Samuli Hannuksela

Unity 5- ja Unreal Engine 4 -pelimoottorien vertailu

Tradenomi

Syksy 2016

KAJAANIN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

KAJAANIN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

TIIVISTELMÄ

Tekijä: Samuli Hannuksela

Työn nimi: Unity 5- ja Unreal Engine 4 -pelimoottorien vertailu

Koulutusala: Tradenomi, tietojenkäsittely

Asiasanat: Unity, Unreal, pelimoottori, peli, ohjelmointi, peliala

Opinnäytetyössä vertaillaan Unity 5- ja Unreal Engine 4 -pelimoottoreita keskenään niiden erojen, samanlaisuuksien ja hyvien ja huonojen puolien selvittämiseksi. Opinnäytetyössä suoritetaan myös kolme käytännön tutkimusta.

Pelimoottorien ominaisuuksien vertailussa Unity 5 on useamman ohjelmoijan käytettävissä, sillä Unity 5 tukee useampaa ohjelmointikieltä. Unreal Engine 4 taas sisältää enemmän erikoistuneita työkaluja ja kuvallisen ohjelmointijärjestelmän. Muuten pelimoottorit ovat karkeasti samanveroisia.

Käytännön tutkimusten mukaan Unity 5 käytti huomattavasti vähemmän resursseja kuin Unreal Engine 4 ja pystyi laskemaan enemmän fysiikkaa hidastumatta. Suurin ero oli muistien käytössä. Unity 5 käytti huonoimmassakin vertailukohdassa vain kaksi kolmasosaa Unreal Engine 4:n muistinkäytöstä. Unreal Engine 4 pystyi kuitenkin näyttämään enemmän hiukkasia ilman hidastumista kuin Unity 5.

Pelimoottorien työkalut ja ominaisuudet mukaan lukien Unity 5 sopii paremmin vähän resursseja omaaviin laitealustoihin ja fysiikkaan pohjautuviin sovelluksiin. Unreal Engine 4 sopii paremmin graafisesti vaativiin sovelluksiin.

KAJAANIN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

ABSTRACT

Author: Samuli Hannuksela

Title of the Publication: Unity 5 and Unreal Engine 4 game engine comparison

Field of study: Bachelor of Business Administration, Information Technologies

Keywords: Unity, Unreal, game engine, game, programming, games industry

This paper will compare Unity 5 and Unreal Engine 4 game engines to find out their differences, similarities as well as their pros and cons. It will compare their properties and tools and test the game engines with three experiments.

In the comparison between the game engines, Unity 5 is more accommodating for programmers, since it supports multiple programming languages. Unreal Engine 4 contains more specialized tools and a visual programming tool. Aside from those differences, the engines are roughly equivalent.

According to the experiments Unity 5 used considerably less resources than Unreal Engine 4 and could process more physics without slowing down. The largest difference was with the memory usage between the two game engines. Unity 5 used two thirds of the memory used by Unreal Engine 4 at the smallest point of difference. On the other hand, Unreal Engine 4 could show much more particles than Unity 5 could without slowing down.

The paper's conclusion is that Unity 5 works better for situations where less resource usage is needed or more physics needs to be calculated, whereas Unreal Engine 4 works better for graphically intensive programs.

KAJAANIN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

SISÄLLYS

SYMBOLILUETTELO

SYMBOLILUETTELO

Ambient occlusion: Tekniikka, jossa lasketaan eri pintojen taustavaloisuus peittyvyyden perusteella. Mitä peittyneempi pinta on, sitä vähemmän taustavaloa siihen loistaa. Esimerkiksi suljettu putki, joka pimenee syvemmällä.

Android: Suosittu älypuhelimissa käytetty käyttöjärjestelmä ja laitealusta.

Epic Games: Unreal Engine -pelimoottoreita kehittävä yritys.

Grafiikka: Yleistermi tietokoneissa käytettäviin kuvallisiin asioihin ja tietokoneen ruudulla näytettäviin kuviin.

Hertsi: Taajuuden yksikkö. Yksi tietokoneiden osissa käytetty nopeutta kertova yksikkö, jota käytetään yleensä muodossa megahertsi (MHz, miljoona hertsiä) tai gigahertsi (GHz, miljardi hertsiä).

Jälkikäsittely: (Tässä tutkielmassa käytettynä) Yleisesti peleissä käytetty ruudulle piirron vaihe. Näytönohjain käsittelee muistissa olevaa jo piirrettyä grafiikkaa ennen ruudulle näyttämistä. Käytetään yleensä tehosteisiin.

Kuvannin: (Tässä tutkielmassa käytettynä) Englannin sanasta ”renderer”. Tarkoittaa tietokoneen grafiikkaa kaksi- ja kolmiulotteisista malleista piirtävää ohjelmaa, joka toimii ennen varjostinta.

Lykätty varjostus: (Tässä tutkielmassa käytettynä) Englannin sanasta ”deferred shading”. Tekniikka, jossa varjostin laskee eri varjostuksen osat erikseen varjostimen tehostamiseksi.

Occlusion Culling: Tekniikka, jossa täysin toisten läpinäkymättömien kappaleiden peitossa olevat kappaleet jätetään piirtämättä.

Resurssi: Yleistermi tietokoneen käyttämiin osiin, joita on rajallinen määrä käytettäväksi.

Sarjallistus: (Tässä tutkielmassa käytettynä) Englannin sanasta ”serialization”. Muuttujille tehtävä toimenpide, jossa muuttujien muoto käännetään pelimoottorin

käyttämästä muodosta helpommin siirrettävään muotoon. Käytetään esimerkiksi tallentamisessa ja tietoverkon yli siirtämisessä.

Screen Space Ambient Occlusion: Erillinen tekniikka tehokkaaseen taustavalaistuksen laskemiseen varjostimessa.

Sprite: Kaksiulotteinen erillisistä pikseleistä koostuva kuvatietorakenne.

Unity Technologies: Unity-pelimoottoreita kehittävä yritys.

Varjostin: (Tässä tutkielmassa käytettynä) Englannin sanasta ”shader”. Tarkoittaa tietokoneen grafiikkaa ruudulle piirtävää ohjelmaa, joka toimii juuri ennen ruudulle piirtoa.

Videomuisti: Näytönohjaimen käyttämä muisti. Yleensä sijaitsee näytönohjaimessa.

Ylikellotus: Suorittimelle ja näytönohjaimelle tehtävä toimenpide, jossa osa säädetään toimimaan nopeammin kuin alun perin tarkoitettu.

1 JOHDANTO

Tutkielmassa vertaillaan Unity Technologiesin Unity 5 -pelimoottoria ja Epic Gamesin Unreal Engine 4 -pelimoottoria keskenään. Tässä opinnäytetyössä otetaan huomioon kummankin pelimoottorin ominaisuudet ja työkalut ja suoritetaan kolme kokeellista tutkimusta. Tutkielman tarkoituksena on selvittää Unity 5 ja Unreal Engine 4 -pelimoottorien erot ja samanlaisuudet ja erilliset hyvät ja huonot puolet.

Opinnäytetyön kirjoituksen aikana Unity 5 ja Unreal Engine 4 ovat kaksi suurinta kaupallista pelimoottoria tarjolla ja kummatkin kilpailevat pitkälti samoista asiakkaista. Kummatkin pelimoottorit tarjoavat tuen kaikille moderneille pelialustoille ja eri rahoitustason ja -menetelmän pelinkehittäjille. Kummatkin tarjoavat karkeasti samat työkalut ja ominaisuudet yleisiin tarkoituksiin. Suuria eroja ei ole paljon ja nekin ovat lähinnä tietyissä tilanteissa merkittäviä.

Pelimoottorien ominaisuudet ja työkalut selvitetään pelimoottorien omista ohjeistuksista, tehokkuus ja käytettävyys sen sijaan tutkitaan käytännön mittauksilla. Selvitysten ja tutkimusten jälkeen pelimoottoreita verrataan toisiinsa ja päätellään niiden hyviä ja huonoja puolia.

Opinnäytetyön on tarkoitus auttaa kehittäjiä, jotka eivät ole varma kumpi pelimoottori sopisi heidän peliprojektiinsa paremmin. Antamalla selkeän kuvan kummastakin pelimoottorista ja niihin sopivista projekteista tutkielma auttaa muita kehittäjiä valitsemaan heille sopivamman pelimoottorin.

