaliaikaisesti renderöitävissä ympäristöissä visualisoinnin tehostamiseksi. Ne kattavat laajan kirjon tekniikoita, alkaen hienovaraisista säädöistä, kuten värin korjauksesta, ja päättyen monimutkaisiin simulaatioihin, kuten tulipaloihin, tuhoihin tai ympäristöilmiöihin. VFX toimii silta ammattina taidollisen näkemyksen ja teknisen suorituksen välillä, millä pystyy rakentaa immersiivisiä kokemuksia, jotka hämärtävät rajan taiteen ja todellisuuden välillä. Nykyisessä pelimaailmassa tyylitellyn VFX:n merkitys on kasvanut, tarjoten ainutlaatuisia esteettisiä mahdollisuuksia, jotka viehättävät yleisöä. (Anastasiia Chabanova 2022.)

2.2 Reaaliaikaiset efektit

Edellisessä osiossa kävimme läpi VFX:n määrittelystä, tässä kappaleessa käsitellään reaaliaikaisten efektien teoriaa.

Peliteknologian kehittyessä reaaliaikaiset efektit ovat tulleet alan standardiksi, ajettuna innovaatioilla reaaliaikaisessa renderöinnissä ja laitteistokapasiteeteissa. Toisin kuin esirenderöidyt efektit, jotka rendataan etukäteen ja käytetään elokuvissa tai cinematic videoissa, reaaliaikaiset visuaaliset tehosteet generoidaan dynaamisesti pelin aikana, mikä mahdollistaa interaktiiviset ja adaptiiviset kokemukset (Mad VFX 2025). Tämä muutos on tehnyt VFX:stä olennaisen osan tuotantoputkia, ja studiot luottavat yhä enemmän työkaluihin, kuten Unreal Engineen prosessien tehostamiseksi. Lisäksi tyyliteltyjen efektien kysyntä on noussut rinnan pelien, kuten Hades ja Genshin Impact, suosion kanssa, jotka yhdistävät taiteellista luovuutta tekniseen tehokkuuteen. Kuvassa 1 on esitelty ympäristöefektien vaikutus tunnelmaan ja visuaalisen tarinankerrontaan Genshin Impactissa. Nämä pelit osoittavat, kuinka efekteillä pystyy nostaa tarinankerronnan ja pelaajan sitoutumisen korkealle tasolle huolellisesti suunnitelluin visuaalisin elementein, jotka sopivat pelin esteettiseen ja narratiiviseen tavoitteeseen. (Aguiar Gabriel 2023).

Kuva 1. Kuvakaappaus Genshin impactista (RPG fan 2025)

2.3 VFX:n Luokitus

Tässä luvussa esitellään pelien visuaalisten tehosteiden päätyypit, jotka voidaan luokitella käyttötarkoituksen ja simuloitavan ilmiön perusteella. Veselinovikj Bojan luokittelee visuaaliset efektit seuraaviin päätyyppeihin:

- **Ympäristöefektit (environment effects)**
  Ympäristöefektit simuloivat luonnonilmiöitä, kuten sadetta, tuulta, tulta ja vettä.
- **Toimintaefektit (Action effects)**
  Parantavat pelaajan kokemusta dynaamisilla animaatioilla, kuten aseiskuilla, taikoilla tai hahmojen kyvyillä. Nämä vaikutukset riippuvat usein hiukkasjärjestelmistä ja materiaalien vuorovaikutuksista.
- **Pintaefektit (Surface effects)**
  Muuta materiaalin ominaisuuksia simuloidaksesi kulumista, kiiltoa tai maagisia hehkuja.
- **Jälkikäsittelyefektit (Post-Process effects)**
  Käytä koko näytön muokkauksia, kuten valon hohto, syväterävyyttä tai väriluokitusta, parantaaksesi yleistä estetiikkaa.
- **Käyttöliittymäefektit (UI&HUD effects)**

  Käyttöliittymän elementit, esim. Elämäpalkki. Voi olla staattisena tai täysin dynaamisella systeemillä.

2.4 Tyylitellyt ja realistiset efektit

Yleensä visuaaliset tehosteet voi jakaa kahteen kategoriaan: tyylitellyt ja realistiset. Realistiset efektit pyrkivät toistamaan todellisessa maailmassa havaittuja yksityiskohtia korostaen tarkkuutta ja uskollisuutta. Toisaalta tyylitellyt efektit asettavat etusijalle taiteellisen ilmaisun, jossa käytetään liioiteltuja muotoja, eloisia värejä ja yksinkertaistettua geometriaa.

Tyylitelty grafiikka eroaa realistisesta radikaalisti: sen tavoitteena ei ole esittää maailmaa mahdollisimman tarkasti, vaan luoda omaperäisiä ja mieleenpainuvia ilmeitä. Kirkkaat, voimakkaat värit, epätavalliset muodot ja liioitellut siluetit sekä luova, taiteellinen tunnelma tekevät tyylitellystä ulkoasusta monien kehittäjien ja

pelaajien suosikin. Toinen tyylitellyn taiteen vahvuus on sen ajattomuus. Tyylitellyillä peleillä on tapana säilyttää vetovoimansa ja näyttää ajattomilta, mikä auttaa niitä näyttämään yhtä tuoreilta vuosienkin päästä. (Rocket Brush Studio 2025.)

2.5 Esimerkit tyylitellyistä efekteistä peleissä

Aiemmin käsiteltiin reaaliaikaiset efektit ja määriteltiin tyylitellyt ja realistiset efektit, nyt käsitellään tarkemmin tyylitellyt efektit peleissä.

Minun suosikkini esimerkki tyylitellyistä peleistä on Naruto Storms pelisarja (Kuvassa 2). Naruto Storm pelisarjan tehosteet toimivat, koska ne jäljittelevät anime tyyliä tarkasti: voimakkaat siluetit, kirkkaat värit ja selkeät ääriviivat tekevät liikkeistä ja taikuudesta helposti luettavia ja näyttäviä. Synkronointi äänitehosteiden ja animaatiokaappauksien kanssa vahvistaa efektien tunteikasta vaikutusta.

Kuva 2. Naruto Storm Connections kuvakaappaus (Bandai Namco, 2023)

Hyvä esimerkki tyylitellyistä efekteistä on peli Hades. Tämä peli käyttää 2D efektien ja 3D mallien yhdistelmää, missä hahmot ja ympäristö ovat 3D malleja ja toimintaefektit ovat 2D (Näkyy kuvassa 3). Hades:n efektit hyödyntävät tietyn

menetelmän - Spritesheet animaatiot. Hadeksen VFX on toteutettu käsinmaalatuilla tekstuureilla ja pehmeillä värigradienteilla, jotka sulautuvat saumattomasti pelin taiteelliseen tyyliin. Pienet partikkelit, valosäteet ja reaktiot, kuten sirpaleet tai savu, tehostavat taistelun rytmiä ja pitävät visuaalisen ilmeen selkeänä ja yhtenäisenä.

Kuva 3. Hadesin toimintaefektit (Supergiant Games, 2020).

Kuvassa 4 näkyy käyttöliittymä Diablo:ssa, reunoilla on elementit mitkä tarkoittavat elinvoimat (vasemmalla) ja mana (oikealla). Tämä yksinkertainen efekti on tehty yhdistämällä pelihahmon parametrit ja 2D maskit. Diablon terveys ja manapallot toimivat, koska ne yhdistävät selkeän värikoodauksen dynaamisiin animaatioihin ja minimoivat ruudun häiriötekijät. Punaiset ja siniset pallot välittävät välittömästi elintärkeän tiedon - terveyden ja manan - universaalisti ymmärrettävillä sävyillä, kun taas hienovaraiset partikkelipulssit ja täyttöanimaatiot tarjoavat jatkuvaa reaaliaikaista palautetta ilman, että ne häiritsevät pelattavuutta.

Kuva 4. Diablo:n Terveys ja manapallot (Blizzard, 2023)

2.6 Animaation periaatteet VFX:ssä

Tässä käydään läpi animaation periaatteita visuaalisissa efekteissä.

Animaation periaatteet auttavat tuomaan eloa ja uskottavuutta myös visuaalisiin tehosteisiin, sillä niiden avulla säädellään liikettä, rytmiä ja muotoa. JS Qi Mad VFX:stä määrittelee animaatioperiaatteet näin:

Squash and Stretch

Tavoitteena on korostaa kohteen massaa ja elastisuutta muodon muuttuessa puristuessa ja venyessä, mikä antaa tehosteelle painon ja joustavuuden tuntua.

Anticipation (Eteneminen)

Ennakointi valmistaa katsojan huomion päätoimintoon kasvattamalla odotusta. Esimerkiksi ennen räjähdystä nähdään pieni jännitys, joka korostaa lopullisen iskun voimaa.

Staging (Sommittelu)

Tehoste kannattaa sommitella siten, että liike, muoto ja väri ohjaavat katsojan huomion olennaiseen, esimerkiksi korostamalla räjähdyksen keskusta kontrastilla ja liikkeellä.

Follow Through ja Overlapping Action (Jälkiliike ja päällekkäisliike)

Jälkiliike varmistaa, että tehosteen osa jatkaa liikettään painovoiman tai inertiaefektin mukaisesti, ja päällekkäisliike saa erilliset osat liikkumaan eri ajoituksella, mikä lisää realismia.

Slow In and Out (Hidastukset alussa ja lopussa)

Liikkeen alkuun ja loppuun lisätään useampi ruutu hidastamaan ja pehmentämään toimintaa, jolloin tehoste ei vaikuta äkilliseltä vaan sulavalta ja luonnolliselta.

Arcs (Kaaret)

Luonnollinen liikerata on kaarenmuotoinen; tehosteet näyttävät pehmeämmiltä ja sulavammilta, kun esimerkiksi savupilvi tai valonsäde kaartuu loogisesti.

Secondary Action (Toissijainen liike)

Päätehosteen tueksi lisätään pieniä, täydentäviä liikkeitä, kuten kipinöitä räjähdykseksi tai leijuvaa pölyä, jotka rikastuttavat kokonaisuutta ilman, että ne vievät liikaa huomiota.

Timing (Ajastus)

Ruudunvaihto tai partikkelinopeus määrittää tehosteen rytmin ja tuntuman. Oikea ajoitus tekee tehosteesta vakuuttavamman ja hallitumman.

Exaggeration (Liioittelu)

Tehosteita korostetaan reunoilta tai intensiteetiltään yli todellisen rajan, jotta ne erottuvat ja jäävät mieleen tehokkaammin ilman, että niistä tulee liian ylimitoitettuja.

Osa näiden animaatioperiaatteista tulen käyttämään käytännön osuuden Esimerkkiefekti 4 luomiseen.

2.7 Efektit pelimoottoreissa

Useita pelimoottoreita on saatavilla, ja ne tarjoavat laajan työkaluvalikoiman VFX:n luomiseen, palvellen eri tarpeita ja taitotasoja. Unreal Engine erottuu edukseen nodepohjaisella Material Editorilla sekä Niagara partikkelijärjestelmällä, joiden avulla VFX artistit voivat luoda yksityiskohtaisia ja dynaamisia tyyliteltyjä tehosteita. Unity on toinen suosittu valinta, se tarjoaa VFX Graphin, joka mahdollistaa reaaliaikaiset partikkelisimulaatiot ja shader mukautukset, tehden siitä ihanteellisen sekä indie kehittäjille, että suuremmille studioille. Godotin joustava shader kieli ja partikkelijärjestelmät tarjoavat kevyitä mutta tehokkaita ratkaisuja tyyliteltyihin efektiin, sopien erinomaisesti sekä 2D että 3D projekteihin.

Tässä opinnäytetyössä hyödynnän Unreal Engineä VFX:n luomiseen, ja kaikki tulevat teoreettiset käsitteet sekä menetelmät sovelletaan nimenomaan tähän ympäristöön.

2.8 Teorian Yhteenveto

Teorian luvussa käsittelin VFX:n perusmääritelmiä ja luokittelua, erotin realistiset ja tyylitellyt efektit, sekä selitin materiaalien ja shaderien roolit. Sen lisäksi, esittelin esimerkit tyylitellyistä peleistä ja animaation periaatteet visuaalisten efektien luomisessa. Tämä luo vankan pohjan käytännön esimerkeille, joissa teoria siirtyy suoraan toteutukseen ja havainnollistuu konkreettisina, reaaliaikaisina efektinä Unreal Enginessä.

3 VFX:n tuotantomenetelmät Unreal Engine 5:ssa

Edellisessä luvussa käsiteltiin lyhyesti VFX:n teoriaa: VFX:n määritelmä, tyylitellyt efektit peleissä ja efektien luokitus. Tässä luvussa kerrotaan tarkemmin miten visuaaliset efektit toimivat, mitä menetelmiä käytetään materiaalien luomisessa ja kerrotaan Unreal Engine:n partikkelimoottorista – Niagara ja visuaalisesta skriptaustyökalusta Blueprints.

VFX yleisesti koostuu kahdesta pääkomponentista: partikkelijärjestelmästä ja materiaalista. Partikkelijärjestelmä vastaa tehosteiden käyttäytymisestä ajan ja tilan suhteen. Se määrittelee, milloin partikkelit syntyvät, kuinka kauan ne elävät, miten ne liikkuvat ja milloin ne poistuvat. Materiaali tai shader puolestaan huolehtii tehosteen visuaalisesta ulkoasusta: se määrittää partikkelien värin, läpinäkyvyyden, tekstuurit ja pinnan yksityiskohdat. Yhdessä nämä kaksi osaa muodostavat reaaliaikaisen VFX:n ytimen, jossa partikkelit tarjoavat dynamiikan ja materiaalit vastaavat ulkoasusta ja muodonmuutoksista. Materiaalipuolella nodepohjainen Material Editor tarjoaa joustavan käyttöliittymän tekstuurien, funktioiden ja matemaattisten operaattoreiden yhdistämiseen. (Epic games, Unreal Engine Documentation 2025).

3.1 Materiaalit ja Sävyttimet

Nyt keskitymme Unreal Engine:n materiaaleihin ja erilaisiin menetelmiin, mitkä mahdollistavat niiden luomista.

Unreal Enginen materiaalit ovat korkeantason resursseja, jotka määrittävät objektien pintojen ominaisuudet yhdistämällä tekstuurit, parametrien arvot ja muut syötteet yhdeksi kokonaisuudeksi. Materiaali editori tarjoaa nodepohjaisen käyttöliittymän, jonka kautta taiteilijat voivat rakentaa ja muokata materiaaleja visuaalisesti ilman, että heiltä vaaditaan HLSL koodin kirjoittamista. Shaderit (Sävyttimet) ovat GPU:lla ajettavia ohjelmia, jotka prosessoivat verteksi ja pikselidataa materiaalieditorissa määriteltyjen ohjeiden mukaisesti lopullisen kuvan renderöimiseksi. Käytännössä materiaali kätkee alleen shaderien monimutkaisuuden ja

toimii käyttäjäystävällisenä kerroksena, jossa visuaaliset ominaisuudet kuten hehku, läpinäkyvyys ja normaalikartat säädetään graafisessa ympäristössä. Materiaalit rakennetaan yhdistämällä Material Expressions nodeja ja välittämällä tulokset syötteisiin (Epic games, Unreal Engine Documentation 2025).

Seuraavaksi syvennyn materiaalieditorin tekstuurikarttoihin ja niiden rooliin visuaalisten efektien toteutuksessa.

Tekstuurikartat

Unreal Enginessä tekstuurikartat ovat keskeisessä roolissa materiaalien pinnan ominaisuuksien määrittelyssä. Niiden avulla taiteilijat voivat luoda yksityiskohtaisia ja visuaalisesti rikkaita pintoja ilman, että täytyy lisätä monimutkaista geometriaa. Materiaalia rakennettaessa päämateriaalinode (Kuvassa 5) toimii keskuksena, johon erilaiset tekstuurikartat liitetään omiin tuloihinsa. Jokainen syöte vaikuttaa materiaalin lopulliseen visuaaliseen ilmeeseen eri tavoin. (Epic games, Unreal Engine Documentation 2025).

Alla on listattu tärkeimmät tekstuurikarttojen tulot, joita hyödynnetään erityisesti tyyliteltyjen efektien luomisessa:

- **Base Color:**

Perusväri määrittää materiaalin yleisvärin. Periaatteessa perusvärin tulee edustaa pinnasta heijastuvaa hajavaloa, josta on vähennetty kaikki peiliheijastukset.

- **Metallic**:

Arvo 0 = epämetallinen, 1 = metalli, vaikuttaa valon heijastukseen ja specular malliin.

- **Specular**:

Spekulaariset kartat lisäävät esineiden pinnalle tasaisuutta, jolloin pinta heijastaa enemmän valoa suoraan, mikä lisää pinnan peilivaloa.

- **Roughness**:

Ohjaa pinnan karheutta ja heijastuvuuden terävyyttä, tummat arvot tekevät pinnasta kiiltävämmän.

- **Emissive**:

Emissive (Hehku) kartta asetetaan Emissive Color tuloon, jolloin materiaalin määritellyt alueet näyttävät säteilevän valoa ilman ulkoista valaistusta.

- **Opacity / Opacity mask**:

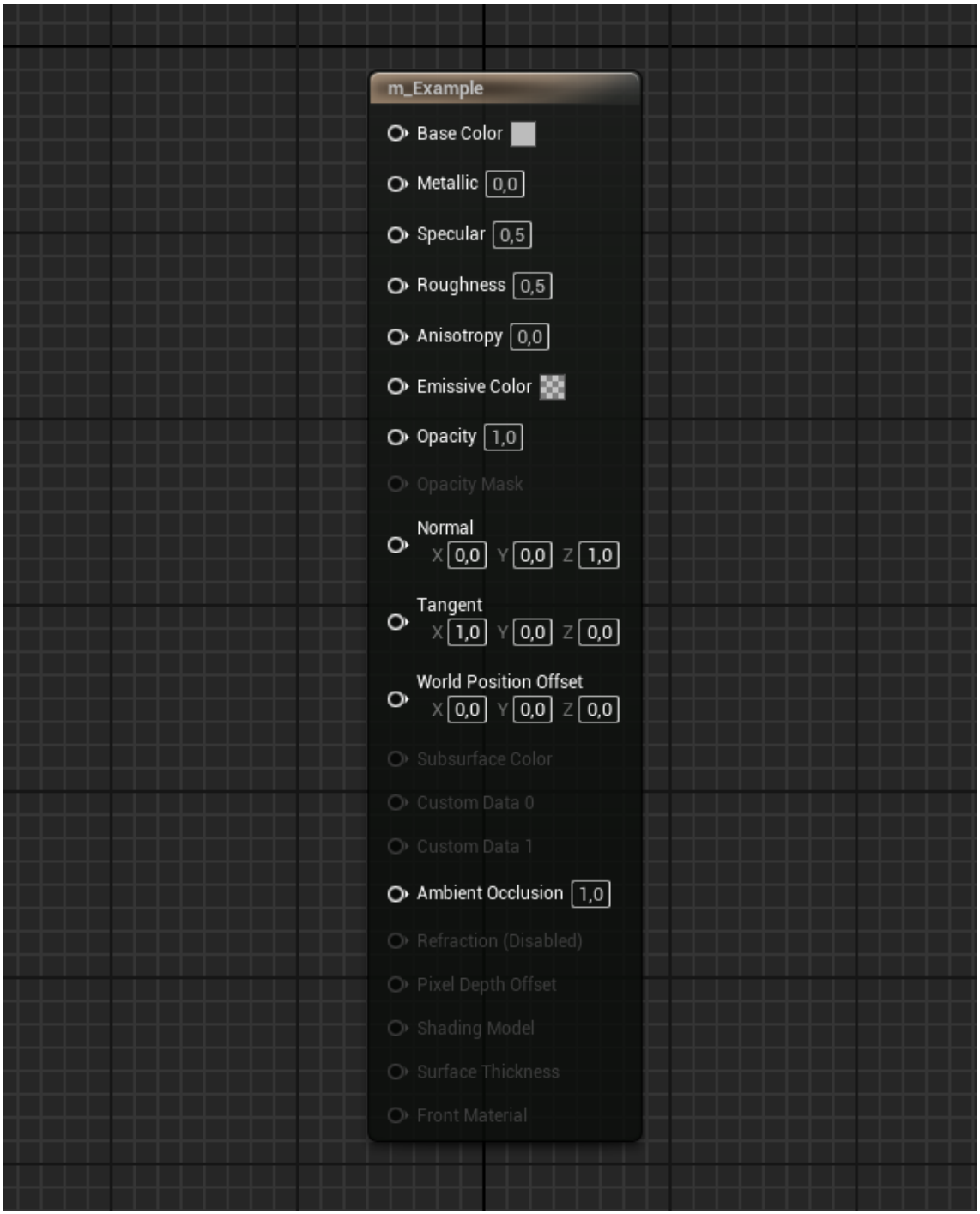
Läpinäkyvyys (Opacity) säätää materiaalin läpinäkyvyyttä sallien pehmeät siirtymät täysin läpinäkymättömän ja täysin läpinäkyvän välillä, kun Läpinäkyvyysmaski (Opacity Mask) käyttää mustavalkoista tekstuuria määrittääkseen tietyt alueet joko täysin läpinäkymättömiksi (valkoinen) tai täysin läpinäkyviksi (musta) ilman pehmeitä siirtymiä.

- **Normal**:

Normal kanava määrittää pinnan korkeus ja syvyysvaikutelmat käyttämällä värillistä tekstuuria, joka kuvaa pintaa kohtisuorassa olevien normaalivektorien suuntaa, luoden näin illuusion kolmiulotteisista yksityiskohdista ilman todellisia geometrisia muutoksia.

- **World Position Offset (Displacement):**

Displacement kanava (WPO) muuttaa meshin verteksien sijaintia maailman tilassa, mahdollistaen todellisen geometrisen muodonmuutoksen, kuten pintojen liikuttamisen tai dynaamisten muotojen luomisen suoraan materiaalissa.



Kuva 5. Materiaalinode (Filipp, 2025)

Artisti käyttää tekstuurikarttoja ja maskeja kerroksittain materiaalissa tuodakseen esiin eri tasoisia yksityiskohtia, jolloin pinnat näyttävät monimutkaisilta ja visuaalisesti rikkailta. Esimerkiksi kohinatekstuuriin perustuvan karheuskartan yhdistäminen liukuvärimaskiin voi simuloida kulunutta tai epätasaista pintaa, mikä lisää tyyliteltyihin efekteihin syvyyttä ja luonnetta. Unreal Enginen Material

Editorissa näiden karttojen parametreja hienosäädetään materiaalikohtaisesti, jotta lopputulos täyttää halutut visuaaliset tavoitteet. Kun näitä syötteitä hyödynnetään oikein, taiteilija voi luoda sekä näyttäviä että reaaliaikaan optimoituja materiaaleja. Tehokkaiden VFX tehosteiden tekeminen vaatii useiden eri tekniikoiden hallintaa, joista kerrotaan tarkemmin seuraavissa kappaleissa. Käytännön osuudessa keskitytään ainoastaan Color, Emission, Opacity ja Displacement kanaviin.

3.2 Tärkeät menetelmät materiaaleissa

Tässä luvussa luetellaan tärkeät menetelmät materiaalien tuotantomenetelmät.

- **UV:n muokkaus**: Skaalaus, Vieritys, Kierto, Vääristely.
- **Tekstuurit: Maskit**: Värit, Kohinatekstuurit, Atlakset.
- **Verteksien manipulointi**: Mallin pinnan syrjäytyminen.
- **3D mallien käyttö**: Tiettyjen 3D pintojen ja mallien.
- **Aikajana**: Tiettyjen parametrien muuttumista ajan perusteella.

Seuraavissa luvuissa käydään läpi tarkemmin jokainen menetelmä ja sen sovellus Unreal Enginen materiaalieditorissa ja Niagara partikkelijärjestelmässä.

3.3 3D mallien käyttö VFX:ssä

Perinteisesti partikkelijärjestelmät käyttävät kaksiulotteisia Spritejä, jotka ovat kameraan suunnattuja tasoja. Niitä käytetään esimerkiksi liekkien, savun tai kipinöiden kaltaisten efektien luomiseen. Spritejen avulla voidaan helposti toteuttaa kevyitä ja visuaalisesti vaikuttavia tehosteita, jotka toimivat hyvin erilaisissa pelitilanteissa.

Kuitenkin, kun halutaan lisätä tehosteisiin kolmiulotteisuutta ja syvyyttä, voidaan käyttää mesh partikkeleita. Mesh partikkelit mahdollistavat monimutkaisempien ja realistisempien efektien luomisen, kuten esimerkiksi pyörivät 3D objektit, räjähdykset tai taikapallot. Niiden avulla voidaan myös toteuttaa volumetrisia tehosteita, jotka reagoivat ympäristöön ja valaistukseen luonnollisemmin.

Unreal Enginen Niagara:ssa mesh partikkeleiden käyttö edellyttää Mesh Renderer moduulin lisäämistä partikkelijärjestelmään. Tämän avulla voidaan määrittää käytettävä 3D malli ja siihen liitettävä materiaali. Lisäksi voidaan säätää partikkelien käyttäytymistä, kuten pyörimistä, skaalausta ja elinkaarta, jotta saavutetaan haluttu visuaalinen lopputulos.

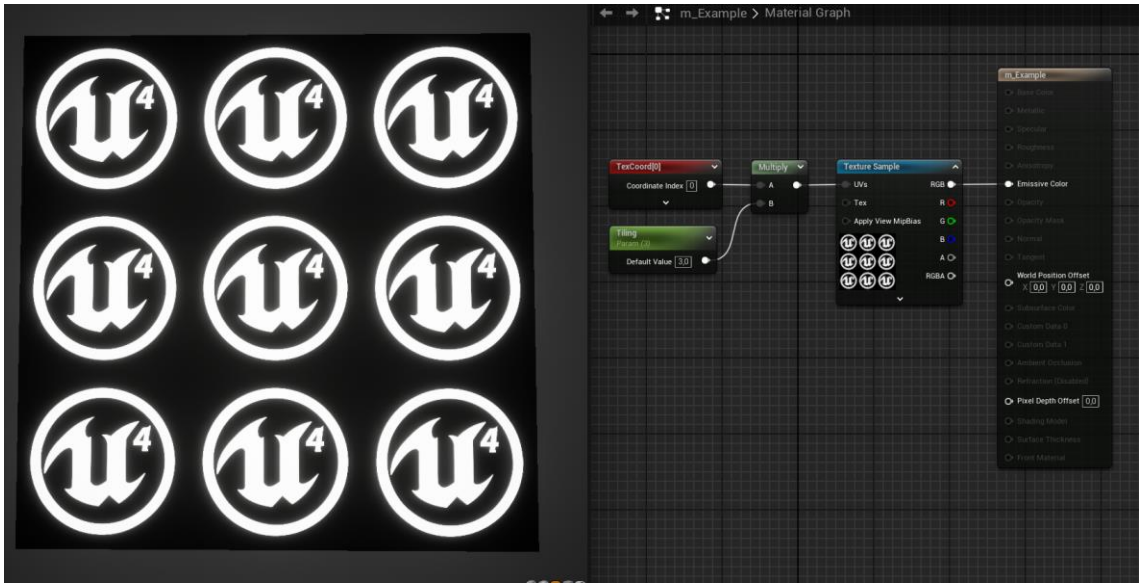
Yksi esimerkki mesh partikkeleiden käytöstä on tekstuurien vieritys 3D malleilla. Tämä tekniikka mahdollistaa dynaamisten ja jatkuvasti muuttuvien pintojen luomisen, jotka voivat esimerkiksi simuloida virtaavaa vettä tai liikkuvaa energiaa. Tällöin tekstuurin UV koordinaatteja manipuloidaan materiaalieditorissa, jotta saavutetaan haluttu liike ja vaikutelma. (Epic games, Unreal Engine Documentation 2025).

3.4 UV Kartoituksen manipulointi tekniikat

UV kartoitus on kriittinen prosessi 3D mallinnuksessa ja materiaalien luomisessa, koska se määrittää, miten 2D tekstuureja sovelletaan 3D malleihin. Oikea UV kartoitus varmistaa, että tekstuurit ovat saumattomasti linjassa geometrian kanssa välttäen vääristymiä tai venymistä (3Dcoat 2025). Unreal Enginessä UV manipulointitekniikoiden avulla taiteilijat voivat dynaamisesti muuttaa materiaalien tekstuurikoordinaatteja luoden tehosteita, kuten vieritystä, vääristymää ja skaalausta. (Epic games, Unreal Engine Documentation 2025). Alla on yleiskatsaus yleisimmistä UV kartoitus ja manipulointitekniikoista, joita käytetään materiaalien luomisessa:

Tekstuurin laatoitus

Unreal Enginessä tekstuurin skaalaus määrittää, kuinka suuri tai pieni tekstuuri näyttää olevan suhteessa mallin pintaan. Materiaali editorissa tämä saavutetaan kertomalla tai jakamalla UV koordinaatteja. Esimerkiksi UV koordinaattien pienentäminen (esim. kertomalla arvoilla suuremmilla kuin 1) saa aikaan sen, että tekstuuri toistuu useammin pinnalla, mikä on hyödyllistä esimerkiksi tiiliseinien tai vedenpinnan kaltaisten toistuvien kuvioiden luomisessa. Tämä menetelmä mahdollistaa yksityiskohtaisten pintojen luomisen ilman, että tarvitsee lisätä geometrista monimutkaisuutta. Lisäksi Unreal Enginen materiaalieditorissa on mahdollista käyttää parametrisoituja arvoja, jolloin tekstuurin skaalausta voidaan säätää dynaamisesti materiaalinstanssien kautta. (Autodesk 2025). Kuvassa 6 on esitelty kolme kertaa laatoitettu tekstuuri.



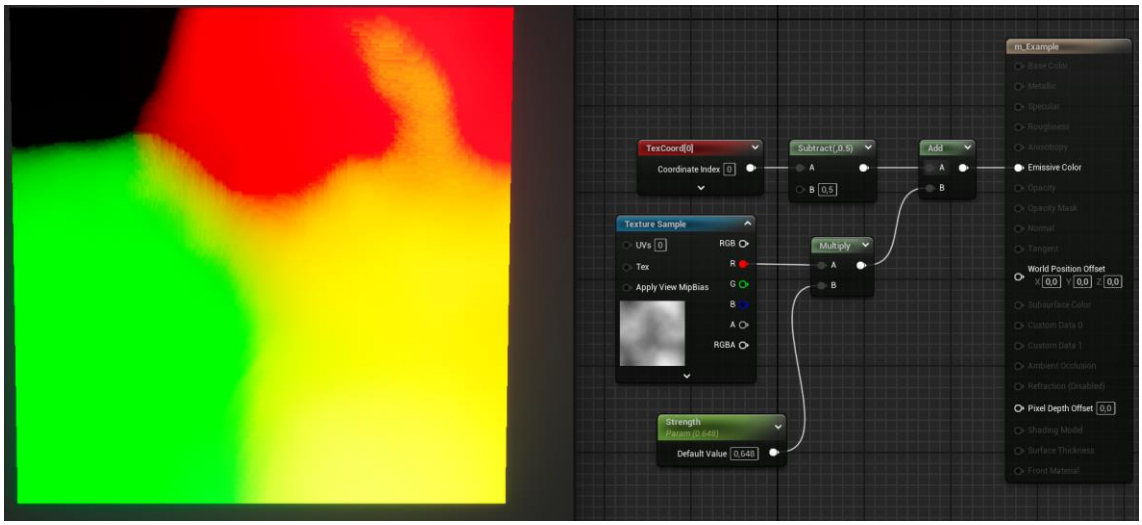
Kuva 6. Tekstuuri laatoitettu kolme kertaa (Filipp, 2025)

Tekstuurin vieritys

Unreal Enginessä tekstuurin vieritys, eli UV panning, on tehokas tekniikka, jolla saadaan aikaan liikkuvia visuaalisia efektejä. Tässä menetelmässä tekstuurin UV koordinaatteja siirretään ajan myötä, mikä luo vaikutelman liikkeestä. Tätä käytetään yleisesti esimerkiksi virtaavan veden, laavan tai liukuhihnojen kaltaisten efektien luomiseen. (Epic games, Unreal Engine Documentation, 2025).

UV Vääristymä

Tekstuurin vääristys (UV distortion) on tehokas menetelmä, jolla voidaan luoda dynaamisia ja visuaalisesti vaikuttavia efektejä, kuten lämpöväreilyä, portaaleja tai maagisia hohteita. Tämä saavutetaan manipuloimalla UV koordinaatteja erilaisilla kohinatekstuureilla ja matemaattisilla operaatioilla, kuten lisäämällä tai kertomalla arvoja (Kuvassa 7). Esimerkiksi Perlin tai Voronoi kohinatekstuureja voidaan käyttää UV koordinaattien siirtämiseen, mikä simuloi tulen tai savun kaoottista liikettä. Tällainen lähestymistapa mahdollistaa monimutkaisten ja elävien pintojen luomisen ilman raskaita animaatioita tai lisägeometriaa. (Jasper Flick 2018).

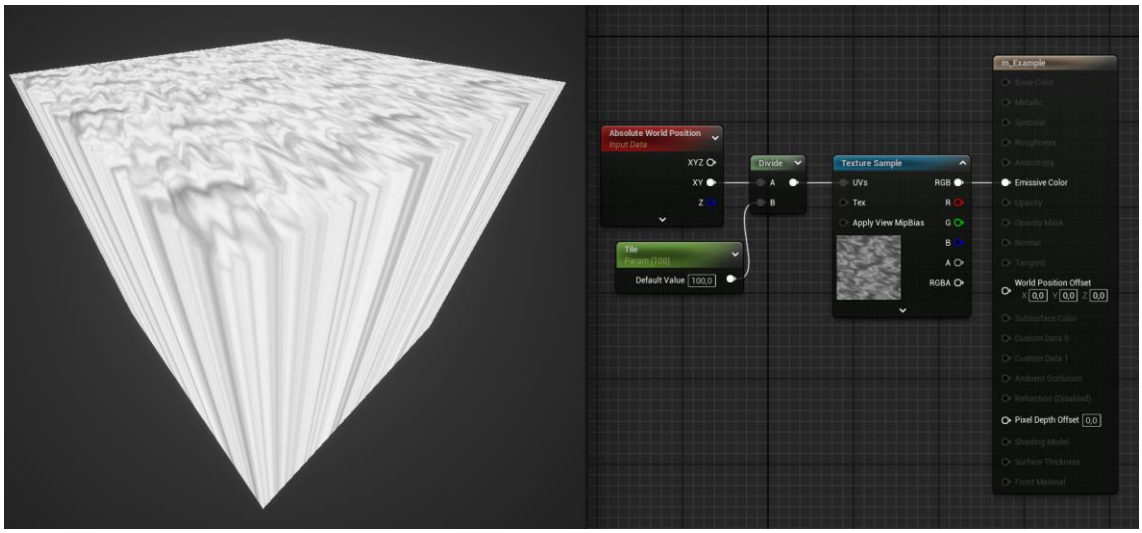


Kuva 7. UV:n vääristely perlin kohinatekstuurilla (Filipp, 2025)

Planar Projection

Tasoprojektio on UV kartoitustekniikka, jossa tekstuuri projisoidaan mallin pinnalle yhdestä suunnasta, ikään kuin valo heijastuisi tasaisesti kohteeseen (Kuvassa 8). Tämä menetelmä sopii tasaisille tai lähes tasaisille pinnoille, kuten lattioille, seinille tai kyltteihin. Tasoprojektion avulla voidaan nopeasti ja tarkasti kohdistaa tekstuureja ilman monimutkaista UV kartoitusprosessissa.

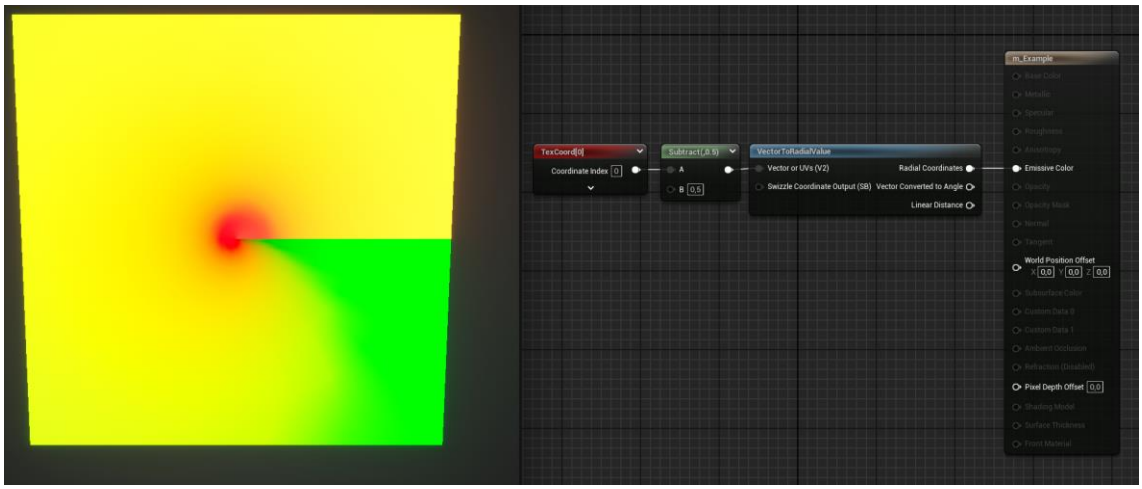
Jos tasoprojektiota käytetään kaarevilla tai monimutkaisilla pinnoilla, saattaa esiintyä venymistä tai vääristymiä tekstuurissa. Tämä johtuu siitä, että projektion suunta ei vastaa pinnan muotoa kaikilta osin. (Autodesk 2025).



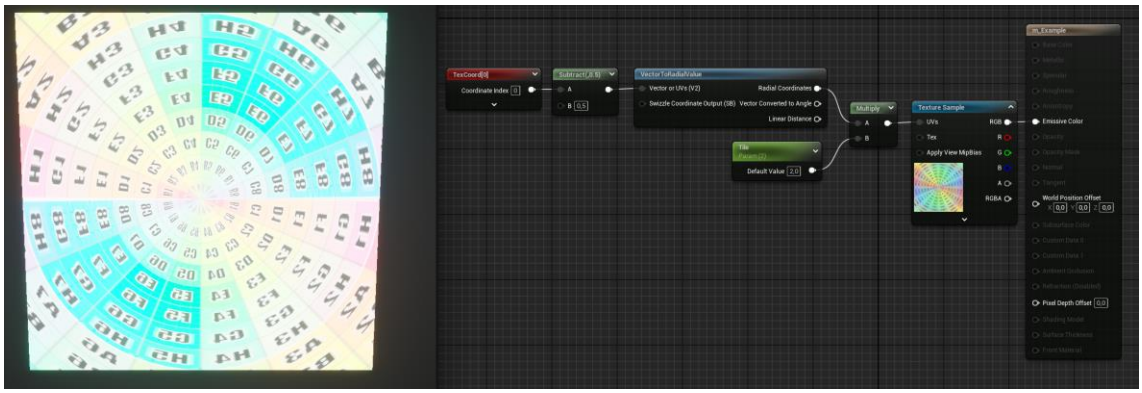
Kuva 8. Planaarinen UV kartoitus (Filipp, 2025)

Radial UV mapping

Radiaalinen UV kartoitus on tekniikka, jossa perinteiset kartesiolaiset UV koordinaatit muunnetaan polar eli sädekoordinaateiksi. (The blog at the bottom of the sea 2013). Tämä mahdollistaa efektien, kuten auringonsäteiden, linssiheijastusten tai pyörteisten vääristymien, luomisen, jotka säteilevät ulospäin keskipisteestä (Kuvat 9 ja 10).



