

Johanna Puirava

## Partikkeliefektin luonti Unity-pelimoottorilla



Tradenomi  
Tietojenkäsittely  
Kevät 2021



KAMK • University  
of Applied Sciences

## Tiivistelmä

**Tekijä:** Puirava Johanna

**Työn nimi:** Partikkeliefektin luonti Unity-pelimoottorilla

**Tutkintonimike:** Tradenomi (AMK), Tietojenkäsittely

**Asiasanat:** partikkeliefekti, visuaalinen efekti, peligrafiikka, Unity

Tämän opinnäytetyön tarkoitus oli luoda pelikäyttöön soveltuva partikkeliefekti portfolioa varten, ja tehdä dokumentti, josta löytyy efektin luontiin hyödyllistä teoriaa ja käytännön tietoa efektin rakentamisesta Unity-pelimoottorissa.

Partikkeliefektit ovat visuaalisia tehosteita, joita voidaan käyttää videopeleissä ja elokuvissa. Efektien avulla voidaan luoda dynaamista grafiikkaa. Opinnäytetyössä käydään läpi, kuinka efektejä voidaan käyttää peleissä monilla eri tavoin. Efekteillä voidaan visualisoida pelimekaniikkoja tai parantaa pelin grafiikoita. Efekteillä voidaan myös ohjata pelaajan katsetta pelimekaanisesti tärkeisiin tapahtumiin pelissä. Työssä myös syvennytään partikkeliefektien teoriaan muodon, värin, ajoituksen ja performanssin pohjalta.

Työn käytännön osuudessa luotiin taikateemainen räjähdys efekti. Efekti suunniteltiin opinnäytetyön teoriaosuudessa läpi käytyt asiat huomioiden. Efekti rakennettiin Unity-pelimoottorissa vaiheittain. Ensimmäisenä tehtiin efektin odotusosa, jonka jälkeen siirryttiin luomaan efektin tärkein osa eli räjähdys. Viimeisenä luotiin projektiili. Viimeisenä käytännön osuudessa efekti yhdistettiin ja optimoitiin, jotta sitä voitaisiin käyttää peleissä.

Projektiosuudessa käytiin läpi, kuinka luodaan tekstuureja, 3D-malleja ja varjostimia efektejä varten. Työssä käytiin läpi erilaisia parametreja, joita käytetään partikkeliefektien tekemiseen, ja kuinka näitä parametreja voidaan käyttää eri tapauksissa. Projektiosuudesta lukija voi seurata efektin tekoa, ja kokeilla luoda itse samankaltainen efekti.

## **Abstract**

**Author:** Puirava Johanna

**Title of the Publication:** Creating a Particle Effect in Unity

**Degree Title:** Bachelor of Business Administration, Business Information Technology

**Keywords:** particle effect, visual effect, game graphics, Unity

The objective of this thesis was to create a game ready particle effect for portfolio use, and to create a document where one can find useful theoretical and practical information on particle effects and how to create them in Unity game engine.

Particle effects are visual effects which can be used in movies and video games. Dynamic graphics can be created with particle effects. The document goes through different ways how effects can be used in video games. Effects can be used to visualize game mechanics or to improve a game's visuals. They can also be used to guide a player's view to game mechanically important action. The document covers four most important subjects for making particle effects, shape, colour, timing, and performance.

In the practical part of the document a magic themed explosion effect was created. The effect was designed using things learned in the document's theory part. The effect was built step by step in Unity game engine. First the effect's anticipation part was created, after that the most important part of the effect, the climax, was created. The effect's projectile part was created last. Lastly all the effect's parts were combined to a single effect, and the effect was optimized so it would be ready to be used in a video game.

The practical part of the document goes through how to create textures, 3D-models, and shaders for particle effects. The work also reviews different kinds of parameters that are used to create the effect, and how to they can be used in different scenarios. The reader can follow along the building of the effect in the practical part and try to create a similar effect for themselves.

## Sisällys

1	Johdanto .....	1
2	Partikkeliefekti.....	2
3	Partikkeliefektien perusteet .....	4
3.1	Muoto .....	4
3.2	Väri .....	5
3.3	Ajoitus.....	7
3.4	Performanssi .....	9
4	Taikaefektin suunnittelu.....	11
4.1	Iterointi.....	11
4.2	Konsepti.....	12
4.3	Raakile .....	12
5	Efektin luonti .....	14
5.1	Odotus .....	14
5.2	Räjähdykys .....	19
5.3	Projektiili.....	26
6	Efektin viimeistely.....	31
7	Yhteenveto .....	34
	Lähteet .....	36

## Symboliluettelo

Adobe Photoshop	Kuvankäsittelyohjelma
Blender	3D-mallinnusohjelma
Korkeapolygoninen	3D-malli, jossa on suhteellisen korkea määrä polygoneja.
Matalapolygoninen	3D-malli, jossa on suhteellisen alhainen määrä polygoneja.
Piirto	Adobe Photoshop ohjelman tasotehoste, piirtää ääriviivan kuvan reunalle.
Polygoni	Muodostavat mallin 3D-grafiikassa.
Resoluutio	Kuvan tarkkuus, joka kertoo pikselien määrän kuvassa.
Skene	Näkymä Unity-pelimoottorissa.
Skripti	Lyhyt tietokoneohjelma
Sprite	2D-kuva peligrafiikassa
Ulkoinen hehku	Adobe Photoshop ohjelman tasotehoste, joka luo hehkun kuvan reunojen ulkopuolelle.
Unity	Pelimoottori, jota käytetään efektin luontiin.
Varjostin	Ohjelma, joka kertoo näytönohjaimelle, miten kukin pikseli tulee piirtää.
VFX	Lyhenne visuaaliselle efektille.

## 1 Johdanto

Opinnäytetyön aiheeksi valitsin partikkeliefektit, sillä ne ovat minua kiinnostava aihealue, josta toivoisin oppivani lisää. Opinnäytetyössä luotava efekti tehdään sekä portfolioa että oman kokemuksen ja taidon kehittämistä varten.

Partikkeliefektit ovat visuaalisia tehosteita, joita voidaan käyttää videopeleissä. Ne ovat keskeinen osa pelejä, ja niillä on todella monenlaisia eri käyttötarkoituksia pelin grafiikoiden parantamisesta pelimekaniikkojen visualisointiin.

Ensimmäisenä työssä käydään läpi mitä partikkeliefektit ovat, ja miten niitä yleensä käytetään videopeleissä. Tämän jälkeen siirrytään käsittelemään partikkeliefektien teoriaa neljältä eri osa-alueelta. Teoriaosuudessa käydään läpi muotokieltä, väriteoriaa, partikkeliefektien ajoitusta ja performanssia. Teoriaosa antaa lukijalle hyvät valmiudet luoda efektejä teoriapohjalta.

Työn tavoitteena on luoda pelikäyttöön soveltuva partikkeliefekti alusta loppuun. Efekti on taika-teemainen räjähdys efekti, joka voisi toimia esimerkiksi fantasiapelissä pelaajahahmon kykynä. Efektin luominen aloitetaan suunnittelemalla efektin ulkonäkö, jonka jälkeen siirrytään luomaan se Unity-pelimoottorissa. Työssä tutustutaan Unityn partikkelijärjestelmäeditoriin ja käydään läpi sen sisältämiä parametreja ja niiden erilaisia funktioita. Työssä käydään läpi myös tekstuurien luontia Adobe Photoshop ohjelmalla ja 3D-mallinnusta Blender ohjelmalla. Opinnäytetyön projektiosio antaa lukijalle ohjeita efektin luontiin Unityssä.

## 2 Partikkeliefekti

Partikkeliefekti on visuaalinen tehoste, jota voidaan käyttää elokuvissa tai videopeleissä. Partikkeliefektien avulla voidaan luoda dynaamista grafiikkaa, joka ei välttämättä onnistuisi 3D-mallilla, kuten tulta, vesiputouksia tai räjähdysisiä (kuva 1). Myös 2D-grafiikan peleissä partikkeliefektit ovat hyödyllinen työkalu. Esimerkiksi ruutu ruudulta animoitu räjähdys on joka kerta täysin samannäköinen, mutta luomalla sama efekti partikkeliefektinä saadaan räjähdykseen helposti vaihtelua ja mielenkiintoa.



Kuva 1. Esimerkkikuva partikkeliefektistä [1].

Partikkeliefekti koostuu yhdestä tai useammasta partikkelijärjestelmästä. Partikkelijärjestelmä generoi partikkeleita, jotka ovat pieniä kuvia eli Spritejä tai 3D-malleja.

Partikkelit ovat pieniä, yksinkertaisia kuvia tai malleja, joita partikkelijärjestelmä näyttää ja liikuttaa suurissa määrissä. Jokainen partikkeli edustaa pientä osaa nestemäisestä tai amorfisesta kokonaisuudesta, ja kaikkien partikkelien vaikutus luo vaikutuksen kokonaisuudesta. Käyttäen savupilveä esimerkkinä jokaisella partikkelilla olisi pieni savutekstuuri, joka muistuttaa pientä pilveä itsessään. Kun monta näitä pikku pilviä kootaan yhteen skenen alueelle, on kokonaisvaikutus suurempi ja tilavuutta täyttävä pilvi. [2.]

Näitä monimutkaisia efektejä kontrolloidaan määrittämällä yksittäisten partikkelien käyttäytymistä käyttämällä erilaisia ominaisuuksia, kuten lähtöasento, nopeus ja elinikä [3].

Partikkeliefekteillä on todella monia eri käyttötarkoituksia peleissä. Niillä voidaan esimerkiksi visualisoida pelimekaniikoita tai tehdä pelistä visuaalisesti mielenkiintoisempi. Pelimekaniikkaefekteiksi kutsutaan yleensä efektejä, jotka suoraan visualisoivat pelimekaniikkoja. Tällaisia efektejä ovat esimerkiksi hahmon kykyefektit kuten loitsut. Efekteillä voidaan myös parantaa pelin grafiikoita. Esimerkiksi lisäämällä aseella ampumiseen piipun välähdysefekti. Graafisesti realistisessa pelissä voidaan partikkeliefektejä käyttää lisäämään pelin realismia entisestään, esimerkiksi luomalla pölypilviefekti auton renkaiden taakse.

Efekteillä voidaan ohjata pelaajan katsetta tärkeisiin pelimekaniikkoihin tai mielenkiintoisiin pelimaailmasta löytyviin objekteihin. Jos pelaajaa kohti on tulossa kaksi hyökkäystä, joista toinen on normaali isku ja toinen paljon vahinkoa tekevä kyky, pelaajan tulisi näistä ensin huomata vaarallinen hyökkäys, jotta pelaaja voi yrittää väistää sitä ajoissa. Katseen ohjaamiseen voidaan käyttää kahta erilaista tekniikkaa, visuaalista tärkeyttä ja painopistealuetta.

Efektin visuaalisen tärkeyden tulisi heijastaa sen pelimekaanista tärkeyttä. Visuaalista tärkeyttä voidaan kontrolloida monin tavoin. Suurentamalla efektiä saadaan helposti lisättyä sen tärkeyttä, ja myös muokkaamalla efektin väriä, kirkkautta tai muotoa voidaan vaikuttaa efektin visuaaliseen tärkeyteen.