2 PELIMOOTTOREISTA YLEENSÄ

Pelimoottori on pelin pohjana oleva ja taustalla toimiva ohjelmisto, joka tarjoaa selvän paketin valmiita osia ja järjestelmiä pelin ohjelmiston keskeisiin rakenneosiin. Esimerkiksi kaikki kolmiulotteiset pelimoottorit tarjoavat kuvanninjärjestelmän, joka piirtää pelimaailman grafiikan käyttäjälle pelin määrittelemällä tavalla, ja käyttäjän syötteen huomioivan osan. [1, s. 11–13]

Pelimoottorin tarjoamat osat yksinkertaistavat ja erottavat osien toiminnan ja pelin toiminnan toisistaan, mikä auttaa pelinkehittäjiä keskittymään peliin ja sen tarpeisiin kehityksessä. Tavallisesti pelimoottori tarjoaa kuvanninjärjestelmän lisäksi myös osat käyttäjän syötteen lukemiseen, muistista lataamiseen, fysiikan mallintamiseen, käyttöliittymään luomiseen, animointiin ja äänen käyttämiseen pelissä. Moni pelimoottori mahdollistaa myös pelinkehittämisen monelle eri pelialustalle samanaikaisesti ilman suuria muutoksia peliin. [2, s. 1]

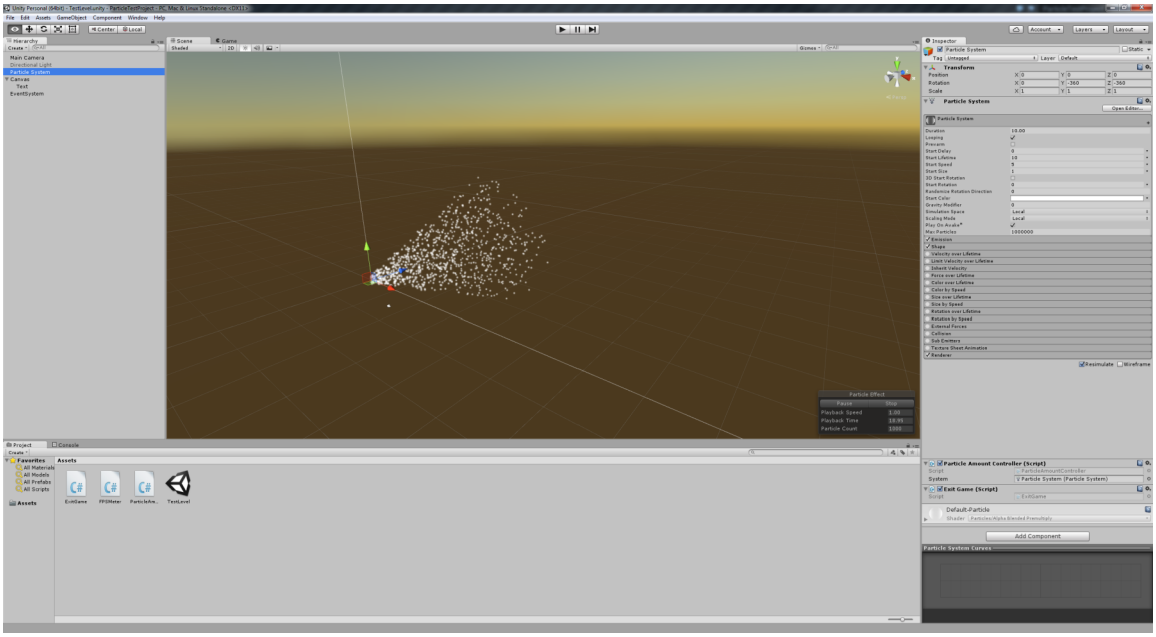
Pelimoottorin tarjoamat osat ovat yleensä erotettu pelistä ja pelin omista järjestelmistä kohtuullisen selkeästi, mutta tämä erotus ei ole välttämättä millään tavalla selkeä tai edes käytännössä olemassa. Jos peliin käytettyä moottoria on melkein mahdotonta käyttää toisen pelin pelimoottorina, on mahdollisesti järkevämpää puhua kyseisestä kokonaisuudesta pelkästään pelinä ja jättää pelin ja pelimoottorin erotus tekemättä. Esimerkiksi pelimoottoriin kiinni ohjelmoitu pelilogiikka voi tehdä pelimoottorista käyttökelvottoman muissa peleissä. [1, s. 11–13]

Pelimoottorit voidaan karkeasti jakaa kolmeen osaan: tee-se-itse-pelimoottoreihin, joiden päälle pelinkehittäjien pitää kuitenkin pitkälti rakentaa pelimoottori itse, vaikka joitain pohjarakenteita on jo valmiina mutta jotka antavat samalla eniten vapautta pelinkehittäjille; lähes valmiisiin pelimoottoreihin, jotka rajoittavat pelisuunnittelua jonkin verran, mutta antavat kattavan paketin eri osia pelinkehittäjille ja eivät vaadi suurta määrää viimeistelyä; valmiisiin pelimoottoreihin, joissa pelinkehittäjien ei tarvitse käyttää juuri ollenkaan ohjelmointikieliä pelimoottorin ohjelmoimiseen peliä varten mutta joita on hankala muokata pelisuunnittelun tarpeisiin. Kummatkin opinnäytetyön pelimoottoreista ovat valmiin ja lähes valmiin pelimoottorin välimaastossa. [2, s. 2]

3 UNITY 5 -PELIMOOTTORI

Unity 5 on Unity Technologiesin kolmas maaliskuuta 2015 julkaisema pelimoottori ja yhtiön uusin julkaistu versio Unity-pelimoottoreiden sarjassa [3]. Unity-pelimoottorien suurin etu muihin moottoreihin on laaja valikoima eri alustoja, joissa Unityä voi käyttää [4].

Kuvassa 1 on Unity 5 -pelimoottorin muokkaimen pääikkuna avattuna tämän tutkielman hiukkaskokeiluohjelman projektiin. Kuvassa näkyy muokkaimen viisi osaa. Keskellä näkyy muokkaimen päänäkymä muokkaimessa avattuun pelin kenttään, ja vasemmalla ovat avatussa kentässä olevat kappaleet. Vasemmalla sinisellä taustalla oleva kappale, kuvassa näkyvä hiukkasjärjestelmä, on valittuna. Alaosassa näkyvät olemassa olevat tallennetut kappalepohjat, joita voi käyttää pelin rakennuksessa. Oikealla näkyvät valitun kappaleen eri rakenneosat ja niiden ominaisuudet. Oikealla alhaalla näkyy esinäkymä valitusta kappaleesta, joskin kuvassa ei näy juurikaan muuta kuin esinäkymän tyhjä tausta.



Kuva 1. Unity 5 -pelimoottorin muokkain avattuna tämän tutkielman hiukkaskokeiluohjelmaan.

3.1 Unity-pelimoottorien historiaa

Unity Technologies julkaisi ensimmäisen version Unitystä vuonna 2005 [5]. Ensimmäisessä versiossa pelimoottorin tärkeimpinä ominaisuuksina olivat toimiva animaatiojärjestelmä kolmiulotteisille kappaleille ja kaksiulotteisille kuville, tuki yleisimmille äänen tallennusmuodoille, monipuolinen fysiikkamallinnus, muuttujien tehokas sarjallistus ja C#-ohjelmointikielen käyttö. Unity 1.5 lisäsi tuen pelimoottorin käyttämiseen internet-selaimessa ilman erillistä asennusta [6, Osa Unity 1.5]. [7]

Unity 2.0 sisälsi uudet käyttöliittymätyökalut ja lisäsi tuen DirectX 9.0 -kirjastolle, videoiden toistamiselle, verkkopelien tekemiselle ja reaaliaikaisille varjoille [6, Osa Unity 2.0]. Unity 2.1 lisäsi suoralatauksen mahdollisuuden pelin maaston ja kappaleiden ominaisuuksien osalta ja toi samalla käynninaikaisen animoinnin ja valaistuksen pelimoottoriin [6, Osa Unity 2.1]. Unity 2.6 toi Profiler-työkalun, jolla voi tarkastella pelin resurssienkäyttöä käynnin aikana, ja kattavamman animointijärjestelmän [6, Osa Unity 2.6].

Unity 3.0 lisäsi tuen Android-käyttöliittymälle ja päivitti pelimoottorin fysiikanmallinnusta ja kuvantavaa osaa sekä muita osia [6, Osa Unity 3.0]. Unity 3.2 lisäsi mahdollisuuden tarkastella Profiler-työkalulla erillisten peliversioiden resurssienkäyttöä ja lisäsi uuden esitehdyn vesimallin sekä uusia kuvatehosteita, kuten syväterävyysaluetehosteen [6, Osa Unity 3.2].

Unity 4.0 toi tuen reaaliaikaisten varjojen luomiseen, eli pelin sisällä olevat varjot voivat muuttua suoraan pelin tapahtumien mukaan, ja uuden Mecanim-animaatiojärjestelmän, jolla voi helpommin animoida pelin eri kappaleita ja olioita [8]. Unity 4.6 lisäsi uudet käyttöliittymätyökalut rakenneosiin pohjautuvien käyttöliittymien tekemiseen ja uuden tapahtumaviestijärjestelmän [9].