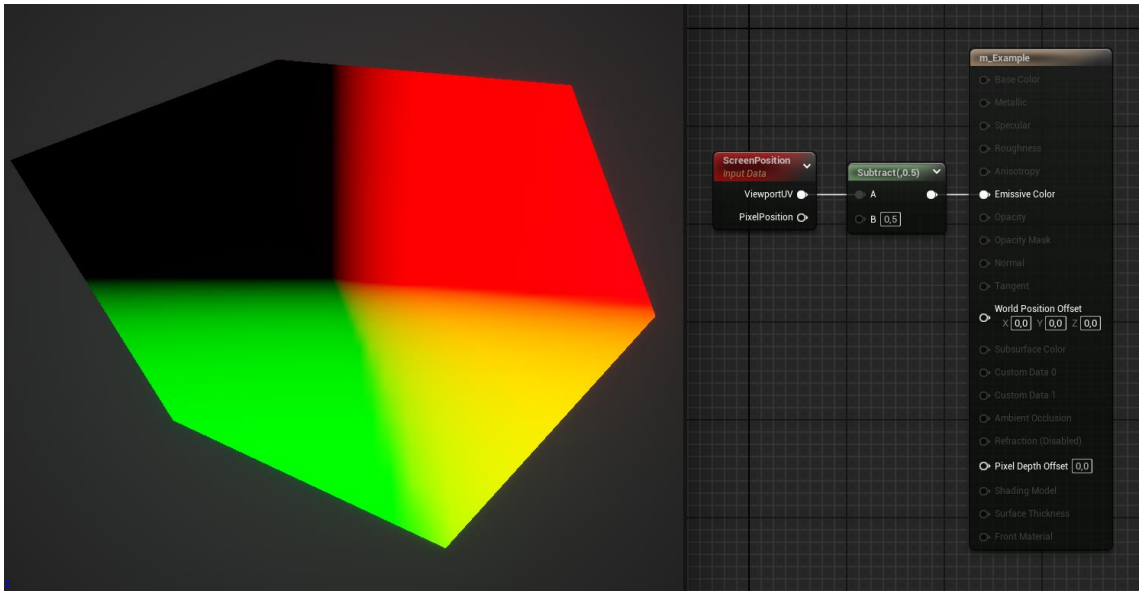
Kuva 9. Radiaalinen UV kartoitus (Filipp, 2025)



Kuva 10. Radiaalinen UV kartoitus kuviotekstuurissa (Filipp, 2025)

Screen UV Mapping

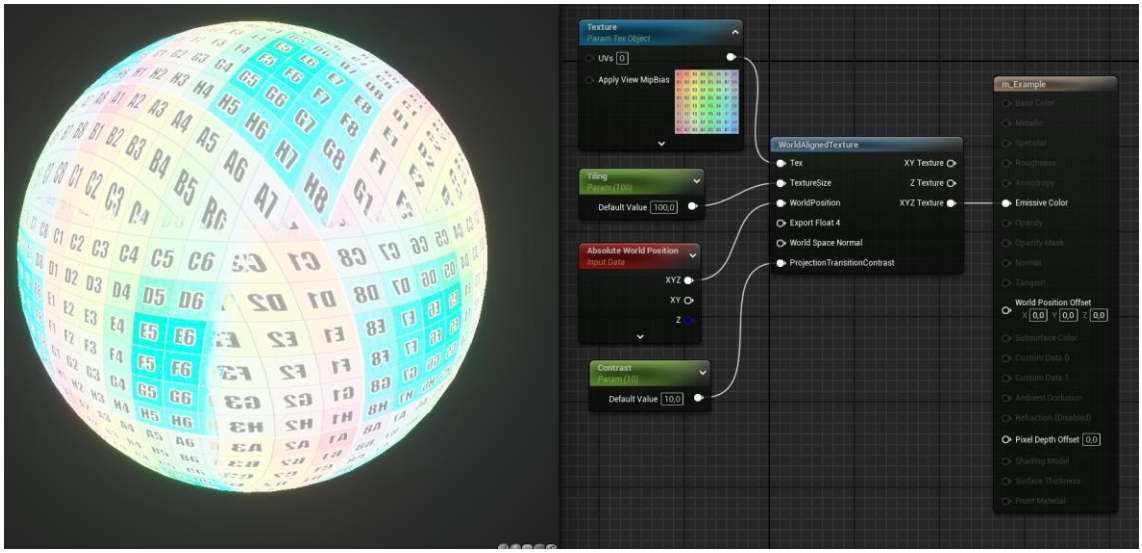
Näytön tilan UV kartoitus on tekniikka, jossa tekstuurien UV koordinaatit perustuvat näytön koordinaatistoon (screen space) sen sijaan, että ne käyttäisivät objektin omaa UV karttaa. Tämä menetelmä on erityisen hyödyllinen visuaalisissa efekteissä, kuten näytön tilan heijastuksissa, taittumisissa ja vääristymissä, koska se mahdollistaa tekstuurien ankkuroinnin kameranäkymään riippumatta objektin geometriasta tai UV asettelusta. (Kuva 11). (Ronja’s tutorials 2019).



Kuva 11. Näytön sidotut UV:t kuutiolla (Filipp, 2025)

Triplanar Mapping

Triplanaarinen kartoitus on tehokas UV kartoitustekniikka, joka poistaa perinteisen UV kartoittamisen tarpeen projisoimalla tekstuureja objektin kolmen pääakselin (X, Y ja Z) suuntaisesti. Tämä lähestymistapa yhdistää kunkin akselin tekstuurinäytteet pintanormaalien perusteella, mikä varmistaa saumattoman tekstuurin soveltamisen jopa monimutkaisille tai epätasaisille geometrioille. Triplanaarinen kartoitus on erityisen hyödyllinen maastoissa, orgaanisissa muodoissa tai objekteissa, joiden UV kartoitus on haastavaa, sillä se estää näkyvät saumat tai venymät, joita voi esiintyä perinteisellä UV kartoituksella. (Ronja's totorials 2018). Triplanaarinen kartoitus on esitelty kuvassa 12.



Kuva 12. Triplanaarinen kartoitus pallolla (Filipp, 2025)

3.5 Tekstuurit ja Maskit

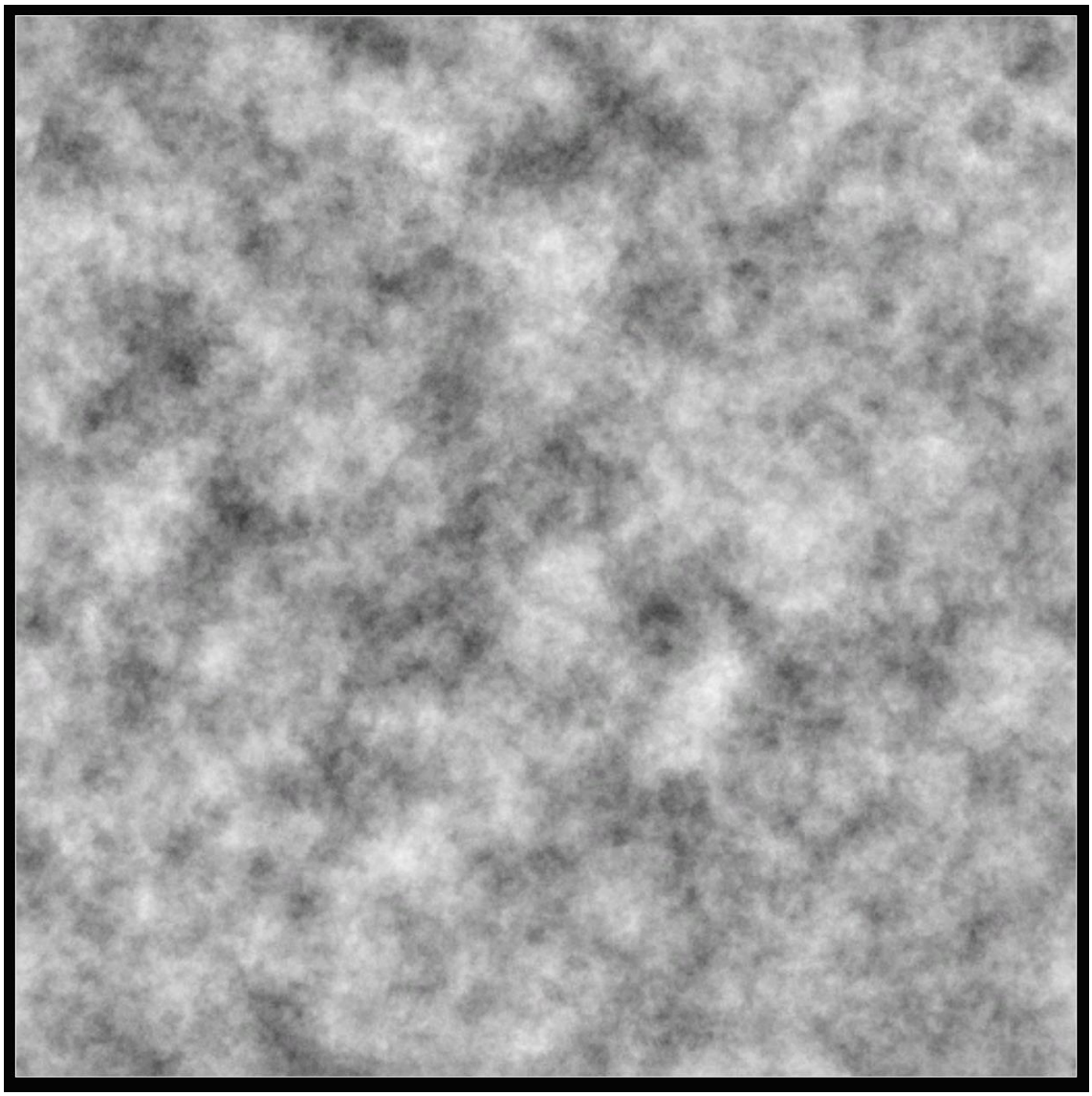
Unreal Enginessä mustavalkoiset tekstuurit ja maskit ovat keskeisiä työkaluja visuaalisten efektien luomisessa. Näitä käytetään muun muassa läpinäkyvyyden hallintaan, värien sekoittamiseen ja materiaalien yhdistämiseen. Maskit voivat olla erillisiä harmaasävykuvia tai yksittäisiä kanavia (R, G, B tai A) tekstuu-

reista, jotka rajoittavat efektin vaikutusalueen materiaalissa. Usein maskit sisällytetään tekstuurien kanaviin, kuten diffuusiotekstuurin alfa kanavaan, mikä vähentää tarvittavien tekstuurinäytteiden määrää ja parantaa suorituskykyä.

Harmaasävykuvat toimivat tehokkaina maskeina, jotka määrittävät, mitkä alueet ovat alttiita erilaisille sekoitus ja muokkaustoiminnoille. Maskeja käytetään laajasti visuaalisissa efekteissä, sillä ne mahdollistavat materiaalien ja efektien sulavat siirtymät, kuten liukuvärjäykset, häivytykset ja valikoidut vääristymät. (Epic games, Unreal Engine Documentation, 2025.)

Kohina Tekstuuri

Kohinatekstuureja käytetään laajasti luomaan orgaanisia ja dynaamisia efektejä, kuten savua, tulta, pilviä ja maagisia ilmiöitä. Menetelmät, kuten Perlin ja Voronoi kohinatekstuurit (Kuvassa 13), mahdollistavat monimutkaisten tekstuurien luomisen ilman korkearesoluutioisia kuvia. Esimerkiksi Voronoi kohinaa voidaan käyttää luomaan pilvimäisiä muotoja tai halkeilevia pintoja, ja sen yhdistäminen muihin kohinatyyppeihin, kuten gradienttikohinaan, lisää visuaalista monimuotoisuutta. Kohinatekstuureja voidaan käyttää opacity mask syötteenä, jolloin saadaan aikaan dynaamisia läpinäkyvyysmuutoksia. Unreal Engine:ssä kohinatekstuureja voidaan yhdistää muihin tekstuureihin ja ajallisiin nodeihiin, kuten Time ja Sine nodet, luomaan animoituja efektejä. Lisäksi kohinatekstuureja voidaan käyttää värien sekoittamiseen ja materiaalien yhdistämiseen, mikä lisää visuaalista monimuotoisuutta ja realismia.

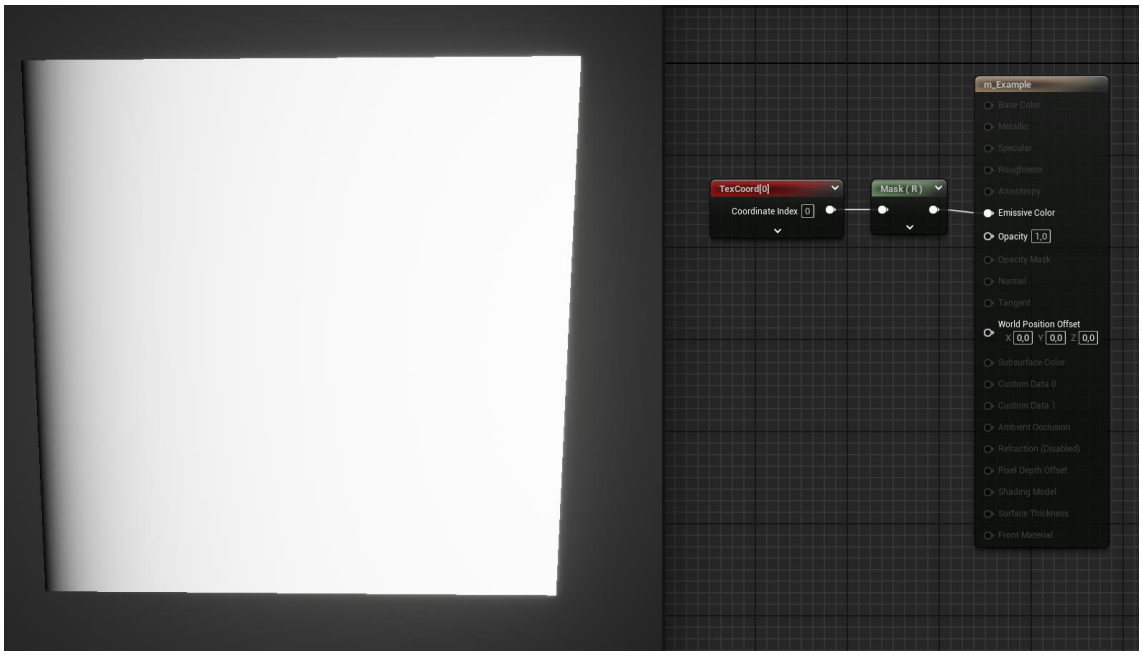


Kuva 13. Perlin kohinatekstuuri (Filipp, 2025)

Gradientit

Gradient maskit ovat tehokas työkalu materiaalien ja visuaalisten efektien hallintaan. Ne mahdollistavat sujuvat siirtymät eri materiaalien tai efektien välillä, kuten häivytykset, liukuvärjäykset ja valikoidut vääristymät. Gradientteja voidaan luoda käyttämällä erilaisia funktioita, kuten LinearGradient kuvassa 14, RadialGradientExponential ja DiamondGradient, jotka perustuvat tekstuurikoordinaatteihin ja tarjoavat säädettäviä parametreja gradientin muotoon ja tiheyteen.

Näitä maskeja voidaan käyttää esimerkiksi läpinäkyvyys maskina, jolloin saadaan aikaan dynaamisia läpinäkyvyysmuutoksia, tai värien sekoittamiseen ja materiaalien yhdistämiseen.

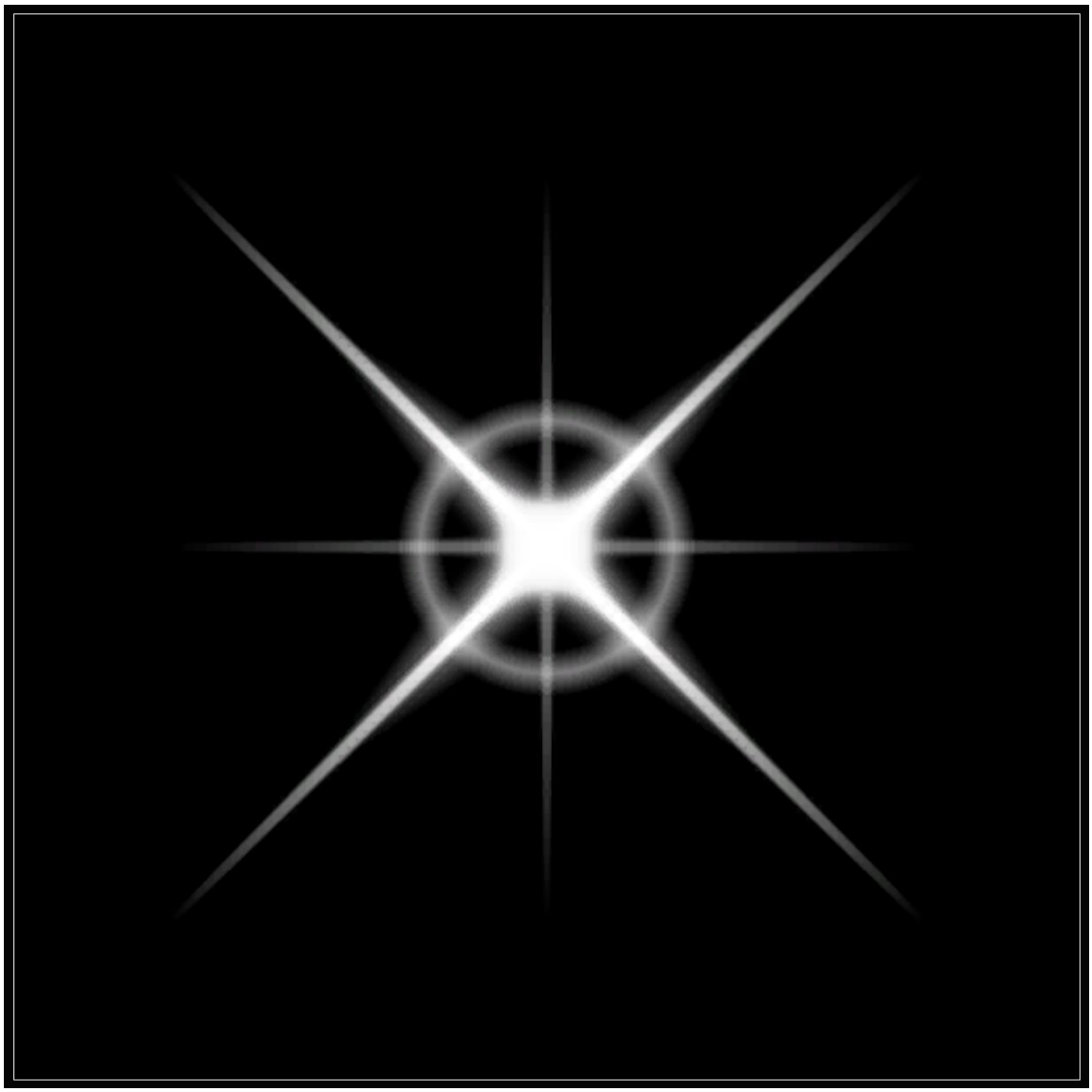


Kuva 14. Gradient maski (Filipp, 2025)

Sprite tekstuuri

Sprite on suosittu valinta videopeleissä, koska sitä on helppo animoida ja muokata. Ne ovat myös suhteellisen pieniä tiedostokooltaan, mikä tekee niistä ihanteellisia peleihin, jotka vaativat nopeita latausaikoja. Spritejä on käytetty videopeleissä jo vuosikymmenten ajan, ja niiden käyttö jatkuu todennäköisesti edelleen tulevaisuudessa. (Mad VFX, viitattu 2025)

Sprite tekstuurit ovat erityisen hyödyllisiä 3D efekteissä ja animaatioissa. Niiden alfa kanavaa voidaan käyttää maskina määrittämään, mitkä osat Spriteistä ovat näkyviä. Unreal Enginessä on mahdollista käyttää mukautettuja Sprite materiaaleja, joissa yhdistetään useita tekstuureja tai maskeja luomaan monimutkaisempia efektejä. Kuvassa 15 on esitelty tähti Sprite tekstuuri.



Kuva 15. Sprite tekstuuri (Filipp, 2025)

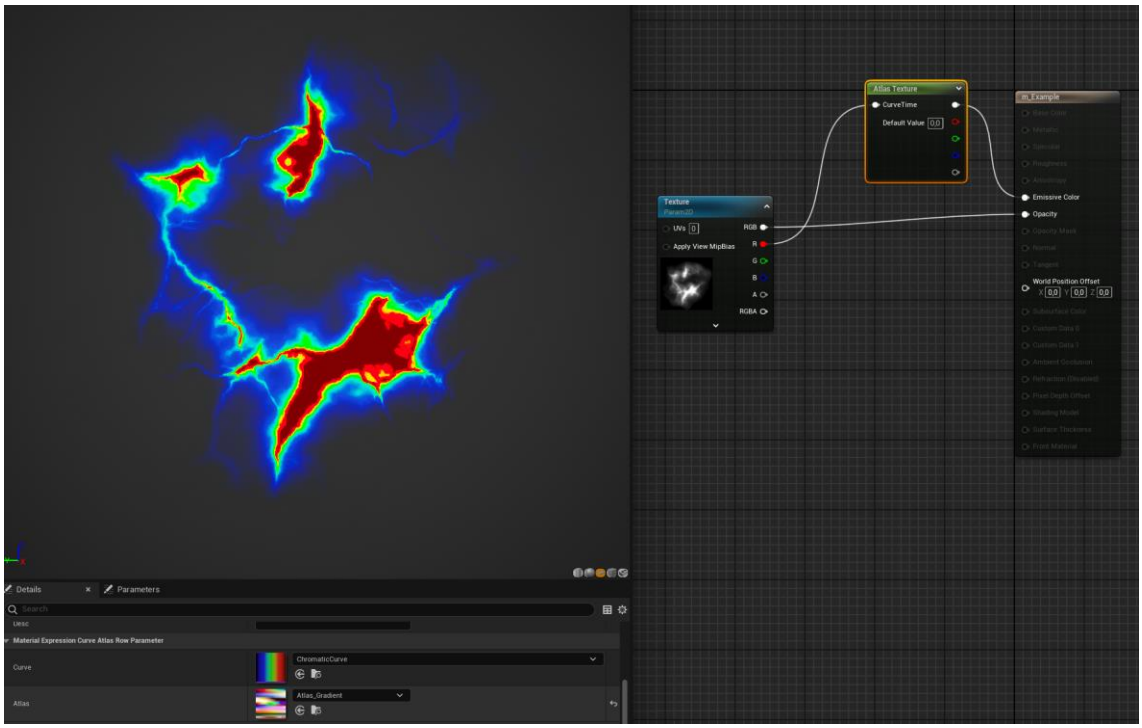
Väriliukukartoitus

Väriliukukartoitus tarkoittaa harmaasävykuvan sävyjen muuntamista värillisen liukuvärin mukaisiksi. Tämä tekniikka on erityisen hyödyllinen tyyliteltyjen efektien, kuten lämpökarttojen, maagisten hehkujen tai animoitujen värimuutosten luomisessa. Koska käytetään vain harmaasävykuvia, voidaan säästää tekstuurimuistia pakkaamalla useita mustavalkoisia kuvia yhteen tekstuuriin. Lisäksi värien vaihtaminen on joustavaa ja nopeaa, sillä liukuvärin muuttaminen vaikuttaa välittömästi kaikkiin siihen perustuviin efekteihin. Gradient värikartoitus on myös proseduraalinen, joten shader muokkaukset voivat vaikuttaa väreihin, mikä

mahdollistaa dynaamisten ja muuttuvien efektien luomisen. Tämä tekee siitä tehokkaan ja joustavan tavan lisätä visuaalista monimuotoisuutta ja tyylikkyyttä taiteeseen. (Epic games, Unreal Engine Documentation, 2025.)

Kun käytetään mustavalkoista kuvaa ja värillistä liukuväriä, kuvan sävyt muunnetaan vastaaviksi väreiksi annetun liukuvärin mukaisesti. Toisin sanoen, jokainen mustavalkoisen kuvan sävy korvataan liukuvärin vastaavalla värillä. (Tobias Noller 2025.)

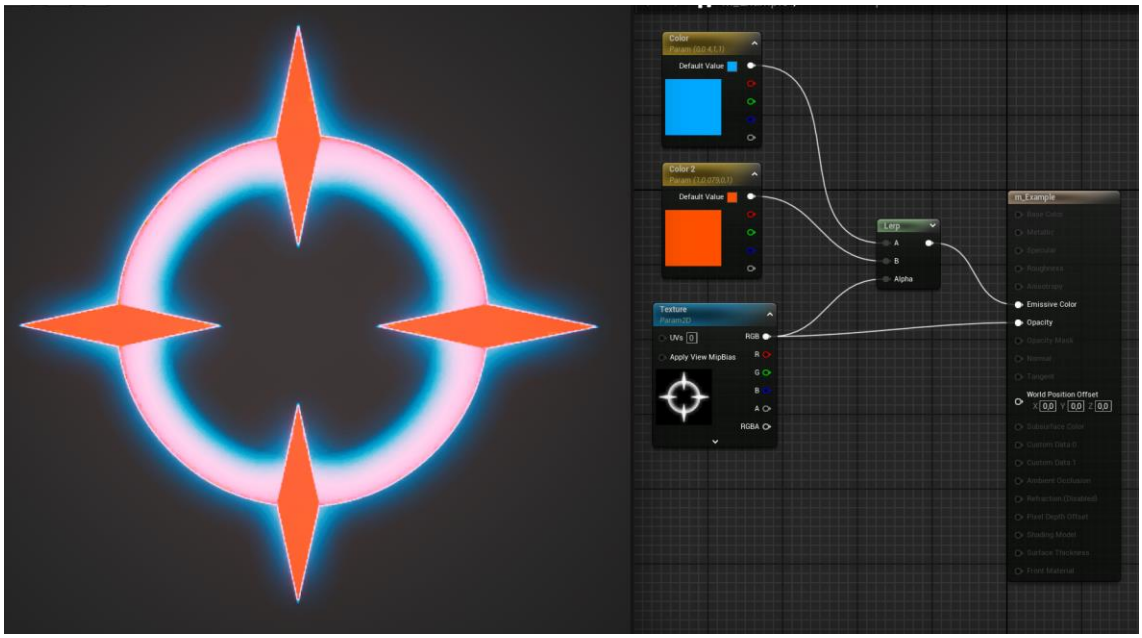
Esimerkki värikartoituksesta on esitelty kuvassa 16.



Kuva 16. Väriliukukartoitus (Filipp, 2025)

Lineaarinen interpolointi

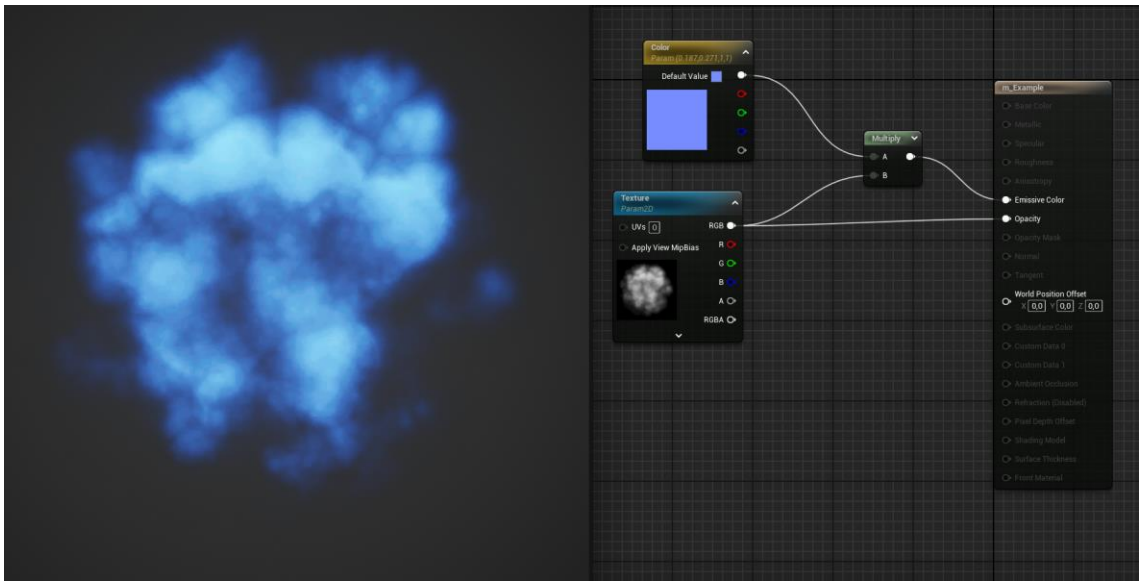
Color Lerp (lineaarinen interpolointi) mahdollistaa kahden värin sulavan siirtymisen toisiinsa määritetyn arvon perusteella. Tämä tekniikka on hyödyllinen efekteissä, jotka vaativat asteittaisia värimuutoksia, kuten häivytyksissä tai sykkivissä valoissa. Materiaalieditorissa ”Lerp” Node yhdistää kaksi väriä annetun alpha arvon mukaan, jolloin saadaan aikaan tasainen värinsiirtymä (Kuva 17). (PrismaticaDev 2022).



Kuva 17. Lineaarinen interpolointi käytössä (Filipp, 2025)

Värin kertominen

Tekstuurin väriä voidaan säätää kertomalla sen väriarvot tietyllä värivektorilla Multiply (Kertolasku) noden avulla (Kuva 18). Tämä menetelmä on yksinkertainen ja tehokas tapa sävyttää tekstuuri halutulla värillä, mikä mahdollistaa yhtenäisten värimuutosten tekemisen koko efektiin. Esimerkiksi kohinatekstuurin sävyttämiseksi voit liittää tekstuurin Multiply noden toiseen syötteeseen ja halutun värin toiseen syötteeseen, ja ohjata tuloksen materiaalin Base Color tuloon. Tämä lähestymistapa on kevyt ja mahdollistaa nopean värien vaihtamisen ilman uusien tekstuurien luomista.



Kuva 18. Värin kertaus (Filipp, 2025)

3.6 World Position Offset (WPO)

World Position Offset (WPO) on materiaalieditorin ominaisuus, jonka avulla voidaan siirtää mallin verteksiä maailman koordinaatistossa tai pinnan normaalien suuntaisesti. Tämä mahdollistaa esimerkiksi kasvillisuuden, kankaan tai nesteiden animoimisen ilman raskaita luuranko tai fysiikka animaatioita. WPO:n avulla voidaan luoda kevyitä ja dynaamisia efektejä, kuten tuulessa huojuvia puita tai aaltoilevaa vettä, manipuloimalla verteksiä shaderin tasolla. Esimerkiksi SimpleGrassWind funktiolla voidaan lisätä yksinkertainen tuulianimaatio kasvillisuuteen säätämällä tuulen voimakkuutta ja nopeutta. Tämä lähestymistapa on tehokas ja suorituskykyinen tapa lisätä liikettä ja elävyyttä ympäristöihin ilman monimutkaisia animaatioita. (Epic games, Unreal Engine Documentation, 2025.)

3.7 Muut menetelmät

Flow Maps: Flow mapit ovat tekstuureja, jotka ohjaavat tekstuurikoordinaattien siirtymistä pinnalla, simuloiden virtausta. Ne ovat erityisen hyödyllisiä luomaan realistisia efektejä, kuten virtaavia jokia, laavavirtoja tai energiakenttiä. Flow mapit tarjoavat tarkempaa hallintaa verrattuna perinteisiin Panner nodeihin, mahdollistaen monimutkaisempien virtausten simuloimisen.

Vertex Painting: Verteksiväritystekniikalla taiteilijat voivat maalata painoarvoja suoraan meshien verteksiin, vaikuttaen materiaalien käyttäytymiseen paikallisesti. Tämä mahdollistaa esimerkiksi eri materiaalien, kuten ruohon, mudan ja kallion, saumattoman yhdistämisen samassa meshissä ilman erillisiä UV karttoja.

RVT (Runtime Virtual Texture): RVT mahdollistaa materiaalien ja tekstuurien dynaamisen yhdistämisen ja jakamisen eri objektien välillä reaaliajassa. Se on erityisen hyödyllinen suurissa maisemissa, joissa tarvitaan tehokasta ja laadukasta materiaalien yhdistämistä ilman suorituskyvyn heikkenemistä. RVT:llä pystyy tehdä polkuja ympäristössä tai yhdistää objektien materiaalit maan materiaaliin kanssa.

Tesselation: Tessellointi lisää geometrian yksityiskohtia jakamalla pintoja pienempiin osiin, mahdollistaen realistisemmat pinnanmuodot. Unreal Engine 5:ssä perinteinen tessellointi on korvattu Nanite teknologialla, joka mahdollistaa erittäin yksityiskohtaisen geometrian renderöinnin tehokkaasti.

(PrismaticaDev 2020–2025)

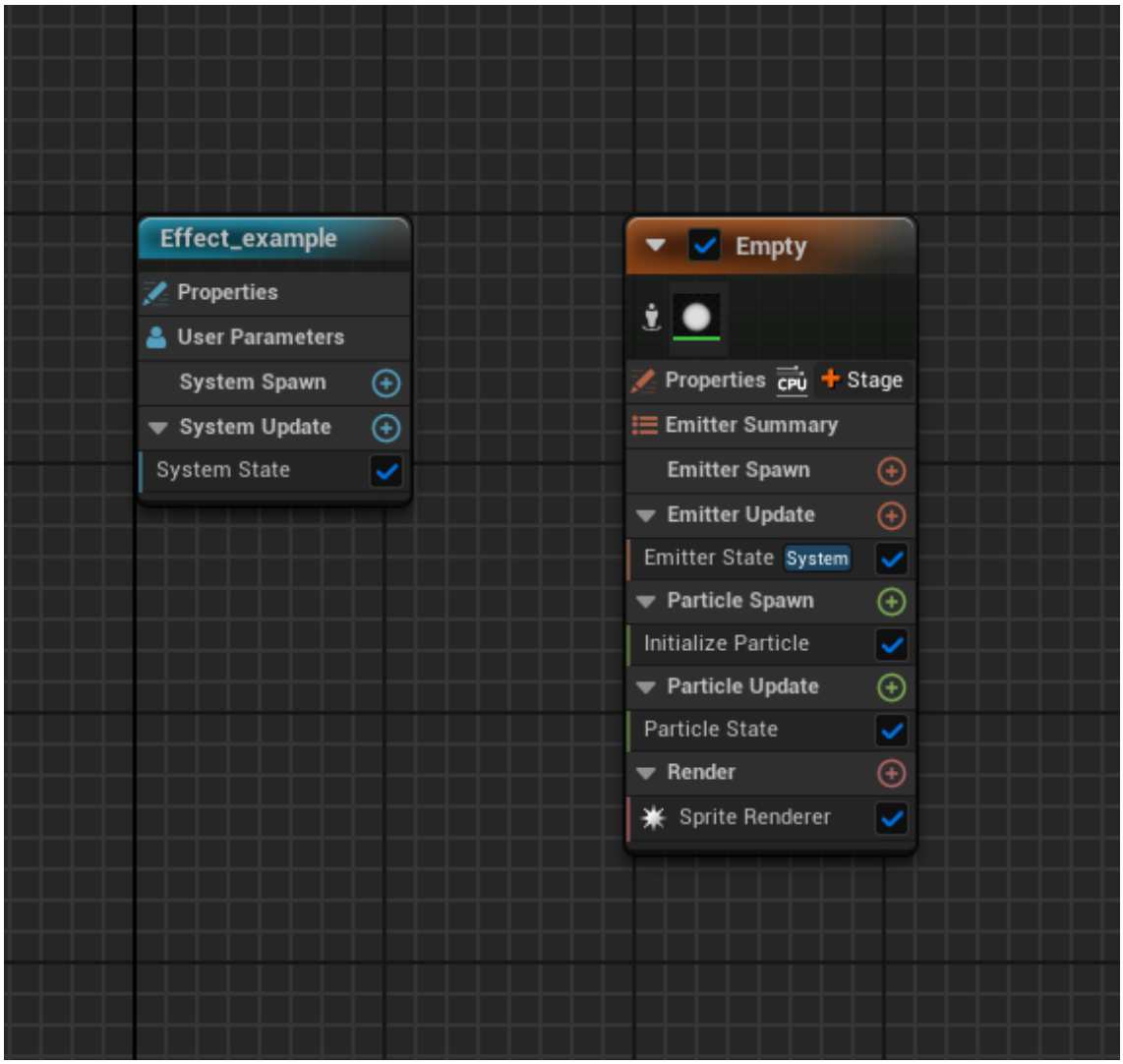
Tässä osiossa oli käsitelty erilaiset menetelmät ja tavat luoda materiaalit Unreal Engine:ssa, seuraavassa osassa kerrotaan Niagara partikkelimoottorista.

3.8 Partikkelimoottorit ja Niagara

Niagara on Unreal Enginen seuraavan sukupolven VFX järjestelmä, joka yhdistää joustavan, nodepohjaisen käyttöliittymän partikkeliefektien luomiseen ja teknisen taiteilijan mahdollisuuden laajentaa järjestelmää ilman ohjelmoijan apua. Järjestelmä on suunniteltu modulaariseksi ja uudelleenkäytettäväksi, mikä tehostaa efektien rakentamista ja ylläpitoa. (Niagara Overview, Unreal Engine Documentation, 2025).

Niagara tarjoaa reaaliaikaisen esikatselun ja tiiviin integraation Blueprints työkaluihin, minkä ansiosta kehittäjät voivat sitoa partikkelisysteemejä suoraan pelilogiikkaan

Kuvassa 19 on esitelty tyhjä partikkelisysteemi. Jokainen partikkelisysteemi koostuu yleiskatsaus nodesta ja hiukkaslähteestä. Yleiskatsausnode tarjoaa korkean tason asetukset ja emitter node (hiukkaslähde) määrittelee partikkelien syntymisen, elinkaaren ja käyttäytymisen. (Niagara Overview, Unreal Engine Documentation, 2025). Esimerkiksi nuotion efektissä voi olla erilliset emitterit liekeille, savulle ja kipinöille.



Kuva 19. Tyhjä partikkelisysteemi (Filipp, 2025)

Jokaisessa hiukkaslähteessä on kuusi moduulia: Emitter spawn (hiukkaslähteen luonti), Emitter update (hiukkaslähteen päivitys), Particle spawn (partikkelin luonti), Particle update (partikkelin päivitys), Event handler (tapahtumakäsittelijä) ja Render (Renderöinti). (Niagara Overview, Unreal Engine Documentation, 2025).

Emitter Spawn

Tämä moduuli määrittelee mitä tapahtuu, kun hiukkaslähteen luodaan ensimmäisen kerran. Tätä moduuli on tarkoitettu alkuperäisten asetusten ja oletusarvojen määrittämiseen. (Niagara Overview, Unreal Engine Documentation, 2025).