Painopistealue on kohta, johon pelaajan katse hakeutuu. Efektin painopistealue voidaan luoda lisäämällä kirkkautta tärkeään kohtaan efektissä tai lisäämällä liikettä efektin tärkeään kohtaan. Painopistealue voi olla pelaajaa kohti tulevan ammuksen kärki, tässä tapauksessa kärjestä on hyvä tehdä kirkkaampi kuin loppuosa ammuksesta, jotta pelaaja varmasti huomaa tulevan uhkan. Painopistealue voi olla myös alueellisen kyvyn reuna. Reunan olisi hyvä olla selkeästi erottuva, jotta pelaaja näkee, missä kulkee kyvyn raja. Jos koko alue olisi yhtä kirkas, eivät alueen rajat erottuisi pelaajalle yhtä selkeästi.



### 3 Partikkeliefektien perusteet

Partikkeliefektien suunnittelu ja toteutus on pitkä ja moniosainen prosessi. Effektien tekeminen on yksi monipuolisimmista pelin kehityksen osa-alueista. Niiden luomiseen voi tarvita piirustustaitoa, 3D-mallinnusta, animointia ja jopa ohjelmointia. Efektit voivat olla todella erilaisia pelikohdaisesti, ja niiden tulisi olla pelin graafisen tyylin mukaisia. Kuitenkin yleisesti neljä asiaa, jotka kannattaa ottaa huomioon partikkeliefektin luodessa, ovat muoto, väri, ajoitus ja performanssi. Näiden perusteiden osaaminen ja ymmärtäminen auttaa luomaan parempia ja näyttävämpiä efektejä.

#### 3.1 Muoto

Effektin muoto välittää tärkeää informaatiota pelaajalle sen pelimekaanisesta tarkoituksesta. Effektin muoto kulkeekin yleensä käsi kädessä muotokielen kanssa.

Muotokieli on käsite, jota käytetään taiteessa ja animaatioissa kommunikoimaan merkitystä meille tuttujen muotojen perusteella. Kun sitä käytetään hahmo-, objekti- ja ympäristösuunnittelussa, muodot voivat kertoa tarinan, näyttää persoonallisuutta ja herättää emotionaalisen reaktion katsojassa sanoja käyttämättä. [4.]

Muotokieltä voidaan hyödyntää efekteissä samaan tapaan kuin muussakin grafiikassa. Vahinkoa tekevissä iskuissa on hyvä käyttää teräviä muotoja, sillä ne ovat aggressiivisen ja vaarallisen näköisiä. Pyöreät muodot ovat turvallisia, ja niitä käytetään monesti puolustavien kykyjen efekteissä, esimerkiksi kilvissä.

Effektin siluetin tulisi olla selkeä ja nopeasti ymmärrettävä. Mitä suurempi efekti, sitä monimutkikkaampi siluetti sillä voi olla. Jos kyseessä on nopeasti liikkuva efekti, kuten projektiili, kannattaa liike ottaa huomioon siluetissa. Nopeasti liikkuvat objektit vääristyvät ja venyvät, joten efektin siluettia kannattaa venyttää. Tämä auttaa luomaan nopeuden tunnetta.

Kun efektiä luodaan jonkin olemassa olevan ilmiön pohjalta, on tärkeää tulkita sen muodon määritteleviä tekijöitä. Esimerkiksi salaman muoto luonnossa on rosainen ja ohut. Jos efekti ei jäljittele näitä asioita, pelaaja ei välttämättä tunnista sitä salamaksi.

### 3.2 Väri

Värillä voidaan vaikuttaa suuresti efektin ulkonäköön ja teemaan. Erilaisilla väriyhdistelmillä voidaan muuttaa efektin ulkonäköä ja tarkoitusta. Efektiä suunnitellessa on hyvä ymmärtää väriteoriaa. Yksi väriteorian keskeisimpiä käsitteitä on väriympyrä (kuva 2). Väriympyrän avulla väriteorian ymmärtäminen ja soveltaminen on helpompaa.



Kuva 2. Väriympyrän koostuminen [5].

Analogisia värejä ovat värit, jotka asetetaan vierekkäin väripyörään. Ne sopivat yleensä hyvin yhteen ja luovat sileitä ja rauhallisia kuvioita. Analogiset värit ovat harmonisia ja miellyttäviä silmille. Analogiset värit ovat myös yleisiä luonnossa. [6.]

Efekteissä käytetään usein analogista väriteemaa, sillä se on helppo ja vaikuttaa luonnolliselta.

Komplementti- tai vastavärit ovat värejä, jotka ovat väriympyrän vastakkaisilla puolilla toisistaan. Komplementtiväriteemaa ei käytetä efekteissä yhtä usein kuin analogista. Vastavärejä voi kuitenkin hyödyntää efekteissä antamaan niille kontrastia ja syvyyttä.

Kun efektissä on kaksi komplementtiväriä, yhden näistä väreistä täytyy olla toissijainen väri. Kun yhdessä efektissä on kaksi vastakkaista väriä, nuo värit kilpailevat aina ollakseen ensisijainen elementti. [7.]

Efekteissä värin valoisuusarvo voi olla tärkeässä osassa erityisesti, jos kyseessä on kykyefekti, joka olisi tärkeä erottaa muusta ympäristöstä. Tekemällä efektistä kirkkaamman saadaan se erottumaan selkeämmin hahmojen ja ympäristön seasta, ja pelaajalle tärkeä informaatio välittyy paremmin.

Myös efektin väriteemaa suunniteltaessa on hyvä tulkita luontoa. Jos kyseessä on jokin luonnollinen ilmiö, tulisi efektin värityksen olla samankaltainen, jotta pelaaja tulkitsee efektin oikein. Esimerkiksi jäätaian väri olisi hyvä olla kylmän vaalean sininen ja valkoinen, jotta pelaaja tunnistaa sen jääksi. Toisaalta tästä ohjeesta voi poiketakin: esimerkiksi tulitaika on monessa pelissä sinistä eikä tulen punaista niin kuin tuli yleensä on. Vaihtamalla väriä voidaan esimerkiksi luoda voimakkaampi versio kyvystä tai tehdä siitä mielenkiintoisempi.

Yleinen tapa määrittää efektin väri peleissä, joissa efektejä on paljon, on määrittää se efektin funktion mukaan. Jos kyseessä on esimerkiksi parantava kyky, on sen väri yleensä vihreä tai keltainen. Myös kyvyn teeman mukaan voidaan määrittää sen väritys (kuva 3). Esimerkiksi myrkyloitsut ovat yleensä keltaisen vihreitä.

VISUAL EFFECTS ELEMENTS												
Primary Elements	Fire	Water	Ice	Air	Earth	Nature	Arcane	Sand	Poison	Dark	Light	Lightning
Colors some effects have more or less colors in order to emphasis depth and to create more contrast. These are some Default colors.												
Secondary Elements are added for complexity and level of feedback	Smoke Sparks Embers	Waves Foam Splashes	Snow Smoke Sparkles	Dust Wind	Dust Dirt	Leaves Dust	Runes Pattern Sparkles	Dust Sparkles	Liquid Fume Bubbles	Liquid Fume	Sparkles Rays	Sparks Blast
Influences determine which aspect of the element is affected them the most	Chaotic Motion Shape Color	Wave Motion Material Color	Shape Material Color	Wave Motion Shape Color	Shape Color Snap Motion	Shape Color Growth Flow	Color Shape Motion	Shape Flow Color	Color Material Flow	Color Flow Color	Color Shape Motion	Shape Forking Motion Color

Kuva 3. Esimerkkejä väristä ja muista elementeistä teeman mukaan [8].

Videopeleissä, joissa pelaajat ovat vastakkain, voidaan efektin värillä näyttää pelaajalle, kumman tiimin kyky on kyseessä. Esimerkiksi Overwatch-pelissä (kuva 4) sama efekti näyttää erilaiselta riippuen siitä kummassa tiimissä olet. Joissain peleissä pelaajahahmot ovat täysin samanlaisia, mutta erivärisiä. Tällaisissa peleissä pelaajilla on myös samat efektit. Pelaajien efektit voidaan erotella toisistaan samoin kuin pelaajahahmot eli vaihtamalla efektin väri samanväriseksi kuin pelaaja. Joskus kuitenkin pelaajien efektit jätetään tarkoituksella identtisiksi, sillä tällöin pelaaja ei voi olla varma, kuka hänen kimppuunsa hyökkäsi ja pelistä tulee hektisempi, mutta myös mahdollisesti viihdyttävämpi.



Kuva 4. Overwatch-pelissä Reinhardt-hahmon kilven väri on punainen vastustajalla.

### 3.3 Ajoitus

Ajoitus on tärkeää visuaalisissa efekteissä, ja se palvelee kriittistä roolia merkityksellisen liikkeen ja visuaalisen kiinnostuksen luomisessa efekteille. Tapa, jolla efekti muuttuu sen eliniän aikana, tarjoaa olennaista visuaalista informaatiota sen funktiosta. [7.]

Ajoitusta voidaan kutsua myös efektin animaatioksi tai liikkeeksi. Efektin ajoitus koostuu yleensä kolmesta osasta, jotka ovat odotus, huippukohta ja hälveneminen (kuva 5). Yksinkertaisemmissa efekteissä ei välttämättä ole tällaista jaottelua.



Kuva 5. Efektin vaiheet räjähdyksessä [8].

Odotus on efektin alku, tällöin ei vielä pelimekaanisesti yleensä tapahdu mitään. Luonnossa mikään ei ilmene tyhjästä, ja tätä kannattaa noudattaa efekteissään. Odotus antaa pelaajalle tiedon tulevasta tapahtumasta ja lisää efektin jännittävyyttä. Mitä pelimekaanisesti tärkeämpi efekti, sen pidempi odotus yleensä efektissä on, sillä tämä antaa pelaajalle aikaa reagoida suuriin tapahtumiin pelissä. Ilman odotusta efekti saattaa tuntua luonnottomalta, ja tyhjästä ilmenevä efekti voi yllättää pelaajan ja tuntua epärealistiselta. Jos efektin tarkoitus on kuitenkin yllättää pelaaja, ei odotusta tarvita. Myös realistisissa efekteissä ei aina odotusta ole, sillä esimerkiksi realistisen räjähdysen alku tapahtuu niin nopeasti, että räjähdys tuntuu alkavan suoraan huippukohdasta.

Huippukohta on efektin päätoiminto. Tämä on efektin tärkein kohta, sillä efektiin liitetyt pelimekaniikat tapahtuvat tässä vaiheessa. Huippukohta on näyttävin osa efektiä. Kaikissa efekteissä ei ole kaikkia vaiheita, jolloin ne periaatteessa sisältävät vain huippukohdan.