3.2 Ominaisuudet

Unity 5 -pelimoottorin ohjelmointi perustuu joko C#-, JavaScript- tai Boo-ohjelmointikieliin. Suurin osa käyttäjistä käyttää C#-kieltä ja huomattava vähemmistö JavaScript-kieltä. Boo-kieli on edelleen tuettu pelimoottorin

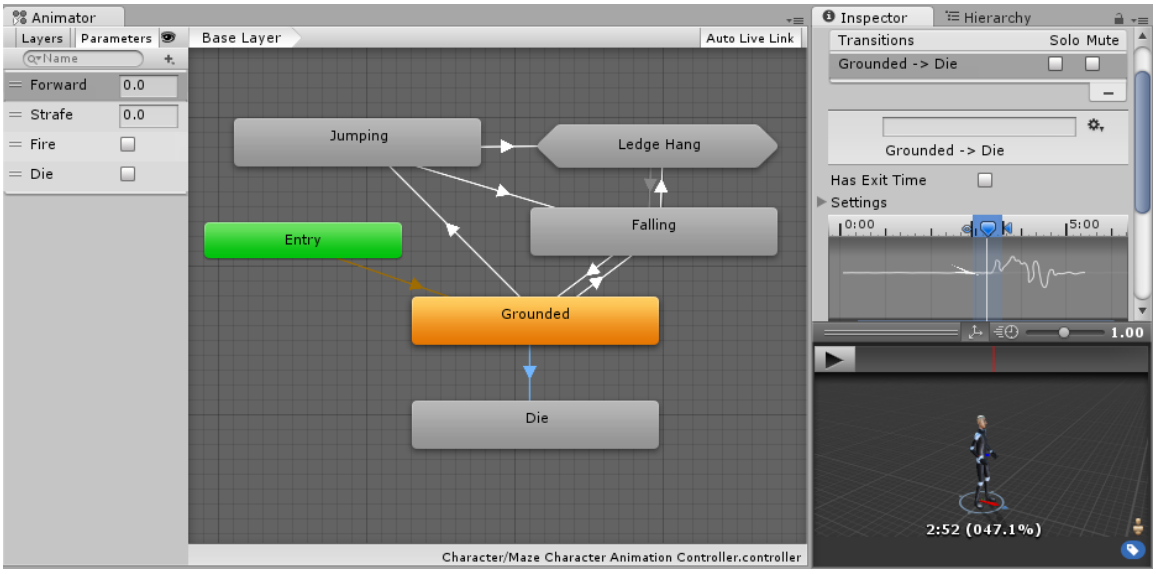
ohjelmointikielen kääntäjän mukaan, mutta Unity Technologies on lopettanut Boo-kielen ohjeiden tarjoamisen Unity 5 -pelimoottorin ohjeistuksista, Boo-kielen käyttäjien pienen määrän takia. [10]

Unity 5 voi myös käyttää C-, C++-, Objective-C- tai Java-ohjelmointikielillä ohjelmoituja lisäohjelmakirjastoja. Verkko-ohjelmoinnissa Unity 5 tukee WWW-toimintojen avulla tietojen hakemista verkkosivuilta ja .NET-pistokkeita. Reitinhakuun voi käyttää Unity 5:n omaa navigointiverkonluontia. [11]

Graafisten ominaisuuksien puolesta Unity 5 tarjoaa fysiikkaan pohjautuvan varjostimen, joka tarjoaa korkean kuvanlaadun voimakkaammille pelialustoille. Unity 5 antaa myös sisäänrakennetun pääsyn matalan tason kuvanninohjelmointiin ja tekstuureille kuvantamiseen. Unity 5 sisältää laajan määrän jälkikäsittelytehosteita, kuten kukoistustehosteen ja kuvan sävytyksen käyttäen kolmiulotteisia tekstuureita. [11]

Animointiin Unity 5 -pelimoottorissa on Mecanim-järjestelmä. Mecanimissa on yksinkertaistettu ja helppokäyttöinen työmenetelmä, joka sopii kaikkiin Unity 5:ssä animoitaviin kappaleisiin. Ihmismäisiä malleja varten Mecanim antaa työkalun sovittaa ihmisanimaatiot kaikkiin ihmismäisiin malleihin. Koko järjestelmän hallintaan on kuvallinen ohjelmointityökalu, joka näkyy kuvassa 2. [12]

Kuvassa 2 näkyy Unity 5:n virallisessa ohjeistuksessa oleva esimerkkikuva ihmismallin animaatioista. Kuvassa näkyy samantapainen rakenne kuin kuvassa 1, mutta ilman tallennettuja kappaleita. Keskellä näkyy eri animaatioita kuvaavat kappaleet ja niiden yhteydet toisiinsa. Vasemmalla näkyy animaation tiloihin ja vaihdoksiin vaikuttavat muuttujat, joiden tilaa voi muokata muualta pelimoottorista. Oikealla ylhäällä näkyy valitun vaihdoksen, sininen nuoli ”Grounded” animaatiosta ”Die” animaatioon, ominaisuudet ja alhaalla näkyy esinäkymä animaatiosta. [12]

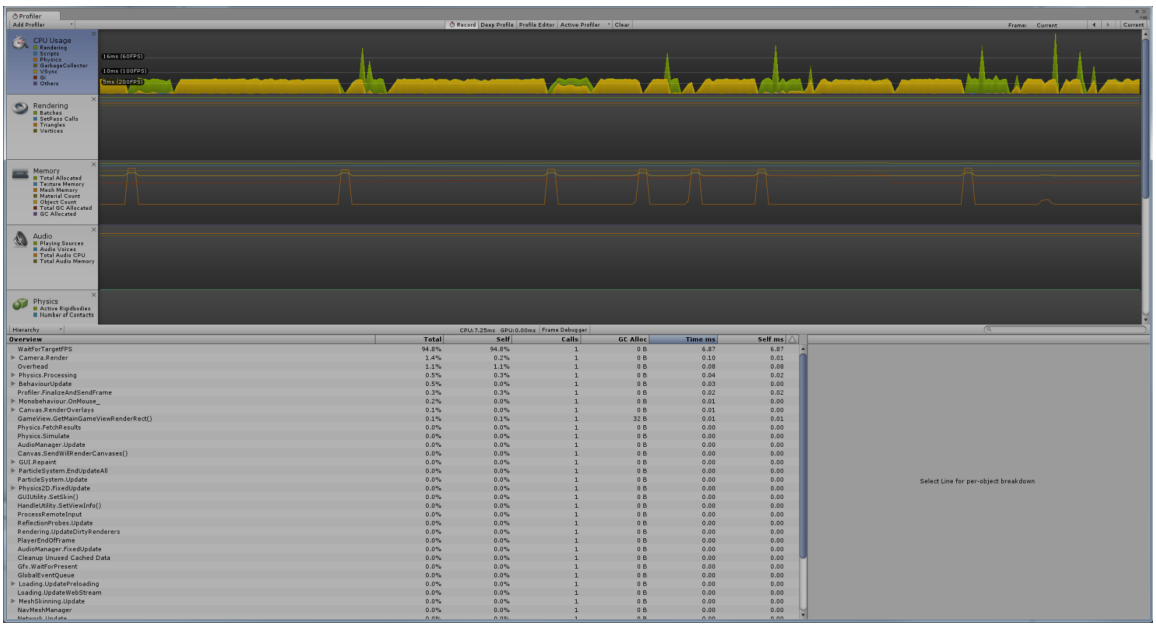


Kuva 2. Unity 5 -pelimoottorin Mecanim-järjestelmän ohjelmointityökalu. [12]

Fysiikanmallinnukseen Unity 5 antaa mahdollisuuden monisäkeiseen mallinnukseen, mikä tehostaa fysiikanmallinnusta monisäkeisyyttä tukevilla alustoilla. Unity 5:n fysiikanmallinnus antaa tarkan törmäyksentunnistuksen, sisäänrakennetun kangasrakenneosan ja rengasmallinnuksen iskunvaimennuksella ja kitkavoimilla. Unity 5 tukee optimoinnin kannalta pysyviä erotteluja pelimaailman paikallaan pysyville rakenteille. Pelimoottori pystyy laskemaan pysyviin rakenteihin liittyvät fysiikat nopeammin. [11]

Unity 5 sisältää tuen myös kaksiulotteisille peleille. Kaksiulotteisen grafiikan käyttämiseen Unity 5 tarjoaa automaattisen spritejen pakkaajan, spritesivujen luojan ja automaattisen spritejen animoinnin. Automaattinen spritejen pakkaaja sisältää myös mahdollisuuden käyttää yleisesti ylimääräiseksi jäävä tyhjä tila hyödyksi. Kaksiulotteinen fysiikanmallinnus tarjoaa laajalti samat tai verrattavat ominaisuudet kuin kolmiulotteinen fysiikanmallinnus [13]. [11]

Optimointiin Unity 5:ssä on mittaustyökalu Profiler, joka näyttää pelin käyttämien resurssien määrän ja käytön lähteen, mikä helpottaa pelin optimointia. Kuvassa 3 näkyy Profiler-työkalu mittaamassa tämän tutkielman hiukkasmittausohjelmaa. Kuvassa näkyy ylhäällä ohjelman eri resurssienkäytöt ja alhaalla näkyy eri komentojen käyttämä aika ja muistinkäyttö. Oikealla alhaalla näkyisi eri kappaleiden käyttämä muistinkäyttö komentoon liittyen ja aika valitun komennon suorittamiseen. [11]



Kuva 3. Unity 5 -pelimoottorin Profiler-mittaustyökalu, joka näyttää resurssien käytön, avattuna tämän tutkielman hiukkaskokeiluohjelmaan.