Emitter Update

Emitter Update moduuli vastaa siitä, mitä tapahtuu joka kehyksellä (frame) emitter tason logiikassa CPU:lla. Näissä moduuleissa määritellään esimerkiksi partikkelien syntymistiheys, elinkaaren hallinta ja muut emitterin tilaan liittyvät muutokset, jotka toistuvat jokaisella päivityksellä. (Niagara Overview, Unreal Engine Documentation, 2025).

Particle Spawn

Tätä moduulia kutsutaan kerran per partikkeli, kun kyseinen partikkeli syntyy ensimmäisen kerran. Tässä määritellään partikkelien alustamiseen liittyviä yksityiskohtia, kuten niiden syntymispaikka, väri, koko ja muut ominaisuudet. (Niagara Overview, Unreal Engine Documentation, 2025).

Particle Update

Tämä moduuli kutsuttu jokaiselle partikkelille jokaisella framella. Siinä määritellään asiat, jotka tarvitsevat muuttua kerrallaan partikkelien vanhetessa. Esimerkiksi, jos partikkelien väri muuttuu ajan myötä, tai jos partikkelit ovat vaikutuksen alaisina voimille kuten painovoima, pyörrevirta (curl noise) tai pisteeseen vetäytyminen. Myös pystyy määrittää partikkeleiden koon muuttuvan ajan kuluessa ja muut parametrit. (Niagara Overview, Unreal Engine Documentation, 2025).

Event Handler

Tapahtumankäsittelijämoduulissa pystyy luoda yhdessä tai useammassa emittorissa tapahtumia, jotka määrittelevät tiettyjä tietoja. Sen jälkeen voi luoda toisissa emittereissa tapahtumia, jotka käynnistävät toiminnon vastauksena tuohon generoituun tapahtumaan. (Niagara Overview, Unreal Engine Documentation, 2025).

Render

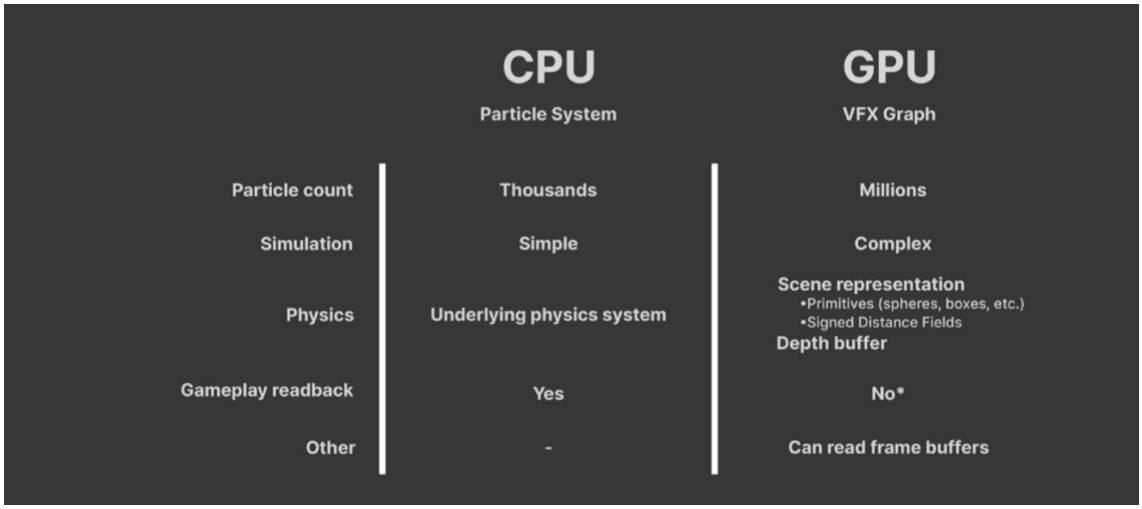
Viimeinen moduuli on Renderöintimoduuli. Siinä määritellään partikkelien näkyvyys ja luodaan yksi tai useampi renderöijä partikkeleille. Oletusvaihtoehto on sprite renderöijä. Se määritellee partikkelit kaksiulotteisina Sprite objekteina. Mesh renderöijää voi käyttää, jos tarkoituksena on määritellä kolmiulotteisen mallin partikkelisi pohjaksi, johon voit soveltaa materiaalia. Käytettävissä on monia erilaisia renderöintityyppejä, kuten Decal, Light tai Ribbon.

Niagara järjestelmän luominen tapahtuu siten, että valitset hiiren oikealla painikkeella Content Browserissa FX > Niagara System ja valitset joko valmiin mallipohjan tai tyhjän järjestelmän. Kun avaat juuri luodun järjestelmän, se toimii konttina, johon voit lisätä yhden tai useamman emitterin joko valmiista malleista tai tyhjistä emittereistä.

Emittereiden sisällä partikkelien syntyminen, käyttäytyminen ja ulkoasu määritellään modulaarisesti. Jokaisella emitterillä on useita moduuliryhmiä, ja uusia moduuleja tuodaan klikkaamalla kunkin ryhmän Plus (+) kuvaketta. Näin voit muokata valmiita ominaisuuksia tai lisätä täysin uusia toimintoja, jolloin suurimmassa osassa tapauksista ei tarvita erillistä kustomoitua moduulisuunnittelua. (Niagara Overview, Unreal Engine Documentation, 2025).

3.9 CPU / GPU partikkelisysteemi

CPU partikkelit tukevat laajaa moduulivalikoimaa, mukaan lukien tapahtumat ja laajennettu fysiikka, mutta skaalautuvat tuhansiin partikkeleihin. GPU partikkelit pystyvät simuloimaan satojatuhansia partikkeleita tehokkaasti, mutta niiden kyvykkyydet ja moduulit ovat rajatummat, esim. tapahtumakäsittely ei toimi GPU simulaatiossa (Kuva 20). (Unity 2024).



Kuva 20. CPU ja GPU partikkelisysteemien vaikutus suorituskyvyn (Unity, 2024)

3.10 Blueprinttien käyttö materiaaleissa ja Niagarassa

Edellisessä osiossa oli käsitelty Niagara Partikkelimoottori, sen rakenne ja moduulit ja nyt siirrymme Blueprint järjestelmään.

Blueprintit mahdollistavat materiaalien ja Niagaran parametrien dynaamisen ohjauksen ilman koodausta. Materiaalien tapauksessa arvoja (esim. skaalaus, väri tai läpinäkyvyys) viedään materiaaliin skalaariparametreina tai vektoreina Material Parameter Collectionien tai dynaamisten materiaaliesiintymien kautta. Niagara järjestelmissä käyttäjän määrittämät muuttujat (User Parameters) asetetaan UNiagaraComponentin kautta Blueprint skripteissä, jolloin partikkelien syntymisnopeus, koko ja muut ominaisuudet mukautuvat pelitapahtumiin. Tällä tavalla samassa Blueprintissa voidaan synkronoida sekä materiaali, että partikkelimuutokset, mikä mahdollistaa yhtenäiset ja reagoivat efektit. (Epic games, Unreal Engine Documentation, 2025.)

Materiaaliparametrien ohjaus Blueprinteilla

Materiaaleihin sidottuja kenttiä voi ohjata dynaamisesti luomalla materiaalista Material Instance Dynamic esiintymän, jonka kautta muuttujat ja vektoriparametreja muokataan reaaliajassa. Vaihtoehtoisesti Material Parameter Collectionin avulla useat materiaalit voivat hyödyntää samaa parametriasettelua. Blueprintissa haetaan viittaus kyseiseen kokoelmaan ja kutsutaan Set Scalar Parameter Value tai Set Vector Parameter Value nodeja arvojen asettamiseksi. Näin esimerkiksi kilven hehkuvoimakkuutta tai väriä voi muuttaa pelaajan etäisyyden mukaan tai pelisignaalin perusteella ilman erillistä C++ koodia. (Epic games, Unreal Engine Documentation, 2025.)

Niagara parametrien ohjaus Blueprinteilla

Niagara hiukkaslähteessä on mahdollista määritellä “User” alkuisia muuttujia, joita Blueprintit voivat muuttaa UNiagaraComponentin Set Niagara Variable nodejen kautta. Käyttäjän määrittämä Float, Vector tai Object parametri luodaan emitterissä ja sijoitetaan haluttuun moduuliin, minkä jälkeen Blueprintissa kutsutaan vastaavaa SetNiagaraVariable funktiota, esimerkiksi Set Niagara Float Parameter nodea. Tällä tavalla partikkelien syntymisnopeus, koko tai väri voidaan säätää suoraan pelitapahtumien perusteella, kuten räjähdyksen voimakkuuden kasvaessa tai tulipalon liekkien laajentuessa. (Epic games, Unreal Engine Documentation, 2025.)

Materiaalien ja Niagara efektien synkronointi

Kun materiaali ja Niagara parametrien ohjaus on perustettu, Blueprintissa voidaan yhdistää molempien järjestelmien päivitys samaan logiikkaan. Esimerkiksi pelaajan lähestyessä tiettyä kohdetta Blueprint voi kutsua sekä Set Scalar Parameter Value nodea emissio parametrille, että Spawn Emitter at Location nodea räjähdysefektille. Lisäksi Blueprintin Timeline node ja Lerp funktio voivat interpoloinnin avulla yhdenmukaistaa materiaalin värin tai hehkuarvon ja partikkelien syntymistiheyden ajallisesti, jolloin visuaalisesti yhtenäinen ja dynaaminen lopputulos saavutetaan saumattomasti. (Epic games, Unreal Engine Documentation, 2025.). Tämä menetelmä on käsitelty Esimerkkityössä 4.

3.11 Optimisaatio

Aikaisemmissa osioissa käsiteltiin Materiaalit, Niagaraa ja Blueprintit, tämän luvun viimeinen aihe on optimisaatio, mikä on erittäin tärkeä VFX:ssä.

Materiaalien ja partikkelijärjestelmien optimointi Unreal Enginessä on ratkaisevan tärkeää sujuvan pelisuorituskyvyn saavuttamiseksi.
Aloittelijoille on tärkeää ymmärtää, että useiden läpinäkyvien kerrosten käyttäminen materiaaleissa voi lisätä ylipiirtoa (overdraw), mikä kuormittaa GPU:ta

huomattavasti. Liialliset piirtokutsut usein seurausta monista uniikeista materiaaleista tai monimutkaisista partikkelijärjestelmistä voivat myös hidastaa pelin suorituskykyä. Muita mahdollisia ongelmia ovat monimutkaiset shader instruktiot, jotka vaativat ylimääräistä laskentatehoa, useat tekstuurihakut, jotka kuormittavat muistiväylää, sekä dynaaminen partikkelikäyttäytyminen, joka aiheuttaa korkeita laskentakustannuksia. Yksinkertaistamalla materiaalien rakennetta (esimerkiksi vähentämällä läpinäkyvyyden kerroksia tai yhdistämällä samankaltaisia materiaaleja), käyttämällä partikkelijärjestelmissä Level of Detail (LOD) tekniikkaa ja ottamalla käyttöön tehokkaita karsintastrategioita, pystyy tasapainottaa efektien visuaalista laatua ja suorituskykyä. (Chris McCole 2023).

3.12 Yhteenveto

Tässä osassa tarkasteltiin pelien VFX:n teoreettisia perustoja ja Unreal Enginen tarjoamia toteutusmahdollisuuksia. Materiaali ja shader tekniikat, kuten UV manipulointi, tekstuurijärjestelyt, maskit ja World Position Offset (WPO), muodostavat efektien visuaalisen ytimen. Lisäksi käsiteltiin eri tekstuurikanavia (Color, Emissive, Opacity, Displacement) ja niiden roolia materiaalieditorissa. Niagara partikkelijärjestelmän modulaarinen rakenne ja sen kuusi päämoduuliryhmää selkeyttävät partikkeliefektien syntyä ja päivitystä, kun taas Blueprint integraatio mahdollistaa materiaalien ja Niagara parametrien dynaamisen ohjauksen pelilogiikan kautta. Lopuksi optimointi ja suorituskykyvinkit osoittavat, miten visuaalinen laatu ja pelin suorituskyky saadaan sopusointuun käytännön konfiguraatioilla. Tämä kokonaisuus luo vankan pohjan käytännön esimerkeille, joissa teoria siirtyy suoraan realistisiksi, reaaliaikaisiksi efekteiksi Unreal Enginessä.

4 Käytännön osio

Luvuissa 7 ja 8 käsiteltiin VFX:n teoriaa ja hyödylliset menetelmät efektien luomisessa Niagarassa ja materiaaleissa, tässä luvussa käytetään nämä menetelmät esimerkkiefektien luomiseen.

Käytännön osiossa toteutan viisi esimerkkiefektiä, jotka soveltavat teoriaosiossa esiteltyjä menetelmiä. Jokainen esimerkki jatkaa edellisen pohjalta lisäämällä asteittain monimutkaisempia tekniikoita, jotta nähdään, miten eri työkalut ja työnkulut yhdistyvät näyttäviin lopputuloksiin. Ensimmäisessä esimerkkityössä käyn yksityiskohtaisesti läpi kaikki efektin luomisvaiheet, ja myöhemmissä esimerkeissä esittelen valmiit noderakenteet ja asetukset. Opastus hyödyntää erilaisia resursseja, kuten Screaming Brain Studio:n ilmaista kohinatekstuuripakettia sekä Kenney:n Sprite tekstuuripakettia, varmistaen selkeän ja toistettavan oppimiskokemuksen.

Esimerkkiefektit ja menetelmät mitkä ovat käsitelty niissä

- Häikäisevä valo
  Radiaalinen UV kartoitus ja säteittäinen gradienttimaski
  Tekstuurin toisto ja vieritys omilla Tiling ja Panner materiaalifunktioilla
  Saumojen korjaus DDX/DDY derivaatioilla
  Värinvaihto Hue Shift nodella

- Tulijälki
  Gradienttikartoitus AtlasTexture nodella
  Maskin hienosäätö SmoothStep funktiolla
  Dynaaminen emissio, erode ja opacity dynaamisten materiaaliparametrien (User Parameters) avulla
  Material Instance Dynamic instanssit nopeaan iterointiin
  Niagara Ribbon Renderer

- Ribbon jälki
  3D ruohomeshin mallintaminen
  Verteksivärigradientti tuulen maskina
  World Position Offset dynaamisella tuuliefektillä

- Per instance satunnaistettu sävymuutos ja world space värivariaatio
  Etäisyyteen perustuva Dither Fade optimointina
  Material Parameter Collection globaalien parametrien hallintaan
  Reroute nodejen käyttö nodeverkoston selkeyttämiseen

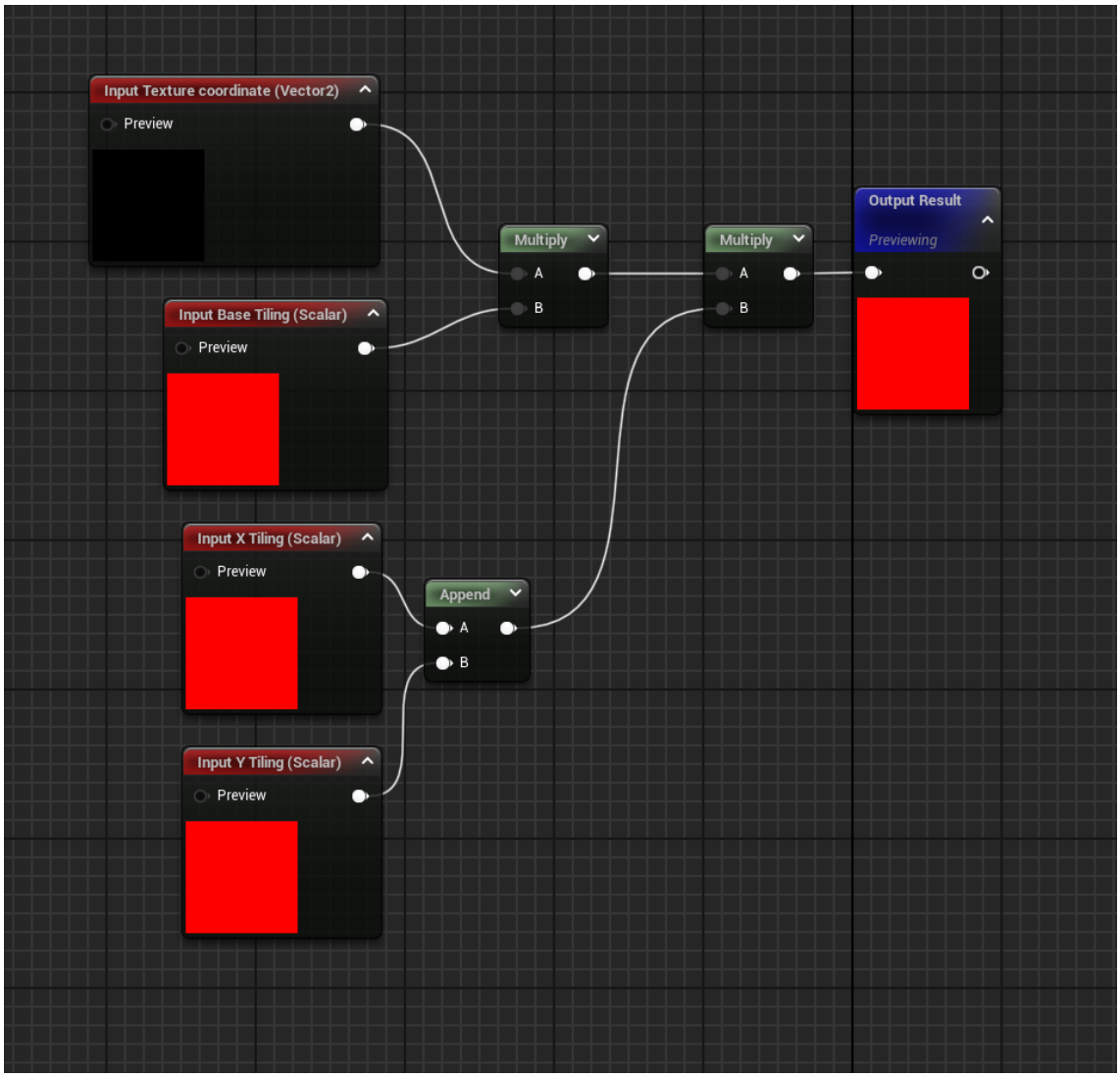
- Tyylitelty räjähdys
  Niagara emitterin kehittynyt partikkelikäyttäytyminen
  Event Handler moduulit interaktiivisiin tapahtumiin
  Blueprint ohjattu materiaaliparametrien animaatio
  Post Process materiaalin käyttö
  Tyylitellyt Sprite partikkelimateriaali
  Niagara Mesh Renderer moduulin käyttö

4.1 Materiaalifunktiot

Seuraava tärkeä menetelmä on materiaalifunktio, tässä osiossa kerrotaan mitä se on ja miten materiaalifunktion voi käyttää, se lisäksi tehdään kaksi materiaalifunktiota, mitkä käytetään esimerkki efekteissä.

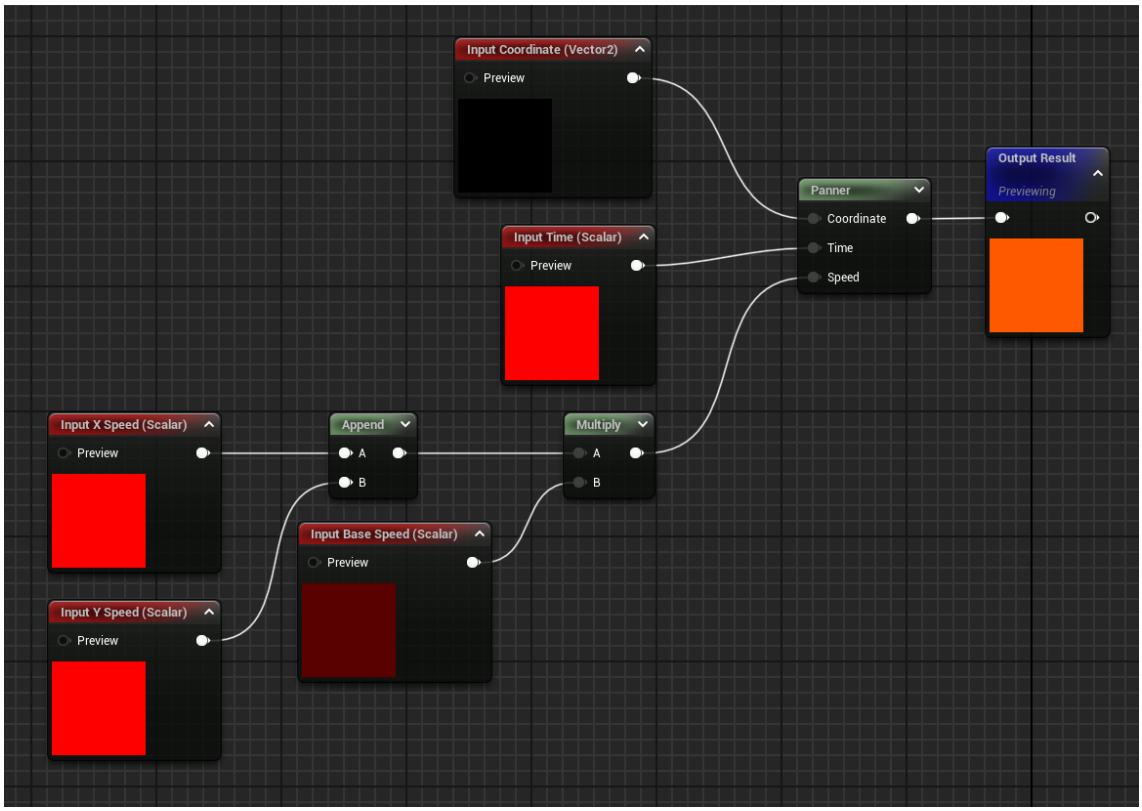
Materiaalifunktiot ovat uudelleenkäytettäviä nodepohjaisia loogisia kokonaisuuksia, jotka kapseloidaan erillisiksi asseteiksi ja joita on helppo liittää muihin materiaaleihin. Ne alentavat monistuvaa työtä, sillä samaa logiikkaa voi käyttää useissa materiaaleissa ilman, että sitä tarvitsee rakentaa alusta asti jokaisessa erikseen. Materiaalifunktioihin tehdyt muutokset päivittyvät automaattisesti kaikkiin niitä hyödyntäviin materiaaleihin, mikä parantaa ylläpidettävyyttä ja vähentää virheiden riskiä. Lisäksi materiaalifunktiot näkyvät omana, kirjastoituna kategorianaan Material Editorin Funktiokirjastossa, mikä nopeuttaa niiden löytämistä ja käyttöä. (Epic Games, Unreal Engine Documentation, 2025). Esimerkkitöitä varten minä tein kaksi materiaalifunktiota: laatoitusfunktio (Kuva 21) ja vieritysfunktio (Kuva 22).

Laatoitusmateriaalifunktio säätää tekstuurin toistuvuutta kertomalla tai jakamalla UV koordinaatteja, jolloin sama tekstuuri toistuu useampaan kertaan pinnalla. Tämä mahdollistaa esimerkiksi tiiliseinien tai ruudukkomallien luomisen ilman, että tarvitsee kasvattaa tekstuurin resoluutiota, ja toimii siten kevyenä tapana lisätä pintaan yksityiskohtia.



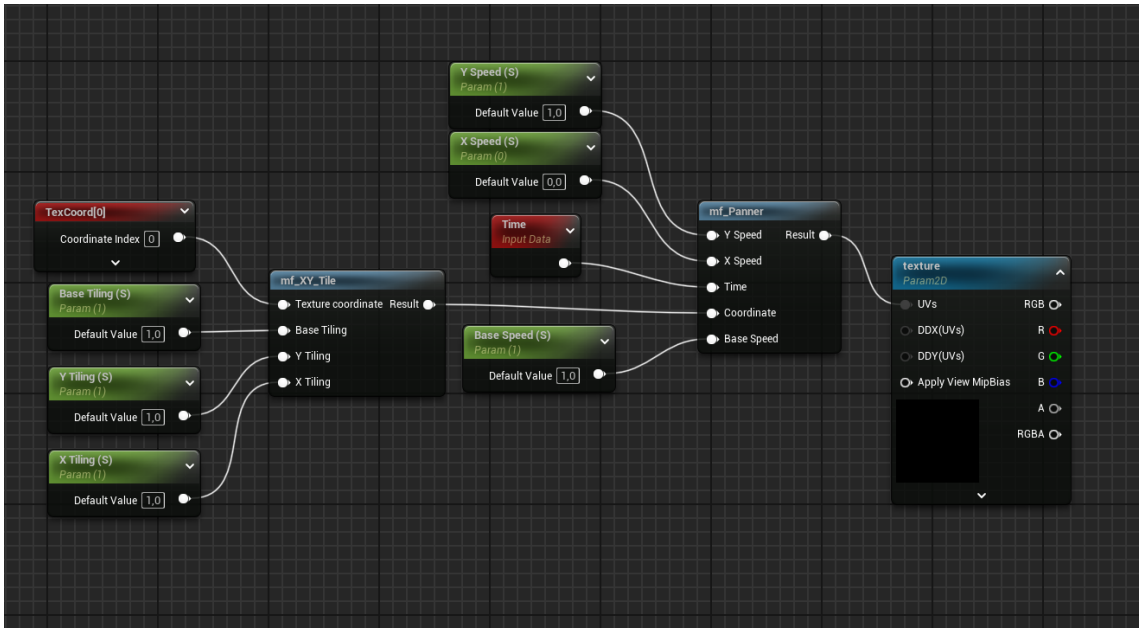
Kuva 21. Tekstuurin laatoitusmateriaalifunktio (Filipp, 2025)

Vieritysmateriaalifunktio siirtää UV koordinaatteja ajan suhteen, jolloin tekstuurit liikkuvat haluttuun suuntaan sujuvasti ja jatkuvasti. Funktio ottaa vastaan Time arvon ja nopeusparametrit, joilla ohjataan tekstuurin vierityksen suuntaa ja vauhtia, ja tuottaa päivitetyt UV koordinaatit kullekin kehykselle.



Kuva 22. Tekstuurin vieritysmateriaalifunktio (Filipp, 2025)

Alla näkyy laatoitus ja vieritys materiaalifunktioiden käyttö materiaalissa (Kuva 23).

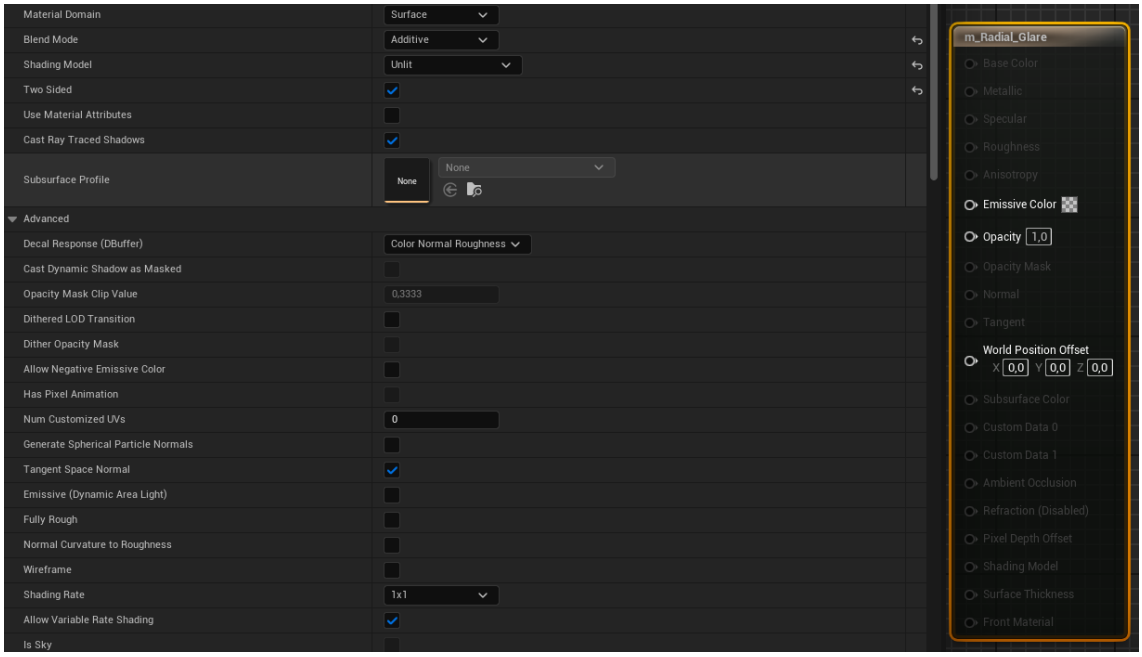


Kuva 23. Materiaalifunktioiden käyttö materiaalissa (Filipp, 2025)

4.2 Esimerkkiefekti 1 – Häikäisevä valo

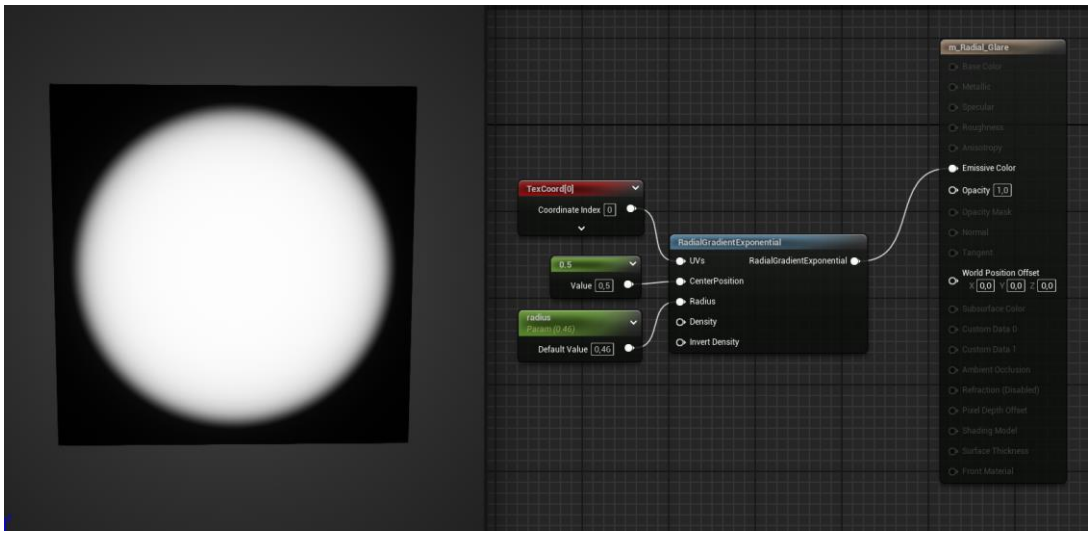
Tämän luvun alussa kerrottiin materiaaligraphin pikanäppäimistä ja materiaalifunktioista, nyt käytetään tätä teoriaa esimerkkiefektien luomiseen.

Ensimmäisessä esimerkissä luodaan säteittäinen hehkuefekti materiaalieditorissa. Prosessi alkaa uuden materiaalin luonnilla Content Browserissa ja sen avauksella. Materiaalin asetuksiksi vaihdetaan Additive blend moden ja Unlit valaistusmalli, se näkyy kuvassa 24.

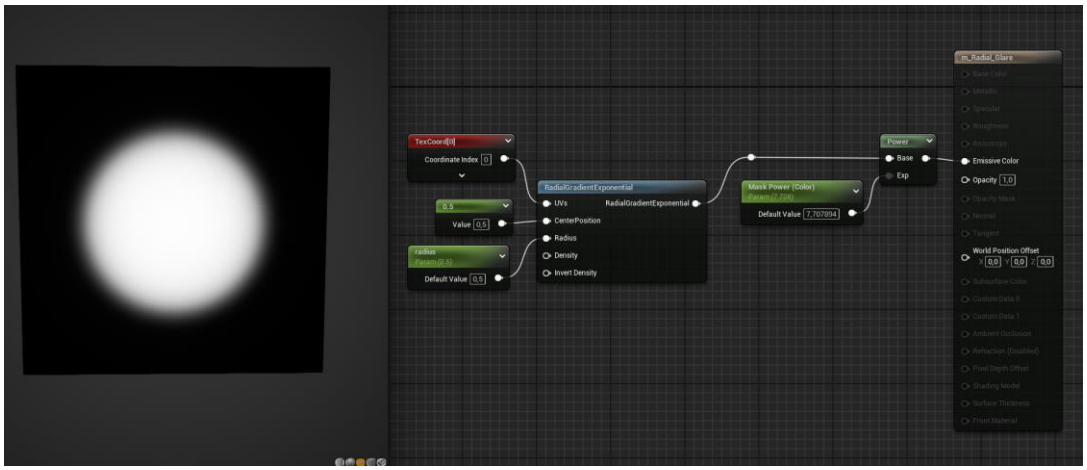


Kuva 24. Materiaalinoden asetukset (Filipp, 2025)

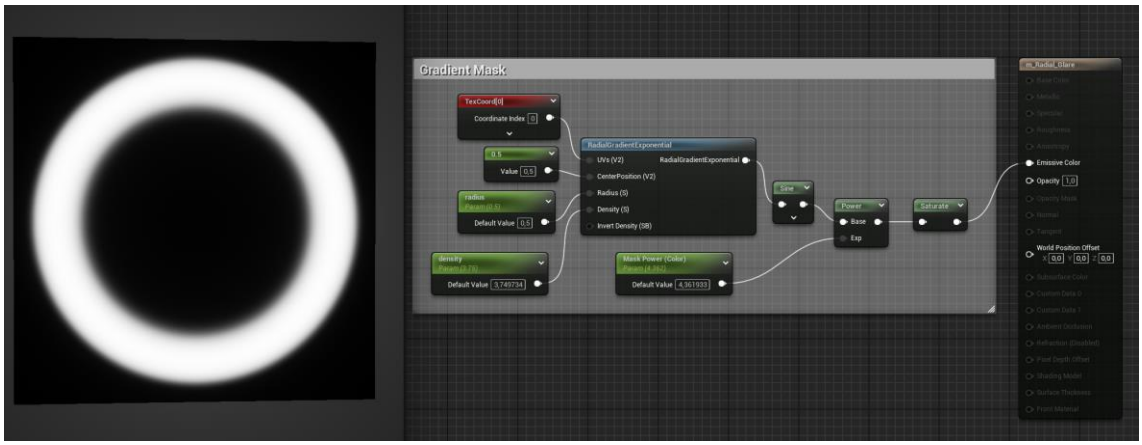
Aluksi luon pallogradienttimaski käyttämällä RadialGradientExponential nodea (Kuva 25) ja säädän sen kaaren jyrkkyyttä Power nodella (Kuva 26). Tämän jälkeen yhdistän Sine ja Saturate nodet gradientin muuntamiseksi rengasmuotoiseksi maskiksi, se näkyy kuvassa 27.



Kuva 25. RadialGradientExponential node (Filipp, 2025)

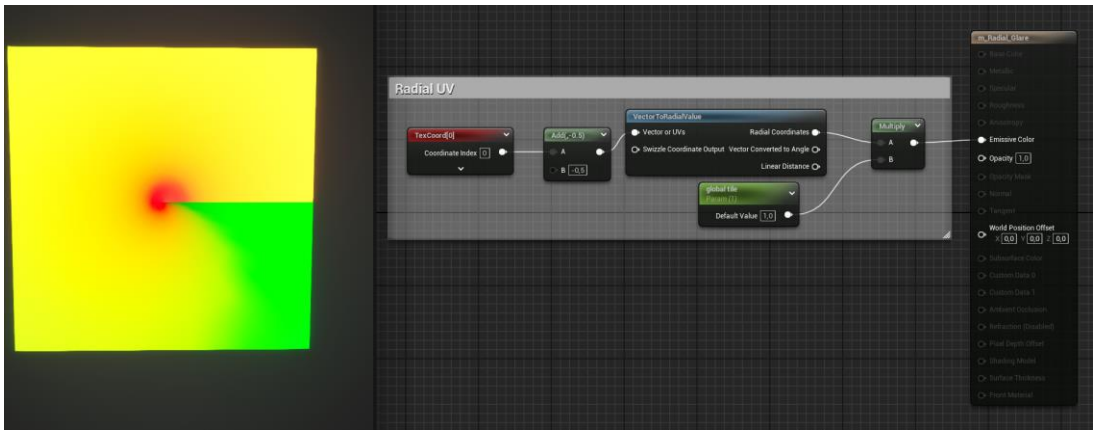


Kuva 26. Litätty Power node (Filipp, 2025)



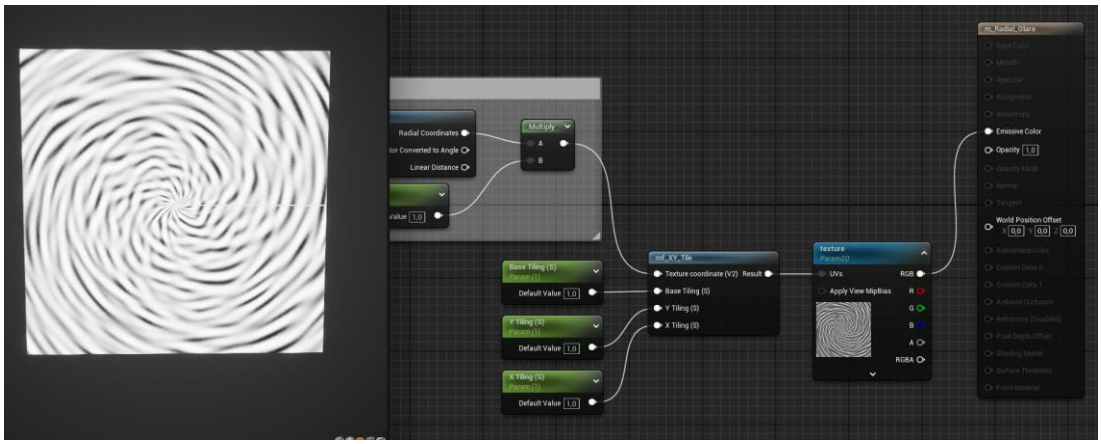
Kuva 27. Sinimaski (Filipp, 2025)

Kun maski on valmis, UV koordinaatit muunnetaan Polar muotoon VectorToRadial noden avulla, jotta tekstuurin vieritys alkaa säteittäin keskipisteestä (Kuva 28).