Hälväminen on efektin loppu. Hälväminen varmistaa, että efekti ei lopu seinään, vaan siitä jää hetkeksi jotain jäljelle, näin efekti tuntuu sulavammalta ja tyydyttävämmältä. Hälväminen voi olla esimerkiksi räjähdysen jälkeen ilmassa leijuvat kipinät ja savu sekä maassa räjähdysen jäljiltä hohtava jälki.

### 3.4 Performanssi

Suorituskykybudjetin hallinta on yleensä vaikeinta VFX- eli efektitaiteilijoilla. Kun mallintajilla ja maailmanrakentajilla on polygonien määrä ja animaattoreilla luumäärä, VFX on todella näkyvästä riippuvaista. Partikkelien päällekkäin piirtyminen voi tappaa performanssin, kun partikkelijärjestelmää katsotaan liian läheltä. Jos kävelet takaperin heittäen kranaatteja, ja koko ruutu täyttyy päällekkäisistä savupilvistä, tippuu ruudunpäivitysnopeus nopeasti. Nämä ongelmat voivat olla erityisen tuhoisia, koska ne kestävät vain muutaman ruudun ajan. Pelaajat eivät välttämättä huomaa pätkimistä, mutta heille jää tunne, että heidän hahmonsa reagoi huonosti raskaassa tilaistelussa. [9.]

Partikkeliefektit voivat olla todella raskaita laitteille. Tämän takia on huomioitava performanssi niitä suunnitellessa. Efektejä voi olla peleissä todella moniakin ruudulla yhtäaikaaisesti ja niiden on toimittava reaaliajassa. Myös kohdealusta vaikuttaa suuresti efektin performanssin optimointiin. Performanssiin voi vaikuttaa kahdella tavalla, suunnittelulla ennen kuin efekti tehdään ja optimoinnilla toteutuksen jälkeen.

Suunnitteluvaiheessa efektien kokonaismäärä on tärkeää ottaa huomioon. Efektit voivat olla performatiivisia yksittäisenä, mutta pelissä luultavasti useampi efekti tapahtuu samanaikaisesti, jolloin pelin performanssi kärsii. Suunnitteluvaiheessa onkin tärkeää asettaa efektit tärkeyskategorioihin. Efektit, joilla on vähemmän pelimekaanista tärkeyttä ja tapahtuvat usein, tulisi olla kevyitä, nopeita ja pieniä. Mitä tärkeämpi ja harvinaisempi efekti, sen raskaampi ja samalla näyttävämpi se voi olla.

Yksi efektin performanssiin vaikuttava tekijä on partikkelimäärä. Laskemalla efektin luomien partikkelien määrää voidaan nopeasti vaikuttaa efektin performanssiin. Partikkelimäärän vähentäminen voi vaikuttaa efektin ulkonäköön negatiivisesti, joten on löydettävä sopiva tasapaino performanssin ja ulkonäön välillä.

Piirtääkseen peliobjektin ruudulle pelimoottorin täytyy antaa piirtokutsu grafiikkasovellusliittymälle. Piirtokutsut ovat usein resurssiraskaita, ja grafiikkasovellusliittymän täytyy tehdä merkittävästi töitä jokaiseen piirtokutsuun, aiheuttaen suorituskykyongelmia prosessoripuolella. Tämä johtuu pääasiassa piirtokutsujen välissä tapahtuvista tilanmuutoksista (kuten toiseen materiaaliin

vaihtaminen), joka aiheuttaa resursseja kuluttavan validoinnin ja käännösvaiheet grafiikkaohjaimessa. [10.]

Tekstuureilla voi vaikuttaa efektin performanssiin. Mitä suuremman resoluution tekstuurit efektissä ovat käytössä, sitä raskaampi efekti on. Varsinkin mobiilipeleissä voidaan käyttää alhaisen resoluution tekstuureja, koska ruudun koko on niin pieni, että yksityiskohtia on hankala nähdä. Tekstuurien resoluution pienentämisen lisäksi efektin performatiivisuutta voidaan parantaa myös tallentamalla tekstuurit yhteen kuvaan eli tekstuuriarkkiin. Näin kaikki partikkelijärjestelmät voidaan laittaa käyttämään tätä tekstuuria, jolloin efektin aiheuttamat piirtokutsut vähenevät.

3D-mallit efekteissä käyvät nopeasti yllättävän raskaiksi. Mallien polygonien määrä vaikuttaa suuresti efektin performanssiin, joten malleista kannattaa tehdä mahdollisimman matalapolygonisia. Yleensä efekteissä voikin käyttää todella matalapolygonisia malleja, ilman että se vaikuttaa efektin ulkonäköön, sillä efektit tapahtuvat yleensä niin nopeasti ja niissä käytetään monesti läpikuultavia tekstuureja.

Myös varjostimilla on vaikutus efektin performanssiin. Yksinkertaiset varjostimet eivät vaikuta kovin suuresti efektin performanssiin, mutta monimutkaisemmat varjostimet voivat nopeasti viedä tehoja. Myös varjostimet aiheuttavat piirtokutsuja. Tämän vuoksi vaikka kaikki efektin käyttämät tekstuurit olisivatkin yhdessä tekstuuriarkissa, niin jos efektissä käytetään useampaa varjostinta, tulee piirtokutsuja lisää jokaiselle varjostimelle.

## 4 Taikaefektin suunnittelu

Efektin suunnittelu aloitetaan rajaamalla efektin teemaa ja käyttötarkoitusta. Tässä tapauksessa luotava efekti on voimakas loitsu, joka aiheuttaa suuren räjähdysen. Räjähdyks alkua odotuksella, jossa tuleva räjähdysen alue näkyy selkeästi. Loitsu luo taika-ammuksen, joka putoaa alueen keskelle käynnistäen räjähdysen. Tämä efekti toteutetaan Unity-pelimootorilla ja on suunnattu tietokone alustalle.

Ensimmäisenä efektiä varten kerätään referenssikuvia. Referenssikuvat voivat olla kaikenlaisia kuvia mitkä sisältävät joitain elementtejä efektistä, esimerkiksi tiettyjä muotoja tai värejä. Kuvat voivat olla myös täysin efektiin liittymättömiä, jos niissä on esimerkiksi sellaista henkeä kuin efektiin haluaisit luoda. Referenssikuvat auttavat hahmottamaan millaista efektiä ollaan luomassa ja antavat suuntaa efektille. Referenssikuvat voivat myös avartaa omaa näkemystä, sillä niistä voi saada sellaisia ideoita, joita ei olisi huomannut edes harkita. Referenssikuvat eivät ole kuitenkaan kopioimista varten, vaan niistä kuuluisi oppia ja laajentaa omaa mielikuvitustaan.

### 4.1 Iterointi

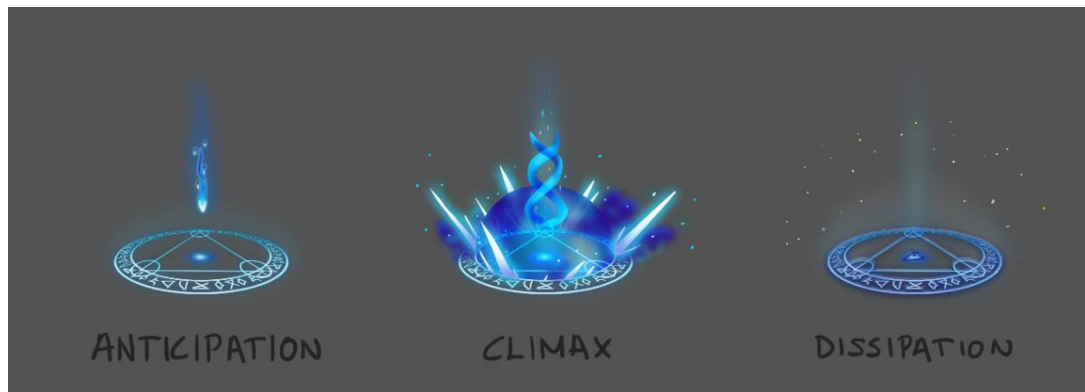
Iteroinnissa tehdään nopeasti erilaisia luonnoksia efektistä. Tällä tavalla saadaan kaikista yleisimmät ideat pois, ja voidaan luoda uniikki ja mielenkiintoinen efekti, sekä saadaan hiottua alkupeleistä ideaa tarkemmaksi. Yksinkertaisempien efektien kanssa voidaan helposti siirtyä suoraan lopullisen konseptin piirtämiseen tai jopa luomaan se pelimootorissa. Tämän efektin konsepti oli jo aika selkeä sitä luonnostellessa, mutta joitain versioita täytyi tehdä loitsuympyrästä ja sen riimuista sekä projektiilista.

Kun luonnokset on tehty ja efektin lopullinen ulkonäkö on selvillä, voidaan siirtyä piirtämään siitä lopullista konseptikuvaa.



## 4.2 Konsepti

Konseptikuvaan kannattaa sisältää efektin päävaiheet. Konseptin voi piirtää myös mustavalkoisena, sillä efektin väriä on helppo muuttaa pelimootorissa. Tässä tapauksessa efektin väritys oli alusta alkaen selkeä, joten konsepti piirrettiin värillisenä. Konseptikuvaan on hyvä myös sisältää tietoa siitä, miten efekti etenee. Esimerkiksi nuolilla voi osoittaa efektin pyörimis- tai kasvamis-suunnan, ja kuvan alle voi kirjoittaa muistiinpanoja efektin liikkeestä. Tämä on erityisen tärkeää, jos efektin suunnittelee ja rakentaa eri henkilöt. Tällöin kaikki tarvittava informaatio välittyy, ja lopputulos on oikeanlainen. Kun konseptikuva on valmis (kuva 6), siirrytään luomaan efektistä raakileversio.



Kuva 6. Selkeät vaiheet valmiissa designissa.

## 4.3 Raakile

Raakile on nopeasti luotu versio efektistä, josta välittyy efektin pääidea ja toiminto. Raakileen luonti ei ole pakollista mutta se on hyödyllinen vaihe, sillä siinä voi tulla esiin ongelmia designissa. Raakilettä voidaan hyödyntää pelissä käyttämällä sitä efektin tilalla. Näin päästään kokeilemaan efektiä pelissä jo ennen kuin sen lopullinen versio on valmis.

Raakilettä varten ei kannata alkaa luomaan uusia tekstuureja, vaan käyttää hyödyksi esimerkiksi Unityn oletustekstuureja tai aiempien projektien tekstuureja. Jos sopivia tekstuureja raakileeseen ei löydy, kannattaa tulevista tekstuureista tehdä nopeat versiot, jotta niihin ei menisi turhaa aikaa

hukkaan. Raakileversioon ei tarvitse sisältää kaikkia efektin osia vaan efektin pääosat. Jos raakileen luonnin yhteydessä ei esiinny efektissä mitään ongelmia, voidaan siirtyä lopullisen efektin luontiin.

## 5 Efektin luonti

Taika efekti luodaan Unity-pelimoottorissa. Efektin luonti jaetaan kolmeen osaan sen helpottamiseksi, Ensiksi luodaan efektin odotusosio, sitten räjähdys ja viimeisenä luodaan efektin odotusvaiheessa taivaalta putoava projektiili.