Unity 5 tukee myös monia optimointitekniikoita, kuten Occlusion Culling -tekniikka kuvantamaan ainoastaan pelaajalle näkyvät kappaleet, automaattinen dynaaminen erottelu siirtämään kuvantimen vaatima tieto näytönohjaimelle tehokkaasti ja lykätty varjostus -tekniikka oikealta näyttävän grafiikan kuvantamiseen tehokkaasti. [11]

3.3 Esimerkkejä

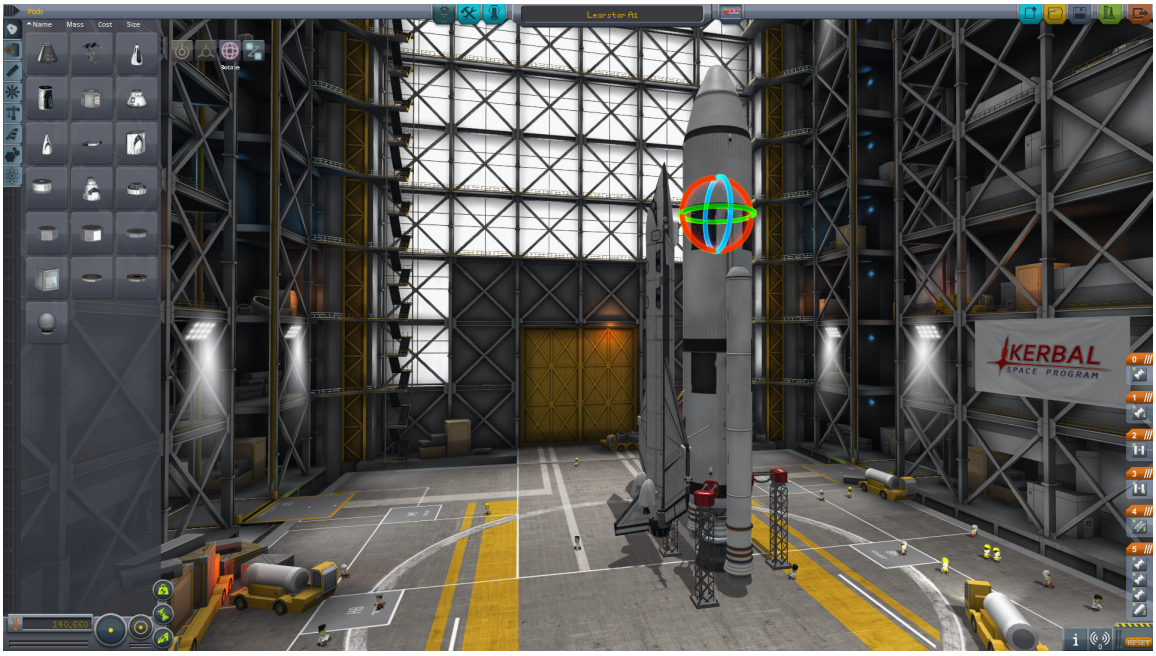
Seuraavaksi esitellään kaksi Unity 5 -pelimoottoria käyttävää peliä. Kumpaakin peliä kehitettiin ensimmäiseksi vanhemmilla Unity-pelimoottoreilla, joista vasta myöhemmin päivitettiin Unity 5 -pelimoottoriin.

3.3.1 Kerbal Space Program

Kerbal Space Program (KSP) on Unity-pelimoottorilla tehty peli avaruusohjelman hoitamisesta Squad-nimiseltä peliyhtiöltä [14]. KSP:n kehitys aloitettiin Unity Technologiesin Unity 3 -pelimoottorilla, joka päivitettiin Unity 4 -pelimoottoriin 14. helmikuuta 2013 [15]. KSP päivitettiin Unity 5 -pelimoottoriin 19. huhtikuuta 2016 [16].

Kerbal Space Programissa on tarkoitus rakentaa oma avaruusohjelma. Pelaaja rakentaa avaruusohjelman käyttämät raketit, alukset ja muut laitteet yhdistämällä osia toisiinsa haluamallaan tavalla. Rakentamisen jälkeen pelaaja myös ohjaa avaruusohjelman lennot itse ilman automaattiohjausta. KSP:n haasteena on sen varsin tarkasti todellisuutta jäljittelevä fysiikanmallinnus aerodynamiikasta kiertoratafysiikkaan ja oikeita raketteja vastaaviin osiin. [17]

Kuva 4 on KSP:n markkinointiin käytetty kuvankaappaus. Kuvassa näkyy KSP:n raketinmuokkainosa. Keskellä näkyy pelaajan rakentama sukkula ja kantoraketti taustalla näkyvässä alusten rakennushallissa. Ylhäällä näkyy aluksen nimi sekä nappulat pikanäppäinvalikon ja käytettyjen astronauttien valikon avaamiseen. Vasemmalla näkyy lista mahdollisista käytettävistä osista avattuna komento-osasivulle ja alavasemmalla näkyy aluksen hinta sekä muutama painike muokkaimen käyttämiseen. Oikealla näkyy osien käynnistysjärjestys eri vaiheissa, joita pelaaja voi käyttää eri osien oikeaan ajoitukseen, kuten raketin käynnistymiseen ja raketin tiellä olevan osan irrottamiseen samaan aikaan. Vaiheiden takana näkyy KSP:n logo lippuna. Viimeisenä oikealla ylhäällä näkyy, vasemmalta oikealle, nappulat uuden aluksen tekemiseen, tallennetun aluksen lataamiseen, nykyisen aluksen tallentamiseen, aluksen laukaisemiseen ja rakennushallista poistumiseen.



Kuva 4. Kuvankaappaus Kerbal Space Program -pelistä. [17]

3.3.2 Homeworld: Deserts of Kharak

Homeworld: Deserts of Kharak (Homeworld) on Unity 5 -pelimoottorilla tehty reaaliaikainen strategiapeli. Blackbird Interactive aloitti pelin kehityksen Unity 2.6 -pelimoottorilla lokakuussa 2010 ja julkaisi lopullisen pelin Unity 5 -pelimoottorilla 20. tammikuuta 2016. [18]

Homeworld: Deserts of Kharak on sotastrategiapeli, jossa pelaaja ohjaa omia sotajoukkoja ja yrittää joko tuhota vastustajat tai saavuttaa muita tavoitteita, pelimuodosta riippuen. Homeworldissa tärkein yksikkö on pelaajan oma emoalus, jossa rakennetaan pelaajan omat sotajoukot kentältä kerättävien resurssien avulla ja joka toimii pelaajan päämajana. Jos pelaajan emoalus tuhoutuu, pelaaja häviää pelin. [19]