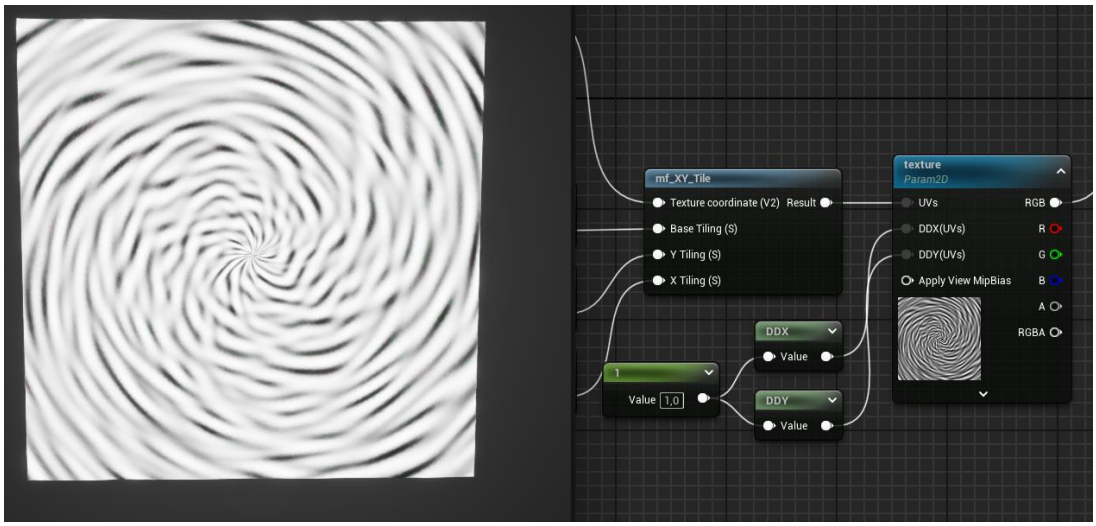


Kuva 28. Radiaalinen UV kartoitus materiaalissa (Filipp, 2025)

Varsinainen tekstuurinäyte lisätään TextureSample nodella. Ennen vieritystä tekstuurin toistoa säädetään omaa Tiling materiaalifunktiota hyödyntämällä, jonka jälkeen UV koordinaatit syötetään TextureSample nodeen UV tuloon (Kuva 29). Sivusaumana esiintyvän rajasaumailmiön korjaamiseksi lasketaan näytteen DDX ja DDY derivaatat ja asetetaan Mip Value Mode arvoksi Derivative (Kuva 30).

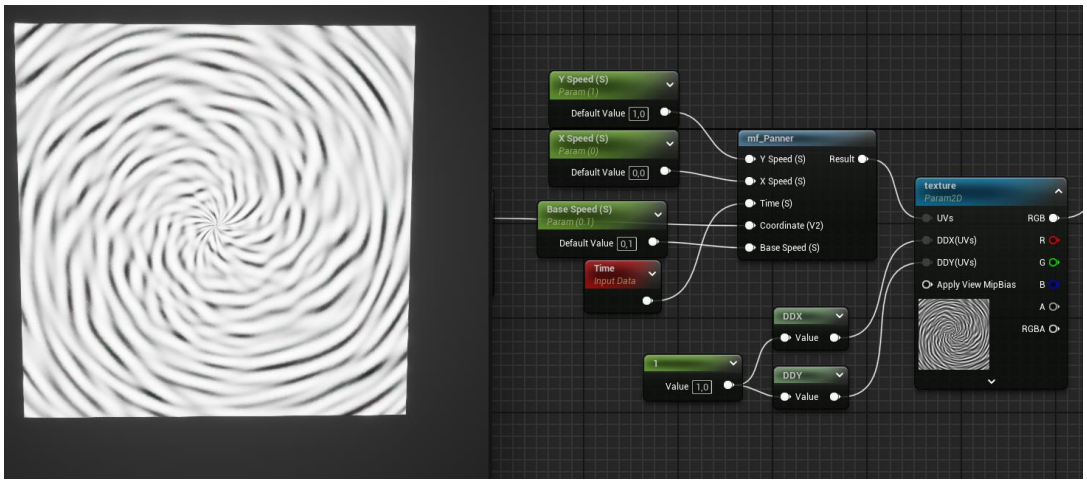


Kuva 29. Tekstuurin laatoitusfunktio materiaalissa (Filipp, 2025)

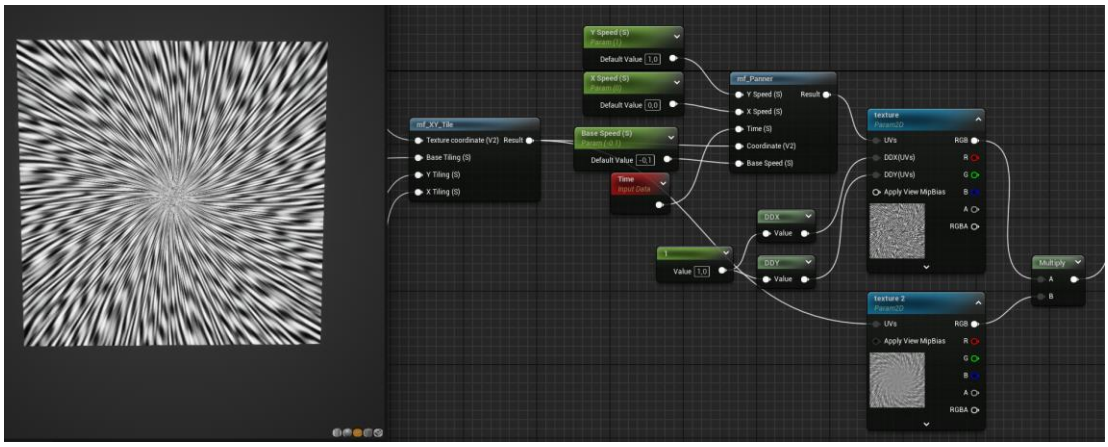


Kuva 30. Derivaatat DDX ja DDY materiaalissa (Filipp, 2025)

Dynaamisuutta lisätään Panner materiaalifunktion avulla, joka siirtää tekstuuria jatkuvasti haluttuun suuntaan (Kuva 31). Tarkempaa pintamallia varten liitetään vielä toinen staattinen tekstuurinäyte, ja molemmat näytteet kerrotaan keskenään Multiply nodella (Kuva 32).

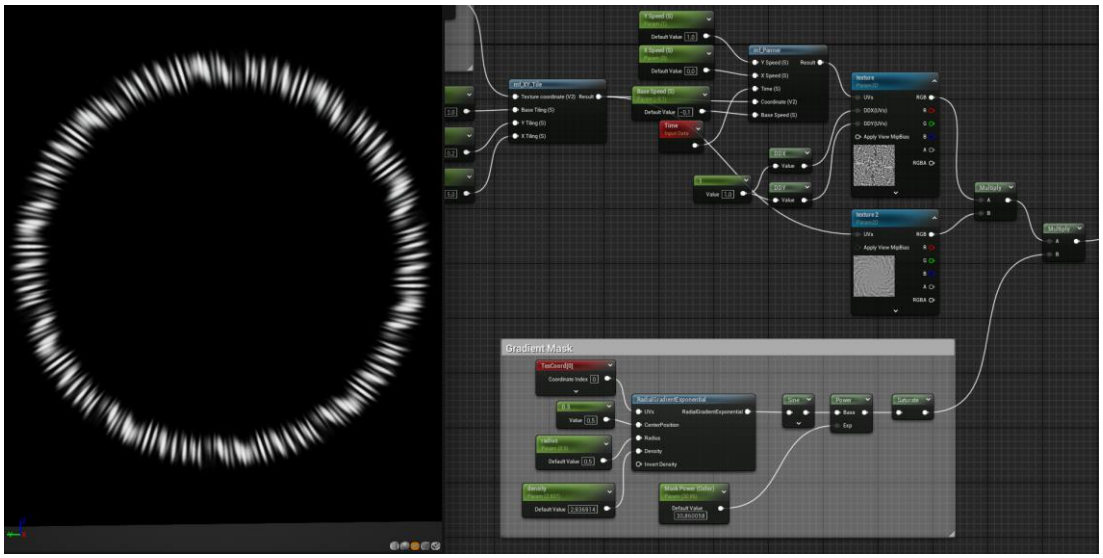


Kuva 31. Vieritysfunktion käyttö (Filipp, 2025)

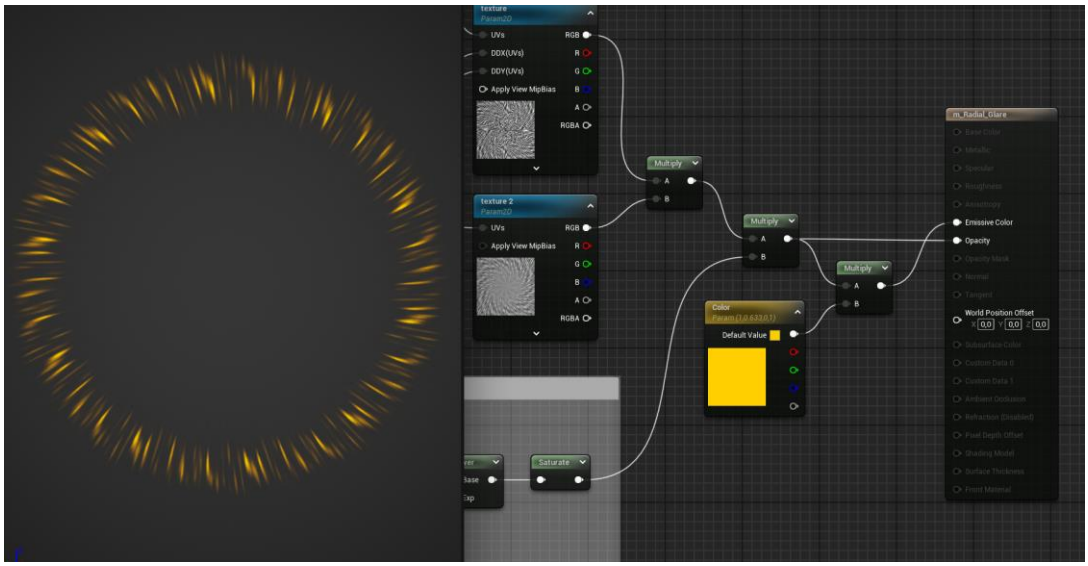


Kuva 32. Tekstuurit kerrottu (Filipp, 2025)

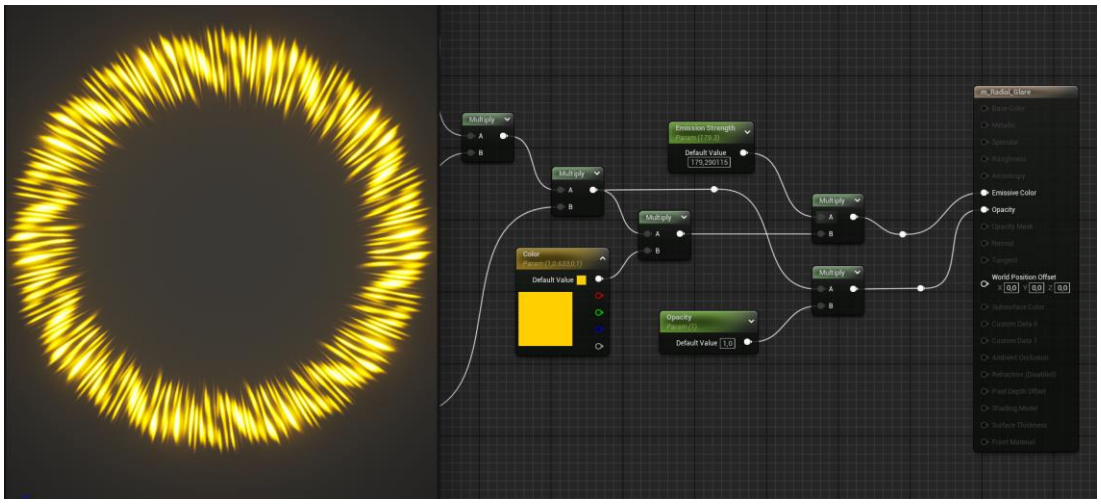
Seuraavaksi yhdistetään kerrotun tekstuurin tulos aiemmin luotuun rengasmaskiin, näin saadan kohdistettu, säteittäinen liike efekti (Kuva 33). Väritys hoidetaan kertomalla tämä tulos vektorikolmioarvolla (Vector 3), jonka jälkeen sama arvo syötetään sekä Emissive Color että Opacity tuloihin materiaalissa (Kuva 34). Lopuksi valmisteluparametreina lisätään kaksi skalaarista parametriä, joilla voidaan säätää hehkun voimakkuutta (emissio) ja läpinäkyvyyttä (opacity) ajon aikana (Kuva 35).



Kuva 33. Tekstuurin ja maskin kerronta (Filipp, 2025)

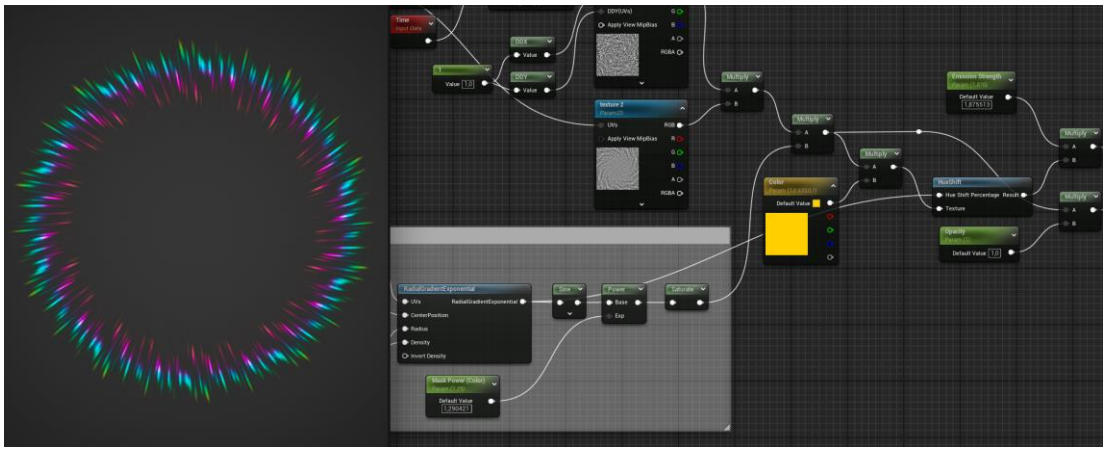


Kuva 34. (Filipp, 2025)



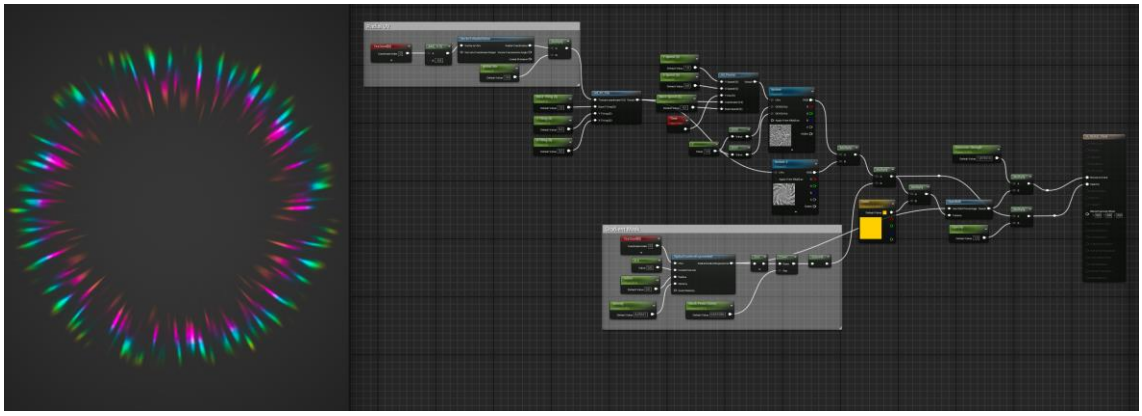
Kuva 35. (Filipp, 2025)

Erikoistehosteena toteutetaan vielä sateenkaarigradientti HueShift noden avulla: radiaalinen maski ohjaa Hue Shift Percentage tulon arvoa siten, että värisävy muuttuu lineaarisesti keskustasta reunoille (Kuva 36).



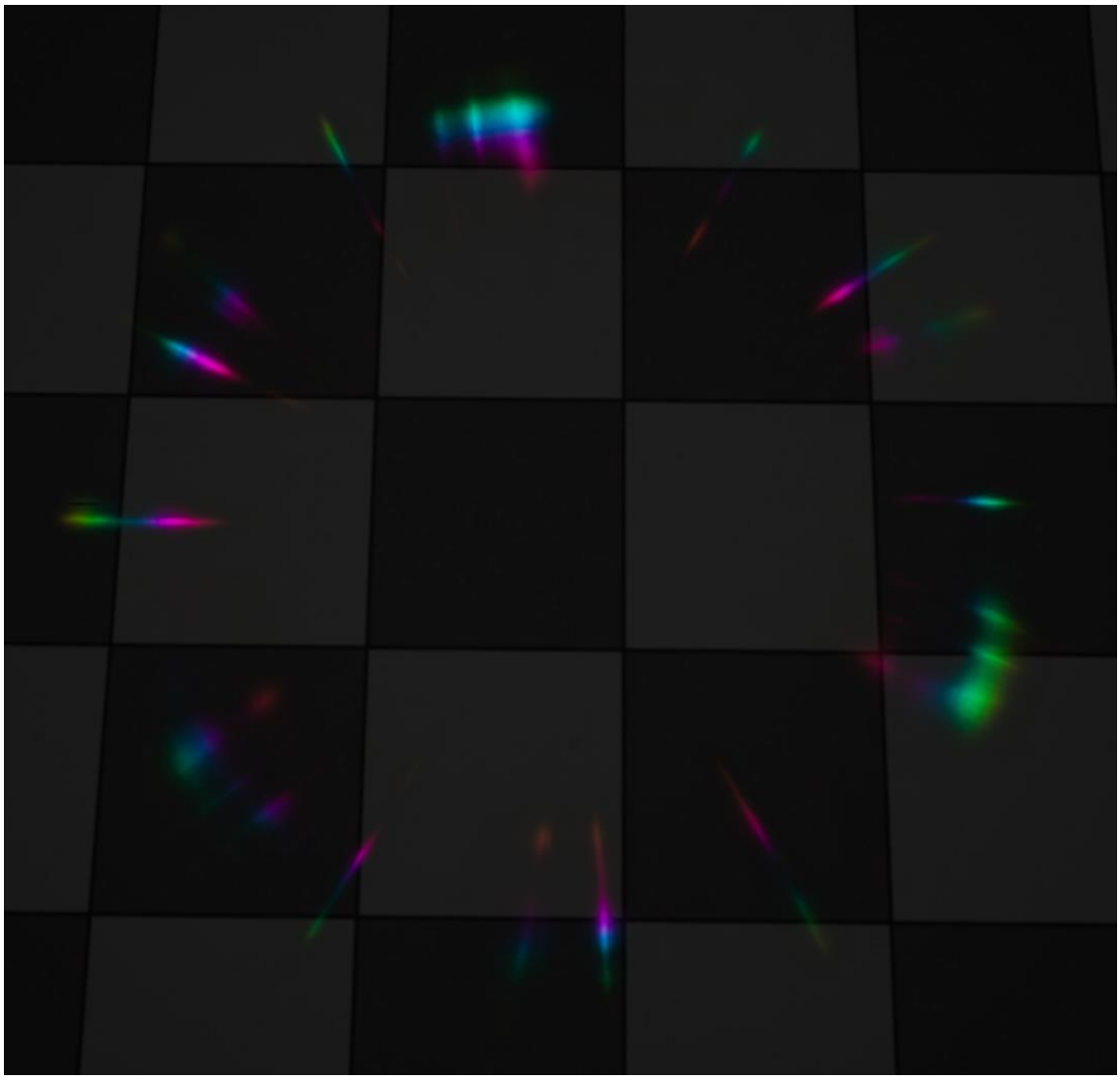
Kuva 36. Sateenkaarigradientti (Filipp, 2025)

Näin valmis materiaali on täysin muokattavissa ja soveltuu kohteiden korostamiseen tai esimerkiksi pelaajahahmon ympärille tulevan auran luomiseen. Valmis Nodekokoonpano näkyy kuvassa 37. Tätä materiaalia hyödynnetään myöhemmässä esimerkissä 4.



Kuva 37. Lopullinen nodekokoonpano (Filipp, 2025)

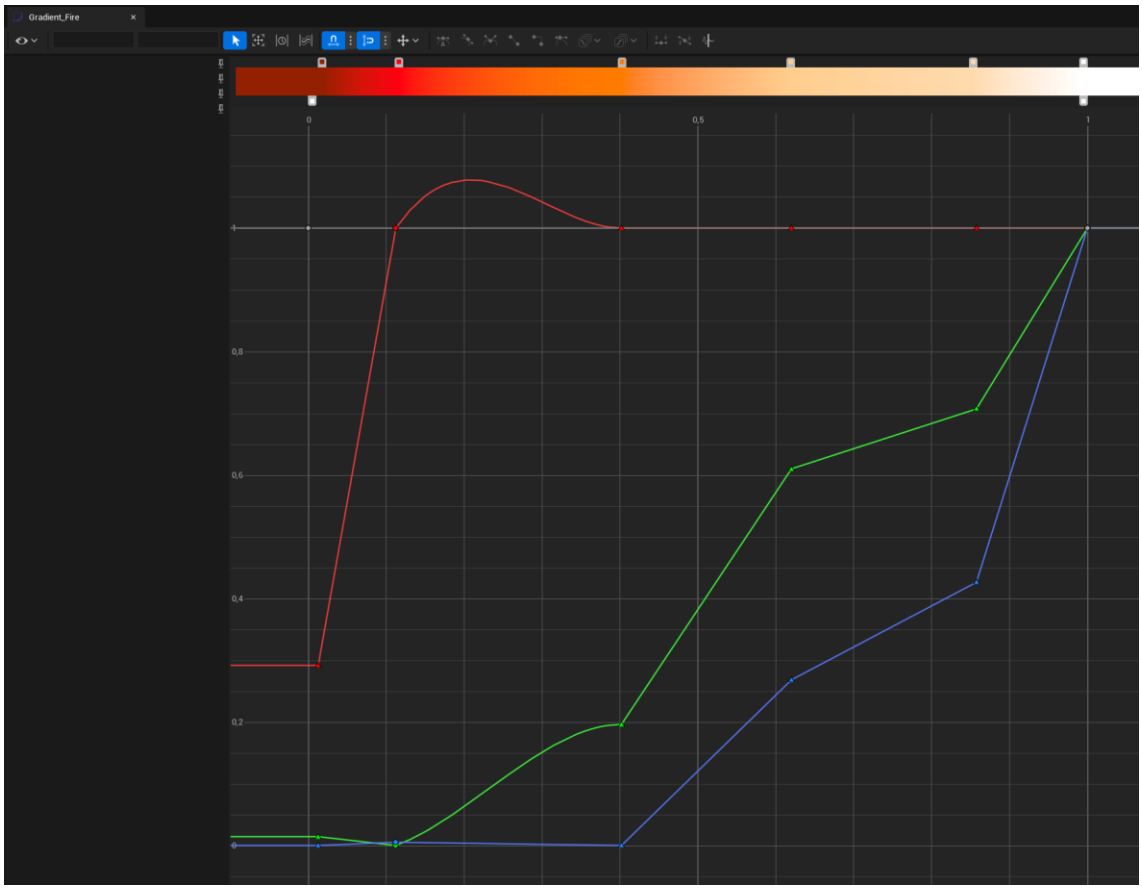
Lopputulos viewportissa näkyy kuvassa 38 ja video efektistä pystyy kastoa avaamalla linkkiä efektiin liitteissä tai klikkaamalla kuvaa 38.



Kuva 38. Lopputulos viewportissa (Filipp, 2025)

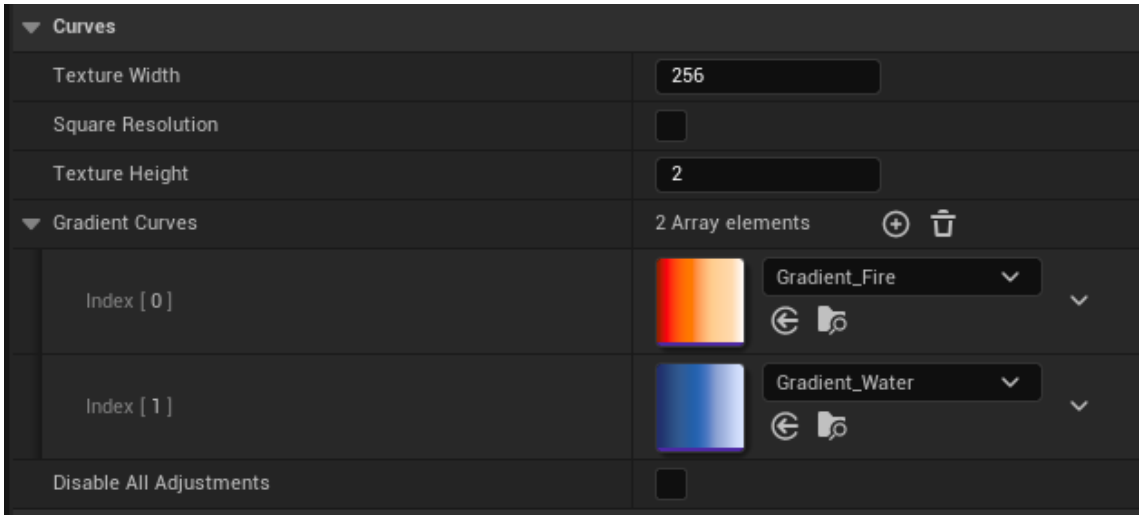
4.3 Esimerkkiefekti 2 – Ribbon jälki

Tässä esimerkissä luodaan ribbon jälkiefekti yhdistämällä materiaalin gradienttikartoitus, vierittävä tekstuurimaski ja Niagara järjestelmä. Materiaalissa hyödynnetään gradienttien hallintaan Atlas tekstuuria sekä värinsiirtymiä säätävää ColorCurve assetia. Värikäyrät (ColorCurve) luodaan Curve Editor työkalulla määrittämällä väripisteitä gradientille, joka tallennetaan käyrätasaukseen (Curve Atlas) 1D tekstuurina, se on esitelty kuvassa 39. Materiaalieditorissa käyrää käytetään Curve Linear Color noden kautta, joka hakee väriarvoja tekstuurikoordinaatin (UV) perusteella.



Kuva 39. Tuligradientti Color curve assettissa (Filipp, 2025)

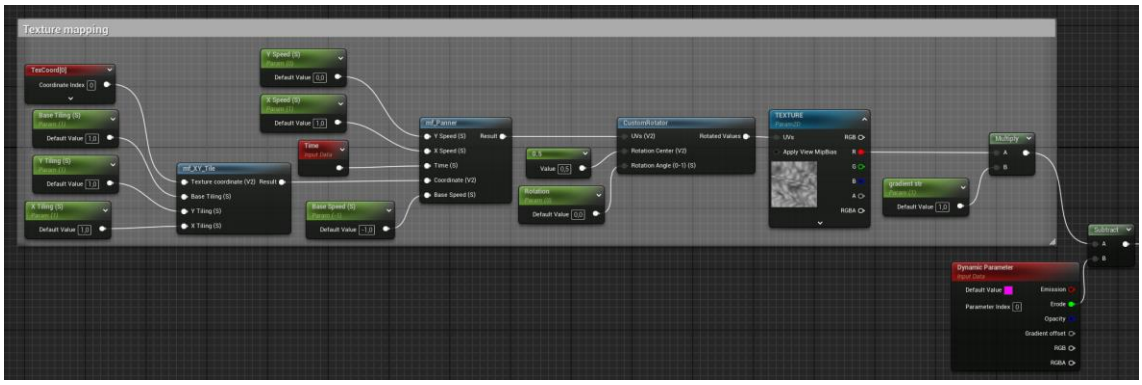
Seuraavana vaiheena on luoda käyräatlaksen (Curve Atlas). Se sisältää useita gradienttikäyriä 1D tekstuureina. Tässä esimerkkiprojektissa luodaan kaksi värikäyrää ja lisätään ne käyräatlakseen (Kuva 40).



Kuva 40. Värikäyrät Käyräkartastossa (Filipp, 2025)

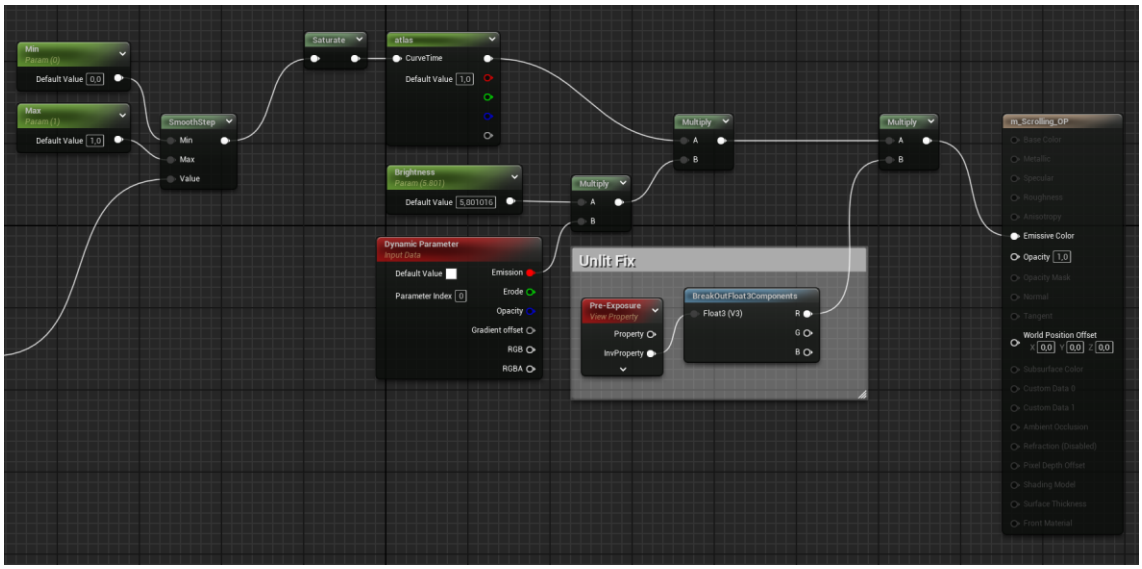
Materiaali

Materiaali alkaa tekstuurin pohjana, johon sovelletaan ensin omaa Tiling funktioni toistamaan atlas tekstuuria UV koordinaateissa ja sen jälkeen Panner funktiota liikuttamaan tekstuuria haluttuun suuntaan (Kuva 41). Maskin säätöä varten käytetään Smoothstep nodea yhdessä AtlasTexture noden kanssa, jotta värien ja mustavalkoisten maskien rajat saadaan teräviksi (Kuva 42).



Kuva 41. Tekstuuri noden funktiot (Filipp, 2025)

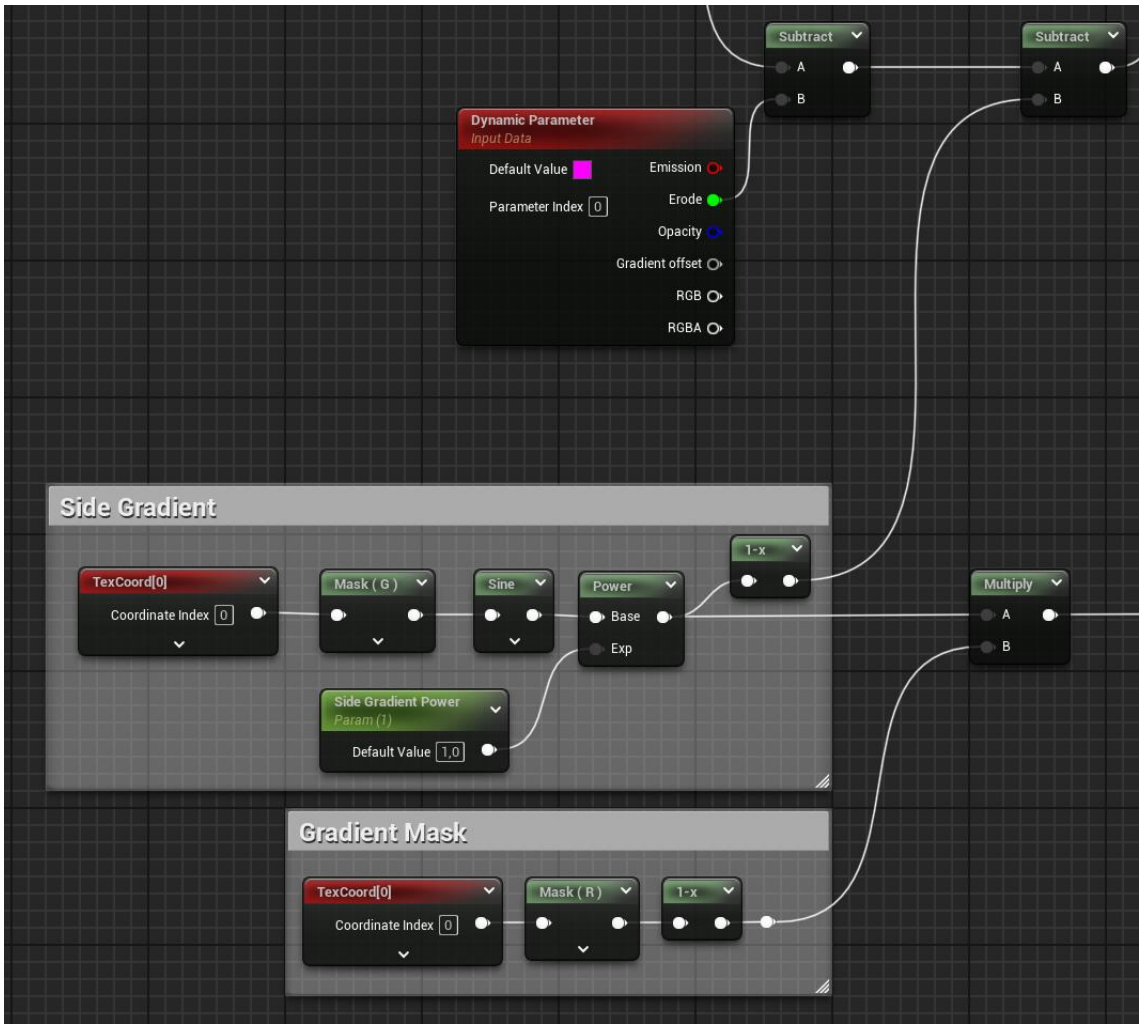
Tämän jälkeen maskin tulos kerrotaan dynaamisella “Emission” parametrilla, joka on sidottu niagara hiukkaslähteen Dynamic Material Parameters moduuliin (Kuvassa 42). Dynaamiset materiaaliparametrit mahdollistavat arvojen muuttamisen ajonaikaisesti, jolloin emitteri voi säätää hehkun voimakkuutta tai muutospisteitä ilman erillistä koodia. Unlit materiaalien värien vakauden takaamiseksi sovelletaan aiemmin kuvattua altistuskorjaussolua, joka poistaa automaattisen valotuksen vaikutuksen. Tästä aiheesta kerron tarkemmin esimerkkityössä 3.



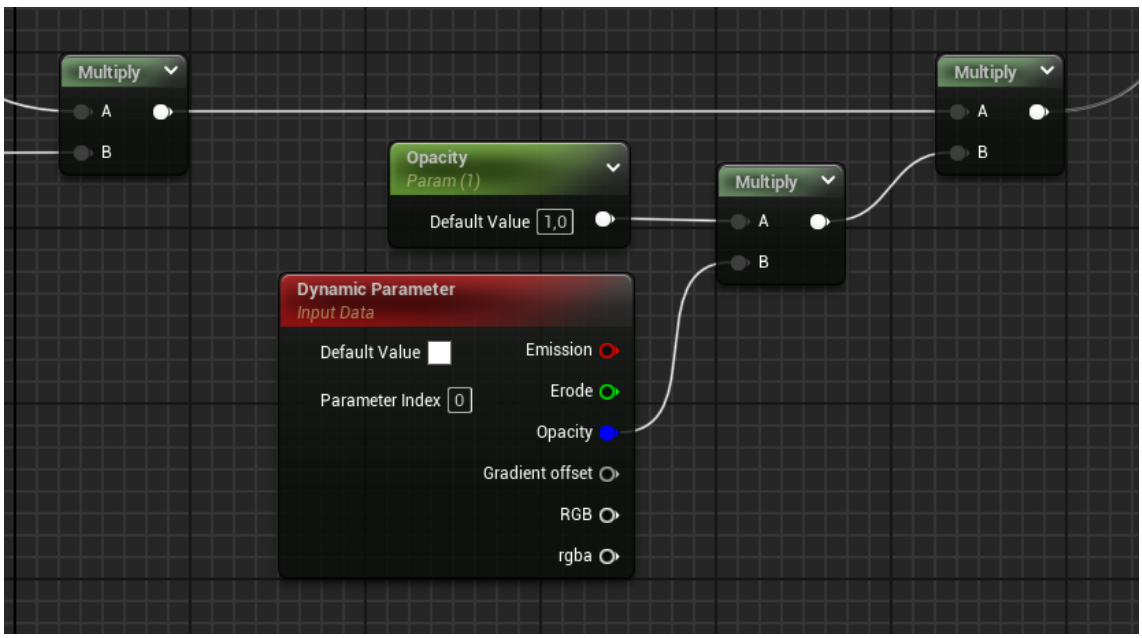
Kuva 42. Maskin säätö ja väritys (Filipp, 2025)

Reunojen erotteluun rakensin sivuttaisen UV gradientin käyttämällä G kanavan maskia yhdessä Sine noden kanssa (periodi 2), mikä tuottaa tekstuurimaskin kulkeutumista reunalta sisäänpäin. Tätä kapeaa sivumaskia vahvistetaan R kanavan UV gradientilla, jolloin reunavaikutus etenee yhdeltä sivulta toiselle (Kuva 43).

Lopuksi reunamaski kerrotaan Opacity maskin kanssa ja sen jälkeen dynaamisella parametrilla Niagara partikkeleita varten, mikä saa jäljen haalistumaan luonnollisesti matkan varrella (Kuva 44).