### 5.1 Odotus

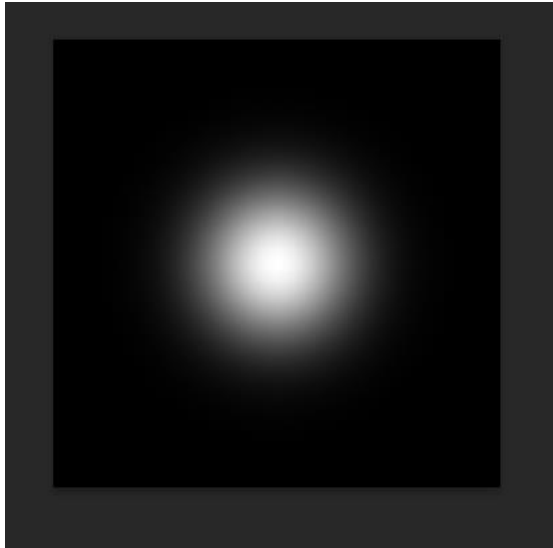
Ensimmäisenä luodaan efektin odotusosa. Odotuksessa loitsuympyrä pyörii, ja sen keskellä on hehku kohdalla mihin projektiili tulee putoamaan. Lisäksi pienemmät taikapartikkelit pyörivät loitsuympyrän yläpuolella. Odotukseen tarvitaan kolme eri tekstuuria, ja kaksi varjostinta.

Ensimmäisenä luodaan tyhjä peliobjekti, jonka sisään partikkelijärjestelmät luodaan, ja luodaan sinne ensimmäinen partikkelijärjestelmä. Tästä partikkelijärjestelmä tulee hehku kohdassa, johon projektiili tulee iskemään. Koska kyseessä on paikallaan oleva partikkeli, asetetaan partikkelin nopeus nolnaan, ja otetaan myös muotoparametri pois päältä sillä sitä ei tarvita. Partikkeli halutaan piirtää maata vasten horisontaalisti, joten piirtäjä asetuksista muutetaan piirtotila horisontaaliksi tauluksi. Säteilyparametri muokataan niin, että partikkelijärjestelmä luo vain yhden partikkelin. Hehkun halutaan kasvavan, kun projektiili lähestyy sitä, joten siihen käytetään koko yli elämän -parametria. Väri yli elämän -parametrilla muokataan partikkelin väri muuttumaan tummasta kirkkaammaksi loppua kohti. Lisäksi partikkeli asetetaan alussa täysin läpinäkyväksi, jotta se tulee esiin hitaasti. Näin partikkeli ei tunnu ilmestyvän tyhjästä.

Partikkelijärjestelmässä voitaisiin käyttää Unityn oletuspartikkeli tekstuuria, mutta koska on hyvä käytäntö luoda itse käytettävät tekstuurit, luodaan uusi tekstuuri. Luotava tekstuuri on samankaltainen hehkuva pallo kuin oletuspartikkeli, mutta kirkkaampi ja efektin visuaaliseen tyyliin sopivampi. Tällaista tekstuuria käytetään todella monissa partikkeliefekteissä, ja tässäkin efektissä sitä tullaan käyttämään suurimpaan osaan partikkelijärjestelmistä.

Tekstuuri luodaan käyttäen Photoshop-ohjelmaa. Se luodaan Photoshopin pehmeällä pyöreällä siveltimellä. Suurella siveltimellä klikataan kerran kuvan keskelle, jolloin kuvaan piirtyy pehmeä

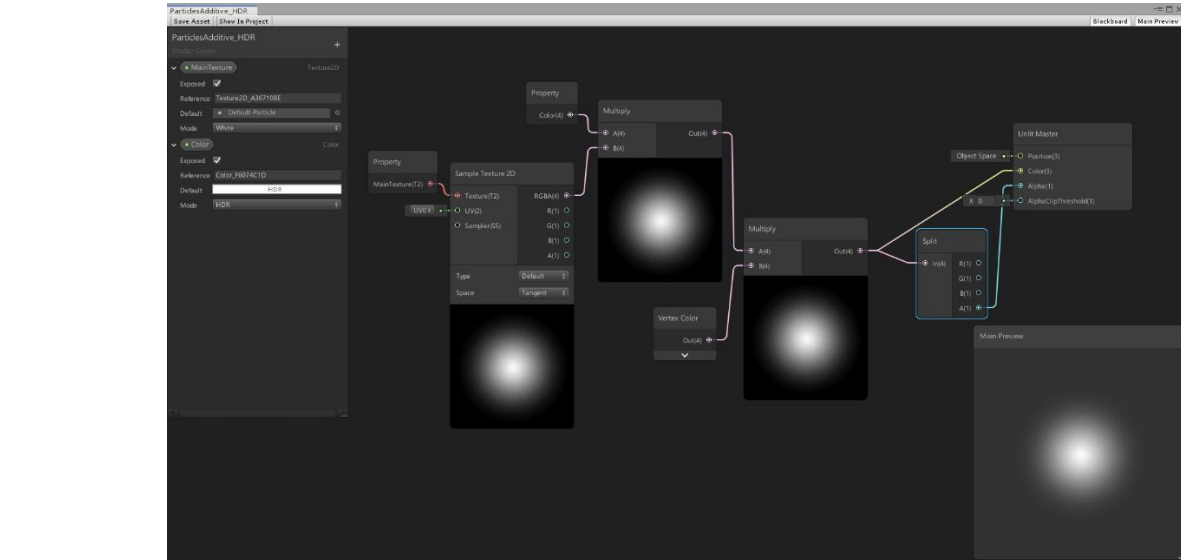
reunainen ympyrä (kuva 7). Taso keskitetään, jotta tekstuuri olisi kuvan keskellä, ja tekstuuri tallennetaan.



Kuva 7. Valmis ympyrä tekstuuri

Uudelle tekstuurille tarvitaan vielä uusi varjostin, joka tekee siitä hohtavamman. Tähän tarkoitukseen sopii lisäävä varjostin. Lisäävä varjostin on yleinen varjostin, jota voidaan käyttää melkein kaikenlaisissa efekteissä. Lisäävä varjostin lisää partikkelin värin taustan värin päälle, joten se on hyvä läpikuultavien ja hehkuvien partikkelien luomiseen. Lisäävä varjostin on todella käytännöllinen esimerkiksi taika- tai tuliefekteissä. Lisäävä varjostin ei ole kovin hyvä tummien värien kanssa, sillä tällä varjostimella partikkelit muuttuvat läpinäkyvämmäksi mitä lähempänä mustaa partikkelin väri on, ja jos partikkeli on täysin musta, se muuttuu läpinäkyväksi.

Lisäävä varjostin luodaan käyttämällä Unityn varjostin kaavio -työkalua. Lisäävän varjostimen pintatila asetetaan läpinäkyväksi ja sekoitustila lisääväksi. Kun lisäävä varjostin on valmis (kuva 8), voidaan se asettaa käyttöön uudelle tekstuurille. Tätä varjostinta käytetäänkin melkein kaikkiin partikkelijärjestelmiin tässä efektissä.



Kuva 8. Valmis lisäävä varjostin Unityn varjostin kaaviolla tehtynä.

Efektiin lisätään vielä toinen hehku taivaalle kohtaan, josta projektiili putoaa. Tämän hehkun ansiosta projektiili ei ilmene tyhjistä vaan ikään kuin jonkinlaisesta taikaportaalista tai pilvestä.

Efektin tärkein osa on loitsuympyrä, sillä se on sen keskeisin elementti, ja se välittävää efektin taikateemaa parhaiten. Loitsuympyrän taika synnyttää projektiilin ja täten aiheuttaa räjähdysten. Ensimmäisenä loitsuympyrää varten luodaan loitsuympyrä tekstuurit. Loitsuympyrässä on ulomainen ringi, jossa on taika riimuja ja sisällä kolmio, jonka kärjissä on pienet ympyrät. Koska kolmion ja ringin halutaan pyörivän ja kasvavan eri suuntiin, ne täytyy tehdä erillisinä tekstuureina.

Loitsuympyrä tekstuurit (kuva 9) luodaan samaan tiedostoon, sillä ne on helpompi piirtää päällekkäin, ja kun ne lisätään efektiin ne menevät päällekkäin oikein, eikä niiden kokoa tarvitse alkaa hienosäätää. Photoshop -ohjelmassa on soikio- ja kolmiotyökalut, joilla saadaan nopeasti tehtyä muodot. Riimut piirretään vapaalla kädellä ympyrän sisään. Viimeisenä loitsuympyrään lisätään ulkoista hehkua. Ulkoisella hehkulla saadaan efektin tekstuureista hohtavampia ja realistisempia. Loitsuympyrästä ja kolmiosta luodaan vielä toiset paksumpi reunaiset tekstuurit. Näitä tekstuuria käytetään loitsuympyrän varjon luontiin, sillä varjon halutaan olevan suurempi kuin alkupe-  
räinen tekstuuri, jotta se näkyy loitsuympyrän alta. Varjo tekstuurit luodaan monistamalla tasot, joille tekstuurit on luotu, ja suurentamalla muotojen piirron leveyttä. Riimujen kokoa suuren-  
taan käyttämällä piirto tasotehostetta, jonka avulla Photoshop piirtää itsestään riimujen ympä-  
rille.



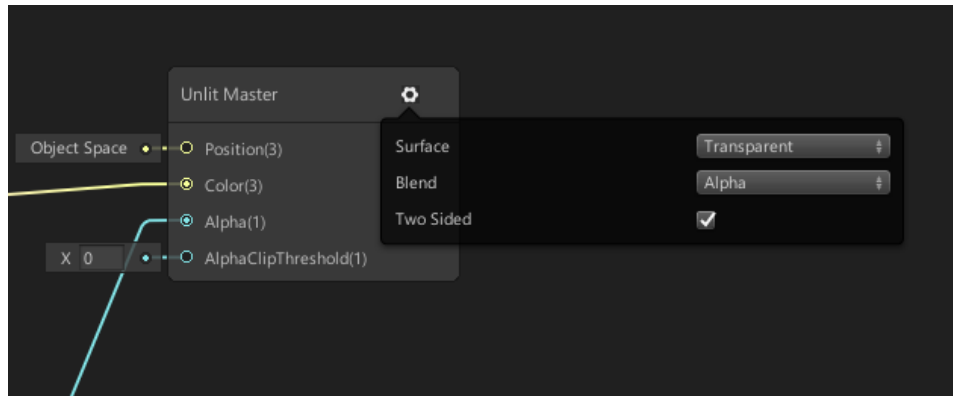
Kuva 9. Vasemmalla loitsuympyrä ja oikealla varjoversio.

Seuraavaksi voidaan Unityssä luoda uudet partikkelijärjestelmät loitsuympyrälle. Asetetaan loitsuympyrä pienenemään sopivan kokoiseksi, ja kolmio taas kasvamaan ympyrään sopivaksi. Ympyrän ja kolmion halutaan pyörivän vastakkaisiin suuntiin ja pysähtyvän lopussa. Loitsuille käytetään rotaatio yli elämän -parametria, joka käyttää käyrää. Käyrällä saadaan kontrolloitua partikkeleita niin, että ne pyörivät alussa nopeaa, mutta hidastuvat loppua kohti, kunnes ne pysähtyvät täysin. Kolmio ja ympyrä laitetaan pyörimään vastakkaisiin suuntiin. Tämä luo mielenkiintoisemman liikkeen kuin samaan suuntaan pyöriminen.