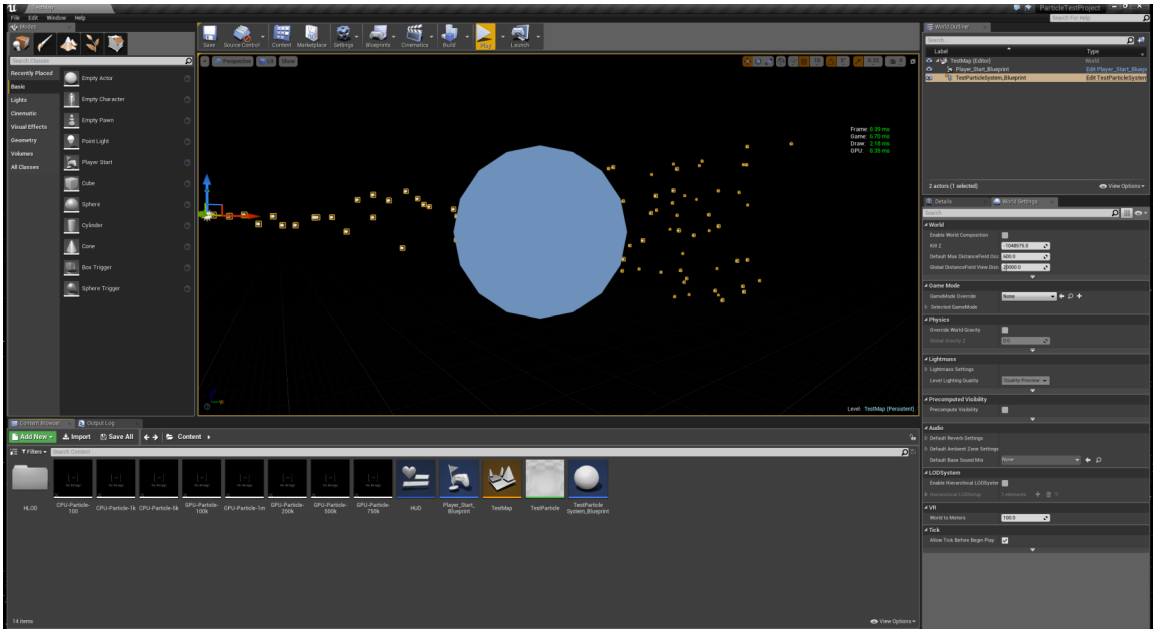
Kuva 5 on Homeworldin markkinointiin käytetty kuvankaappaus. Kuvassa näkyy taistelu keltaisen ja punaisen puolen välillä. Vasemmalla alhaalla keltaisen puolen pelaajan emoalus on piilossa kallion takana kuvan yläosassa olevilta punaisen puolen joukoilta. Kallion solassa eri puolten sotajoukot taistelevat toisiaan vastaan, jättäen jälkeen kuoppia ja romua yhteenotosta. Kuvan oikeassa yläkulmassa näkyy Homeworldin logo.

Kuva 5. Kuvankaappaus Homeworld: Deserts of Kharak -pelistä. [19]

4 UNREAL ENGINE 4 -PELIMOOTTORI

Unreal Engine 4 on Epic Gamesin 19. maaliskuuta 2014 julkaistu pelimoottori ja uusin julkaisu Unreal Engine -pelimoottorien sarjassa. Unreal Engine 4 tarjoaa täyden pääsyn pelimoottorin ohjelmakoodiin, mikä tekee pelimoottorista joustavan ja laajennettavan pelinkehittäjien toimesta. [20]

Kuvassa 6 näkyy Unreal Engine 4 -pelimoottoriin muokkain päävalikko avattuna tämän tutkielman hiukkasmittausohjelman projektiin. Kuvassa näkyy muokkaimen viisi eri osaa. Keskellä näkyy avattu kenttä ja kentässä olevat kappaleet. Kuvassa kappaleita ovat keskellä kameran edessä oleva pelaajan aloitussijainti ja vasemmalta lähtevä hiukkasjärjestelmä. Vasemmalla on lista eri peruskappaleista, joita kehittäjät voivat lisätä kentälle. Lista on kuvassa avattuna yksinkertaisiin kolmiulotteisiin malleihin. Alhaalla näkyy projektin tallennetut kappalepohjat. Oikealla ylhäällä näkyy lista eri kentällä olevista kappaleista, joista ruskean oranssilla taustalla oleva on valittu. Listan alapuolella näkyy valitun kappaleen ominaisuudet ja rakenneosat. Kaikissa valikoissa on kuvassa ylhäällä oleva työkalurivi, joko keskellä tai vasemmalla ylhäällä. Työkalurivissä olevat työkalut vaihtelevat valikosta toiseen.



Kuva 6. Unreal Engine 4 -pelimoottori avattuna tämän tutkielman hiukkaskokeiluohjelmaan.

4.1 Unreal-pelimoottorien historiaa

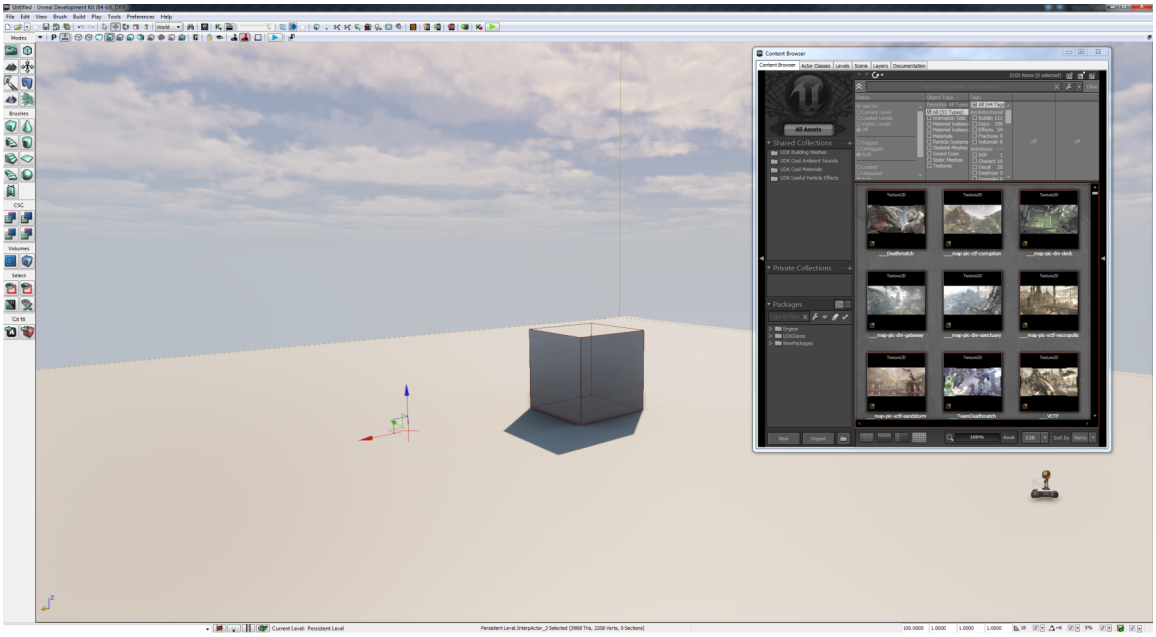
Ennen varsinaisia Unreal Engine -pelimoottoreita Epic Gamesin perustaja ja ensimmäisen Unreal Enginen pääkehittäjä Tim Sweeney oli vuonna 1991 tehnyt ZZT nimisen pelin, jonka pohjalla oli pelimoottorien tapainen rakenne. ZZT:n rakenne mahdollisti 1990-luvun tasolle erittäin laajan käyttäjämuokattavuuden ja toimi konseptuaalisesti Unreal Engine -pelimoottoreiden pohjatyönä. [21, s. 1]

Unreal Engine 1 kehitettiin vuonna 1998 Unreal nimisen pelin moottoriksi. Muihin aikakauden pelimoottoreihin verrattuna Unreal Engine 1 pystyi mallintamaan laajat ulkomaisemat ja yksityiskohtaiset sisämaisemat paremmin ja helpommin kuin muut pelimoottorit. Muut yhtiöt kiinnostuivat käyttämään pelimoottoria itse, mikä kannusti pelimoottorin kehittäjiä keskittymään työkalujen helppokäyttöisyyteen ja teknisten hidasteiden poistamiseen. [21, s. 1–2]

Unreal Engine 2 kehitettiin mahdollistamaan pelien kehittämisen sekä tietokoneille että konsoleille samalla kehitystyökalulla. Pelimoottori sisälsi suoraan tuen Sonyn Playstation 2:lle, Microsoftin Xboxille ja Nintendon Gamecubelle. Unreal Engine 2 julkaistiin vuonna 2002 pelillä America's Army [22]. Unreal Engine 2:een lisättiin laajemmin tukea muille työkaluille. Pelimoottoriin lisättiin sisäänrakennettu tuki Karma-fysiikkamoottorille, uusi hiukkasjärjestelmä ja kuvanninjärjestelmä. Unreal Engine 2:een parannettiin myös Unreal Enginen muokkausohjelman käytettävyyttä ja tehokkuutta [21, s. 2–3]

Unreal Engine 3 oli kehitetty julkaistavaksi samaan aikaan kuin seitsemännen sukupolven konsolit. Pelimoottori julkaistiin vuonna 2006 pelillä Gears of War [23]. Unreal Engine 3 mahdollistaa pelien kehittämisen ilman, että pelinkehittäjien tarvitsee ohjelmoida pelimoottoria, mikä mahdollistaa prototyyppien nopean kehittämisen. Pelimoottoria on kehitetty julkaisun jälkeen ja tuettuja alustoja on lisätty uusien pelialustojen tullessa markkinoille tai suosituiksi. Myöhemmin Epic Games julkaisi Unreal Engine 3:sta ilmaisen rajoitetumman version nimellä Unreal Development Kit (UDK), joka näkyy kuvassa 7. [21, s. 3–4]

Kuva 7 on uusi projekti ilman muokkauksia. Kuvan keskellä näkyy tyhjän projektin kenttä. Oikealla näkyy UDK:ssa mukana tulevia kenttiä ja muuta sisältöä. Ylhäällä ja vasemmalla näkyy monta työkaluriviä.