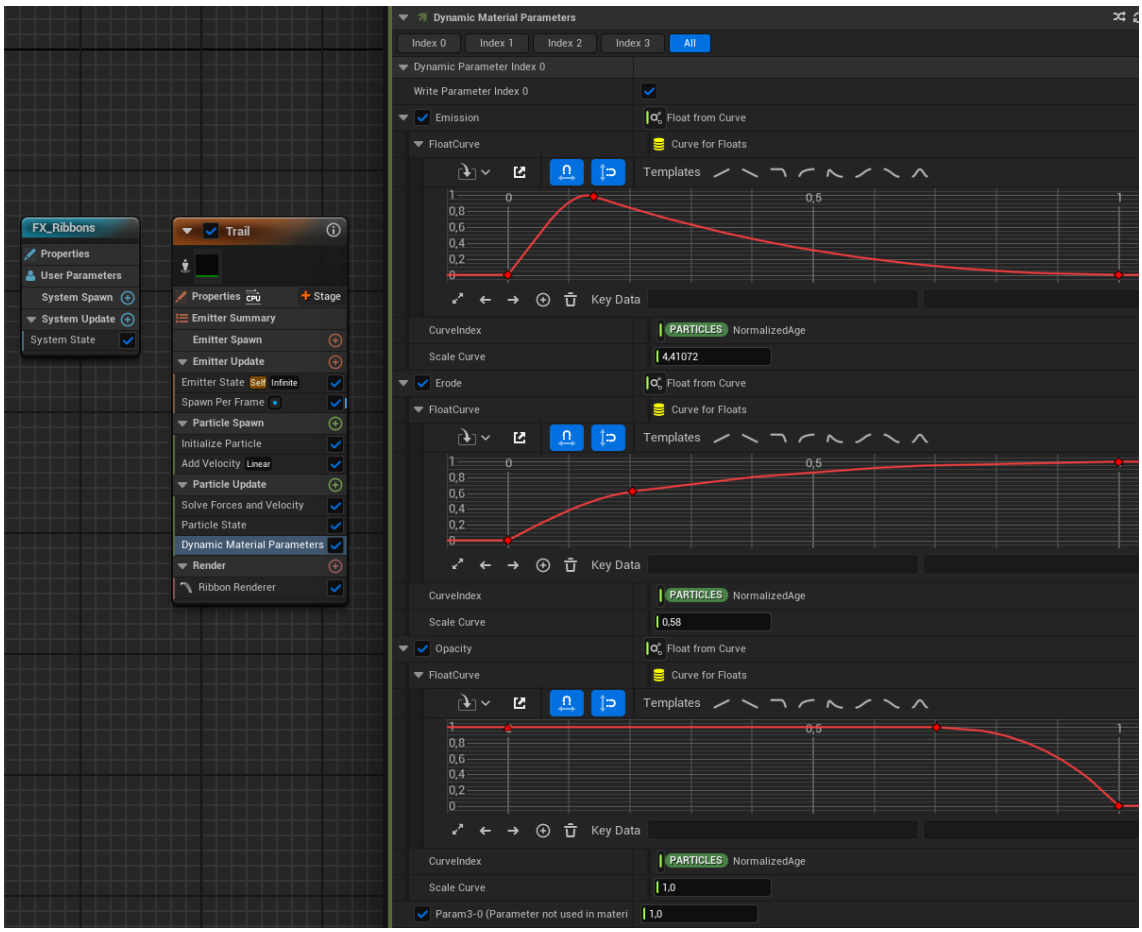
Kuva 43. Tekstuurin reunamaski (Filipp, 2025)



Kuva 44. Materiaalin opacity tulos (Filipp, 2025)

Niagara partikkelisysteemi

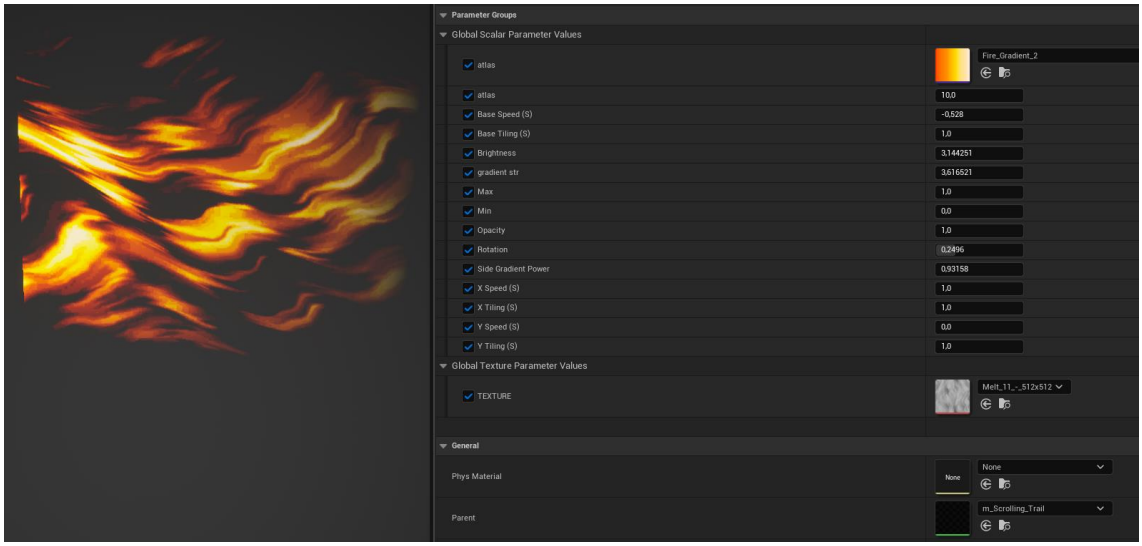
Niagaran puolella luotiin yksinkertainen Niagar systemin, jossa on yksi hiukkaslähde, joka käyttää Ribbon Renderer moduulia. Hiukkaslähde synnyttää partikkelit Spawn Each Frame moduulilla, ja Add Velocity moduulilla ohjataan jäljen suunta ja pituutta. Dynamic Material Parameters moduulilla animoidaan emissio, erode ja opacity parametrit ajan funktiona. Näin materiaali ja partikkelijärjestelmä yhdistyvät saumattomasti luoden dynaamisen, reagoivan nauhatehosteen. Kuvassa 45 on esitelty Dynaamisten parametrien animointi.



Kuva 45. Niagara hiukkaslähde (Filipp, 2025)

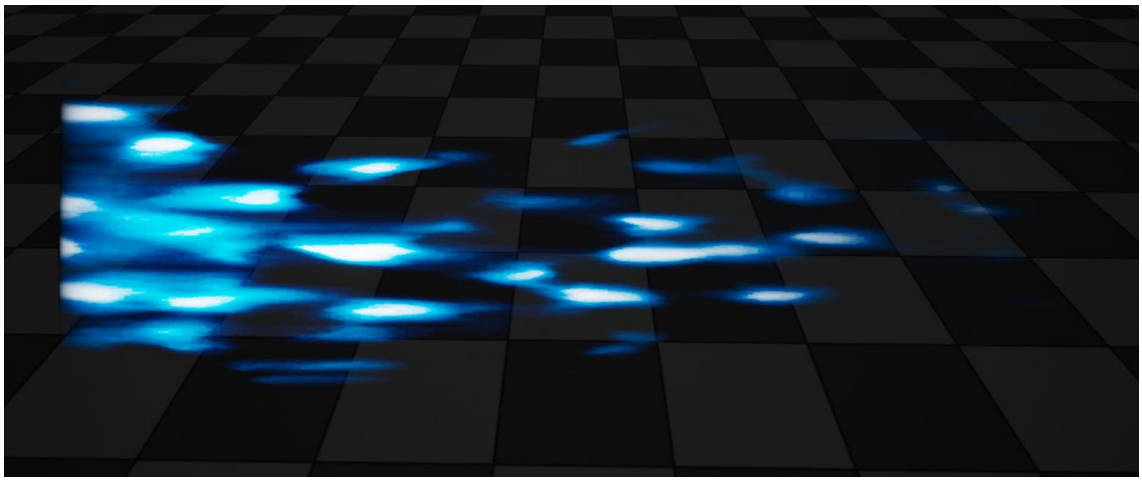
Materiaalin iterointia varten luodaan Material Instance materiaalin, jossa voi vaihtaa atlas värigradientteja ja maskiparametreja lennossa. Instanssien avulla värivaihtoehdot ja tekstuuritesti voidaan tehdä nopeasti ilman, että pohjamateriaalia tarvitsee muokata. Kuvassa 46 näkyy materiaali instanssi, missä pystyy

muokata parametrit ja vaihtaa tekstuurit. Erilaisten materiaali instanssien loppu-tulokset näkyvät kuvissa 47, 48 ja 49.



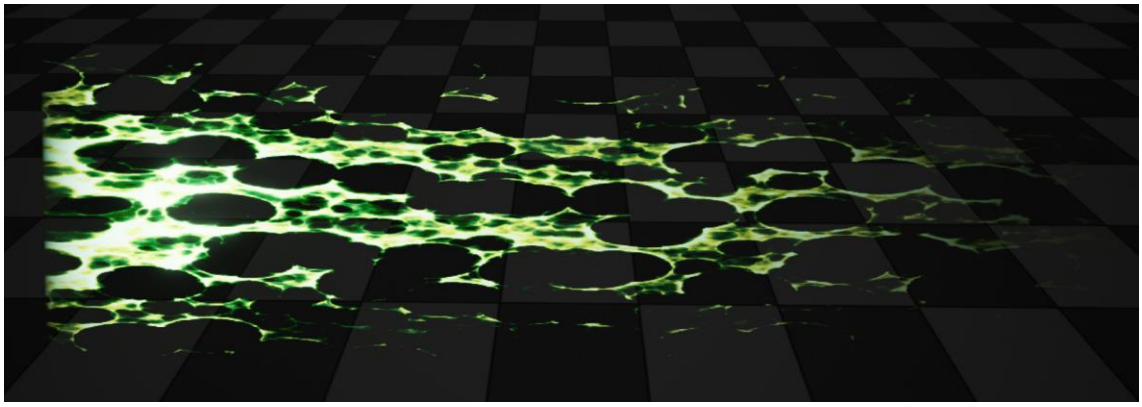
Kuva 46. Materiaali instanssi (Filipp, 2025)

Kuva 47. Tulen värit (Filipp, 2025)



Kuva 48. UV Veden värit (Filipp, 2025)

Ribbon jäljen voi käyttää esimerkiksi aseiden liikeratojen visualisointiin, virtavesien tai laavan kaltaisten virtaavien efektien toteutukseen sekä maagisten energiaratojen korostamiseen. Lopputulos näkyy kuvassa 49 ja video efektistä pystyy kastoa avaamalla linkkiä efektiin liitteissä tai klikkaamalla kuvaa 49.



Kuva 49. Lopputulos (Filipp, 2025)

Tässä luvussa rakennettiin optimoitu, muokattavissa oleva ja visuaalisesti näyttävä jälkiefekti, jossa materiaalin ja Niagara systeemin parametrit on sidottu toisiinsa dynaamisten parametrien avulla. Lopputulos on joustava, skaalautuva ja helposti integroitavissa erilaisiin pelimekaniikkoihin.

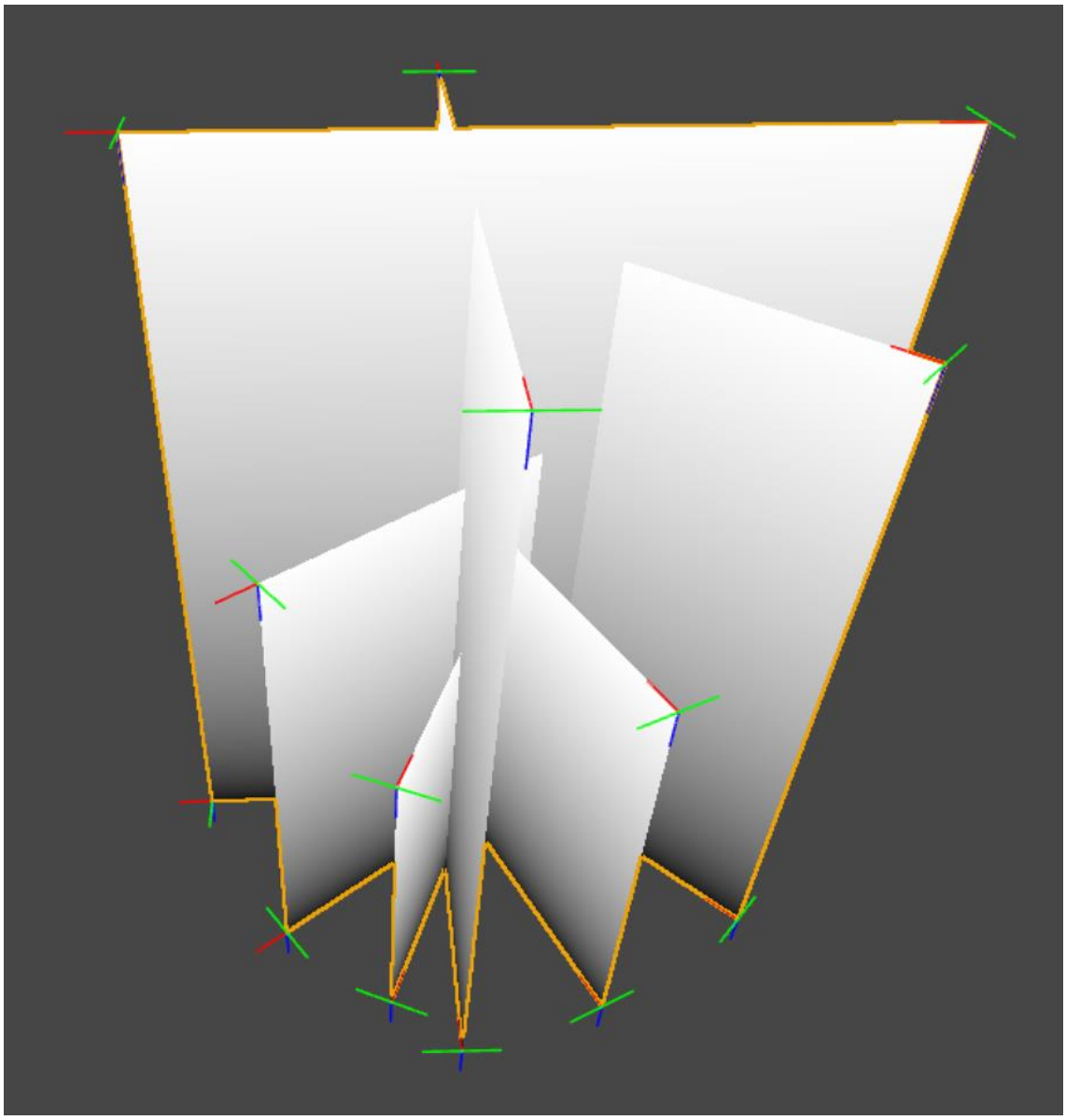
4.4 Esimerkkiefekti 3 – Ruoho

Tässä esimerkissä käsitellään tyylitellyn ruohon luomista Unreal Engine 5:ssä hyödyntäen useita tekniikoita dynaamisen ja visuaalisesti houkuttelevan lopputuloksen saavuttamiseksi.

Materiaalieditorissa käytetään World Position Offset (WPO) ominaisuutta tuulen liikkeen simulointiin. Vierittävä kohinatekstuuria ja HSV:stä saadun 2D vektoria hyödynnetään suunnattujen tuuliefektin luomiseen, ja verteksivärigradientti ohjaa liikkeen voimakkuutta ruohonlehdissä. Tämä mahdollistaa realistisen huojuvan liikkeen, joka reagoi tuulen suuntaan ja voimakkuuteen.

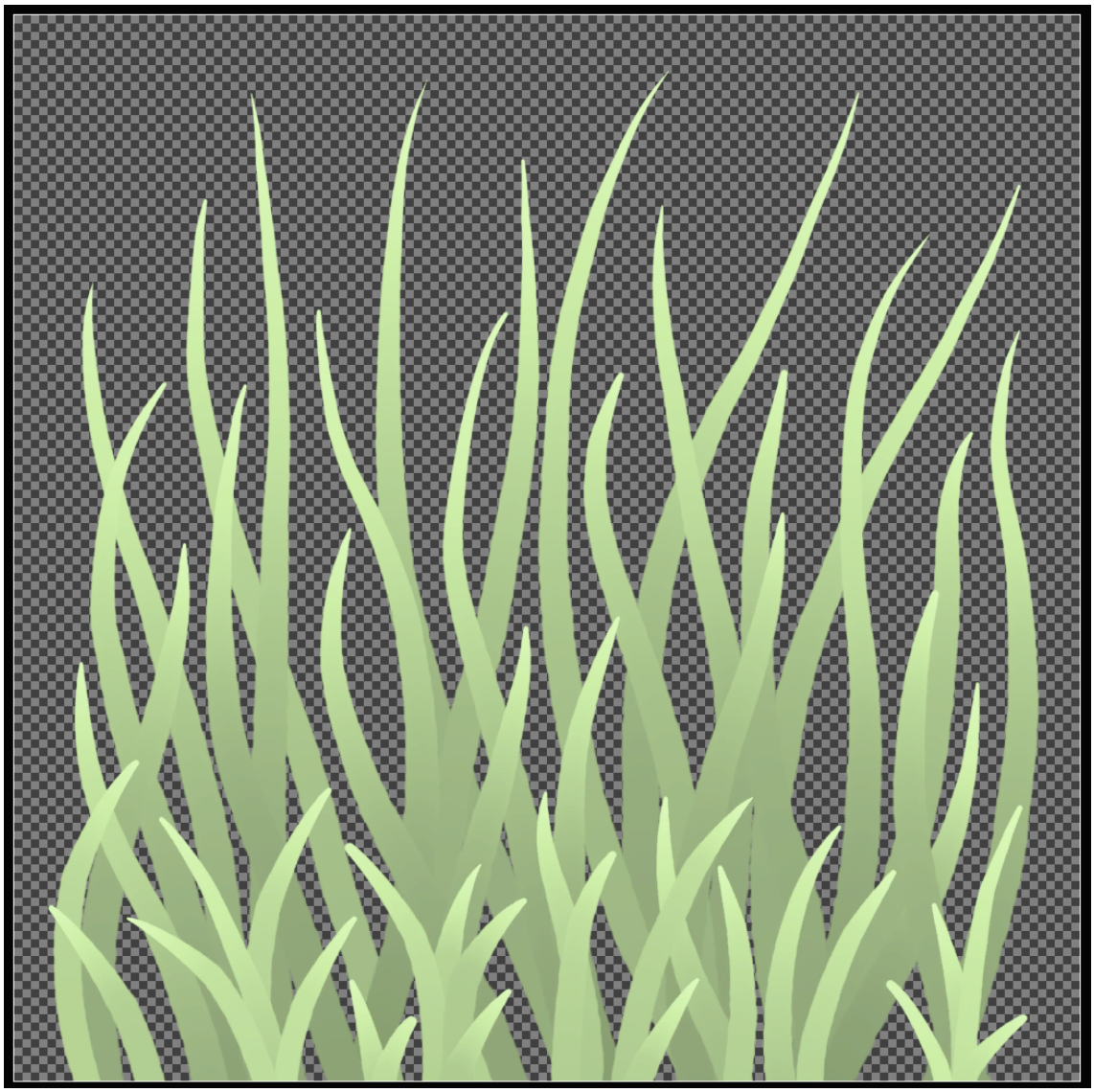
Suorituskyvyn ja visuaalisen laadun optimoimiseksi otetaan käyttöön etäisyyteen perustuva dithered fade tekniikka. Tämä menetelmä vähentää ruohomeshien läpinäkyvyyttä niiden etääntyessä kamerasta, minimoiden piirtokutsun ja varmistaen sujuvan siirtymän yksityiskohtaisten ja kaukaisten alueiden välillä. Näiden yhdistettyjen tekniikoiden avulla saavutetaan tyylitelty ruohotehoste, joka on sekä dynaaminen että tehokas, soveltuen erilaisiin reaaliaikaisiin sovelluksiin.

Prosessi alkaa ruohomeshiin mallintamisella 3D ohjelmistossa. Syvyyden ja volyymin lisäämiseksi meshistä tehdään kopio, jonka normaalit käännetään. Tämän jälkeen kaikkien normaalien suunnat säädetään osoittamaan ylöspäin, mikä varmistaa tasaisen valaistuksen ja varjostuksen ruohonlehdissä. Meshin päälle maalataan pystysuuntainen gradientti verteksiväreillä, joka toimii maskina tuulen vaikutuksen säätelyssä. (Kuva 50).



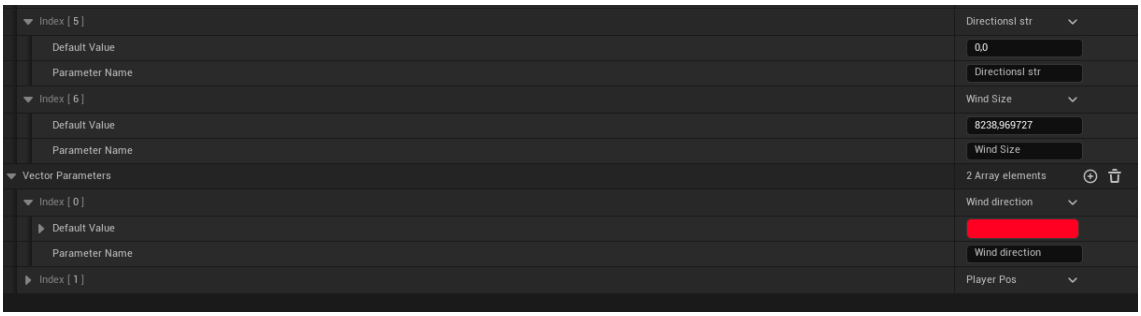
Kuva 50. Ruohomeshi Unreal Engine:ssa (Filipp, 2025)

Ruohotekstuurin pystyy piirtää esim. Kritassa tai Photoshopissa. Ruohotekstuuri tässä esimerkkityössä on omatekemä tekstuuri tehty Kritassa (Kuva 51).



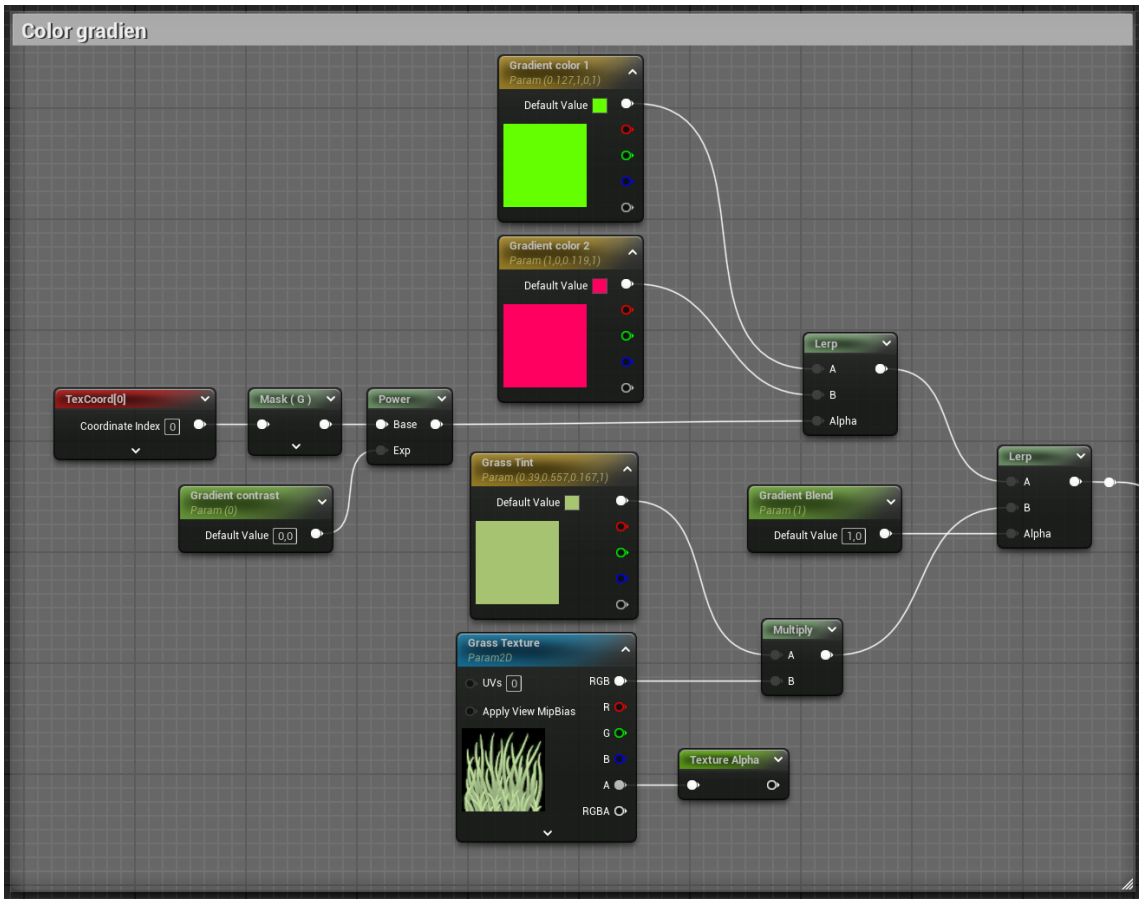
Kuva 51. Ruohotekstuuri (Filipp, 2025)

Ennen varsinaista ruohomateriaalin rakentamista Material Parameter Collection on luotu. Sillä voidaan hallita esimerkiksi tuulen voimaa, värigradientin sävyjä ja dither fade parametrit globaalisilla muuttujilla (Kuva 52). Tämä helpottaa parametrien säätämistä sekä instansseissa että Blueprinteissa ilman, että jokaista arvoa tarvitsee muuttaa erikseen materiaalieditorissa. Tässä esimerkkimateriaalissa kyseiset parametrit haetaan Material Parameter Collection nodella, jolloin niiden arvoja voi säätää Material parameter collection assetissa.

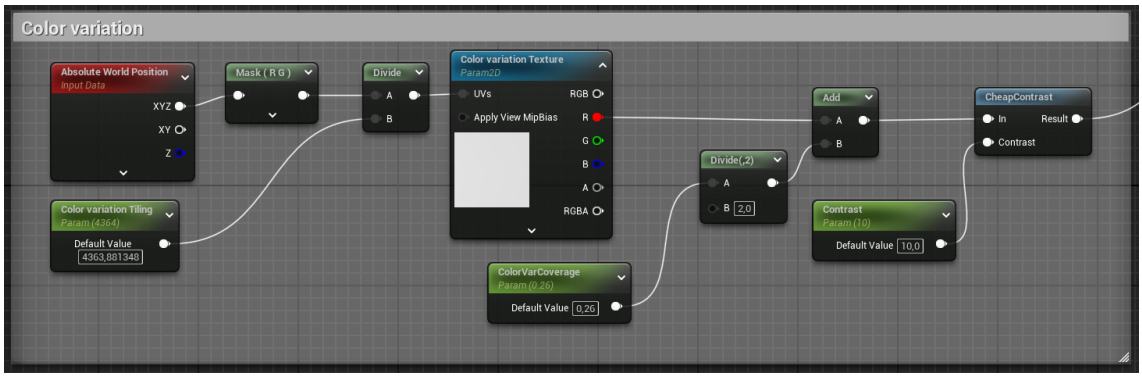


Kuva 52. Tuulen material parameter collection parametrit (Filipp, 2025)

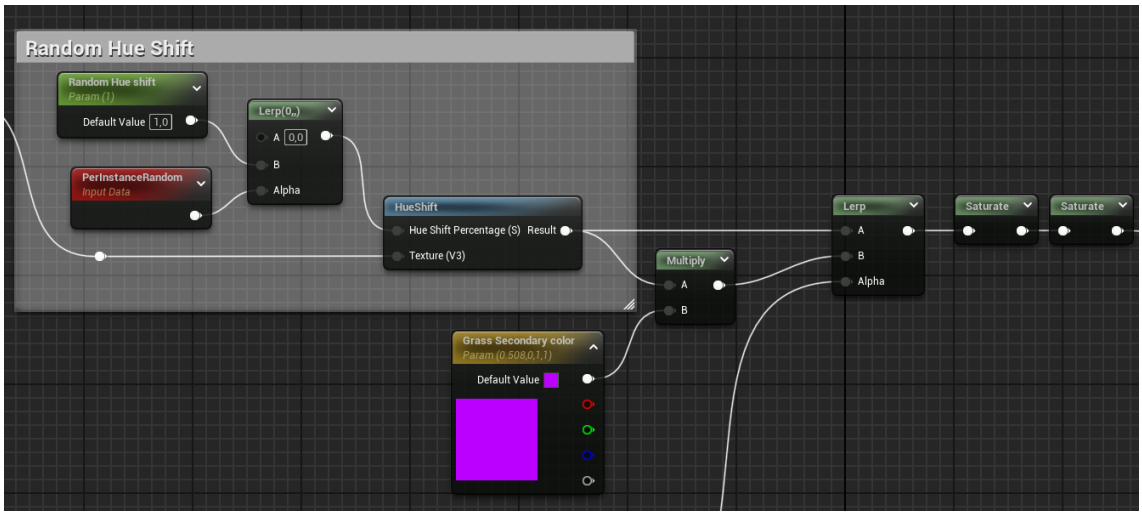
Värivaihtelua saavutetaan UV gradienttivärityksellä ja per instanssi satunnaisvärin säädöillä. UV koordinaattien mukainen gradientti luo perustason värisiirtymän (Kuvassa 53), ja maailmanavaruuden värivaihtelu syventää vaikutelmaa muuttamalla värejä ruohon sijainnin perusteella (Kuvassa 54). Per instanssi satunnaistaminen tuo hienovaraisia värieroja yksittäisten ruohoinstanssien välillä, lisäten luonnollisuutta (Kuvassa 55). Ruohomateriaalin nodeverkkoa pidetään siisteinä hyödyntämällä Reroute nodeja, joilla pitkät ja risteilevät johdotukset pilkotaan lyhyempiin segmentteihin. Näin kokonaisuus pysyy selkeänä ja helppolukuisena, kun nodet on jaettu loogisiin osioihin. Tässä tapauksessa käytän reroute node tekstuurin alpha kanavan tuloksen syötetään reroute nodeen, jota sitten monistetaan opacity tuloksen kanssa (Kuva 53).



Kuva 53. UV pohjainen liukuväritys (Filipp, 2025)

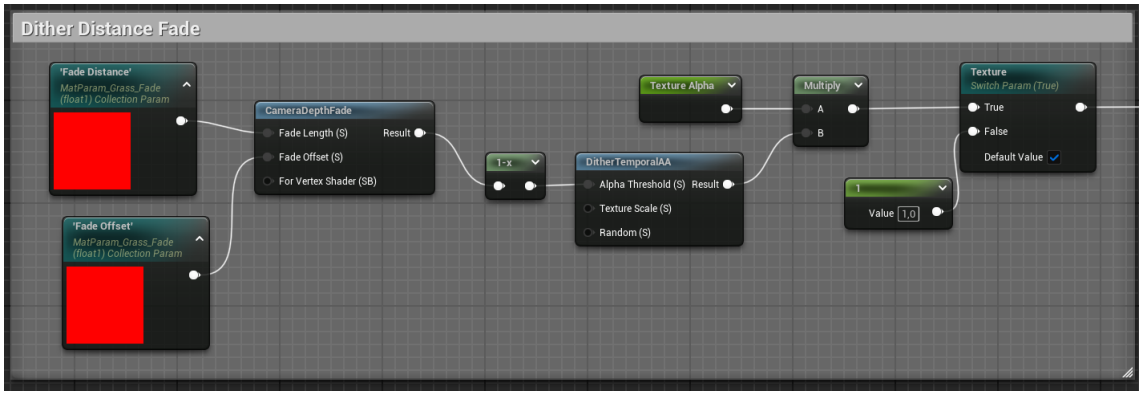


Kuva 54. Värien vaihtelun Nodeverkko (Filipp, 2025)



Kuva 55. Random per instance ja HueShift noden käytössä (Filipp, 2025)

Suorituskyvyn ja visuaalisen laadun optimoimiseksi otetaan käyttöön etäisyyteen perustuva dithered fade tekniikka (Kuva 56). Tämä menetelmä vähentää ruohomeshien läpinäkyvyyttä niiden etääntyessä kamerasta, minimoiden piirtokutsun ja varmistaen sujuvan siirtymän yksityiskohtaisten ja kaukaisten alueiden välillä.

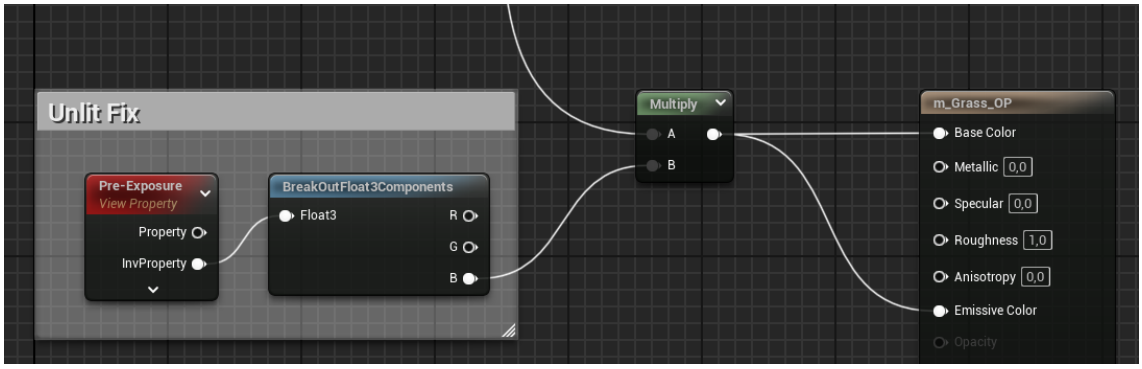


Kuva 56. Dither fade nodekokoonpano (Filipp, 2025)

Unreal Engine pelimoottorissa automaattinen valotuksen säätö (auto exposure) voi vaikuttaa odottamattomasti materiaalien väreihin, erityisesti silloin kun käytetään unlit materiaaleja tai emissiivisiä pintoja. Tämä johtuu siitä, että vaikka unlit materiaalit eivät reagoi valaistukseen, ne ovat silti alttiita kameran valotuksen muutoksille, mikä voi aiheuttaa värien haalistumista tai kirkastumista eri valaistusolosuhteissa. Tämän ongelman ratkaisemiseksi voidaan käyttää View Pro-

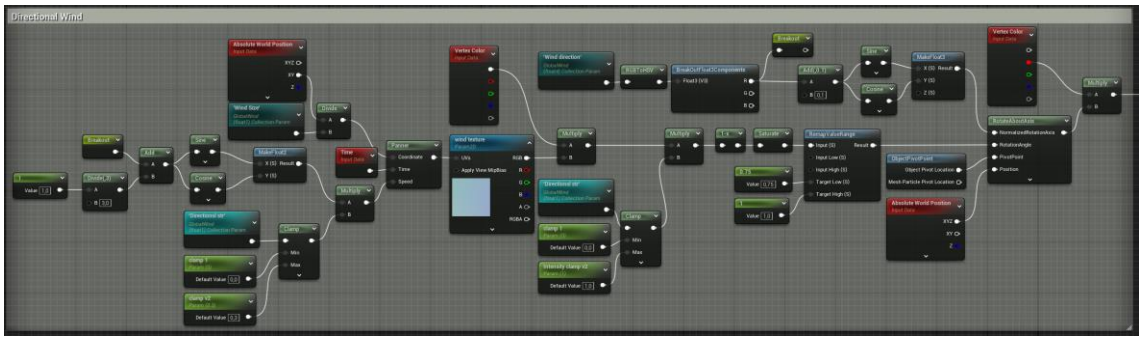
perty nodea, joka asetetaan Pre Exposure arvoon. Kertomalla materiaalin väriarvo tämän Pre Exposure arvon käänteisellä arvolla, voidaan kompensoida valotuksen vaikutus materiaaliin (Kuva 57). Tämä tekniikka auttaa säilyttämään materiaalin värit ja kirkkauden tasaisina riippumatta valotuksen muutoksista. (PROTOWLF 2024.)

Tämä lähestymistapa on erityisen hyödyllinen tilanteissa, joissa halutaan varmistaa materiaalien visuaalinen johdonmukaisuus eri valaistusolosuhteissa, kuten käyttöliittymäelementeissä tai emissiivisissä efekteissä.



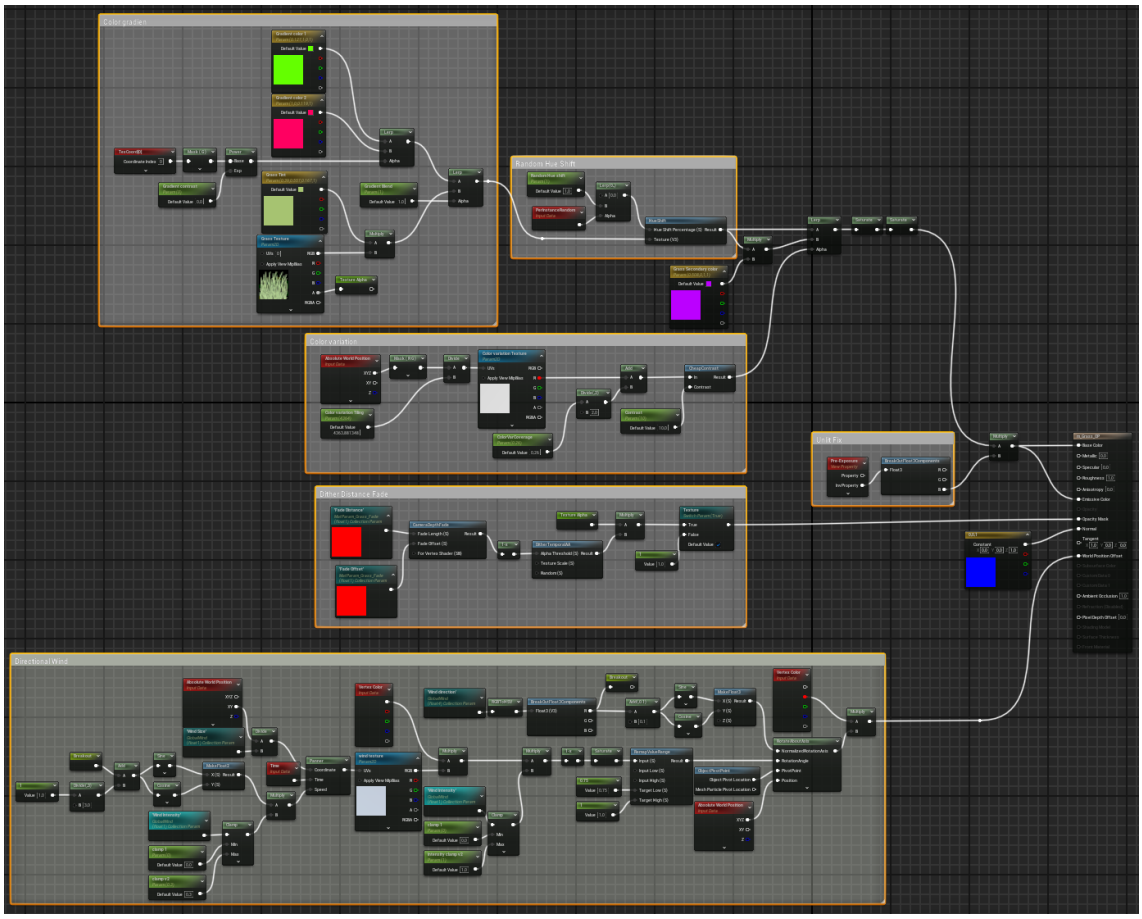
Kuva 57. Unlit materiaalin ratkaisu (Filipp, 2025)

Materiaalieditorissa käytetään World Position Offset (WPO) ominaisuutta simuloimaan ruohon tuulen liikettä. Tuulen suunta määritellään muunnettuna värivektoria HSV värimalliin, josta hue kanavan arvon lisätään Sine ja Cosine funktioiden kanssa. Tulos on 2D vektori, mitä sitten käytetään tuulen maski tekstuurin panner noden nopeusarvojen määrittelyyn. Tämä maski ohjaa RotateAboutAxis funktion parametreja, joka laskee ruohon verteksit kiertymiskulman tuulen voimakkuuden ja suunnan perusteella. Verteksivärigradientti määrittää liikkeen intensiteetin ruohon eri osissa, mikä lisää realistista vaihtelua. Lisäksi clamp ja remap nodet rajoittavat arvojen sallittuja alueita tehokkuuden varmistamiseksi. Lopullinen efekti saavutetaan siten, että jokainen ruohon mesh piste pyörii laskennallisesti määritellyn akselin ympäri, reagoimalla tuulen globaaleihin asetuksiin sekä paikallisiin materiaaliparametreihin (Kuva 58).