Seuraavaksi lisätään taikapartikkelit. Tähän partikkelijärjestelmään voidaan käyttää aiemmin luotua tekstuuria. Partikkelien halutaan kiertävän ympyrää loitsuympyrän yläpuolella, ikään kuin ne saisivat vauhtia loitsuympyrän pyörimisestä. Jotta partikkelit saadaan kiertämään ympyrää, käytetään vauhti yli elämän -parametria. Parametrasta asetetaan sopiva arvo kiertorata kohdan Z-akselille. Jotta efektin taianomaisuus välittyisi paremmin pelaajalle, lisätään sitä asettamalla partikkelijärjestelmään jälki. Jälki piirtää partikkelien perään jäljen sille reitille mitä ne ovat kulkeet. Jäljelle täytyy asettaa oma tekstuurinsa piirtäjä asetuksissa. Jäljelle käytetään samaa tekstuuria kuin partikkeleille. Partikkelien piirtotila vaihdetaan vielä venytetyksi tauluksi, jotta ne näyttävät vauhdikkaammilta. Venytetyllä taululla on kaksi omaa asetusta, jotka vaikuttavat siihen kuinka paljon partikkeleja venytetään. Pituus skaala vaikuttaa suoraan partikkelien pituuteen ja nopeus skaala muuttaa partikkelien pituutta riippuen niiden nopeudesta.

Viimeisenä efektiin lisätään varjot. Varjojen luontiin tarvitaan kuitenkin uusi varjostin, sillä lisäävä varjostin ei voi näyttää mustaa väriä. Tätä varten luodaan sekoitettu alfa varjostin. Sekoitettu alfa varjostin on erityisen hyvä lisäämään kontrastia efektiin.

Sekoitettu alfa varjostin on yksinkertainen luoda, kun lisäävä varjostin on jo tehty, sillä niissä on vain yksi pieni ero. Vanha varjostin voidaan monistaa ja nimetä uudelleen, ja sitten muokataan sen sekoitustila lisäävästä sekoitettuun (kuva 10).



Kuva 10. Vaihdetaan sekoitustila alfaksi.

Kun varjostin on saatu valmiiksi, voidaan siirtyä luomaan varjot. Varjoilla saadaan efektiin kontrastia, ja ne auttavat efektiä erottumaan maasta. Jotta varjot näkyvät oikein, tulee varjopartikkelijärjestelmien väri muuttua mustaksi. Loitsuympyrän varjoihin käytetään siihen luotuja tekstuurireja, ja keskellä olevan hehkun varjon luontiin käytetään samaa tekstuuria, mutta sen partikkelin kokoa nostetaan, jotta se näkyy hehkun alta. Tämän jälkeen efektin odotusosio on valmis (kuva 11).



Kuva 11. Efektin odotusosa on valmis.

## 5.2 Räjähdykset

Efektin räjähdysosa on monimutkikkaampi, ja sisältää useampia partikkelijärjestelmiä. Räjähdykset on efektin pääosa. Räjähdyksessä ilmenevään paineaaltoon ja spiraaliin tarvitaan 3D-malleja.

Räjähdykset aloitetaan samaan tapaan kuin odotus, eli luomalla maahan hehku. Tällä kertaa hehku alkaa kirkkaana ja häviää ajan myötä. Lisäksi hehku asetetaan pienenemään ajan kuluessa. Efektiin luodaan valonvälähdys partikkelijärjestelmä. Valonvälähdys on nopeasti ilmestynyt ja katoava kirkas partikkeli, joka saa efektin tuntumaan enemmän räjähdykseltä.

Seuraavaksi luodaan räjähdysten jälkeinen versio loitsuympyrästä. Räjähdyksen jälkeen loitsuympyrän jälki on palanut maahan kiinni. Loitsuympyrä on räjähdysten aikana latautunut taikaenergialla ja se hohtaa kirkkaasti. Räjähdyksen jälkeen se hiljalleen haalistuu pois, ja siitä jää vain palanut musta jälki maahan. Jäljen luontiin käytetään loitsuympyrän varjotekstuuria, ja sen elinaika asetetaan pidemmäksi kuin hehkuvan ympyrän.

Räjähdykset ei tunnu vielä tarpeeksi voimakkaalta, vaan tarvitaan uusi tekstuurit, joka näyttää enemmän räjähdysten välähdykseltä. Tekstuurin olisi hyvä olla muodoltaan terävä ja korkea, ja sen pitäisi muistuttaa projektiin voimasta ylöspäin lentävää energiaa.

Isku tekstuurin piirtäminen aloitetaan hahmottelemalla sen muoto suurella pehmeällä ja lähes läpinäkyvällä siveltimellä. Kun muoto on hahmoteltu, voidaan tekstuurin lisätä kirkkautta pienemmällä siveltimellä. Seuraavaksi käytetään hankaustyökalua tekemään tekstuurista aidomman näköinen. Lopuksi lisätään tekstuuriin vielä ulkoinen hehku tasotehoste, ja tekstuurit tuodaan Unityyn (kuva 12).





Kuva 12. Valmis isku tekstuuri.

Isku partikkelijärjestelmä voidaan luoda kopioimalla aiemmin luotu välähdys. Vaihdetaan sen materiaali isku materiaaliin. Partikkeli näkyy nyt osittain maan alla, joten sen pivot arvoa Y-akselilla täytyy nostaa piirtäjä osiosta.

Jotta efekti vaikuttaisi luonnollisemmalta, on siinä oltava pieniä eroja joka luomiskerralla. Iskulla saadaan aikaan hyvin vaihtelua asettamalla sen aloituskoko satunnaiseksi kahden vakion välillä. Asetetaan myös 3D aloitus koko päälle, jotta kokoa voidaan muokata joka akselilla erikseen. Sama tehdään myös aloitus rotaatiolle mutta asetetaan se kääntymään vain Y-akselilla. Nyt partikkelijärjestelmä luo partikkelin välillä pelaajaan nähden väärinpäin. Tämän korjaamiseksi muutetaan järjestelmän synnyttämien partikkelien määrää. Nyt isku näyttää räjähdysen alussa tapahtuvalta energian purkaukselta (kuva 13).



Kuva 13. Valmis isku pelimoottorissa.

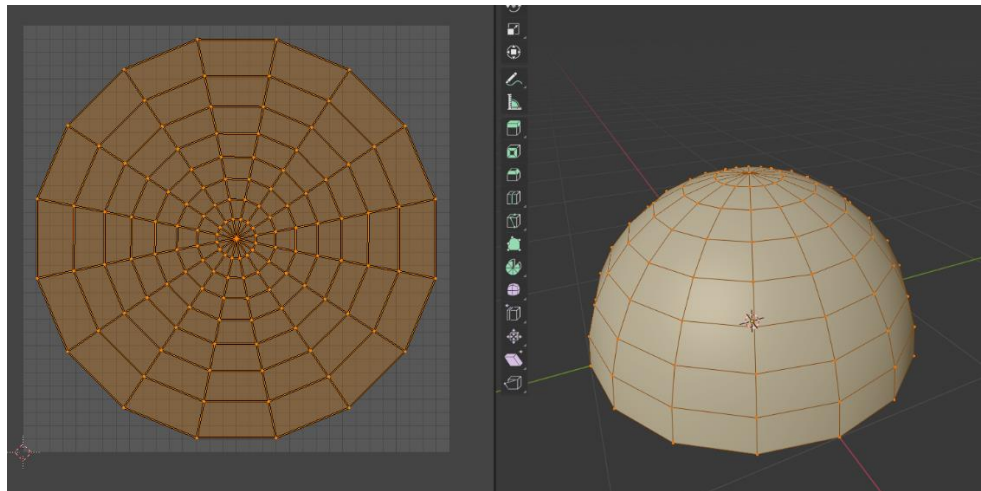
Räjähdyksistä yleensä lentää joitain pienempiä partikkeleita, kuten kipinöitä. Taikapartikkelit luodaan samaan tapaan kuin efektin alussa pyörivät partikkelit. Partikkelien halutaan lentävän todella nopeasti räjähdyskeskeltä pois, ja koska nopeat objektit näyttävät kameralla venyneeltä käytetään näissäkin partikkeleissa piirtotilana venytettyä taulua. Venytystä kontrolloidaan pääasiassa nopeuden mukaan, näin partikkelien pituuteen tulee vaihtelua, kun nopeampaa liikkuvat partikkelit venyvät enemmän. Koska nämä partikkelit liikkuvat niin nopeasti, on niiden elinajan oltava lyhyt, jotta ne eivät kerkeä kovin kauas räjähdyskeskipisteestä ennen katoamista. Efektin designissa kuitenkin nähdään partikkeleita leijumassa vielä efektin lopussa, joten lisätään efektiin toinen partikkelijärjestelmä, jonka partikkelit liikkuvat hitaampaa ja elävät pidempään.

Näiden partikkelien halutaan laskeutuvan räjähdysjälkeen, joten niille asetetaan painovoima. Partikkelien halutaan myös menettävän nopeutta, kun ne lentävät kauemmas räjähdyskeskipisteestä. Tähän käytetään vauhdin rajoitus yli elämän -parametria, jolla voidaan vaimentaa partikkelien nopeutta. Käyttämällä käyrää parametri vaimentaa partikkelien vauhtia vasta tietyn ajan jälkeen. Viimeisenä partikkeleihin asetetaan kohinaparametri, jolla partikkelit saadaan liikkumaan ilmassa satunnaisesti suuntiin, eivätkä ne vain putoa suoraan alas. Tämä lisää efektiin mielenkiintoisuutta ja taianomaisuutta.

Paineaalto, joka etenee nopeasti räjähdysjälkeen, tehdään käyttämällä 3D-mallia. Paineaallon voisi luoda myös Spritenä, mutta koska efektiä halutaan katsella myös ylhäältäpäin, on 3D-

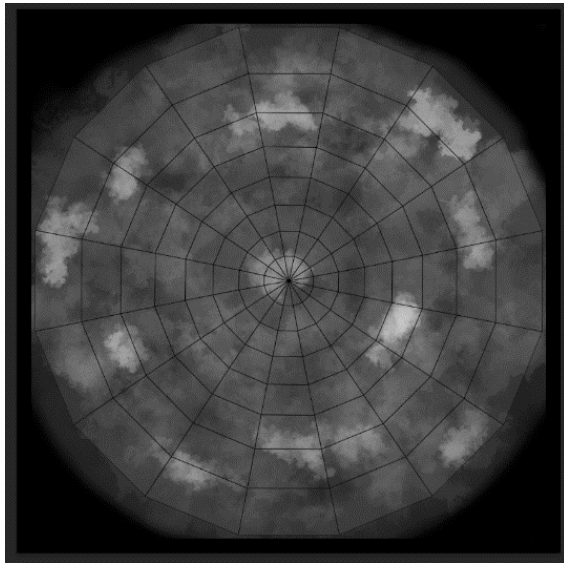
malli parempi vaihtoehto. Koska paineaalto näkyy ruudulla vain todella lyhyen ajan, sen ei tarvitse olla kovin korkeapolygoninen.