Kuva 7. Unreal Development Kitin muokkain.

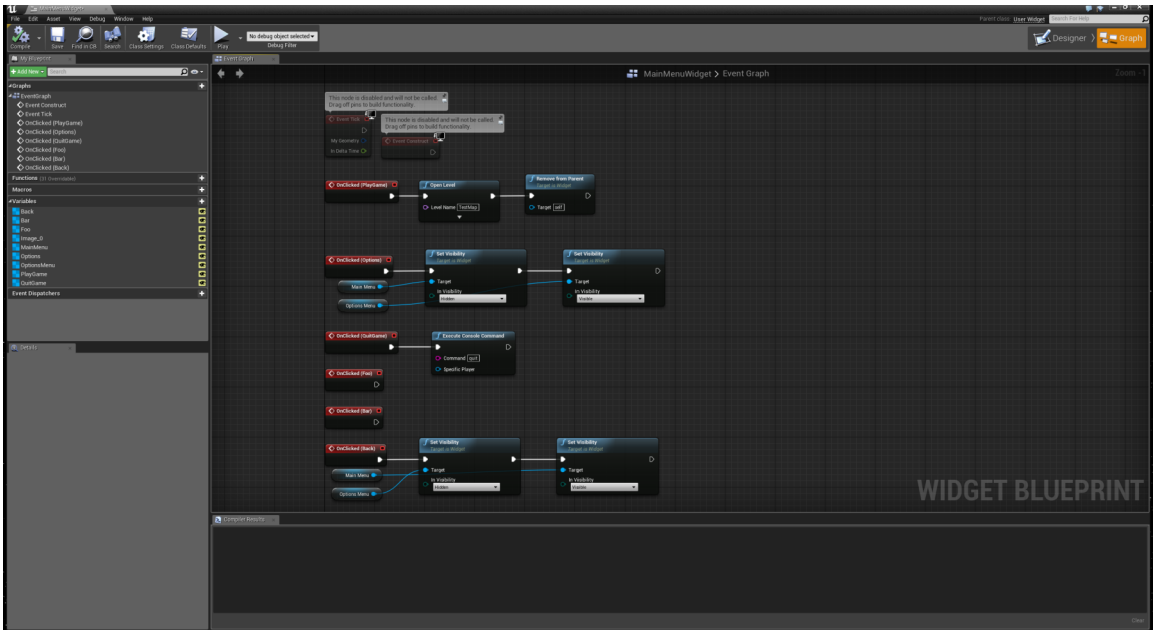
4.2 Ominaisuudet

Unreal Engine 4 tarjoaa ohjelmoimiseen tuen C++-ohjelmointikielelle. Ohjelmoijat voivat muokata Unreal Enginen omaa moottorikoodia ja kytkeä oman koodinsa suoraan moottorin ytimeen, mikä mahdollistaa laajemman muokattavuuden, yleiskäyttöisyyden ja tehokkuuden nostamisen. Ohjelmakoodia voi myös muuttaa pelin ollessa käynnissä, mikä mahdollistaa muutosten näkemisen suoraan ilman pelin uudelleenkäynnistämistä. [24]

Unreal Engine 4:n ohjelmoimiseen voi myös käyttää Unrealin omaa visuaalista ohjelmointijärjestelmää, Blueprintiä. Blueprintiä voi käyttää pelin ohjelmointiin koskematta Unreal Enginen tai pelin ohjelmakoodiin, mikä mahdollistaa yksinkertaisempien toimintojen ohjelmoimisen ilman ohjelmoijia. [24]

Kuvassa 8 on tämän tutkielman käyttöliittymätyökaluja vertailevan ohjelman käyttöliittymän Blueprint-näkymä. Keskellä näkyy Blueprint-järjestelmän kappaleet ja niiden yhteydet toisiinsa. Vasemmalla näkyy lista kaikista

muokattavana olevan Blueprintin komennoista ja muuttujista. Alhaalla näkyy tyhjä virheilmoituslaatikko. Oikealla ylhäällä on nappulat eri valikoiden välillä vaihtamiseen, kuvassa Blueprint-ohjelmointipuolen ja käyttöliittymän suunnittelun välillä.



Kuva 8. Unreal Engine 4 -pelimoottorin muokkaimen Blueprint-osio tämän opinnäytetyön valikkokokeiluohjelmasta.

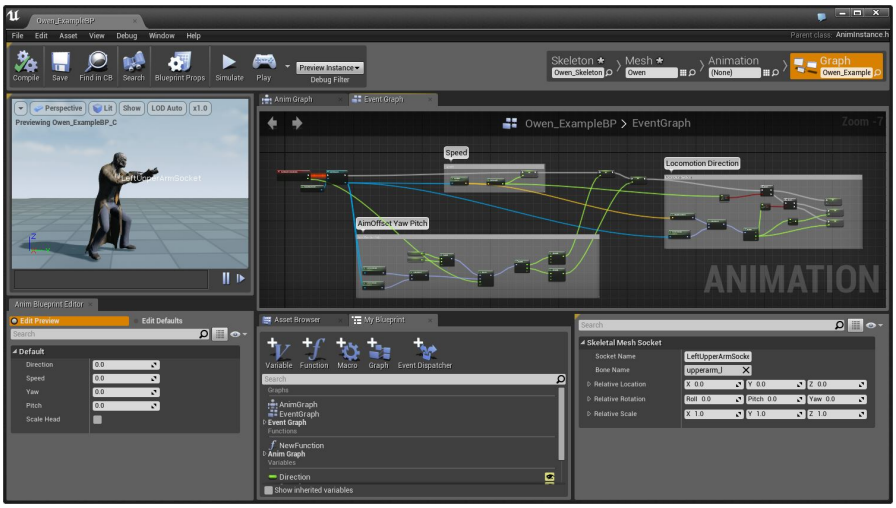
Unreal Engine 4:n kuvannin on Unreal Engine -moottoreille uusi DirectX 11 -kirjastoon pohjautuva kuvanninjärjestelmä. Kyseinen kuvanninjärjestelmä käyttää lykätty varjostus -tekniikkaa valaistuksen laskemiseen ja tarjoaa valmiit ratkaisut läpikuultaville ja valoa levittäville aineille. Kuvannin tarjoaa myös mahdollisuuden käyttää näytönohjaimella mallinnettuja hiukkasia perinteisten suorittimella mallinnettujen sijasta, minkä takia hiukkasia voi olla satoja kertoja enemmän samaan aikaan käytössä kuin perinteisellä järjestelmällä. Näytönohjaimella mallinnetut hiukkaset antavat myös mahdollisuuden käyttää vektorikenttiä, jotka vaikuttavat hiukkasten liikeratoihin. [25]

Unreal Engine 4:n kuvantimen jälkikäsittelytehosteet sisältävät ambient occlusion -tehosteen, joka perustuu Screen Space Ambient Occlusion -menetelmään. Kuvantimen valaistukseen voi myös käyttää ympäristön kuutiokartoitusta, mikä lisää kuutiokarttatekstuurin koko ympäristön valaistukseen ja mahdollistaa tehokkaan ja laadukkaan ympäristövalaistuksen. Kuvannin tarjoaa myös mahdollisuuden linssiheijastuksille, kuvan sävytykselle ja erilaisille

kukoistustehosteille, mallintaen joko silmissä, kameran filmissä tai linssissä tapahtuvaa ilmiötä. [25]

Animointiin Unreal Engine 4 tarjoaa Persona-työkalusarjan. Personalla voi muokata mallinnettujen kappaleiden luurankoja, luurankojen verkkoja ja vastakkeita sekä animaatio-Blueprintejä. Personalla voi esikatsella animaatiosarjoja ja muokata kohteita, sekä sekoittaa animaatioita. Personalla voi myös muokata mallinnetun kappaleen fyysisiä ominaisuuksia. [24]

Kuva 9 on Unreal Engine 4 -pelimoottorin ohjeistuksessa käytetty mallikuva [26]. Kuvassa näkyy Persona-työkalusarjan viisi osaa. Keskellä näkyy Blueprint-ohjelmointiosa animaatioiden yhdistämiseen. Vasemmalla näkyy esinäkymä animaatioista ja sen alapuolella näkyy Blueprintin muuttujat, joita voi muokata esinäkymän muokkaamiseen tai oletusarvojen muokkaamiseen. Keskellä alhaalla näkyy Blueprintin komennot ja muuttujat tai projektin tallennetut kappaleet. Oikealla alhaalla näkyy valittuna olevan istukan ominaisuudet.