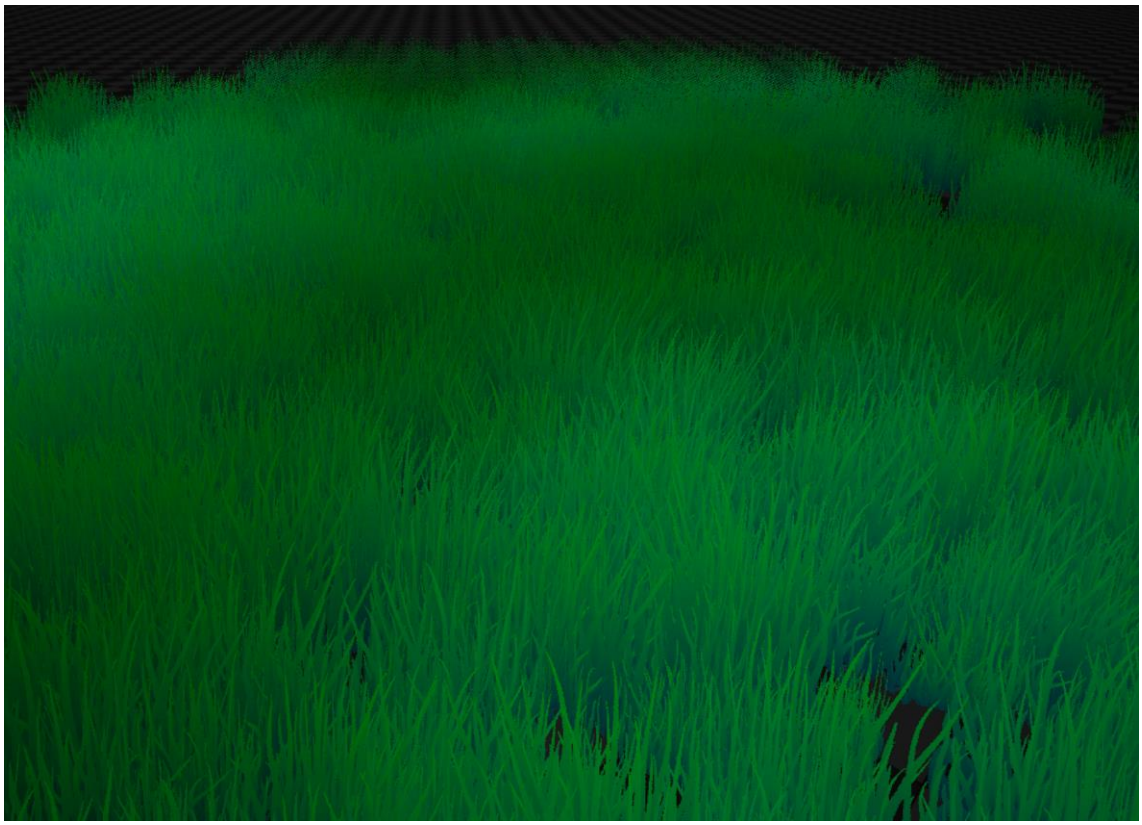
Kuva 58. WPO Tuulen nodekokoonpano (Filipp, 2025)

Ruohomateriaalin luomisessa hyödynnettiin useita tekniikoita: värit yhdistettiin UV koordinaatteihin perustuvan gradientin avulla, minkä lisäksi jokaiselle instanssille lisättiin satunnainen sävymuutos (Hue). Väriä varioitiin myös maailman koordinaattien perusteella, mikä rikastaa kokonaisilmettä. Etäisyyteen perustuva Dither Fade mahdollistaa materiaalin sulavan häivytyksen, mikä optimoi suorituskykyä. Tuuli toimii WPO ratkaisulla, jossa käytettiin mukautettua HSV pohjaista tuulisuunnan ohjausta. Lisäksi materiaalin värit vakautettiin altistuksen vaihteluilta (Exposure Fix) säilyttämään visuaalinen johdonmukaisuus erilaisissa valaistusolosuhteissa. Kuvassa 59 on esitelty lopullinen nodekokoonpano.



Kuva 59. Lopullinen Node käyttöönotto (Filipp, 2025)

Tässä esimerkkityössä luotiin optimoitu ja visuaalisesti tyylitelty dynaaminen ruohomateriaali, joka reagoi tuuleen, säilyttää suorituskykynsä ja tarjoaa laajat mahdollisuudet säätää sen ulkoasua ja käyttäytymistä. Materiaali soveltuu monenlaisiin ympäristöihin ja toimii perustana muille tyylitellyille luonnonefekteille (Kuva 60). Lopputulos näkyy kuvassa 60 ja video efektistä pystyy kastoa avaamalla linkkiä efektiin liitteissä tai klikkaamalla kuvaa 60.



Kuva 60. Ruoho viewportissa (Filipp, 2025)

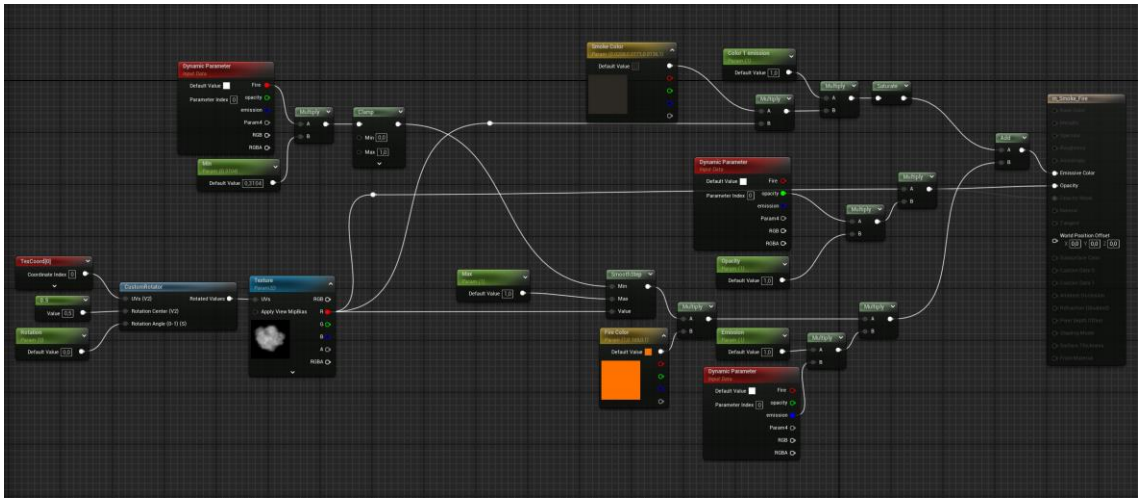
4.5 Esimerkkiefekti 4 – Tyylitelty räjähdys

Tässä esimerkissä tuotetaan räjähdysefektin tuotantoprosessi, joka yhdistää materiaali, Niagara ja Blueprint tekniikat. Työ jakautuu kahteen osaan: itse efektiosuus ja Blueprint integraatio.

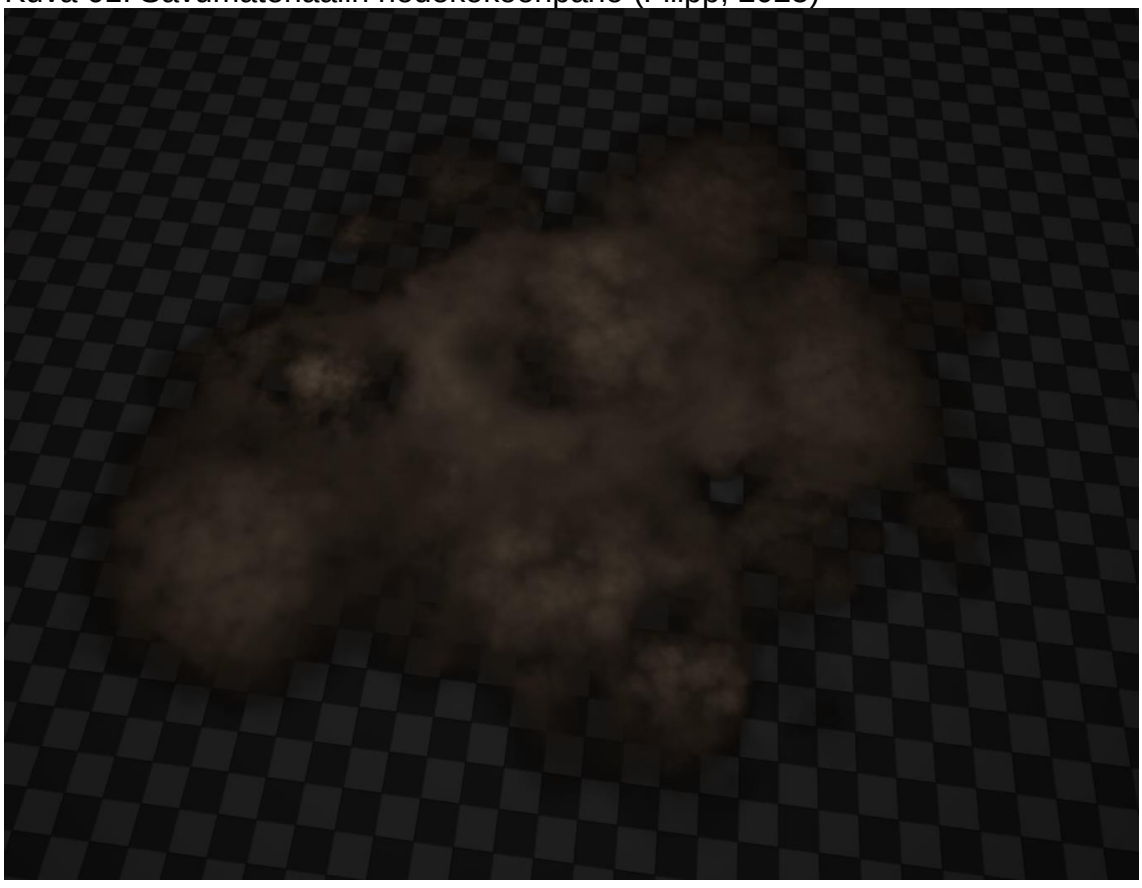
Efektiosuus sisältää ennakointi, räjähdys ja jälkikäsittelyvaiheita. Räjähdyksen toteutus alkaa Niagara systeemin luomisesta, johon lisätään yksityiskohtaisia hiukkaslähteitä.

Ensimmäinen hiukkaslähde vastaa savu ja tuliefektistä. Materiaalieditorissa luodaan tyylitelty savu materiaali, jossa tekstuurinäyte kiertyy CustomRotator noden avulla oikeaan kulmaan. Savu osa peitetään punaisen kanavan maskilla Opacity tulossa ja liitetään Dynamic Material Parameteriksi, jotta Niagara voi säätää sen näkyvyyttä. Savutekstuuriin lisätään Smoothstep maski, joka paljas-

taa sisäisen tuliosuuden, tuliefektin laajenemista ohjataan säätöparametrillä. Lopuksi savu ja tuliosuudet kerrotaan keskenään ja kytketään materiaalin Emissive tuloon. Tämä materiaali toimii pohjana myös muille materiaaleille, sillä material instance instansseilla voidaan vaihtaa tekstuuria ja parametreja joustavasti (Kuva 61).

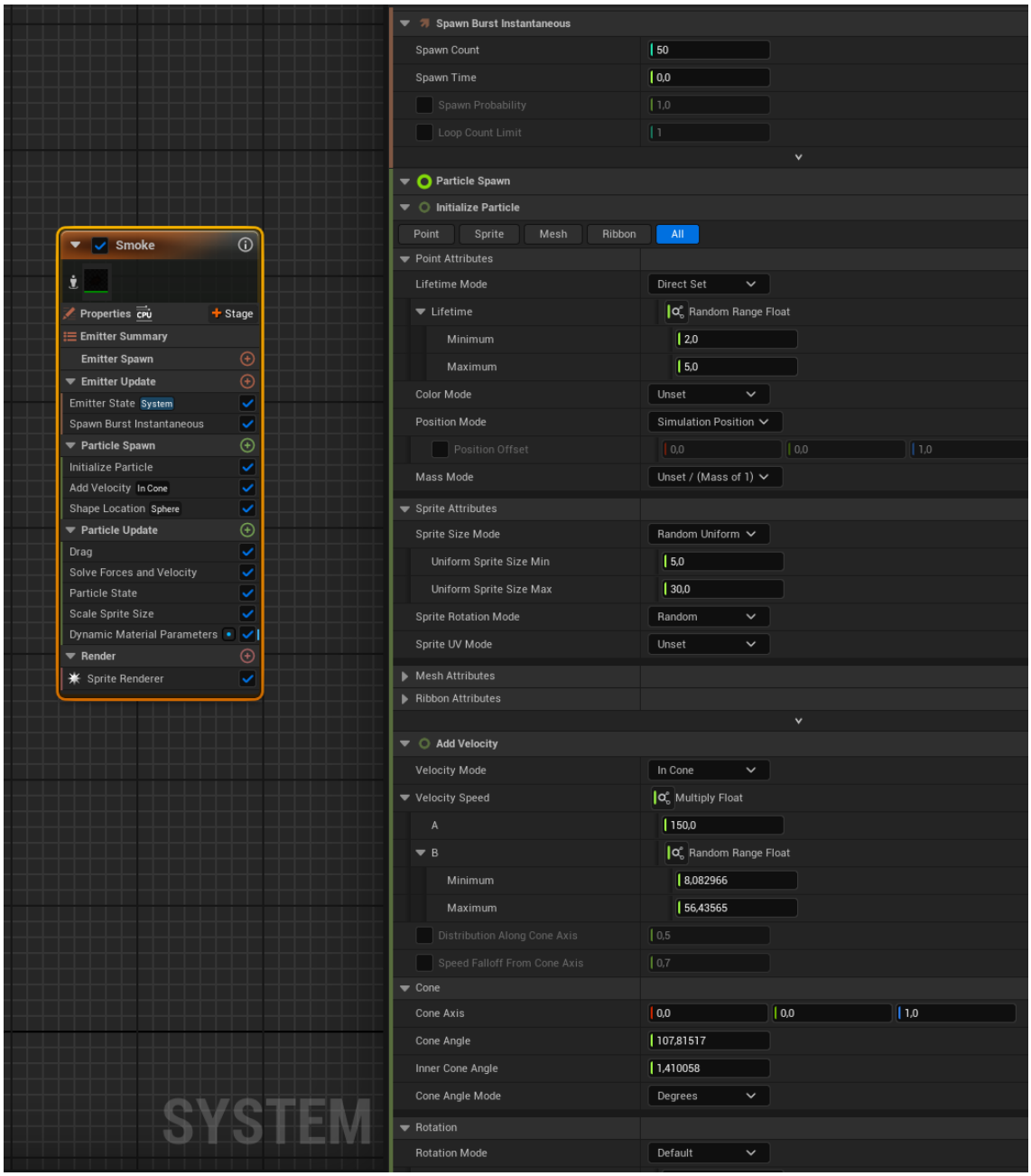


Kuva 61. Savumateriaalin nodekokoonpano (Filipp, 2025)

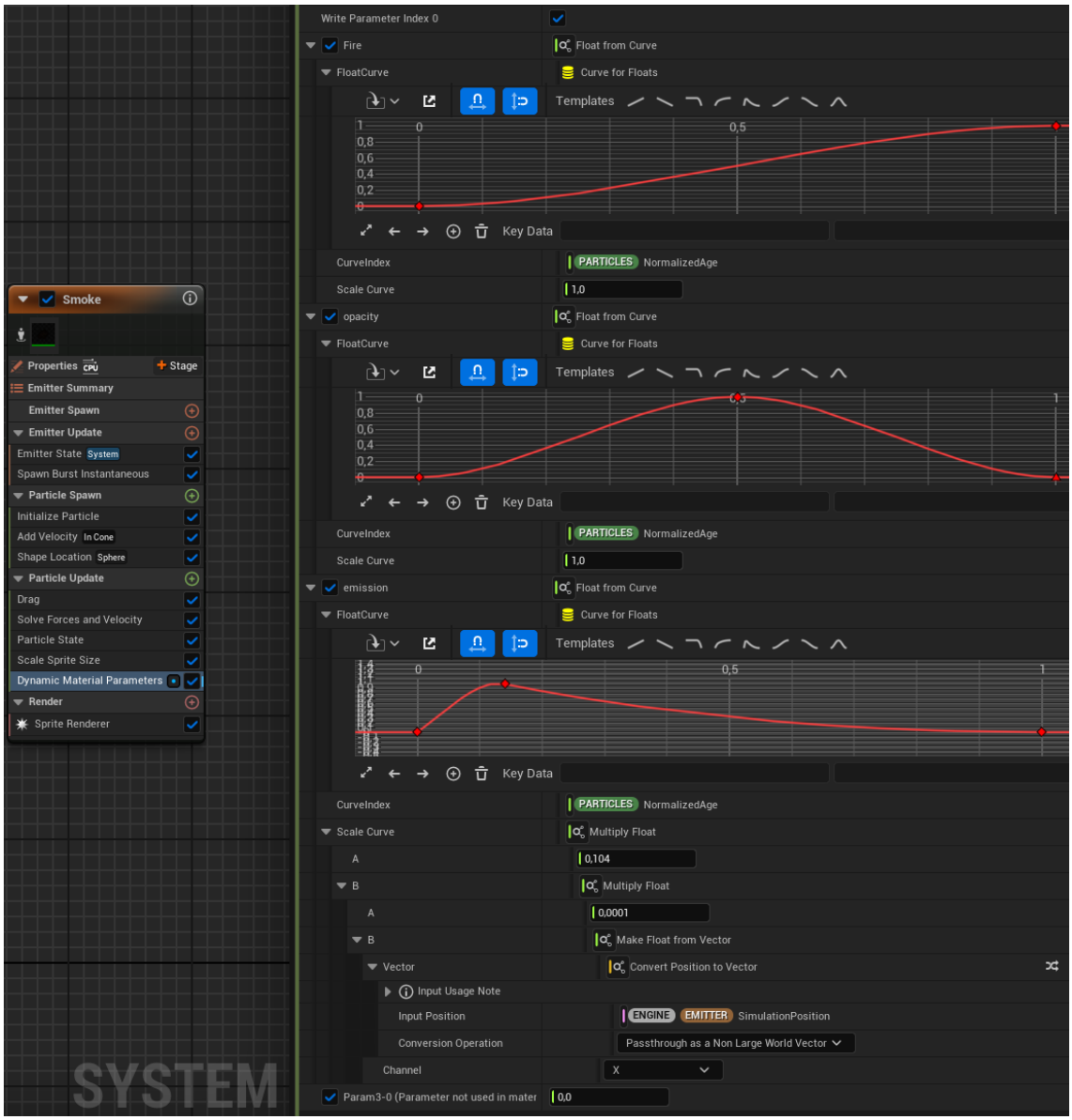


Kuva 62. Savumateriaali viewportissa (Filipp, 2025)

Seuraavaksi luodaan emitterin, missä otetaan käyttöön Dynamic Material Parameters moduuli, jolla ajoitetaan emissio, erode ja opacity parametrien animaatio suoraan hiukkaslähteessä. Näin savun voimakkuus ja palavan tulen elinkaari muotoutuvat reaaliajassa (Kuvat 63 ja 64).

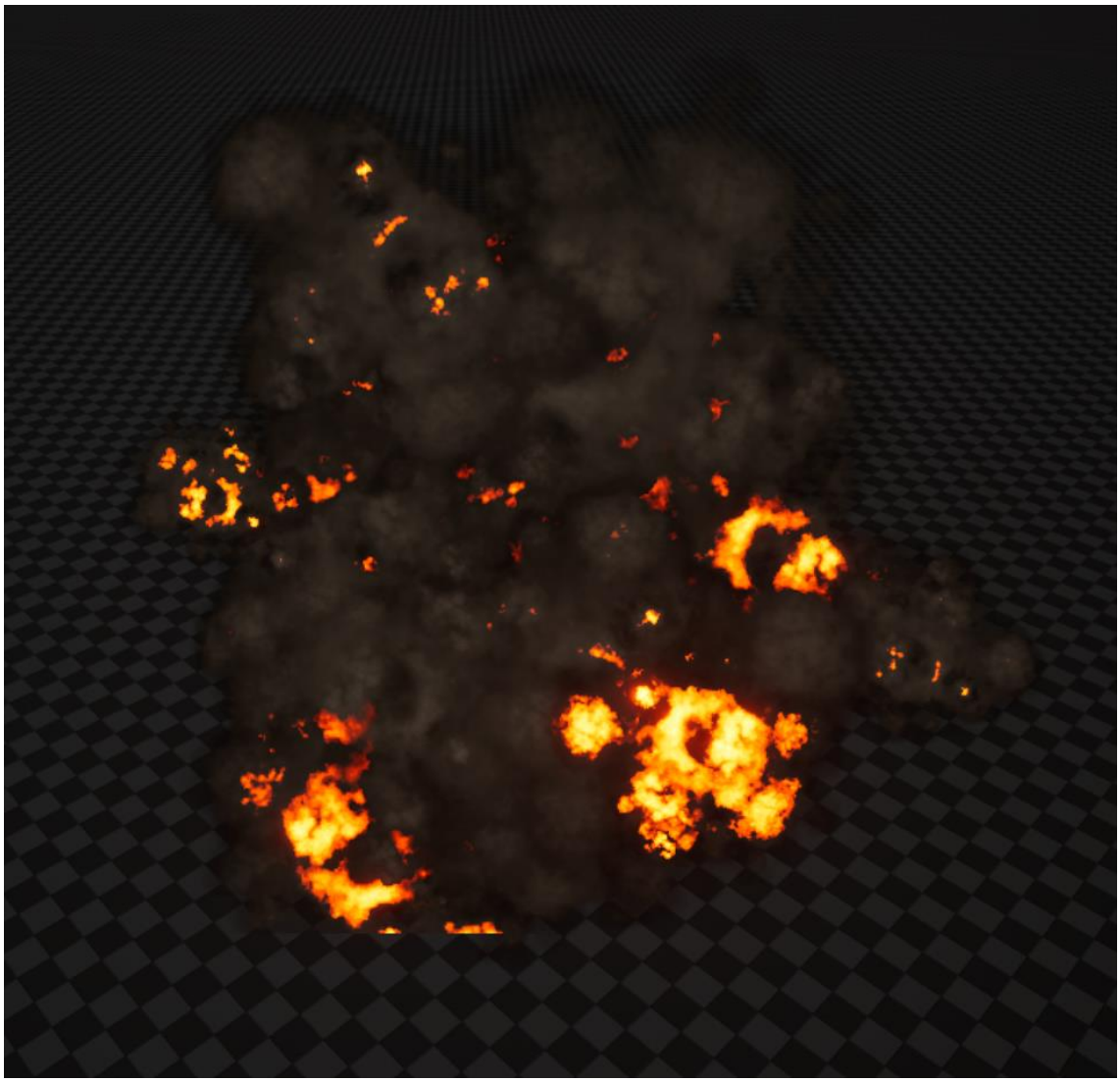


Kuva 63. Savun hiukkaslähde (Filipp, 2025)



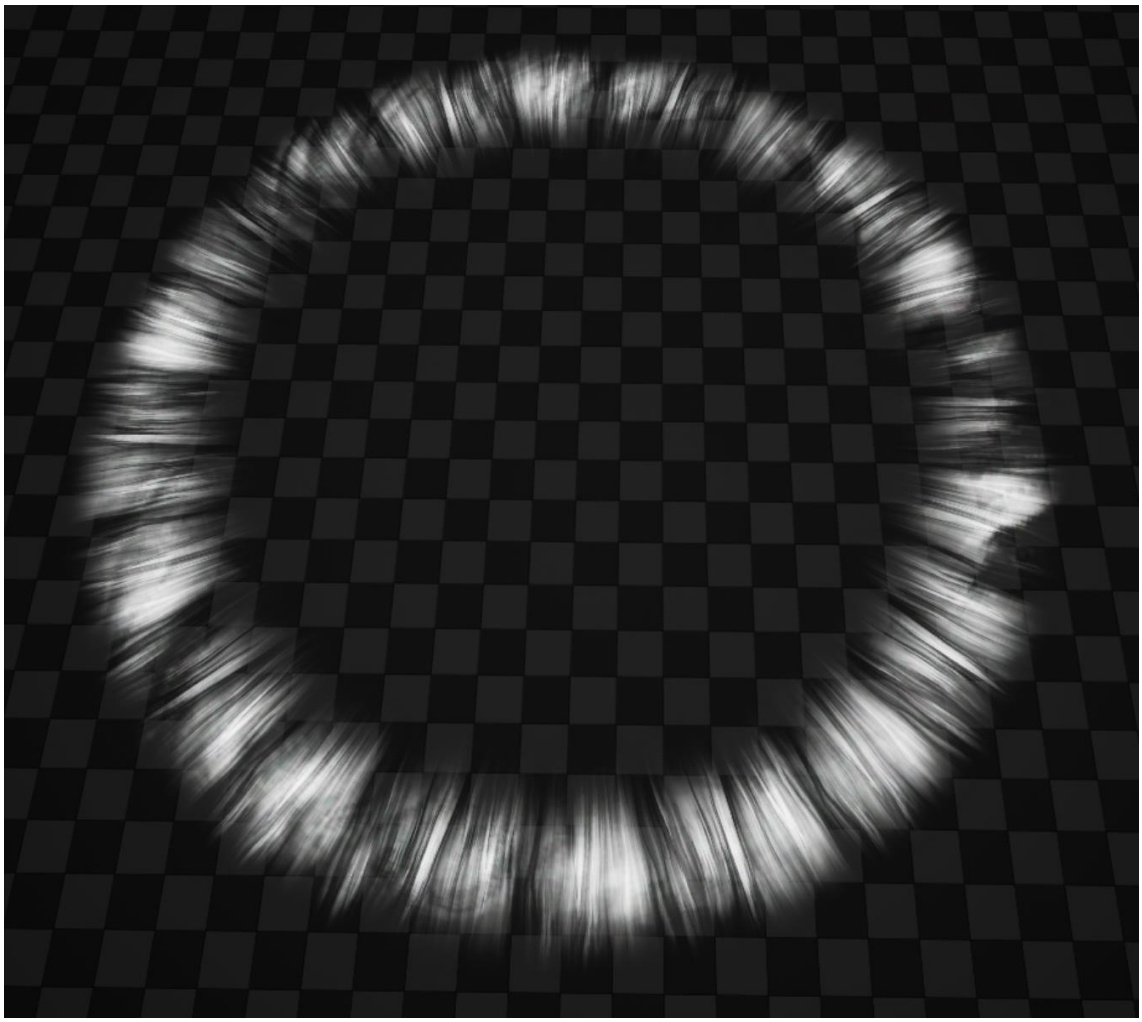
Kuva 64. Savun Dynaamisten materiaalien animointi (Filipp, 2025)

Näiden askeleiden lopputulos on tyylitelty savu/tuli räjähdys, mikä toimii pohjana tässä efektissä (Kuva 68).



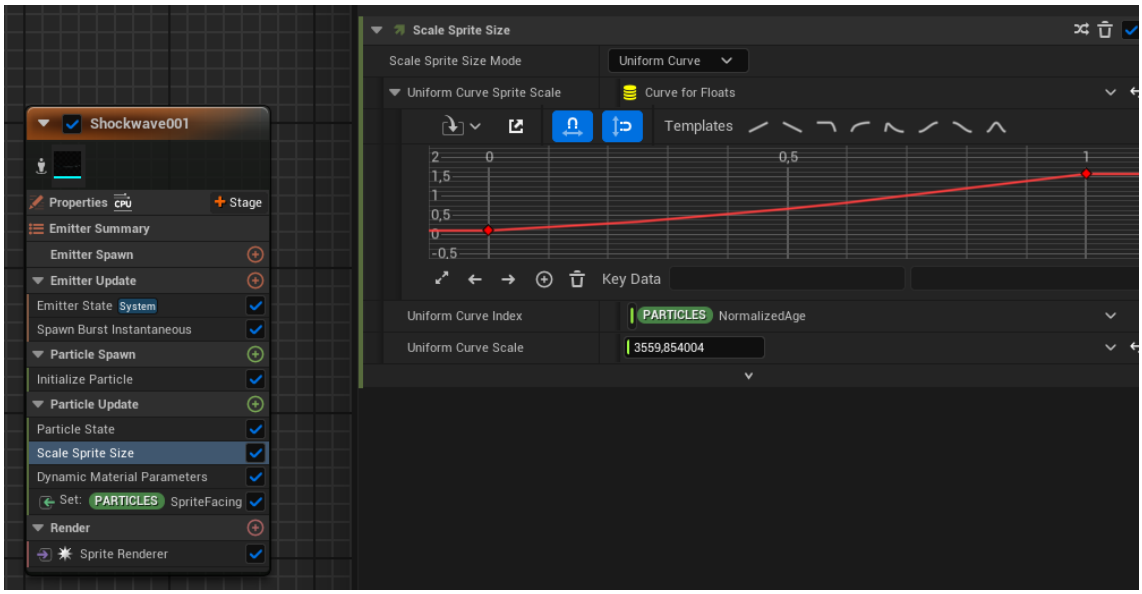
Kuva 65. Savupartikkelit viewportissa (Filipp, 2025)

Paineaalto syntyy laajentuvasta atlas gradientti tekstuurista. Esimerkkiefektissä 1 luotu materiaali instanssoidaan ja parametrien arvoja muokkaamalla saadaan aikaan rengasmainen paineaalto, joka laajenee ulospäin räjähdyksen keskipisteestä (Kuva 66).



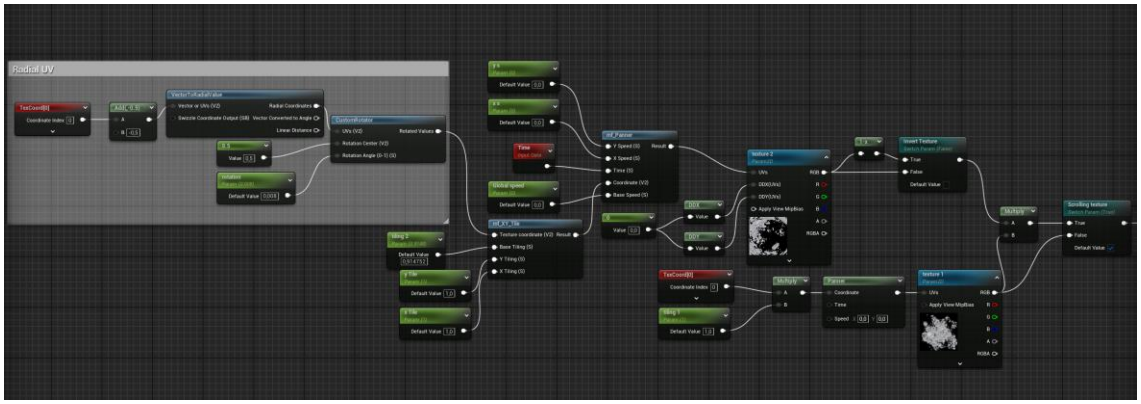
Kuva 66. Paineaalto viewportissa (Filipp, 2025)

Yksi emitteri luo laajentuvan Spriten käyttäen aiemmin tehtyä häikäisevä valo materiaalia. Sprite asetetaan vaakatasoon, ja sen skaalausta animoidaan scale moduulilla, jolloin syntyy räjähdyshetken tyylitelty paineaalto (Kuva 67).

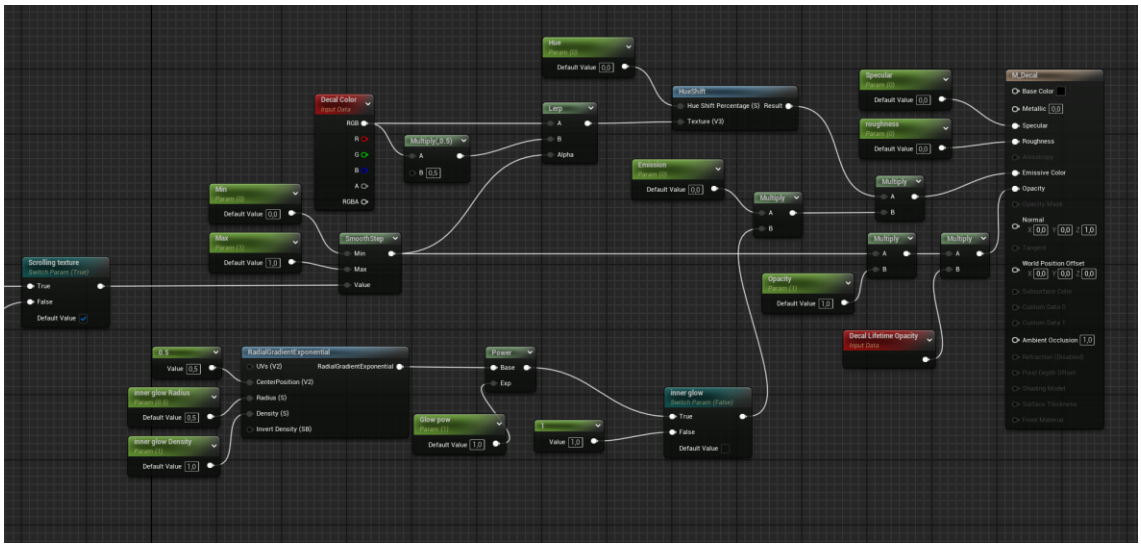


Kuva 67. Paineaallon hiukkaslähde (Filipp, 2025)

Maan halkeama efektiä varten käytetään dekaali materiaalia, jossa yhdistyy säteittäinen UV kartoitus, tekstuurien sekoitus ja Smoothstep maskit halkeamien muotoilemiseen (Kuvat 68 ja 69). Decal Lifetime Opacity ja Decal Color nodet mahdollistavat dekaalin materiaalin animaation Niagarassa (Kuva 69).

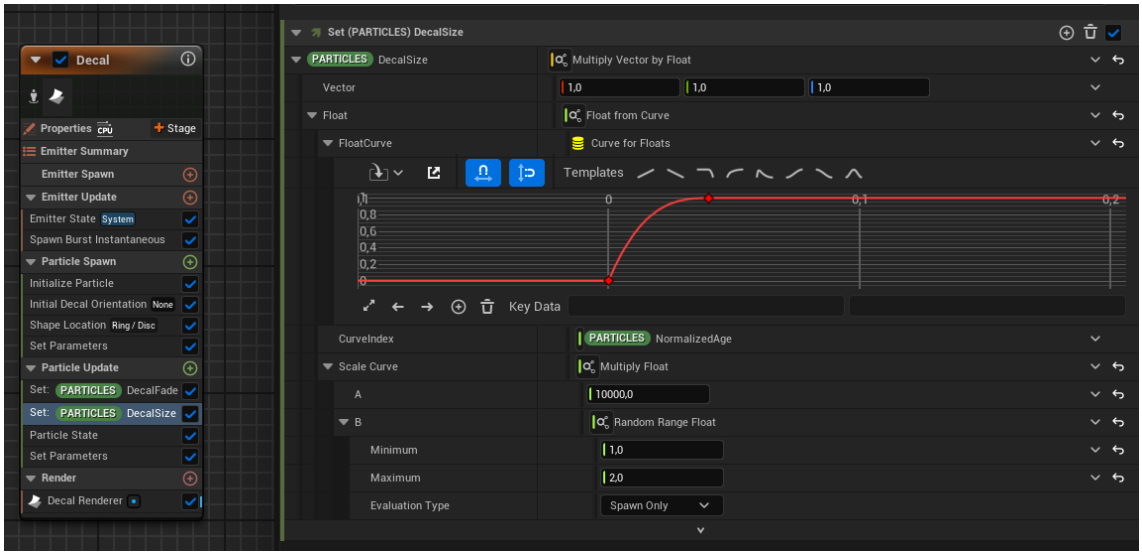


Kuva 68. Decal materiaalin nodekokoonpanon osa 1 (Filipp, 2025)



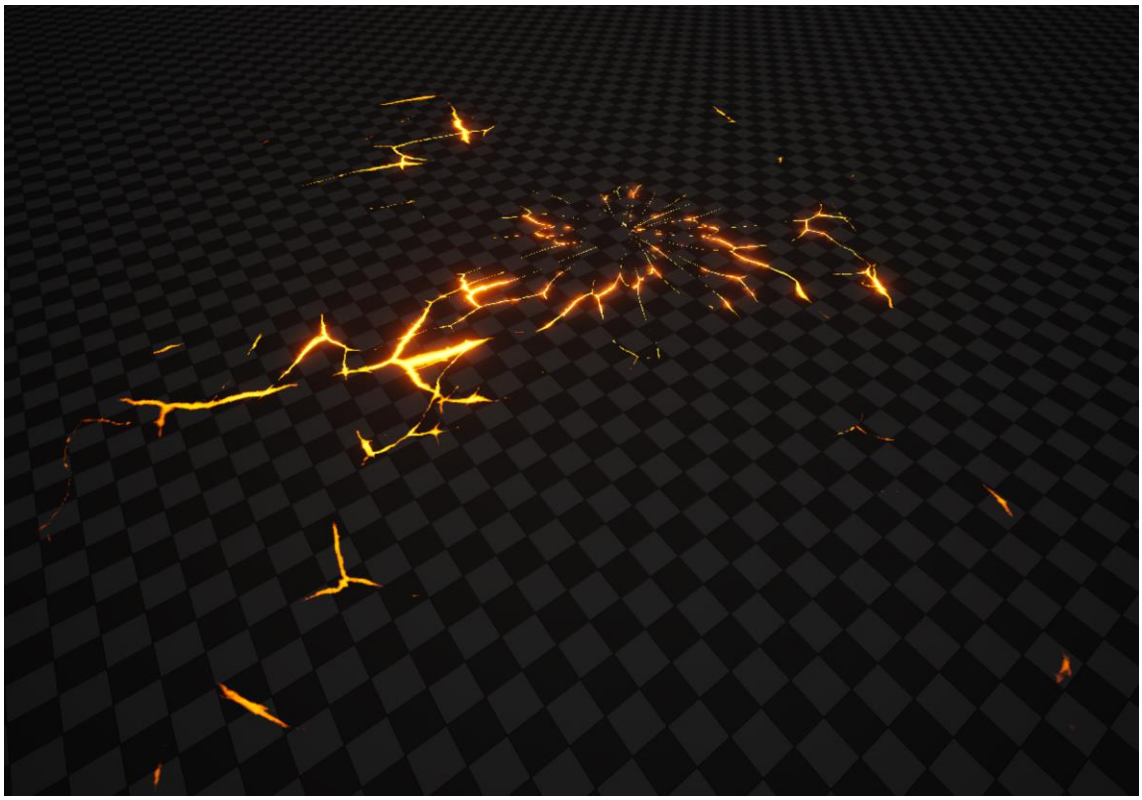
Kuva 69. Decal materiaalin nodekokoonpanon osa 2 (Filipp, 2025)

Niagaran puolessa käytettään aikaisemmin luodut Decal Lifetime Opacity ja Decal Color parametrit. Tässä tapauksessa nämä parametrit asetetaan suoraan moduuleissa (Kuva 70).