Paineaallon 3D-malli on puolipyörä, joka voidaan luoda Blenderissä ympyrän pohjalta. Oletus ympyrä on turhan korkeapolygoninen, joten segmenttien ja renkaiden määrä lasketaan kuuteentoista. Jos niitä on tätä vähemmän, alkaa malli helposti vääristyä pelimoottorissa. Nyt ympyrästä poistetaan puolet muokkaustilassa ja malli on valmis. Malli täytyy vielä UV-kartoittaa, jotta teksturi asettuu mallin päälle oikein (kuva 14). UV-kartta tuodaan ulos Blenderistä, jotta tekstuurin piirtäminen puolipyörälle olisi helpompaa.



Kuva 14. Valmis puolipyörä ja sen UV-kartta.

Paineaallolle täytyy vielä luoda teksturi. Tekstuuriksi sopisi hyvin tasainen värikin koska paineaalto katoaa ja ilmestyy niin nopeasti, mutta tässä tapauksessa halutaan sen olevan mielenkiintoisemman näköinen. Paineaallon UV-kartta tuodaan Photoshopiin, jotta alue jolle teksturi piirretään, voidaan hahmottaa paremmin. Paineaallon teksturi luodaan käyttämällä pilvisivelintä, jolla piirretään eri vahvuuksilla ympäri UV-karttaa (kuva 15). Näin saadaan paineaaltoon mielenkiintoinen teksturi, jossa on vaihtelevia kohtia.



Kuva 15. Puoliympyrän tekstuuri ja UV-kartta.

Puoliympyrä tuodaan Unityyn, mutta sen asetuksia täytyy vielä muuttaa ennen kuin sitä voidaan käyttää partikkelijärjestelmässä. Asetuksista asetetaan materiaalien ja animaatioiden tuominen pois päältä, sillä niitä ei tarvita. Mallin skaala nostetaan asetuksista sataan, sillä muuten se olisi todella pieni Unityssä.

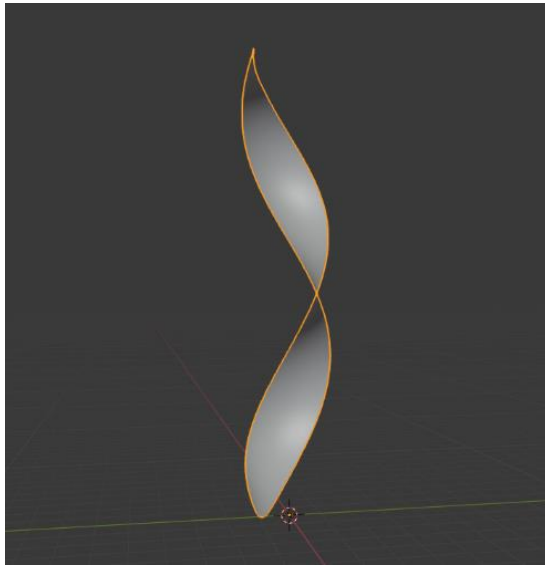
Luodaan uusi partikkelijärjestelmä paineaallolle. Piirtäjäosiosta muutetaan piirtotila verkoksi. Verkko piirtotilalla voidaan asettaa partikkelijärjestelmä käyttämään 3D-objekteja. Valitaan malliksi juuri luotu puoliympyrä, ja materiaaliksi asetetaan puoliympyrä tekstuuri. Paineaalto asetetaan kasvamaan, ja sen väri asetetaan muuttumaan kirkkaan vaalean sinisestä tumman violetiksi (kuva 16). Laitetaan paineaalto myös pyörimään, näin sen tekstuuri ei ole niin staattinen.



Kuva 16. Paineaalto ja partikkelit lisättynä efektiin.

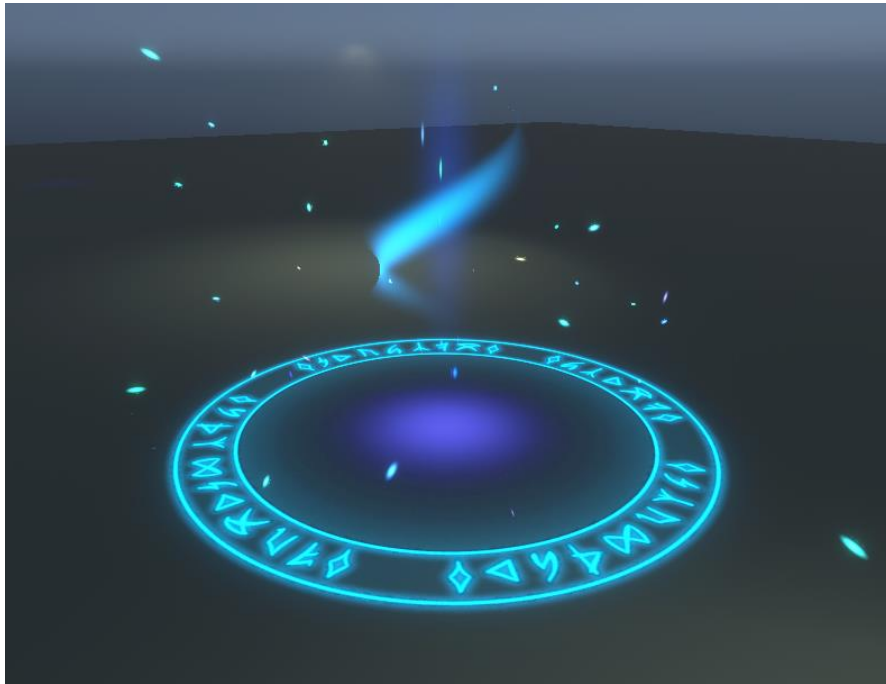
Loitsun lopussa ilmestyy taikaspiraali, joka nousee räjähdyskeskeltä. Spiraalilla saadaan lisättyä efektin taianomaisuutta, ja se antaa efektiin hienon loppusilauksen. Spiraaliin tarvitaan 3D-mallia, joka täytyy ensiksi luoda Blenderissä.

Spiraalin luonti onnistuu helposti käyttämällä ruuvimuokkainta. Ruuvimuokkaimella voidaan generoida mallia haluttuun suuntaan automaattisesti. Muokkaimella voidaan myös määrittää, kuinka suuressa kulmassa mallia käännetään. Luodaan uusi taso, ja liitetään siihen ruuvi muokkain. Ruuvimuokkain asetetaan käyttöön, jolloin voidaan muokata sen luomaa geometriaa. Seuraavaksi poistetaan 3D-mallista muokkaustilassa kaikki paitsi yksi sivu, jolloin jäljelle jää vain spiraali. Spiraali täytyy UV-kartoittaa, jotta tekstuuri asettuisi siihen oikein. UV-karttaa täytyy vielä muokata, sillä se on liian kapea, joten UV-kartta skaalataan neliön muotoiseksi. Jos mallin vie nyt pelimoottoriin, ja käyttää sitä partikkelijärjestelmässä, se aiheuttaa graafisia häiriöitä, sillä siinä ei ole tarpeeksi geometriaa. Geometrian lisäämiseen käytetään alajakopintamuokkainta, joka myös pyöristää mallia sulavammaksi. Alajaon määrä asetetaan kahteen, jonka jälkeen malli voidaan viedä pelimoottoriin (kuva 17).



Kuva 17. Valmis spiraali malli.

Spiraali partikkelijärjestelmän pivot arvoa täytyy nostaa Z-akselilla, jotta se näkyy maanpäällä kokonaan. Se asetetaan myös kasvamaan ajan myötä, jotta se näyttäisi nousevan räjähdyskeskeltä. Spiraali laitetaan pyörimään, jotta se näyttäisi sulavalta taikakiekuralta eikä vain staattisesti nousisi maasta. Sen keskelle luodaan hehku, joka näyttää räjähdyskeskeltä nousevalta valolta ja energialta, ja se myös saa spiraalin näyttämään mielenkiintoisemmalta. Hehkua venytetään Y-akselilla, jotta se näyttäisi pitkältä, ja sen piirtoilaksi asetetaan vertikaali taulu. Vertikaali taulu piirtää partikkelin vain vertikaalisesti, jolloin partikkeli ei käännä katsottaessa ylhäältä päin. Alkuperäisessä designissa myös spiraalin keskellä on pieniä taikahiukkasia, joten lisätään ne efektiin (kuva 18).

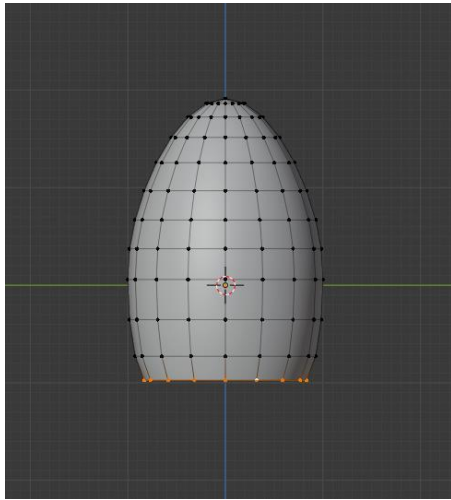


Kuva 18. Valmis spiraali lisättynä efektiin.

### 5.3 Projektiili

Projektiili on taika-ammus, jonka loitsuympyrä luo taivaalle, ja projektiili putoaa loitsuympyrän keskelle aiheuttaen räjähdysen. Projektiiliin tulee olla vaarallisen näköinen ja nopeasti liikkuva. Tätä projektiilia voitaisiin hyvin käyttää myös omana efektinä, joka ammutaan kohti vihollista.

Projektiilin luominen aloitetaan tekemällä sille 3D-malli. Projektiilin malli luodaan samaan tapaan kuin puolisympyrä mutta sitä venytetään pidemmäksi, jotta se muistuttaisi enemmän projektiilin muotoa. Venyttämiseen käytetään Blenderin suhteellinen muokkaus -työkalua, jolla voidaan muokata myös valitun alueen ympäröiviä alueita, säätämällä työkalun vaikutusaluetta. Kun malli on luotu (kuva 19), se UV-kartoitetaan ja UV-kartta viedään Photoshopiin teksturointia varten.



Kuva 19. Projektiilin 3D-malli.

Projektiilin kärki on kirkas ja sen reunat ovat rosoiset. Koska projektiilin kärki on UV-kartan keskellä, tehdään siitä valkoinen ja piirretään keskeltä tähtimäisesti energiajuovia UV-kartan reunoja kohti. Tekstuurin muotoa pehmennetään käyttämällä hankaustyökalua, ja sillä luodaan myös pieniä piikkejä tekstuuriin. Viimeisenä tekstuuriin asetetaan ulkoista hehkua.