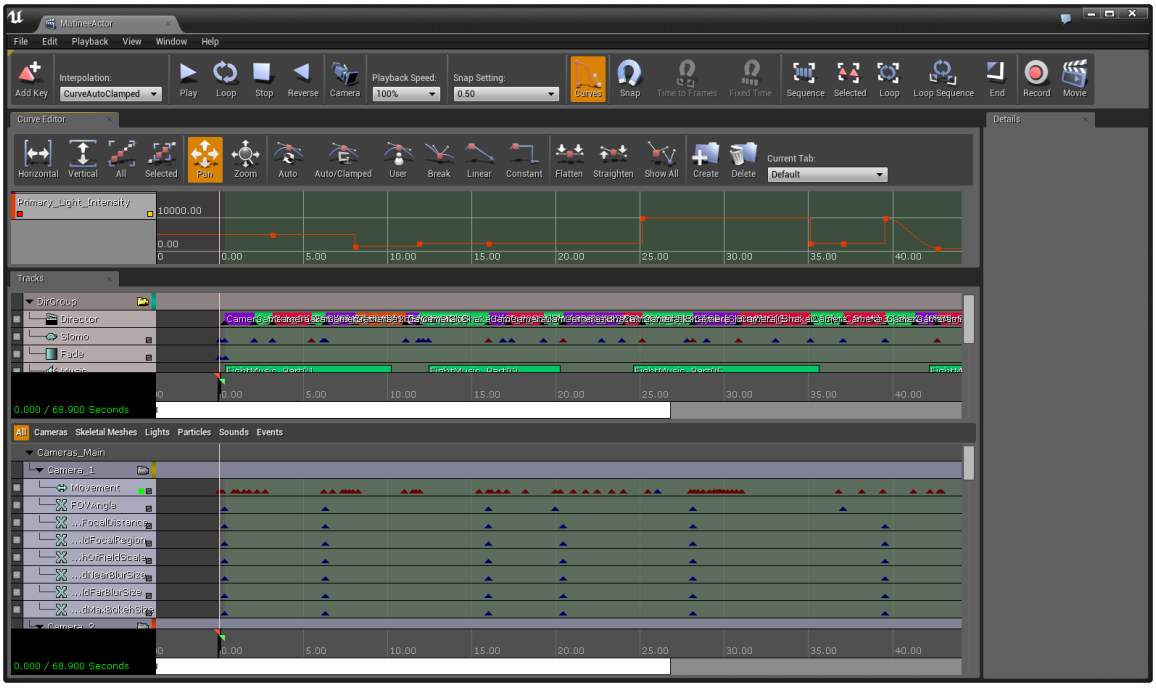


Kuva 9. Unreal Engine 4 -pelimoottorin Persona-työkalusarja. [26]

Matinee-elokuvatyökalusarja tarjoaa elokuvien, välivideoiden ja ennalta määrättyjen kohtausten tekemiseen tarvittavat työkalut. Matinee antaa kehittäjälle kyvyn määritellä ja animoida pelimaailman ominaisuudet pienimpään yksityiskohtaan saakka. Työkalujen käyttö on samantapaista kuin videoeditointiohjelmien käyttö. [24]

Kuva 10 on Unreal Engine 4 -pelimoottorin ohjeistuksessa käytetty mallikuva [27]. Kuvassa näkyy Matinee-elokuvatyökalusarjan valikko keskellä. Valikossa

näkyy kaikki elokuvaan liittyvät tilanteet ja tapahtumat yhteisessä aikajanassa. Valikko on eroteltu ylhäältä alas kurvien muokkaimeen, raitojen muokkaimeen ja kameran muokkaimeen. Oikealla näkyisi valittuna olevan kappaleen ominaisuudet, jos jotain olisi valittuna. Ylhäällä oleva työkalurivi sisältää huomattavasti oleellisempia valintoja ja komentoja kuin muissa valikoissa.



Kuva 10. Unreal Engine 4 -pelimoottorin Matinee-työkalusarja. [27]

Ohjelmointivirheiden poistoon ja pelintestaukseen Unreal Engine 4 tarjoaa Blueprintien suoratestauksen, mikä mahdollistaa arvojen tarkastamisen ja muuttamisen pelin aikana. Unreal Engine 4:ssä voi myös päivittää pelin ja välittömästi esikatsella pelin tilannetta ja pelin aikana kehittäjä voi helposti poistua pelihahmosta ja hallita pelin kameraa suoraan mahdollisen virheen tarkastamiseksi. [24]

4.3 Esimerkkejä

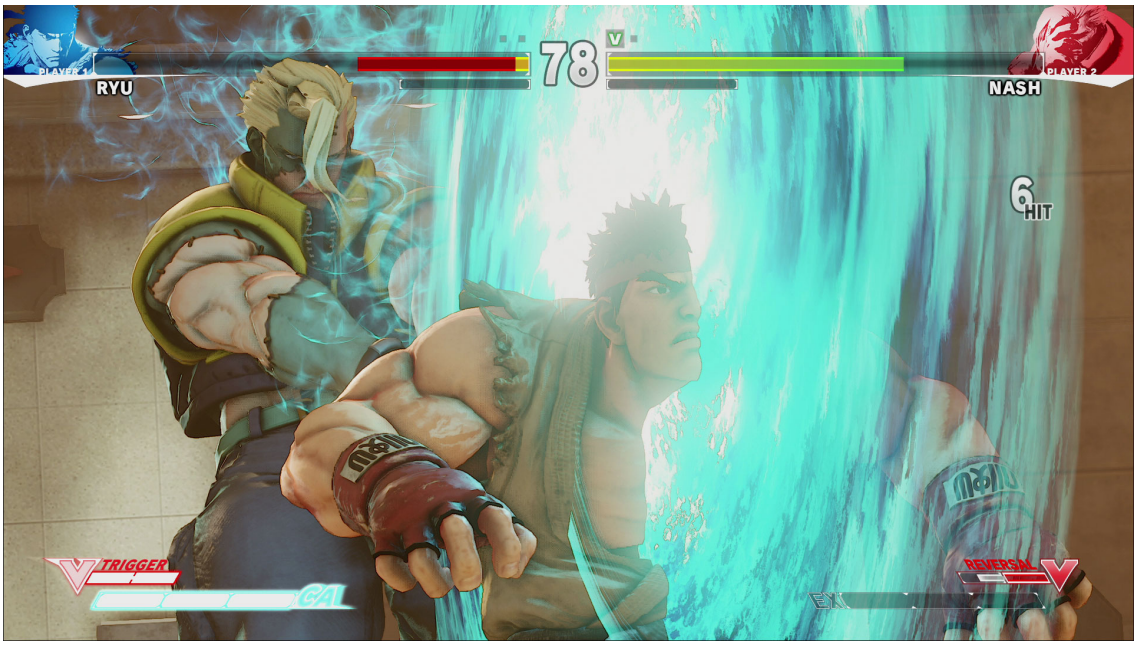
Seuraavaksi esitellään kaksi Unreal Engine 4 -pelimoottorilla kehitettyä peliä. Toisin kuin Unity 5:n esimerkit, kumpikin esimerkkipeli kehitettiin nimenomaan Unreal Engine 4:llä.

4.3.1 Street Fighter V

Street Fighter V on Unreal Engine 4 -pelimoottorilla kehitetty tappelupeli Playstation 4:lle ja PC:lle Capcom-peliyhtiöltä [28]. Street Fighter V julkaistiin 16. helmikuuta 2016 [29].

Street Fighter V on kaksintaistelupeli, jossa kaksi hahmoa taistelee toisiaan vastaan erinäisillä taistelulajeilla ja -tyyleillä. Kummankin pelaajan tarkoitus on iskeä vastapuolen hahmo tajuttomaksi tyhjentämällä heidän terveyspalkkinsa tyhjäksi. Pelissä on myös erikoisliikkeitä, joita voi käyttää erikoismittarin täytyttyä tarpeeksi. Peli on ohi pelimuodosta riippuen joko yhden ottelun jälkeen tai toisen puolen voitettua enemmistö parittomasta lukemasta otteluita. [29]

Kuva 11 on Street Fighter V -pelin markkinointiin käytetty kuvankaappaus. Kuvassa kaksi hahmoa, Ryu ja Nash, taistelevat toisiaan vastaan. Nash on vasemmalla takana oleva hahmo ja on lyömässä erikoisliikkeellä keskellä edessä olevaa Ryuta. Kuvan yläosassa näkyy ottelussa jäljellä oleva aika ja hahmojen terveysmittarit. Kuvan alaosassa on hahmoja vastaavilla puolilla heidän erikoismittarinsa.