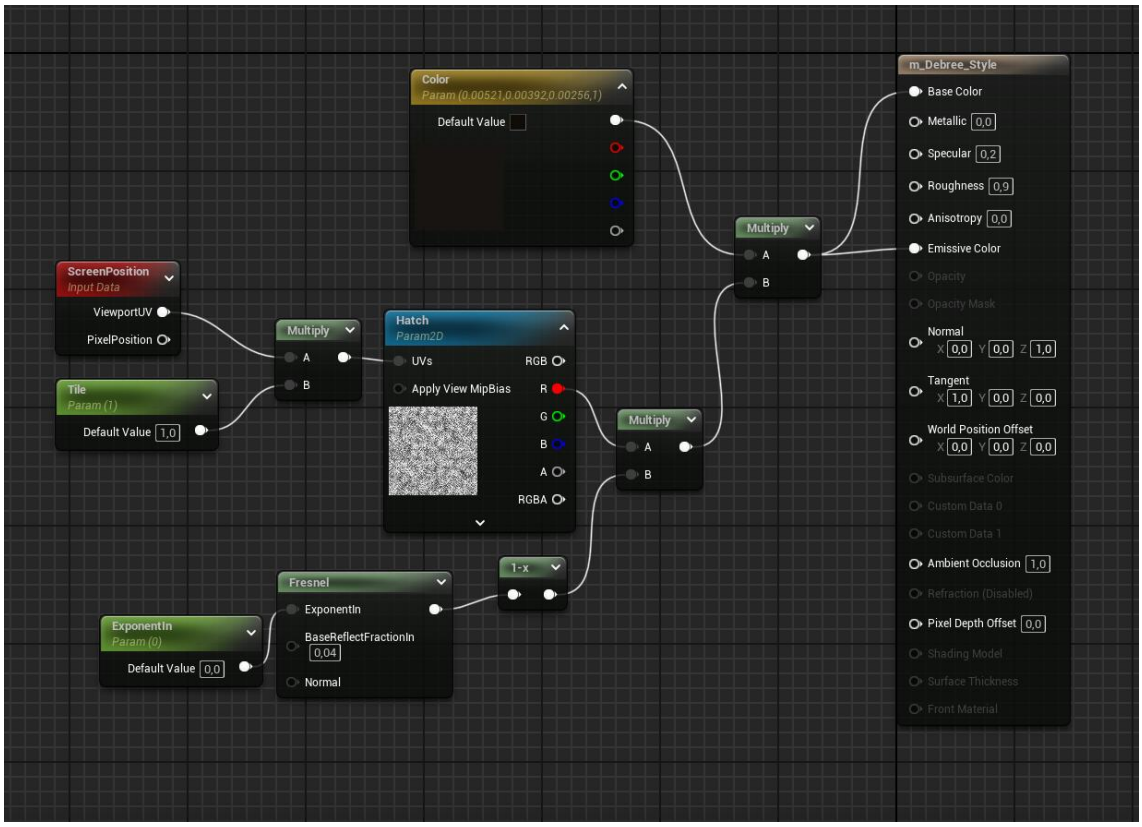
Kuva 70. Maan halkeamaefektin hiukkaslähde (Filipp, 2025)

Lopputuloksena on laajenevat hehkuvat halkeamat (Kuvassa 71).



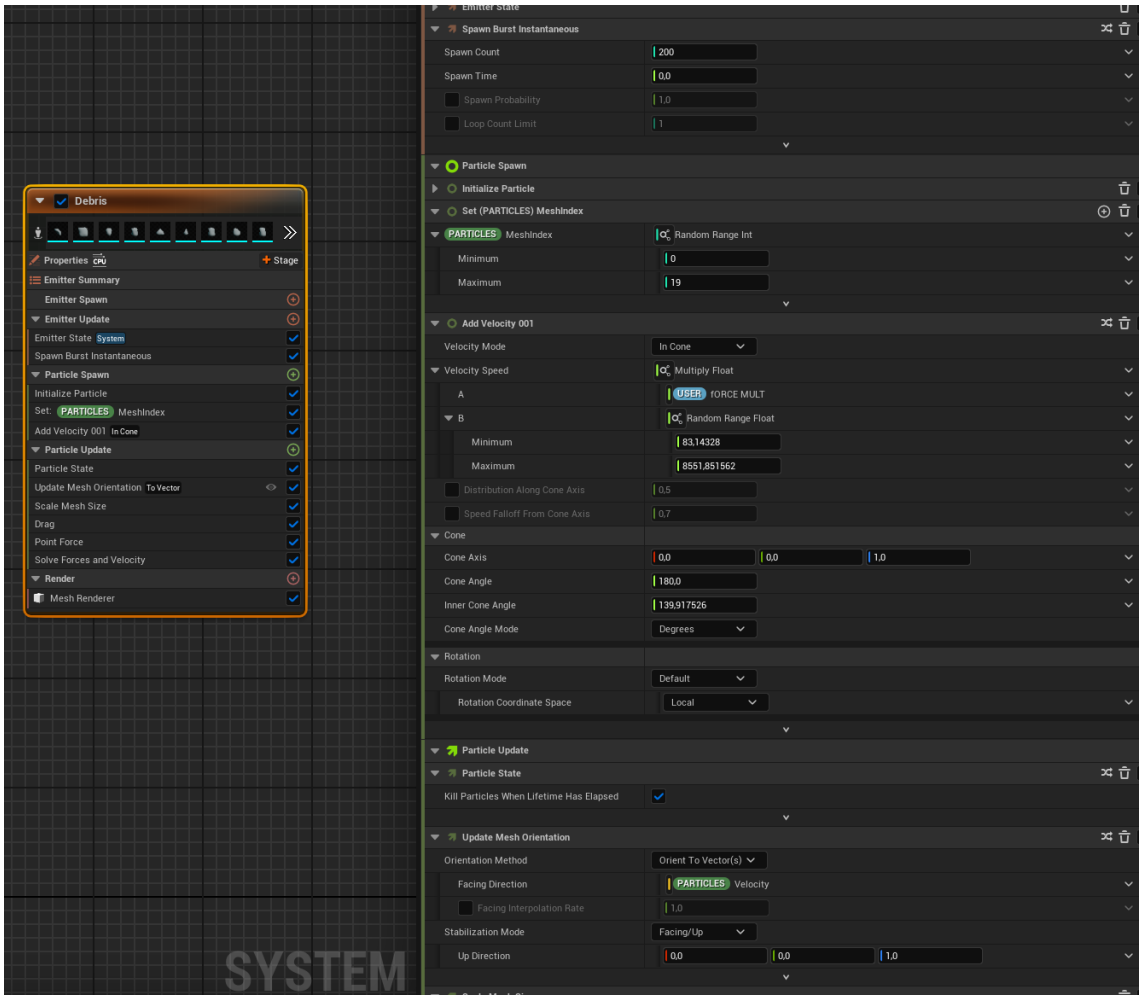
Kuva 71. Halkeamat viewportissa (Filipp, 2025)

Sirpaleiden uskottavuus syntyy räjähdyksessä sinkoutuvista 3D mesh partikkeleista. Sirpaleiden materiaali perustuu ruudun tilan viivavarjostustekstuurin (crosshatch) käyttöön, mikä antaa kiinteän ja tyylitellyn ilmeen (Kuva 72).



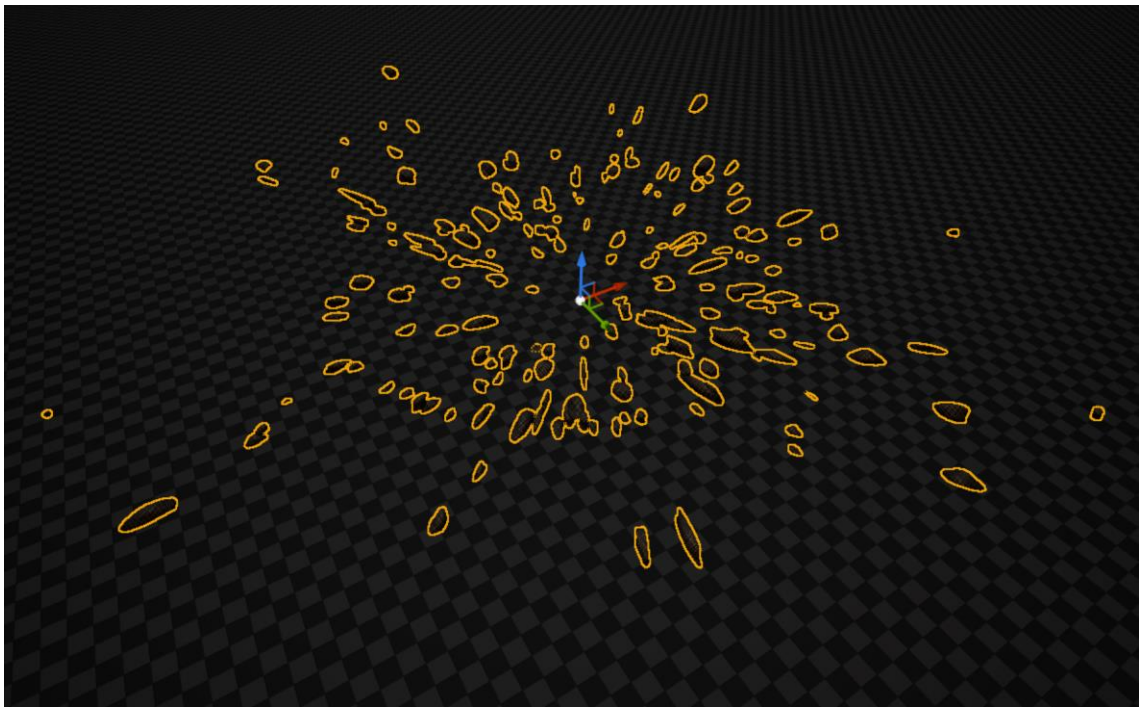
Kuva 72. Sirpaleiden materiaali (Filipp, 2025)

Etukäteen olen mallintanut 20 erilaista low poly kiveä ja käytin ne Niagara mesh renderer moduulissa. Satunnaistin kivien esiintymisjärjestyksen satunnaistamalla partikkeleiden indeksiarvon Set Parameter moduulissa (Kuva 72).



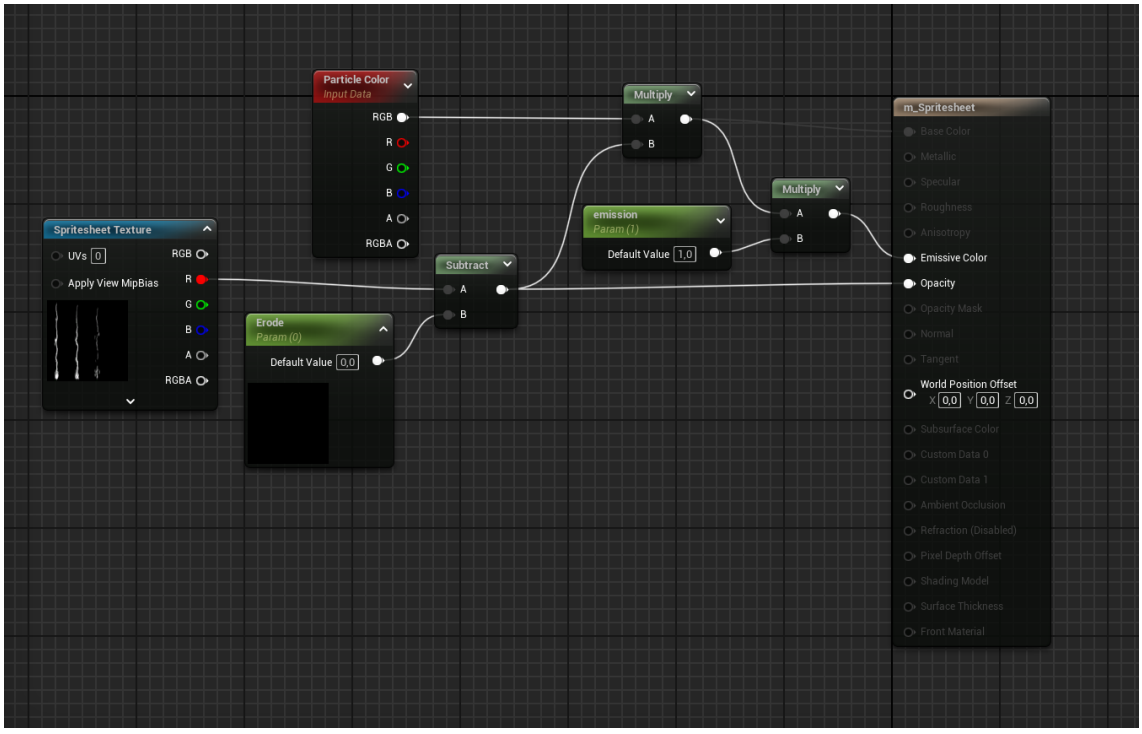
Kuva 72. Sirpaleiden hiukkaslähde (Filipp, 2025)

Lopputulos on keskipisteestä lentävät tyylitellyt kivipartikkelit (Kuva 73).



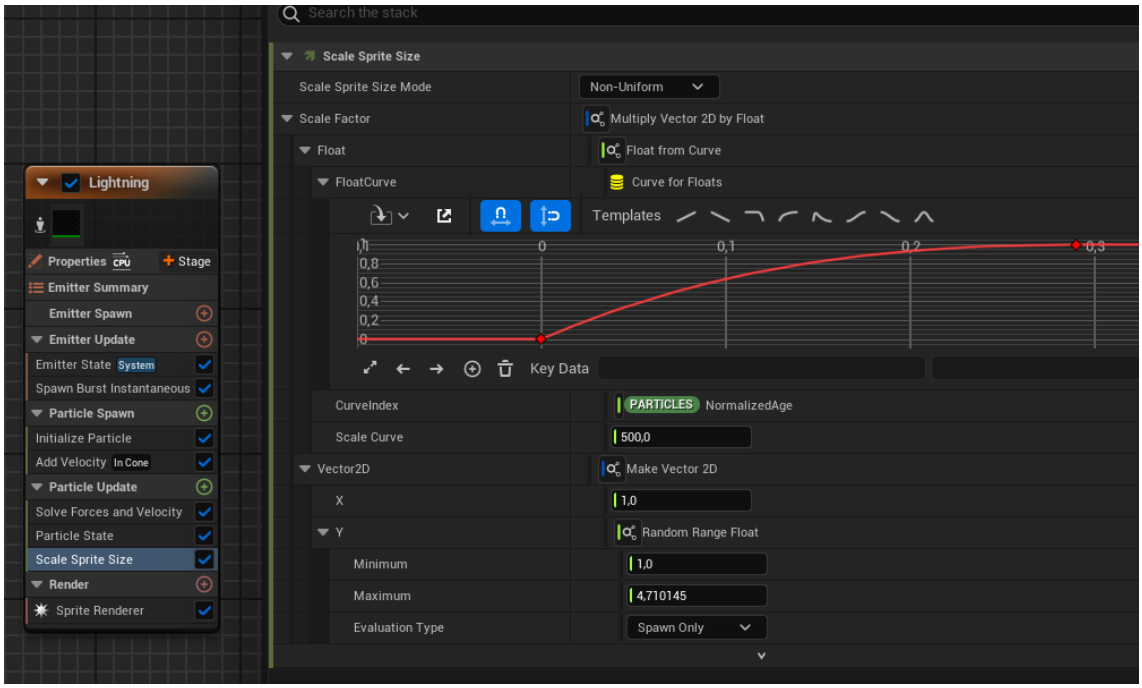
Kuva 73. Sirpaleet viewportissa (Filipp, 2025)

Viimeinen hiukkaslähde on salamat, niitä saa tehtyä yksinkertaisella Sprite materiaalilla, jossa on erode, emission ja color parametrit (Kuva 74).

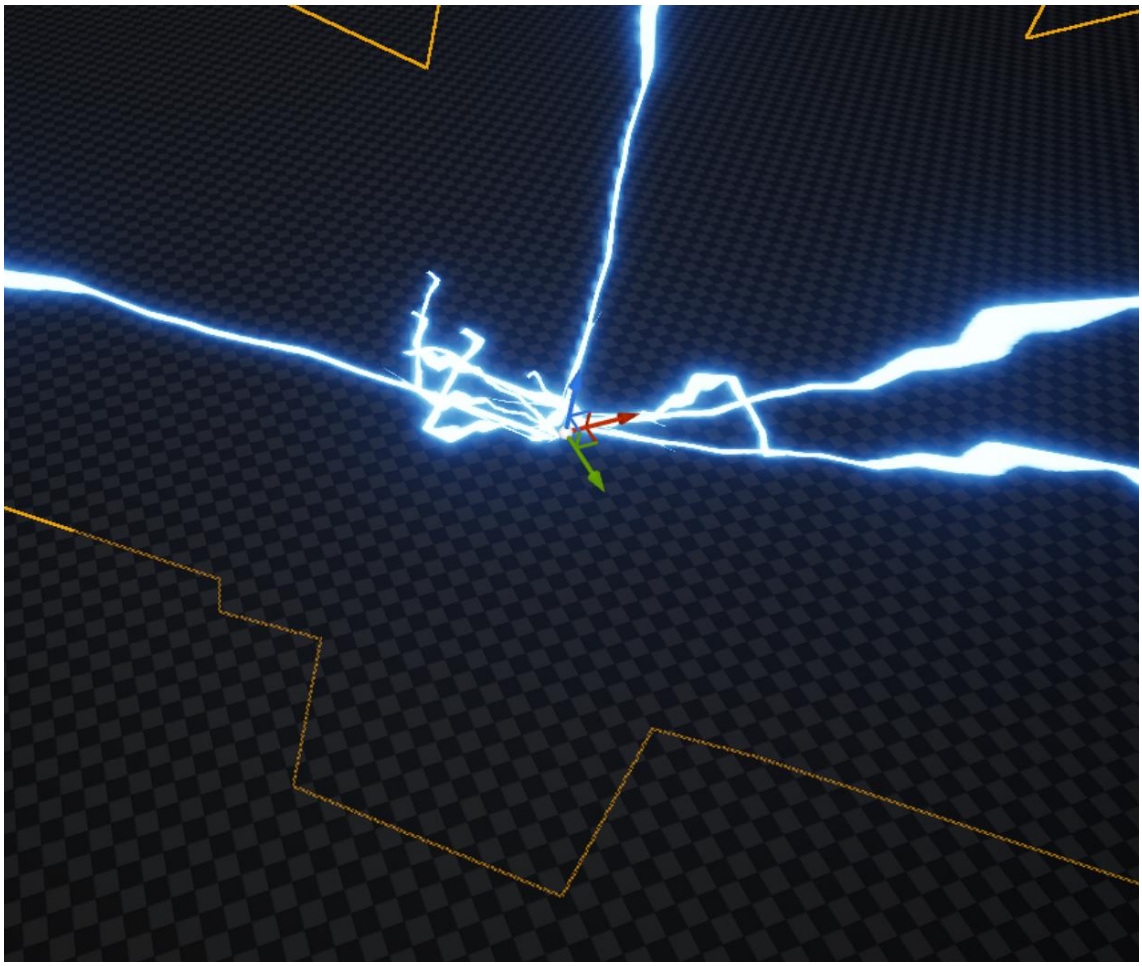


Kuva 74. Yksinkertainen Sprite materiaali (Filipp, 2025)

Hiukkaslähde tuottaa suuria yhtenäisiä Spritejä muutamalle kehykselle iskua korostamaan (Kuvassa 75), ja niiden animaatio synkronoidaan jälkikäsittelymateriaalin kanssa (Kuva 76 ja 92).

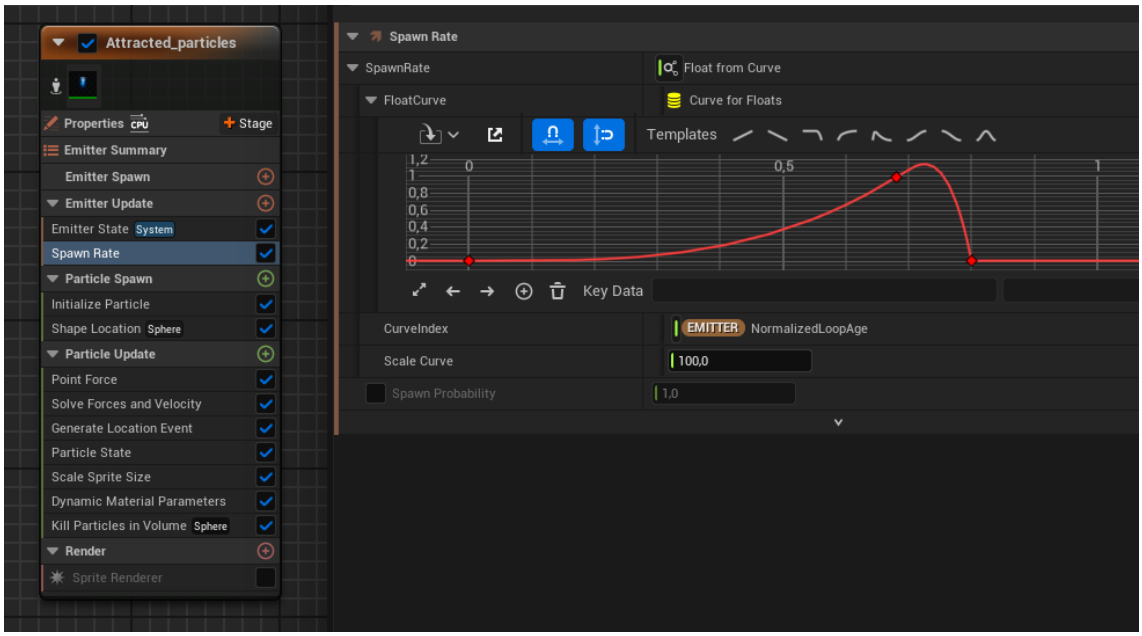


Kuva 75. Salamoiden hiukkaslähde (Filipp, 2025)

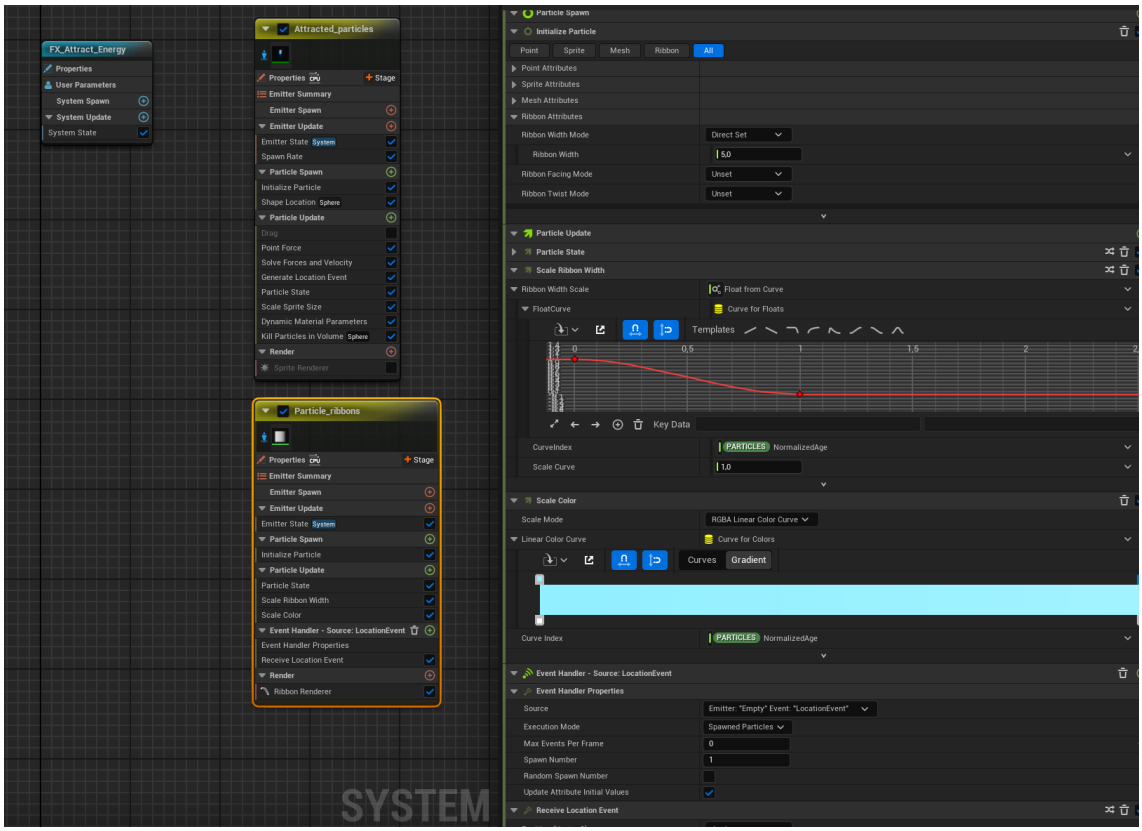


Kuva 76. Salamat viewportissa (Filipp, 2025)

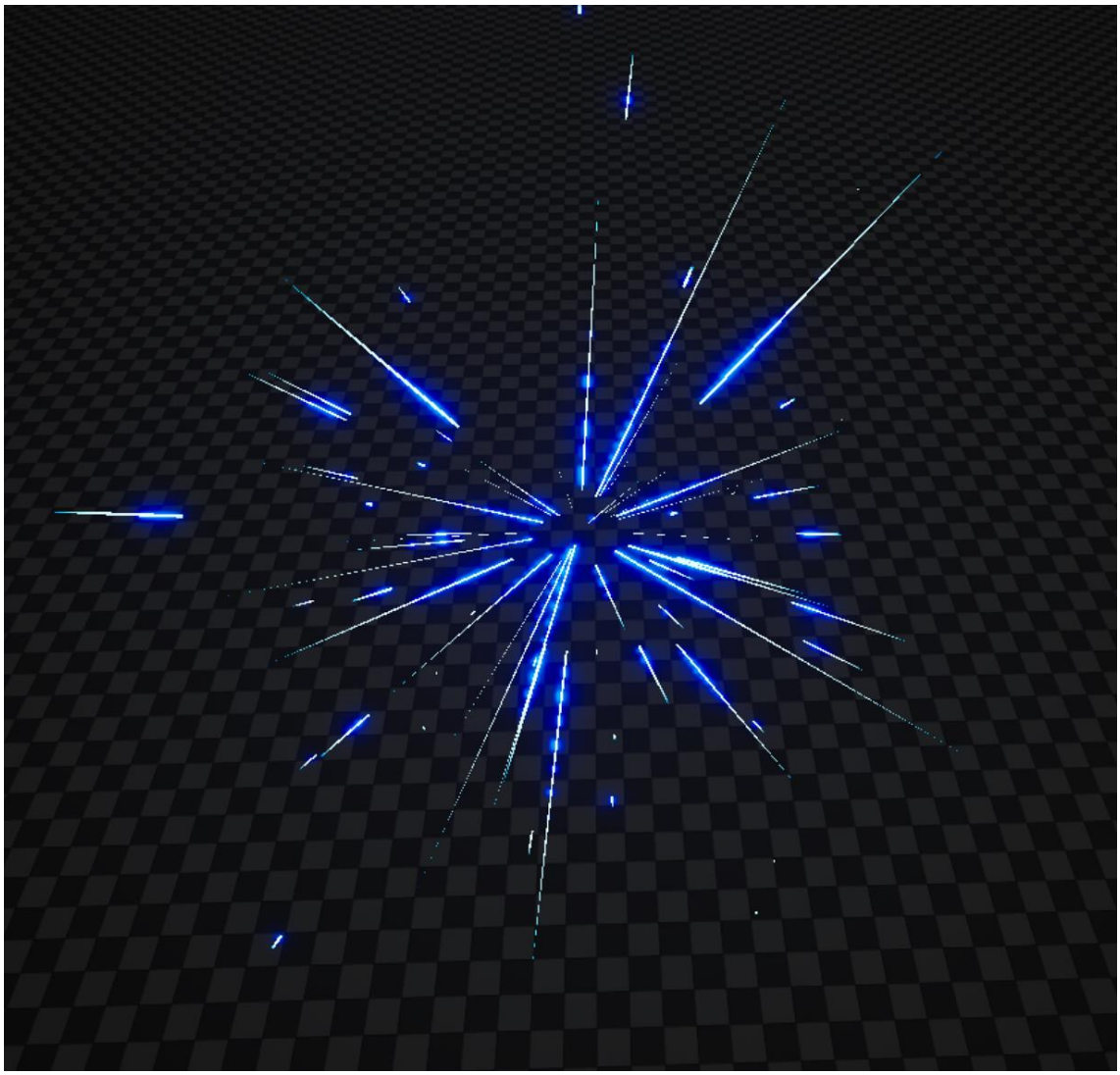
Ennakointiefektissä on kaksi hiukkaslähdettä: liukuvat energianauhat (Ribbon partikkelit) ja keskipisteeseen imevä energia aura. Ensimmäinen hiukkaslähde muodostaa nauhoja käyttäen Ribbon rendereria ja Event Handler moduulia, jolla luodaan Generate ja Receive tapahtumia (Kuva 77 ja 78). Partikkeleiden syntymisnopeuden sidotaan emitterin normaaliin elinkaareen, jolloin energian määrä kasvaa asteittain (Kuvassa 77). Nauhojen värisävyjä animoidaan Scale Color moduulilla (Kuva 78). Lopputulos näkyy kuvassa 79.



Kuva 77. Ribbon partikkeleiden syntymis- ja lokaatio hiukkaslähde (Filipp, 2025)

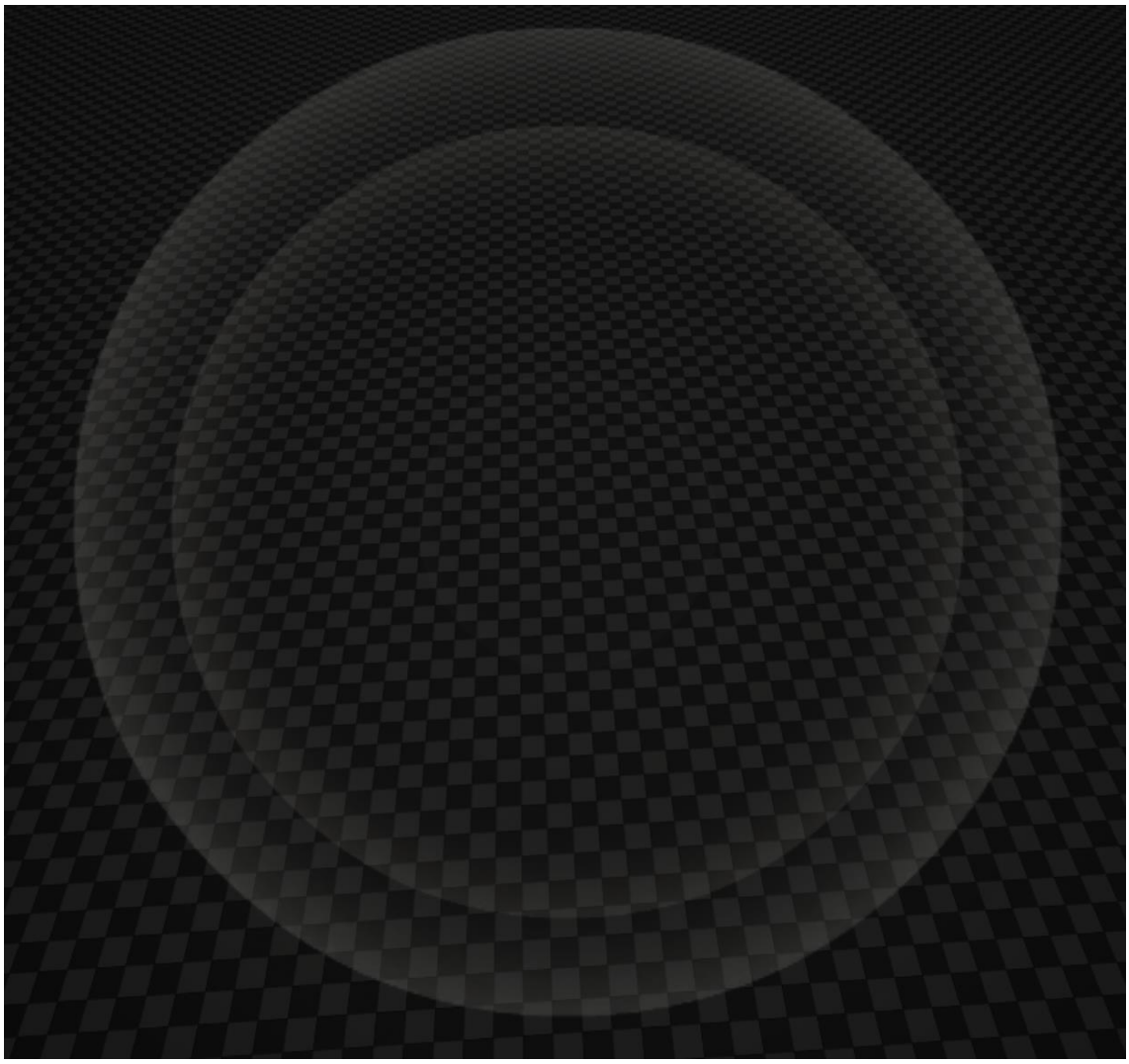


Kuva 78. Ribbon hiukkaslähde (Filipp, 2025)



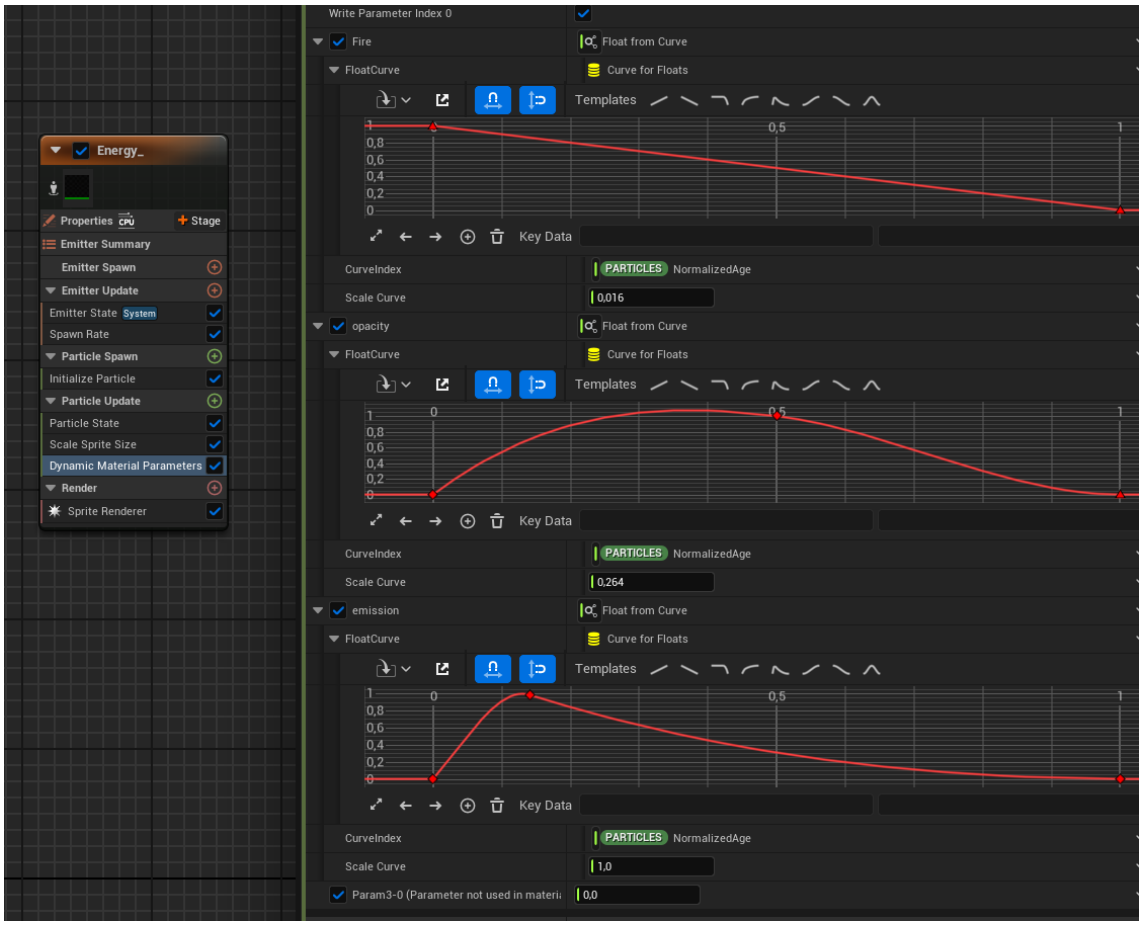
Kuva 79. Sisäänpäin menevät ribbon partikkelit (Filipp, 2025)

Toinen hiukkaslähde tuottaa yhden keskeisen partikkelin samaa materiaalia käyttäen ja animoi sen läpinäkyvyyttä, jolloin syntyy sisäänpäin imevä energia-efekti (Kuva 80).



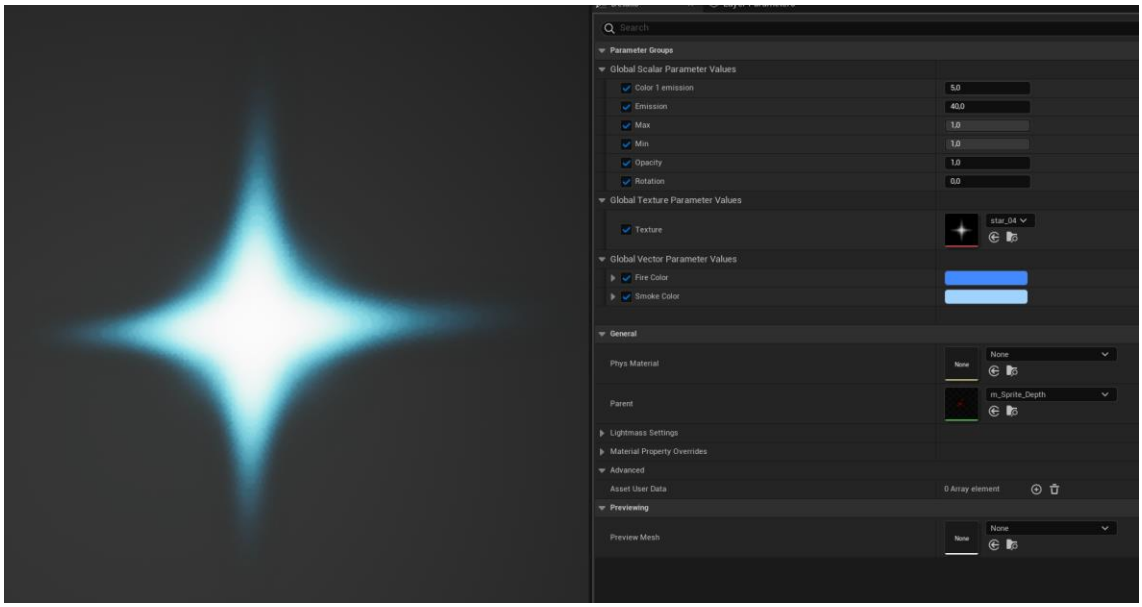
Kuva 80. Sisäänpäin imevä energian auraefekti (Filipp, 2025)

Energian auran hiukkaslähteen moduulit ja parametrit näkyvät kuvassa 81.