Kun malli ja tekstuuri on tuotu Unityyn, voidaan siirtyä tekemään sille partikkelijärjestelmä. Koska projektiilia kontrolloidaan koodin kautta, sille luodaan oma tyhjä peliobjektinsa. Projektiiliin asetetaan käyttöön malli ja tekstuuri. Projektiili asetetaan toistumaan, sillä sen luominen ja tuhoaminen tehdään skriptin avulla. Tämän vuoksi projektiilin väri on myös koko ajan sama. Projektiilin staattisuuden vähentämiseksi asetetaan se pyörimään, jolloin sen tekstuuri näyttää mielenkiintoisemmalta. Jotta projektiili näyttäisi voimakkaammalta ja enemmän taikaenergialta, tarvitaan uusi varjostin muokkaamaan projektiilin tekstuuria.

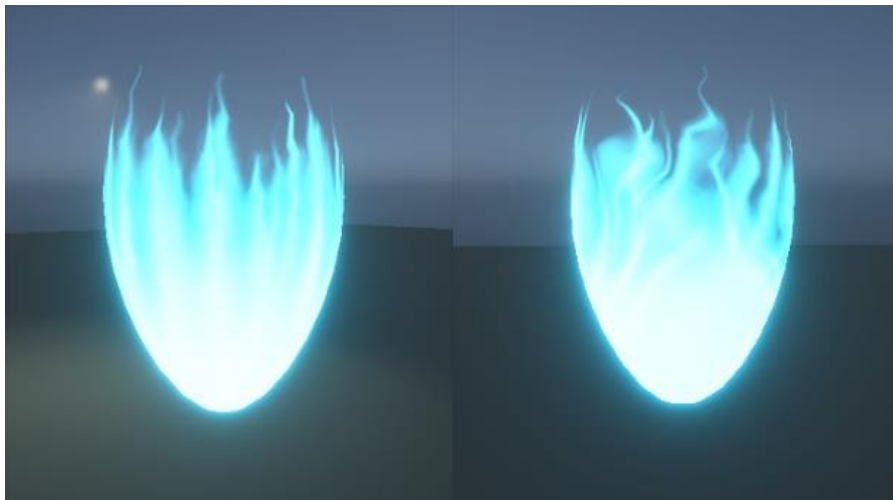
Jotta projektiilista saataisiin mielenkiintoisempi ja elävämpi, halutaan sen tekstuurien liikkuvan. Tätä varten luodaan uusi varjostin, joka vierittää ja vääristää tekstuuria sille asetettujen arvojen mukaisesti. Käyttämällä tällaista varjostinta voidaan säästää aikaa esimerkiksi tuliefektin teossa, sillä tekstuuria ei tarvitse tehdä arkki animaationa, vaan voidaan käyttää vain yhtä tekstuuria, jota vääristellään varjostimen voimin. Vierittävä varjostin on lisäävä varjostin, johon lisätään tekstuurin vieritys ja vääristys, joten se tehdään lisäävän varjostimen pohjalta.

Jotta tekstuuria saadaan vieritettyä, täytyy varjostin kaaviossa käyttää aika ja UV solmuja. Lisäämällä aika ja UV yhteen, ja liittämällä se tekstuuriin saadaan tekstuuri liikkumalla. Jotta tekstuurin



liikkumissuuntaa ja -nopeutta saataisiin kontrolloitua, on kaavioon lisättävä uusi ominaisuus, jolla kontrolloidaan nopeutta.

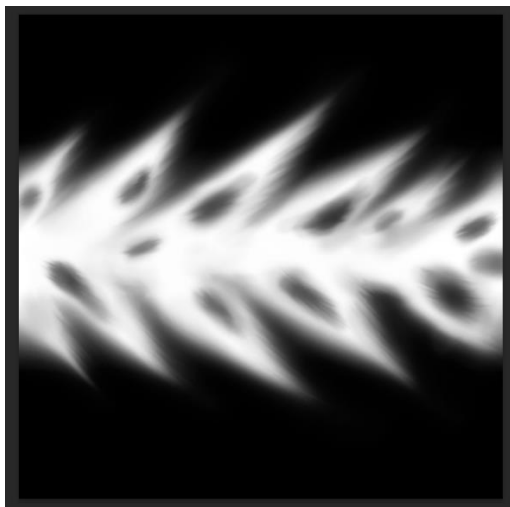
Vääristyksen aikaansaamiseksi, tarvitaan kohina solmua. Lisäämällä aika ja UV solmut toisiinsa, voidaan ne liittää kohina solmuun. Kun kohina liitetään tekstuuriin, saadaan tekstuuri vääristymään. Kaavioon lisätään uudet ominaisuudet, jotta vääristyksen määrää voidaan kontrolloida helpommin. Seuraavaksi uusi varjostin laitetaan projektiilille käyttöön (kuva 20).



Kuva 20. Projektiili ennen ja jälkeen vääristävän varjostimen.

Projektiilin häntä eli jälki voitaisiin luoda käyttämällä partikkelijärjestelmän jälkiparametria tai tekemällä sille oma partikkelijärjestelmä, mutta Unity sisältää paremman vaihtoehdon liikkuvien objektien jäljen luomiseen. Jälkirenderöijä luo jäljen liikkuvan peliojektin perään. Jälkirenderöijä ei toimi, jos partikkelin laittaa liikkumaan partikkelijärjestelmän kautta, sillä tällöin vain yksittäinen partikkeli liikkuu eikä partikkelijärjestelmä. Jotta jälkirenderöijä piirtää jäljen, täytyy peliobjektia, johon se on asetettu lapseksi liikuttaa koodilla tai animaatiolla. Jälkirenderöijä asetetaan projektiilin lapseksi, jotta se liikkuu sen mukana. Jälkirenderöijän luoman jäljen pituutta säädetään aika kohdasta, joka määrittää kuinka pitkään sen luoma jälki elää. Myös jäljen leveyttä ja väriä voidaan muuttaa jälkirenderöijän asetuksista.

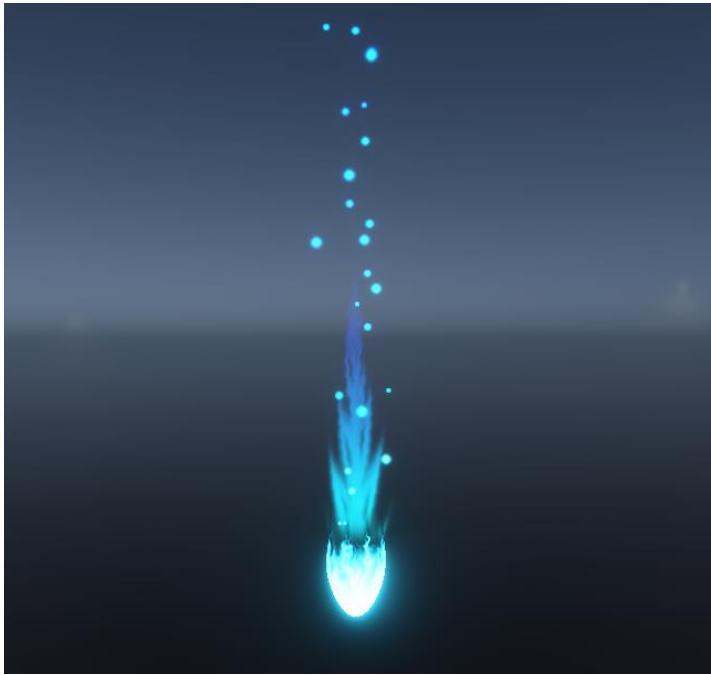
Jäljelle on luotava oma tekstuurinsa (kuva 21). Tekstuuriksi luodaan piikikäs ja hehkuva viiva. Teksturiin luodaan myös reikiä, jotta siinä olisi hieman negatiivista tilaa. Näin jälki on mielenkiintoisemman näköinen. Jälkiteksturiin käytetään samaa varjostinta kuin projektiiliin, näin saadaan jäljestä energisempi ja mielenkiintoisempi.



Kuva 21. Jäljen tekstuuri on valmis.

Projektiili jättää jälkeensä myös pieniä taikapartikkeleita. Partikkeleita varten luodaan uusi partikkelijärjestelmä. Taikapartikkeleiden halutaan syntyvän vain, kun projektiili liikkuu, joten niiden luomiseen käytetään säteilyparametrissa määrä yli matkan -asetusta. Määrä yli matkan luo partikkeleita asetetun määrän, kun partikkelijärjestelmä on liikkunut pelimoottorin yksikön verran. Jotta partikkeleiden liike olisi mielenkiintoisempi eivätkä ne vain leijuisi paikallaan, asetetaan ne liikkumaan satunnaisesti ylös ja alas. Koska partikkelien halutaan putoavan lopulta projektiilin perässä, asetetaan niille painovoima arvo. Partikkeleiden liikkeeseen lisätään vielä satunnaisuutta käyttämällä kohinaparametria pienellä vahvuudella.

Viimeisenä luodaan projektiilin sisälle varjo luomaan kontrastia. Varjo luodaan samoin kuin aiemmin sekoitettu alfa varjostimella. Varjo asetetaan projektiilin sisään, ja siitä tehdään hieman pienempi kuin projektiilista jotta se sopii sen sisälle. Jotta varjo näkyisi projektiilin alta sopivasti luoden kontrastia, sitä venytetään Z-akselilla pidemmäksi kuin projektiili. Tämän jälkeen projektiili on valmis käytettäväksi (kuva 22).



Kuva 22. Valmis projektiili

## 6 Efektin viimeistely

Viimeistelyvaiheessa korjataan vielä efektin ajoitusta, värejä ja muita mahdollisia hiomista vaativia osia. Myös tekstuureja voidaan parannella tässä vaiheessa. Päätin lisätä vielä lopussa efektiin jäljen kohtaan, johon projektiili osuu, jotta räjähdys tuntuisi voimakkaammalta. Kun kaikki efektin osat ovat valmiita voidaan siirtyä yhdistämään efektin osat yhdeksi kokonaisuudeksi.

Efektin odotus ja räjähdys voidaan koota saman peliobjektin sisään. Räjähdysosaan asetetaan oikea määrä viivettä, jotta se käynnistyy juuri odotuksen jälkeen. Tämän jälkeen täytyy ohjelmoida skripti. Skripti aloittaa taikaräjähdysefektin, odottaa puoli sekuntia, ja luo efektin projektiiliin. Projektiili liikutetaan efektin keskelle, ja se tuhotaan skriptillä sen osuessa maahan. Nyt efektin kaikki vaiheet on yhdistetty (kuva 23).

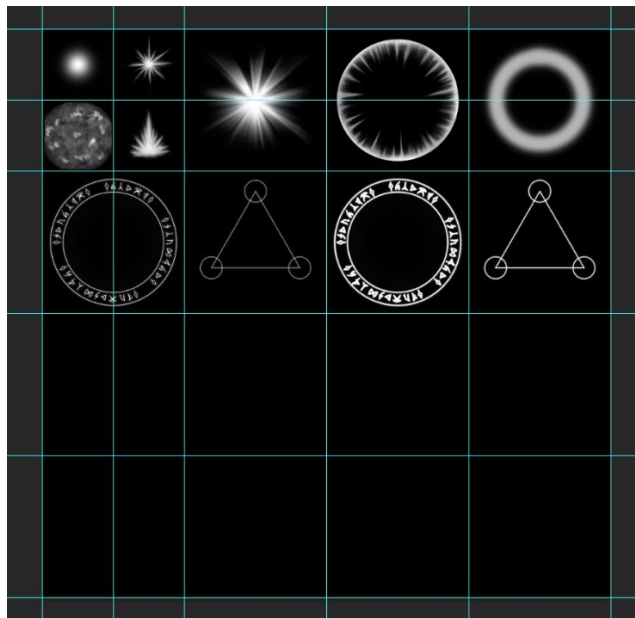


Kuva 23. Efektin osat yhdistettynä.

Viimeisenä efekti täytyy vielä optimoida, jotta sitä voidaan käyttää pelikäytössä. Loitsuefektin luomien partikkelien määrä ei ole kovin suuri, joten partikkelien määrä jätetään ennalleen. Suurin

efektin performanssiin vaikuttava asia on sen tekemät piirtokutsut, joita efekti tekee liian monta ilman optimointia. Efektin tekemien piirtokutsujen määrä voidaan nähdä Unityn statistiikka ikkunasta. Tarkemmin piirtokutsut voidaan nähdä Unityn ruutu testaus ikkunan kautta, jonka kautta voidaan nähdä mikä kunkin piirtokutsun aiheuttaa. Piirtokutsujen vähentämiseksi efektin käyttämät tekstuurin kerätään yhteen suureen tekstuuriarkkiin. Tekstuuriarkki on iso kuva, johon koottaan kaikki tarvittavat tekstuurit.

Tekstuuriarkin luonti on helppoa Photoshop -ohjelmalla. Tekstuuriarkin pohjalle halutaan ruudukko, jotta tekstuurit on helppo asetella paikoilleen. Ruudukon luominen Photoshop ohjelmassa onnistuu käyttämällä apuasettelu toimintoa. Apuasettelulla voidaan luoda apulinjat säännöllisin välimatkoin. Tässä tapauksessa luodaan kahdeksan kertaa kahdeksan ruudukko. Ruudukon koko riippuu siitä, kuinka monta tekstuuria siihen halutaan laittaa. Kun ruudukko on luotu, täytyy tekstuurit vain tuoda tiedostoon ja asetella ruutuihin (kuva 24). Jotkin efektin elementit ovat kuitenkin suurempia kuin toiset, joten ne vaativat suurempi resoluutioisen kuvan näyttääkseen hyvältä. Esimerkiksi efektissä esiintyvät loitsuympyrät vaativat suuremman tekstuurin kuin taikahiukkaset. Tämän vuoksi tehdään osa tekstuureista neljän ruudun kokoisiksi eli neljä kertaa neljä ruudukolle.



Kuva 24. Valmis tekstuuriarkki. Ylhäällä vasemmalla pienemmän resoluution tekstuurit.

Kun tekstuuriarkki on valmis, tuodaan se pelimoottoriin. Koska efektissä on kolme varjostinta, tulee tekstuuriarkista luoda kolme materiaalia: jokaiselle varjostimelle omansa. Tekstuuriarkki

asetetaan käyttöön joka partikkelijärjestelmälle. Oikea tekstuuri valitaan arkista käyttämällä teksturiarkki animaatio -parametria. Tekstuuriarkki animaatiolla voidaan luoda animoituja tekstuurreja efektiin tai saada efekti valitsemaan satunnainen tekstuuri arkin tekstuureista. Tässä tapauksessa halutaan, että partikkelijärjestelmä käyttää aina samaa tekstuuria arkista. Ensimmäiseksi täytyy asettaa arkin koko. Pienemmille tekstuureille arkin kooksi asetetaan kahdeksan kertaa kahdeksan ja suuremmille neljä kertaa neljä. Ruutu yli ajan asetetaan vakioksi, jolloin kohtaan syötetään halutun tekstuurin ruudun numero. On kuitenkin otettava huomioon, että Unityssä ruudut alkavat nolasta.

Tekstuuriarkin käyttöön asettamisen jälkeen voidaan huomata, että efekti aiheuttaa edelleen liikaa piirtokutsuja. Tämä johtuu siitä, että efektin kaikki partikkelijärjestelmät ovat samalla tasolla, joten Unity ei tiedä missä järjestyksessä sen pitäisi renderöidä ne, ja tämä rikkoo piirtokutsuja pienempiin osiin. Jotta Unity osaisi yhdistää piirtokutsut, täytyy jokaiselle partikkelijärjestelmälle määritellä sen piirtäjä asetuksista taso niin, että samaa materiaali käyttävät ovat samalla tasolla. 3D-malleja käyttävät partikkelijärjestelmät eivät voi olla samassa piirtokutsussa Spritejä käyttävien kanssa, joten ne täytyy asettaa eri tasolle, jotteivat ne riko piirtokutsuja. Myös jälkirenderöijä tarvitsee oman tasonsa. Jälkirenderöijä tekee tällä hetkellä monta piirtokutsua, sillä siinä on varjojen luominen päällä. Tämä laitetaan pois päältä, sillä sitä ei efektissä tarvita.

Tällä tavoin piirtokutsujen määrä saatiin nopeasti puolitettua. Piirtokutsuja ennen optimointia saattoi tapahtua jopa kaksitoista, mutta optimoinnin jälkeen niitä tapahtuu enimmillään kuusi. Yksi piirtokutsu saataisiin pois vielä poistamalla jälkiparametri sitä käyttävistä partikkelijärjestelmistä, sillä jälki materiaalina ei voida käyttää teksturiarkkia, joten se lisää yhden kutsun.

Nyt efekti on valmis käytettäväksi peliprojektissa.

## 7 Yhteenveto

Opinnäytetyön tavoitteena oli luoda pelikäyttöön soveltuva, valmis räjähdys efekti taikateemalla. Ennen efektin luontia, tutustuttiin partikkeliefektien teoriaan, jotta efektiä voitaisiin lähteä rakentamaan paremmalta pohjalta.

Teoriaosuudessa tutustuttiin siihen mitä partikkeliefektit ovat ja kuinka niitä voidaan käyttää peleissä. Työssä syvennyttiin myös partikkeliefektien suunnitteluun ja luontiin, ja siihen miten efektin väri, muoto ja ajoitus vaikuttavat. Teoriaosuudessa tarkasteltiin myös efektien performanssia, ja siihen vaikuttavia asioita sekä kuinka performanssia voidaan parantaa. Käytännön osuudessa suunnittelin ja rakensin efektin Unity-pelimoottorissa hyödyntäen teoriaosuudessa käytyjä asioita.

Lopullinen efekti poikkesi hieman alkuperäisestä suunnitelmasta. Suunnitelmassa taikaspiraaleja oli kaksi ja lopullisessa versiossa vain yksi. Suunnitteluvaiheessa olin jo miettinyt kumpaakin vaihtoehtoa, mutta päädyin lopulta laittamaan designiin kaksi. Kuitenkin luodessani efektiä tykästyin yhden spiraalin versioon niin, että päätin pitää efektin sellaisena. Toinen poikkeavuus efektissä on savu. Olin suunnitellut efektin räjähdys synnyttävän savua, mutta efektiä tehdessäni huomasin savun olevan liikaa efektien muiden elementtien lisäksi, ja se peitti ikävästä joitain efektin osia.

Projektiosiossa törmäsin muutamiin ongelmiin efektiä luodessa. Partikkeliefektejä on niin monta tapaa luoda itse pelimoottorissa, joten jouduin usein miettimään mikä olisi paras lähestymistapa juuri kyseisen efektin osan luontiin. Välillä ratkaisu löytyi vain yrityksen ja erehdyksen kautta. Myös monista tekstuureista jouduin luomaan useampia versioita, sillä ne eivät näyttäneetkään lopulta siltä mitä olin kuvitellut.

Työn lopputulokseen olen tyytyväinen, ja työtä tehdessäni opin paljon uusia asioita efektien luonnista ja myös Unity-pelimoottorista. Opin myös paljon varjostimien luonnista. Erityisesti efektien optimoinnista opin todella paljon uutta, varsinkin piirtokutsuista.

Tulevaisuutta ajatellen tästä opinnäytetyöstä on paljon hyötyä, sillä opin paljon teoriaa efektien luonnista opinnäytetyötä kirjoittaessani ja uusia tapoja luoda efektejä.

Työni oli minusta sopivan haastava ja pitkä. Se myös opetti paljon uutta. Jatkossa aion luoda lisää partikkeliefektejä, erityisesti kokeillen erilaisia visuaalisia tyylejä.



## Lähteet

- 1 Make a Particle Explosion Effect. Saatavilla: <https://gamedev.net/tutorials/visual-arts/make-a-particle-explosion-effect-r2701>
- 2 Technologies U. Unity - Manual: What is a Particle System? Saatavilla: <https://docs.unity3d.com/2018.3/Documentation/Manual/ParticleSystemWhatIs.html>.
- 3 Introduction to Particle Systems | cesium.com. Saatavilla: <https://cesium.com/docs/tutorials/particle-systems/>.
- 4 Walt Disney Tips & Techniques Shape Language. Saatavilla: [https://www.waltdisney.org/sites/default/files/2020-04/T%26T\\_ShapeLang\\_v9.pdf](https://www.waltdisney.org/sites/default/files/2020-04/T%26T_ShapeLang_v9.pdf).
- 5 Basic Color Theory. Saatavilla: <https://www.colormatters.com/color-and-design/basic-color-theory>.
- 6 Analogiset ja täydentävät värit - Mikä on niiden ero ja miten niitä käytetään graafisessa suunnittelussa? | Print Peppermint. Saatavilla: <https://www.printpeppermint.com/fi/vastaavat-t%C3%A4ydent%C3%A4v%C3%A4t-v%C3%A4rit%2C-mik%C3%A4-on-ero-ja-kuinka-voin-k%C3%A4ytt%C3%A4%C3%A4-niit%C3%A4-graafisissa-kuvioissani/>.
- 7 The complete guide to creating visual effects within League of Legends. Saatavilla: [https://nexus.leagueoflegends.com/wp-content/uploads/2017/10/VFX\\_Styleguide\\_final\\_public\\_hidpjwx7lqyx0pjj3ss.pdf](https://nexus.leagueoflegends.com/wp-content/uploads/2017/10/VFX_Styleguide_final_public_hidpjwx7lqyx0pjj3ss.pdf).
- 8 VFX Staples: Shape, Color, and Motion. 2018; Saatavilla: <https://80.lv/articles/vfx-staples-shape-color-and-motion/>.
- 9 VFX Production For AAA Video Games. 2017; Saatavilla: <https://80.lv/articles/the-secrets-of-vfx-production-for-doom/>
- 10 Technologies U. Unity - Manual: Draw call batching. Saatavilla: <https://docs.unity3d.com/Manual/DrawCallBatching.html>.