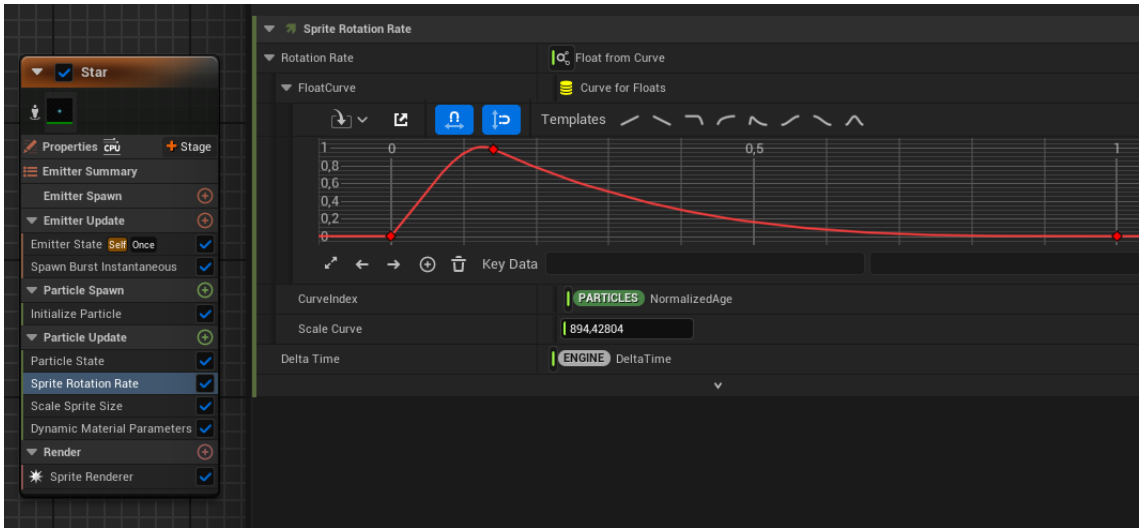


Kuva 81. Energia auran hiukkaslähde (Filipp, 2025)

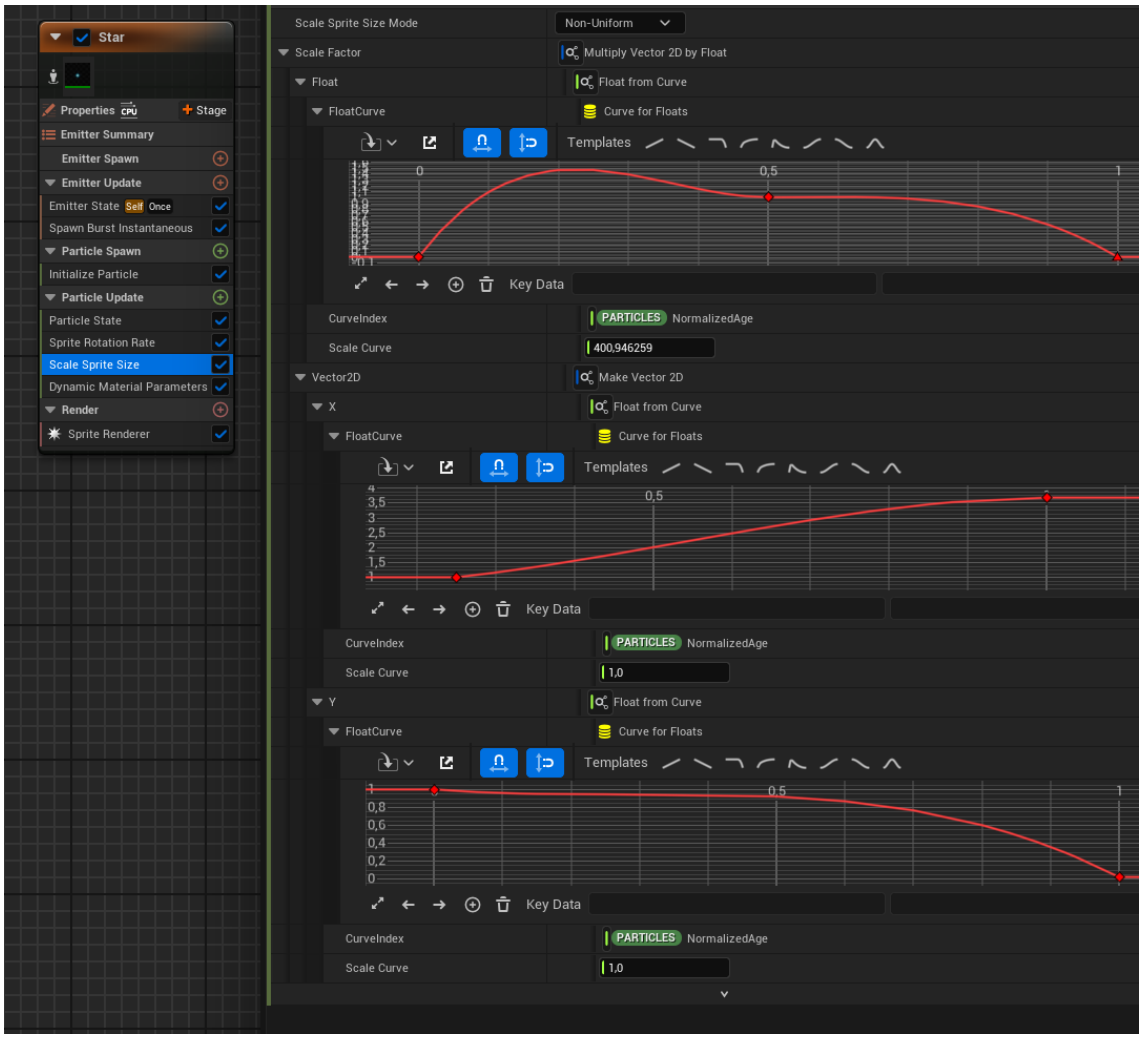
Ennen pääräjähdystä luodaan vilkahdusefekti. Materiaalina käytetään tähtimaskia ja samanlaista Sprite ratkaisua kuin savumateriaalissa (Kuvassa 82). Emitteri hyödyntää Ribbon renderer moduulia, Sprite rotationRate ja Sprite scale moduuleja sekä Dynamic Material Parameters animointia (Kuvat 83 ja 84).



Kuva 82. Tähden materiaali instanssi (Filipp, 2025)

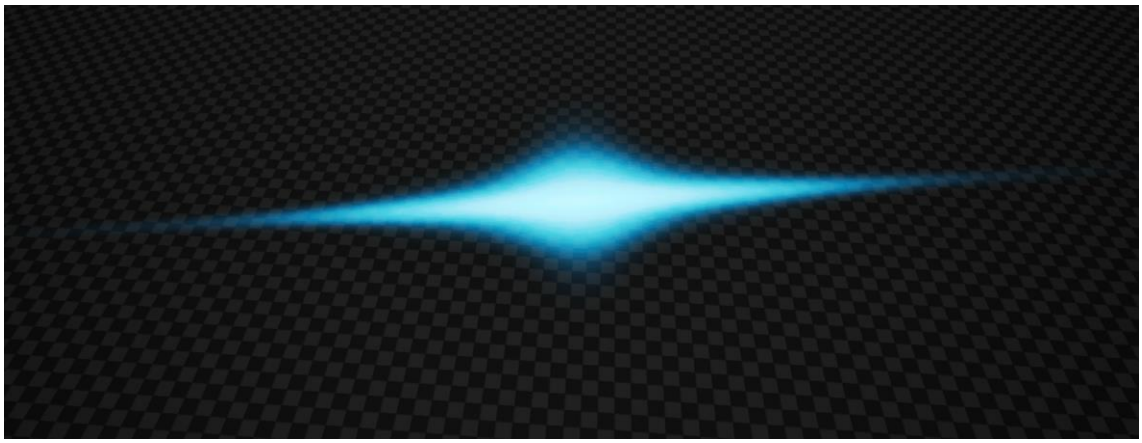


Kuva 83. Tähden RotationRate moduuli (Filipp, 2025)

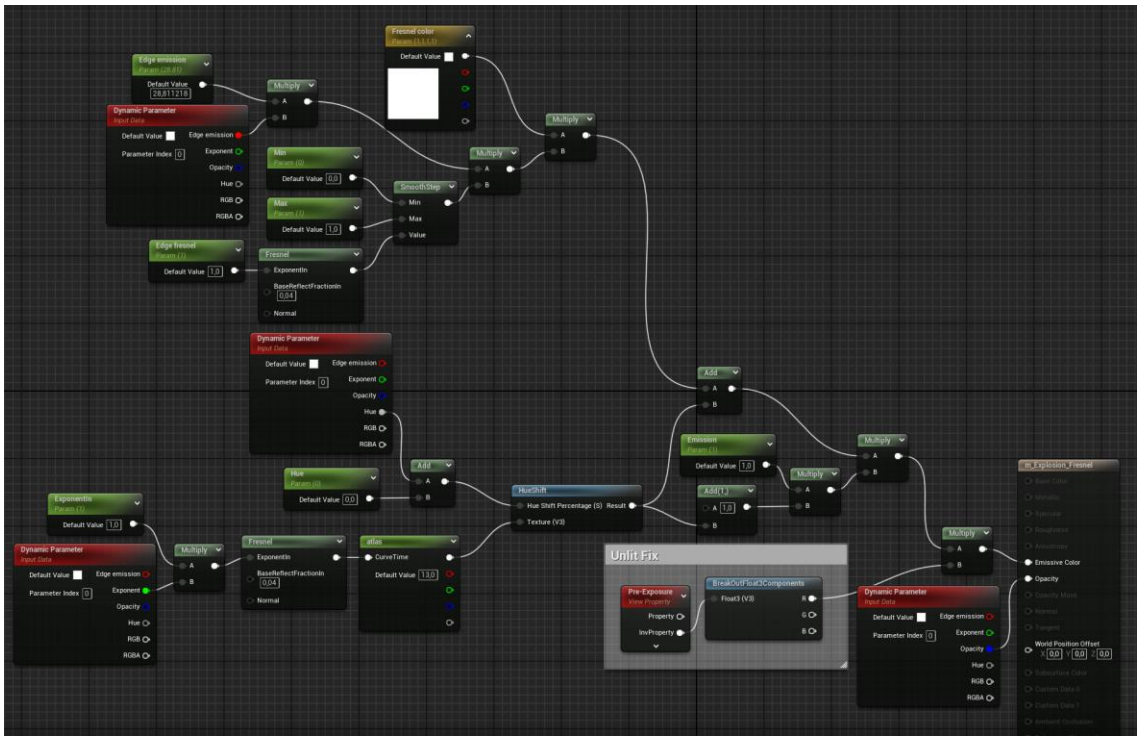


Kuva 84. Tähden skaalaus animaatiokäyrät (Filipp, 2025)

Tämä yhdistelmä seurailee animaation squash ja stretch periaatteita ja luo hetkellisen kirkkautensa ansiosta erottuvan vilkahdushetken (Kuva 85).

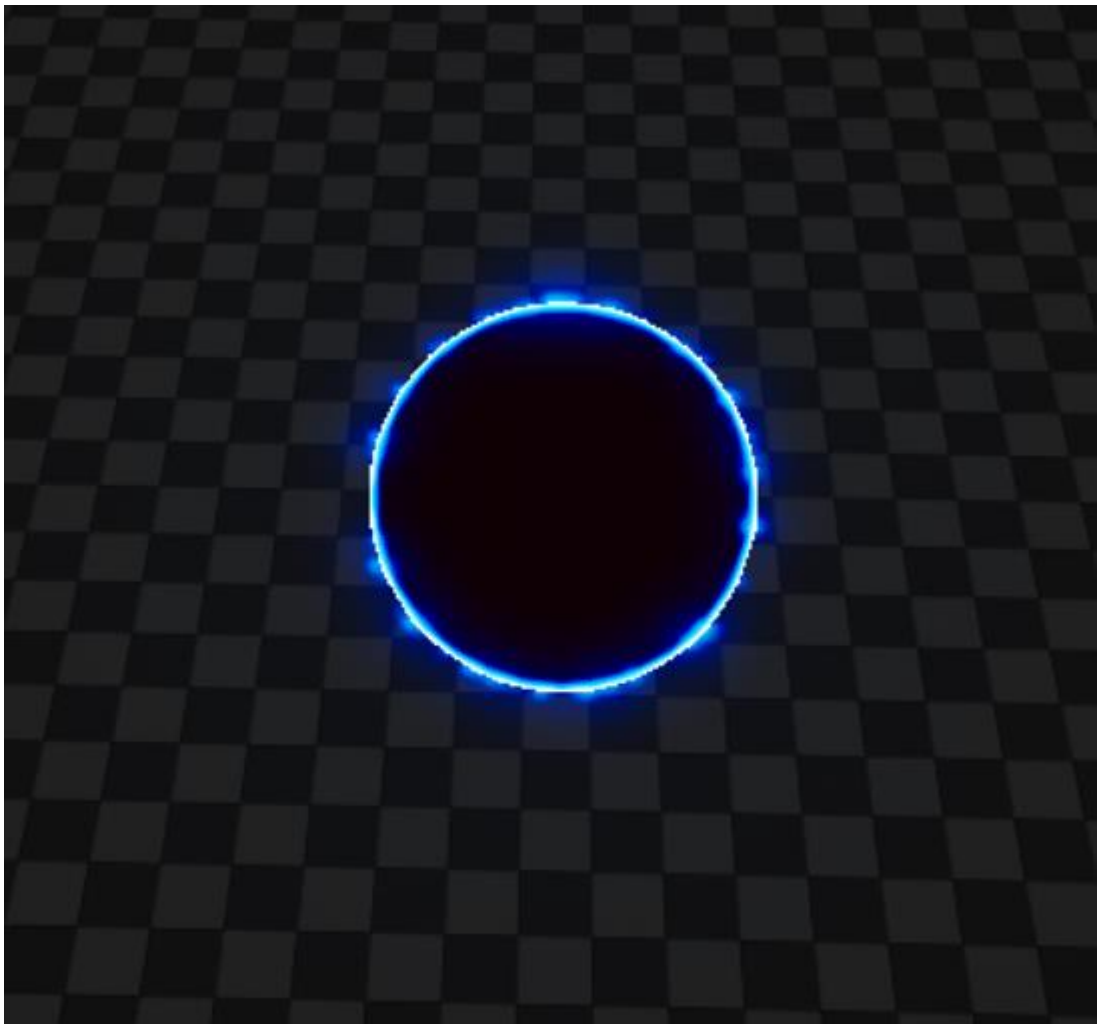

Kuva 85. Vilkahdusefekti viewportissa (Filipp, 2025)

Lopuksi keskuspisteessä laajenee pallomuotoinen ydin. Material Editorissa hyödynnetään atlas gradientteja ja Fresnel sekoitusta reunahehkua varten (Kuva 86). Emitterissa animoidaan ytimen kokoa ja sen väriparametreja, jolloin syntyy vaikutelma räjähdyksen energiasta purkautuvasta ytimestä (Kuva 87).



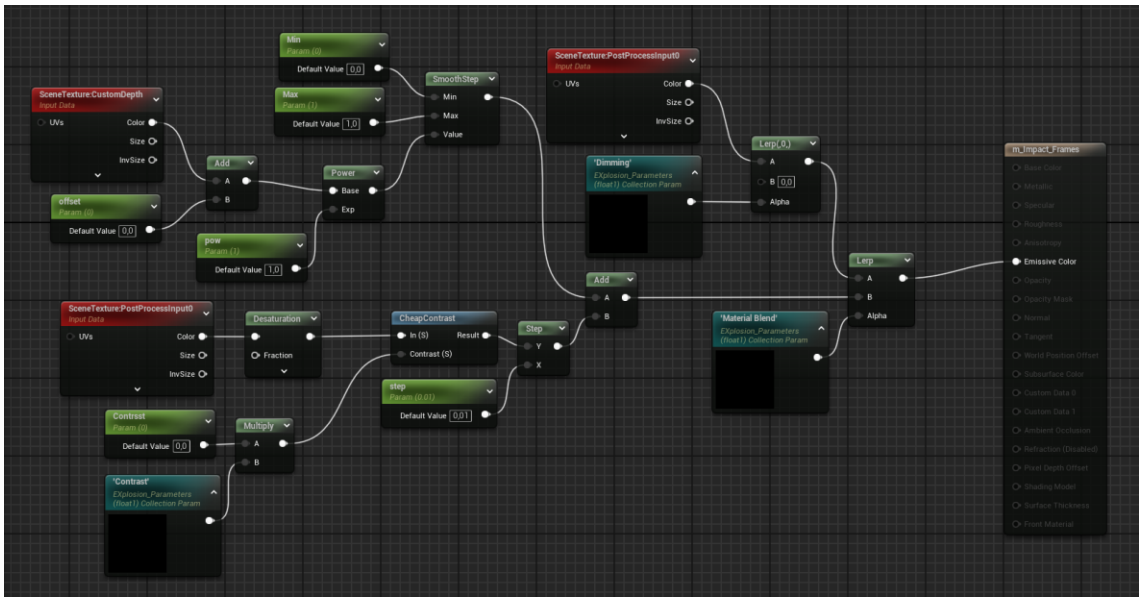
Kuva 86. Ytimen materiaali (Filipp, 2025)

Kuva 87. Ytimen hiukkaslähde (Filipp, 2025)

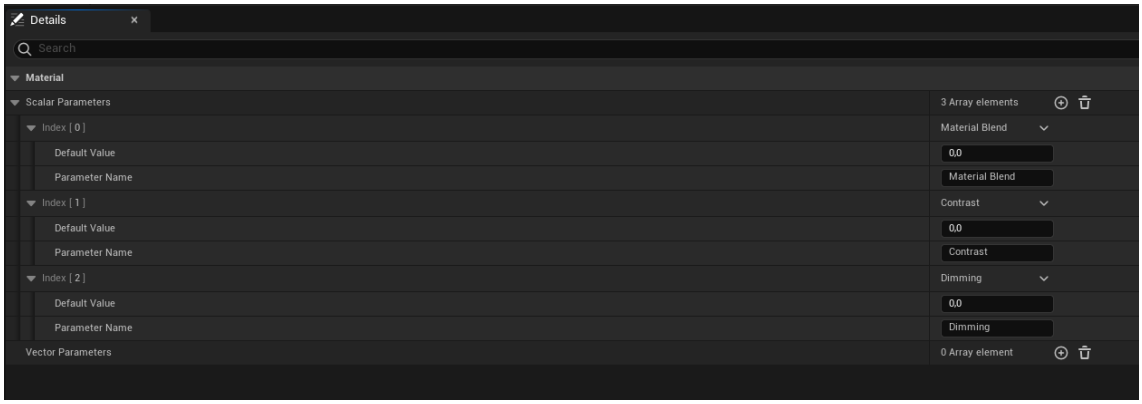


Kuva 88. Ydin viewportissa (Filipp, 2025)

Iskukuvakehys (impact frame) jälkikäsittelymateriaali kiihdyttää animaation dramaattisuutta. Post process materiaalissa. Siinä käytetään Custom Depth maskia, kontrastin korostusta ja Material Parameter Collection asetuksia (Kuvat 89 ja 90), jotka tuottavat voimakkaan mustavalkoisen kontrastiefektin juuri räjähdyksen alussa (Kuva 92).



Kuva 89. Iskukuvakehyksen Post Process materiaali (Filipp, 2025)



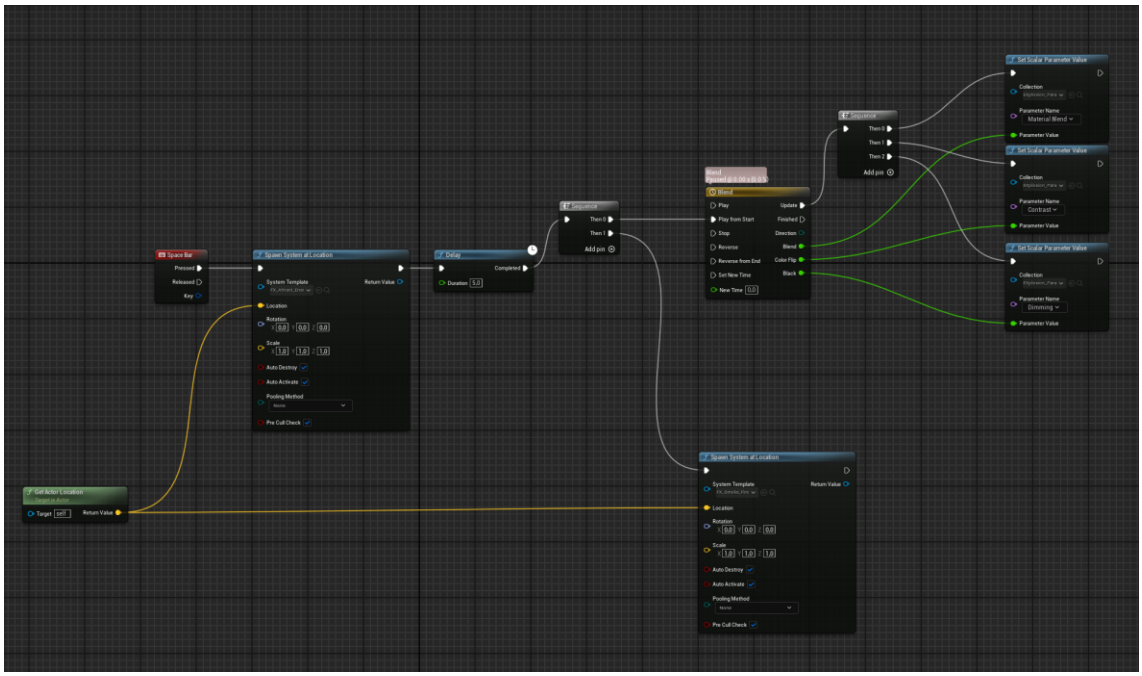
Kuva 90. Iskukuvakehyksen material parameter collectionin parametrit (Filipp, 2025)

Post process materiaalit täytyy lisätä Post Process Volume objektiin, jotta materiaali olisi näkymässä viewportissa (Kuva 91).

Kuva 91. Post process materiaali Post process volumessa (Filipp, 2025)

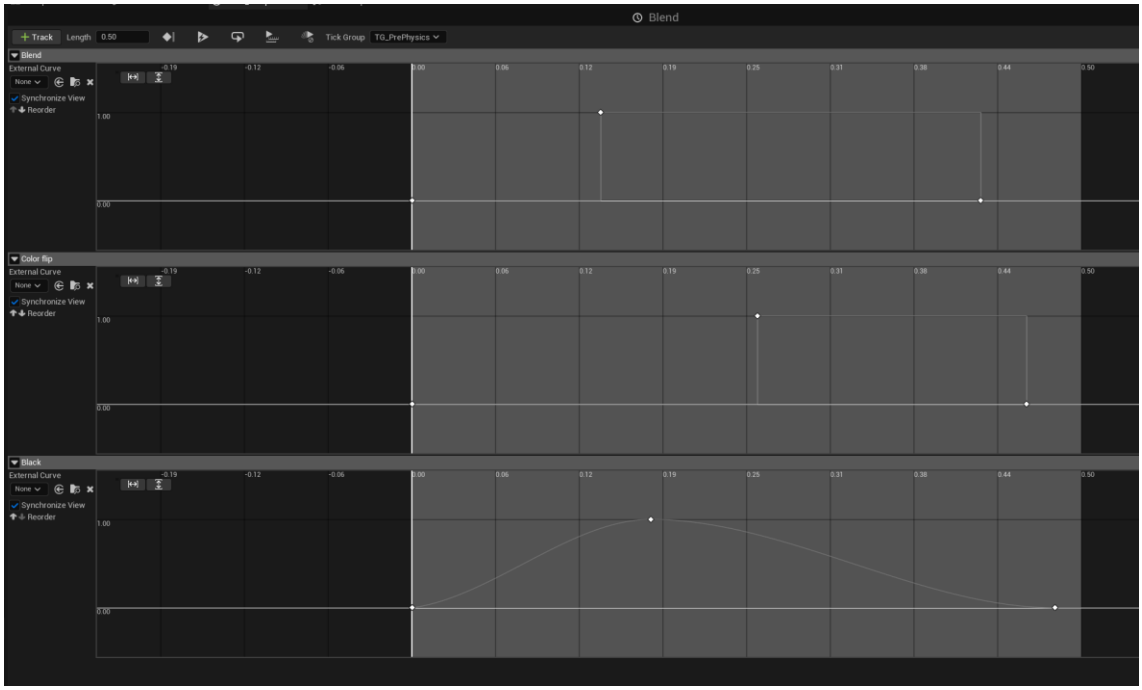
Kuva 92. Iskukuvakehys viewportissa (Filipp, 2025)

Blueprint skripti käynnistää koko efektiketjun painamalla esimerkiksi välilyöntiä. Ensin aktivoituu ennakointijärjestelmä, minkä jälkeen sopivalla viiveellä animaatio ja jälkikäsittelyprosessit käynnistyvät synkronoidusti (Kuva 93). Oikean ajoituksen ansiosta räjähdyksestä syntyy eeppinen, tyylitelty esitys, jossa iskukuvakehys ja varsinainen purkaus kohtaavat animaatioperinteen tavoin.



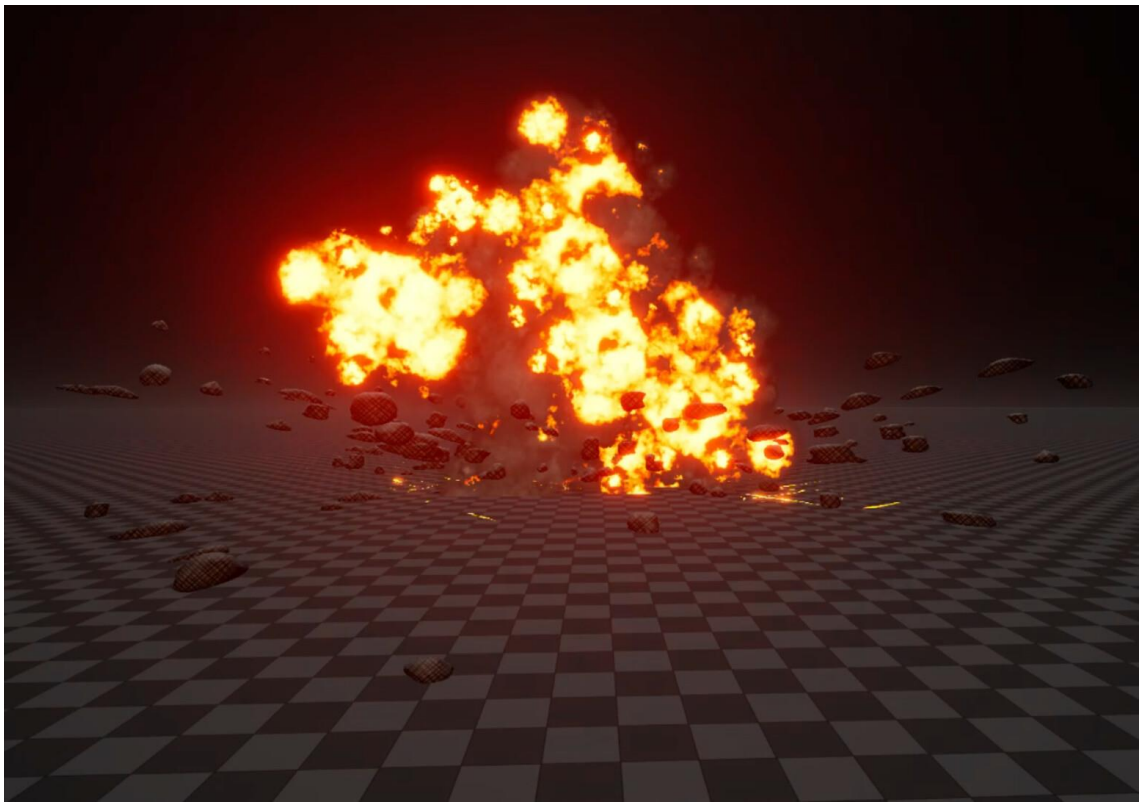
Kuva 93. Lopullinen Blueprint nodekokoonpano (Filipp, 2025)

Blueprintissa Timeline noden avulla kontrastia, himmennystä ja materiaalin näkyvyyttä animoidaan käyttäen sekä tasaisia että bezier keyframe interpolointia. (Kuva 94).



Kuva 94. Timeline noden animaatiokäyrät (Filipp, 2025)

Tässä esimerkissä tutustuttiin laajaan valikoimaan materiaali, Niagara ja Blueprint tekniikoita, joilla saadaan aikaan dynaaminen ja visuaalisesti erottuva räjähdys. Eri vaiheissa käsiteltiin savu ja tuliefektiä, paineaaltoa, dekaalia, sirpaleita, salamointia sekä ennakointi ja iskukuvaruutuefektejä. Lopputulos on muokattavissa, optimoitu ja helposti integroituva osa pelimaailmaa, joka korostaa tyyliteltyä visuaalista kerrontaa pelin keskeisenä elementtinä (Kuva 95). Lopputulos näkyy kuvassa 95 ja video efektistä pystyy kastoa avaamalla linkkiä efektiin liitteissä tai klikkaamalla kuvaa 95.



Kuva 95. (Valmis räjähdysefekti, Filipp 2025)

4.6 Käytännön osion yhteenveto

Käytännön osio on havainnollistanut, miten Unreal Engine 5:n Material Editor ja Niagara partikkelisysteemi yhdessä mahdollistavat tyyliteltyjen reaaliaikaisten tyyliteltyjen visuaalisten efektien luomisen. Ensimmäisessä esimerkissä opittiin radiaalinen UV kartoitus, pallogradienttien maskaus ja tekstuurin vieritys, mikä loi pohjan säteittäisille hehkuvasteille. Toisessa esimerkissä tutkittiin maskien ja

kerrostamisen periaatteita yhdistämällä kohinatekstuurit ja värigradientit dynaamiseksi tulijäljeksi. Kolmannessa lähestyttiin kasvillisuusefektejä hyödyntäen World Position Offset tuuliefektiä, per instanssi satunnaisia värisävyjä ja etäisyyteen perustuvaa haalistumista. Neljännessä esimerkissä taas yhdistettiin materiaali ja Blueprint animaatiot Niagaran kehittyneisiin partikkelimoduulien ominaisuuksiin, jolloin syntyi näyttävä ja räätälöitävä räjähdysefekti.

Näissä harjoituksissa käydyt tekniikat UV manipuloinnista ja tekstuurien muokkauksesta materiaalifunktioihin sekä Niagara integraatioon asti muodostavat tyyliteltyjen reaaliaikaisten VFX efektien selkärangan Unreal Enginessä. Näiden menetelmien hallinta antaa taiteilijoille työkalut luoda sekä teknisesti optimoituja että visuaalisesti vaikuttavia tehosteita, jotka syventävät pelaajan immersiota ja tukevat pelin kerrontaa. Eri tekniikoiden yhdistäminen avaa loputtomia mahdollisuuksia luovuudelle, kun taas ymmärrys niiden suorituskykyvaikutuksista varmistaa tasapainon visuaalisen laadun ja pelin toimivuuden välillä.

5 Opinnäytetyön yhteenveto

Opinnäytetyön tavoitteena oli kannustaa aloittelija ja harrastusartisteja sekä muita VFX tuotannosta kiinnostuneita tutustumaan tyyliteltyjen reaaliaikaisten tehosteiden maailmaan Unreal Engine 5:ssä. Aluksi käsiteltiin VFX:n perusperiaatteet, luokittelut ja teoreettiset lähtökohdat, minkä jälkeen syvennyttiin Unreal Enginen materiaalien, shaderien, tekstuuritekniikoiden ja Niagara partikkelisysteemin ratkaisuihin. Blueprint integraatio puolestaan osoitti, miten efektien parametreja voi muuttaa lennossa ja siten synkronoida materiaalit, partikkeliefektit ja pelilogiikka saumattomasti.

Neljä käytännön esimerkkiä säteittäinen hehku, ribbon jälki, dynaaminen ruohomateriaali ja tyylitelty räjähdys havainnollistivat teoreettisten menetelmien soveltamista askel kerrallaan. Jokainen projekti käsitteli erilaiset menetelmät: UV kartoitukset ja maskit, kohinatekstuurien kerrostus, World Position Offset tuuliefektit, dynaamiset materiaaliparametrit ja Blueprint animaatiot. Näin luotiin paitsi visuaalisesti vaikuttavia, myös teknisesti optimoituja ja helposti muokattavia visuaalisia efektejä.

VFX efektien merkitys peleissä ulottuu esteettisestä houkuttelevuudesta aina pelaajan vuorovaikutustiedon välittämiseen ja maailmankuvan rakentamiseen saakka. Unreal Engine 5:n nodepohjainen Material Editor sekä Niagara järjestelmän modulaarisuus antavat taiteilijalle vapauden kokeilla, yhdistellä ja iteratiivisesti kehittää omia ideoitaan ilman syvää ohjelmointiosaamista. Kokeilemalla ja muokkaamalla voi oppia parhaiten, joten rohkaisen tarttumaan työkaluihin, katselemaan esimerkkejä ja laajentamaan osaamistaan myös esitettyjen menetelmien ulkopuolelle.

Lisäoppimista varten kannattaa vierailla Unreal Enginen virallisen dokumentaation sivuilla, joista löytyy runsaasti tutoriaaleja, ohjeita ja viimeisimmät päivitykset efektityökaluihin. Uskalla siis syventyä, kokeilla ja luoda omia, ainutlaatuisia VFX elämyksiä!

Lähteet

3Dcoat 2025, What is UV Mapping?, https://3dcoat.com/articles/article/what-is-uv-mapping/ (Viitattu 26.04.2025)

Autodesk 2025, Planar UV mapping, https://help.autodesk.com/view/MAYAUL/2025/ENU/?guid=GUID-B6519472-C0ED-4C07-99C6-12107A3509D9 (Viitattu 26.04.2025)

Autodesk 2025, Texture tiling, https://help.autodesk.com/view/MOBPRO/2025/ENU/?guid=GUID-E41CC69C-2D23-4FB2-A794-4D0AB88A818B (Viitattu 26.04.2025)

Aguiar, Gabriel 2023. What is VFX for Games? Verkkovideo 2023. YouTube. 11:14 https://www.youtube.com/watch?v=7rXjWnI-U7I (Viitattu 26.04.2025)

Chabanova, Anastasia 2022, VFX – A New Frontier: The Impact of Innovative Technology on Visual Effects, https://westminsterresearch.westminster.ac.uk/download/8d42780143c5e9a41295a72a9c253dc205a16f512fa59450dc7ab1001d6c30d7/3755820/Final-thesis-submission-Examination-Miss-Anastasiia-Chabanova%20%281%29.pdf (Viitattu 26.04.2025)

Epic Games, Unreal Engine Documentation, 2025, https://dev.epicgames.com/documentation/en-us/unreal-engine/unreal-engine-5-5-documentation (Viitattu 26.04.2025)

Flick, Jasper 2018, Texture Distortion Faking Liquid, https://catlikecoding.com/unity/tutorials/flow/texture-distortion/ (Viitattu 26.04.2025)

Mad VFX, Textures used in VFX for Games, https://www.mad-vfx.com/blogs/textures-used-in-vfx-for-games (Viitattu 26.04.2025)

McCole Chris 2023, Material Optimization in UE4/UE5, https://www.chrismccole.com/blog/material-optimization-in-ue4-ue5 (Viitattu 26.04.2025)

PrismaticaDev 2025, YouTube kanava, https://www.youtube.com/@PrismaticaDev/videos (Viitattu 26.04.2025)

PrismaticaDev 2021, Linear interpolate | 5-Minute Materials [UE4], https://www.youtube.com/watch?v=rFIvxvvkYRw (Viitattu 26.04.2025)

PROTOWLF 2025, [Unreal] Fixing Unlit Materials, https://protowlf.com/unreal/unreal-fixing-unlit-materials/ (Viitattu 26.04.2025)

The blog at the bottom of the sea 2013, Converting To and From Polar / Spherical Coordinates Made Easy, https://blog.demofox.org/2013/10/12/converting-to-and-from-polar-spherical-coordinates-made-easy/ (Viitattu 26.04.2025)

Rocket Brish Studio 9.1.2025, Realism vs. Stylization art in games, https://rocketbrush.com/blog/realism-vs-stylization-art-in-games (Viitattu 26.04.2025)

Ronja’s tutorials 2018, Triplanar Mapping, https://www.ronja-tutorials.com/post/010-triplanar-mapping/ (Viitattu 26.04.2025)

Ronja’s tutorials 2019, Screenspace Textures, https://www.ronja-tutorials.com/post/039-screenspace-texture/ (Viitattu 26.04.2025)

Tobias Noller 2024. Gradient mapping for VFX - Unreal Engine 5 https://tobias-noller.com/articles/ue4-gradient-mapping-vfx/ (Viitattu 26.04.2025)

Unity 2024, The definitive guide to creating advanced visual effects in Unity, https://unity.com/resources/creating-advanced-vfx-unity6 (Viitattu 26.04.2025)

Veselinovikj, Bojan March 14.3.2024. VFX in Gaming: The Ultimate Guide to Visual Effects in Video Games. Blogi. https://borisfx.com/blog/vfx-in-gaming-ultimate-guide-visual-effects-games/ (Viitattu 26.04.2025)

Kuvalähteet

Kuva 1. RPG fan 2025, Genshin impact Screenshots. https://www.rpgfan.com/gallery/genshin-impact-screenshots/ (Viitattu 2025)

Kuva 2. SaiyanLegends 2024. All NEW Boruto Characters Ultimate Jutsus & Transformations (4K 60fps) - Naruto Storm Connections. YouTube. Kuvakaappaus videosta. https://www.youtube.com/watch?v=rbF2Nexpw3k&t=519s (Viitattu 2025)

Kuva 3. Mina Andre 2020, Supergiant Games, X. https://www.google.com/url?sa=i&url=https%3A%2F%2Fx.com%2Fyourfathers-belt&psig=AOvVaw1jYaAHkts-NvjTgd1GyIJSx&ust=1745752291774000&source=images&cd=vfe&opi=89978449&ved=0CBcQjhxqFwoTCNCgyJvI9YwDFQAAAAAdAAAAABAE (Viitattu 2025)

Kuva 4. WOWHEAD 2024, Devolution of the Buff UI in Diablo 4. https://www.wowhead.com/diablo-4/news/devolution-of-the-buff-ui-in-diablo-4-342546 (Viitattu 2025)

Kuva 20. Unity 2024, The definitive guide to creating advanced visual effects in Unity, https://unity.com/resources/creating-advanced-vfx-unity6 (Viitattu 2025)

Kuvat 5–19, 21–95. Kähärä Filipp 2025, Kuvakaappaukset Unreal Engine:sta Snipping Tool työkalulla.

Pelit

Genshin Impact, MiHoYo 2020

Naruto Storm Connections, Bandai Namco 2023

Hades, Supergiant Games 2020

Diablo 4, Blizzard Entertainment 2023

Liitteet

Screaming Brain Studios, Noise Texture pack, 2022. https://screamingbrains-tudios.itch.io/noise-texture-pack

Kenney, Particle Pack, 2023.https://www.kenney.nl/assets/particle-pack

Filipp Kähärä, Artstation, 2025. https://www.artstation.com/szexpert

Esimerkkityöt:

Esimerkkityö 6 – Häikäisevä valo, Filipp Kähärä https://youtu.be/NRszv933-Mc

Esimerkkityö 7 – Ribbon jälki, Filipp Kähärä https://youtu.be/cxiS86Fft2g

Esimerkkityö 8 – Ruoho, Filipp Kähärä https://youtu.be/9K45xN9VBRU

Esimerkkityö 9 – Rähäjdys, Filipp Kähärä https://youtu.be/iCNXv1ygWeM