Kuva 11. Kuvankaappaus Street Fighter V -pelistä. [30]

4.3.2 Battlefleet Gothic: Armada

Battlefleet Gothic: Armada (Battlefleet) on Unreal Engine 4 -pelimoottorilla kehitetty reaaliaikainen strategiapeli, jossa pelaaja ohjaa avaruustaisteluita Warhammer 40K -sarjan universumissa. Pelin kehitti Tindalos Interactive ja peli julkaistiin 21. huhtikuuta 2016 PC:lle. [31]

Battlefleet Gothic: Armada on strategiapeli, jossa pelaaja johtaa avaruuslaivastoa sekä taisteluissa että taisteluiden ulkopuolella. Taisteluissa pelaaja hallitsee laivastoaan ja yrittää tuhota vastapuolen laivaston mahdollisimman pienillä vahingoilla. Taisteluiden välillä pelaaja voi muokata laivastonsa miehistöä, aseistusta, erikoisosia ja muita ominaisuuksia parantaakseen laivaston suorituskykyä. [31]

Kuva 12 on Battlefleet Gothic: Armadan markkinoinnissa käytetty mediakuva, eikä ole suora kuvankaappaus pelistä. Kuvassa näkyy kahden eri laivaston alukset ampumassa toisiaan tykistöllä, mikä on tavallinen asia pelissä. Vasemmalla alhaalla näkyy Battlefleetin logo.

Kuva 12. Mediakuva Battlefleet Gothic: Armana -pelistä. [32]

5 TUTKIMUKSEN TOTEUTUS

Opinnäytetyön tavoitteena on selvittää pelimoottorien käytännön eroja tehokkuudessa ja käytettävyydessä pelimoottorien ominaisuuksien lisäksi. Riippuen kehitettävän pelin kohdealustasta, pelimoottorin käyttämien resurssien määrä voi muodostua pullonkaulaksi kehityksessä ja rajoittaa pelisuunnittelua. Koska pelimoottorien resurssien käyttöä ei voi tarkasti arvioida pelkästään ominaisuuksista, opinnäytetyössä on myös käytännön osio.

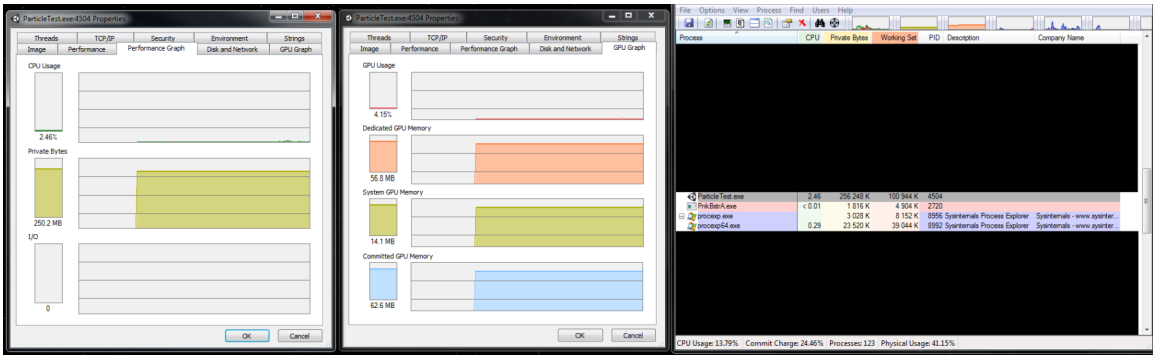
Mittausympäristönä käytettiin noin vuoden vanhaa tietokonetta, jossa on Windows 7 Home Premium -käyttöjärjestelmä, GTX 980 TI -mallinen näytönohjain ja Intel i5 4690K -mallinen suoritin ylikellotettuna 4,2 gigahertsiin.

Tutkimus toteutettiin tekemällä kummallakin pelimoottorilla tiettyyn osa-alueeseen keskittyvä kokeiluohjelma. Nämä osa-alueet ovat hiukkasten laajuus ja niiden tehokkuus, fysiikkamoottorin tehokkuus suurella määrällä fysiikkalaskuja vaativilla kappaleilla sekä käyttöliittymätyökalujen helppokäyttöisyys ja tehokkuus.

Tutkimuksessa käytettiin tehokkuuden mittaukseen tietokoneen resursseja valvovaa ohjelmaa Process Explorer ja kokeiluohjelmissa olevia kuvataajuusmittareita [33]. Unity 5 -pelimoottori ei sisällä suoraan omaa kuvataajuusmittaria, joten Unity 5:n kokeiluohjelmissa käytettiin erikseen ohjelmoitua kuvataajuusmittaria. Unreal Engine 4 taas tarjoaa suoraan sisäänrakennetun kuvataajuusmittarin, joten Unreal Engine 4:n kokeiluohjelmissa käytettiin pelimoottorin omaa kuvataajuusmittaria. Käytännössä kuvataajuusmittareilla ei ole eroja, käytettyä kirjasinta lukuun ottamatta.

Kuva 13 on Process Explorer tämän opinnäytetyön Unityn hiukkasmittausohjelmaan avattuna. Kuvassa on vasemmassa ikkunassa suorittimen käyttö ensimmäisenä, muistin käyttö toisena ja levymuistin lukeminen ja kirjoittaminen kolmantena. Keskimmäisessä ikkunassa on näytönohjaimen käyttö ensimmäisenä, näytönohjaimen oman muistin käyttö toisena, näytönohjaimen yhteisen muistin käyttö kolmantena ja näytönohjaimen

kokonaismuistinkäyttö neljäntenä. Oikea ikkuna on Process Explorerin pääikkuna eri ohjelmien tarkkailuun, mutta kyseistä ikkunaa ei käytetty kuin muiden ikkunoiden avaamiseen tässä opinnäytetyössä.



Kuva 13. Process Explorer avattuna tämän opinnäytetyön Unityn hiukkasmittausohjelmaan.

Hiukkasten tehokkuuden mittauksessa on tavoitteena selvittää pelimoottorien kyky piirtää hiukkasia ilman hidastuksia ja pelimoottorien hiukkasten piirtoon käytettyjen resurssien määrät. Mittauksessa mitattiin ohjelman piirtonopeus ohjelman omalla piirtonopeuden laskimella ja ohjelman käyttämät resurssit Process Explorerilla. Mittaus suoritettiin antamalla ohjelmalle käsky luoda ennalta määrätty määrä hiukkasia ruudulle. Hiukkasten luonnin jälkeen kokeiluohjelman käyttämät resurssit ja toimivuus luettiin ja kirjattiin muistiin.

Fysiikkamoottorin stressitestauksen tavoitteena on selvittää pelimoottoreilla laskettavien fysiikkalaskujen raja ennen hidastumista ja fysiikkalaskentaan käytettyjen resurssien määrä. Mittaus suoritettiin luomalla pelimaailmaan seinillä suljettu alue, jonne luotiin suuri määrä pallon muotoisia kappaleita. Nämä pallokappaleet tiputettiin suljettuun alueeseen ja pakotettiin törmäämään sekä muihin palloihin että alueen seiniin. Palloja luotiin aina tiettyyn rajaan saakka, minkä jälkeen kokeiluohjelman käyttämät tietokoneen resurssit ja kokeiluohjelman toimivuus luettiin ja kirjattiin muistiin. Mittauksessa erotettiin kappaleiden piirtämiseen kulutetut resurssit erottamalla mittaustulokset kameran suunnan mukaan.

Käyttöliittymätyökalujen helppokäyttöisyyden mittauksen tarkoituksena on selventää pelimoottorien käyttöliittymätyökalujen välisiä eroavaisuuksia ja arvioida käyttöliittymien kehittämisen keston eroa pelimoottorien välillä. Helppokäyttöisyys mitattiin tarvittavien rakenneosien ja tarvittavan ohjelmakoodin määrällä. Unreal Engine 4 -pelimoottorissa olevaa Blueprint-järjestelmää mitattiin

siinä vaadittavien palikoiden määrällä. Unity 5 -pelimoottorissa otettiin huomioon käyttöliittymätyökalujen oma menetelmä kutsua komentoja. Kokeiluohjelmien yksinkertaisten valikkojen luomisen ja toimivaksi toteamisen jälkeen rakenneosien lukumäärä ja tyyppi kirjattiin muistiin.

6 TUTKIMUKSEN TULOKSET

6.1 Hiukkaset

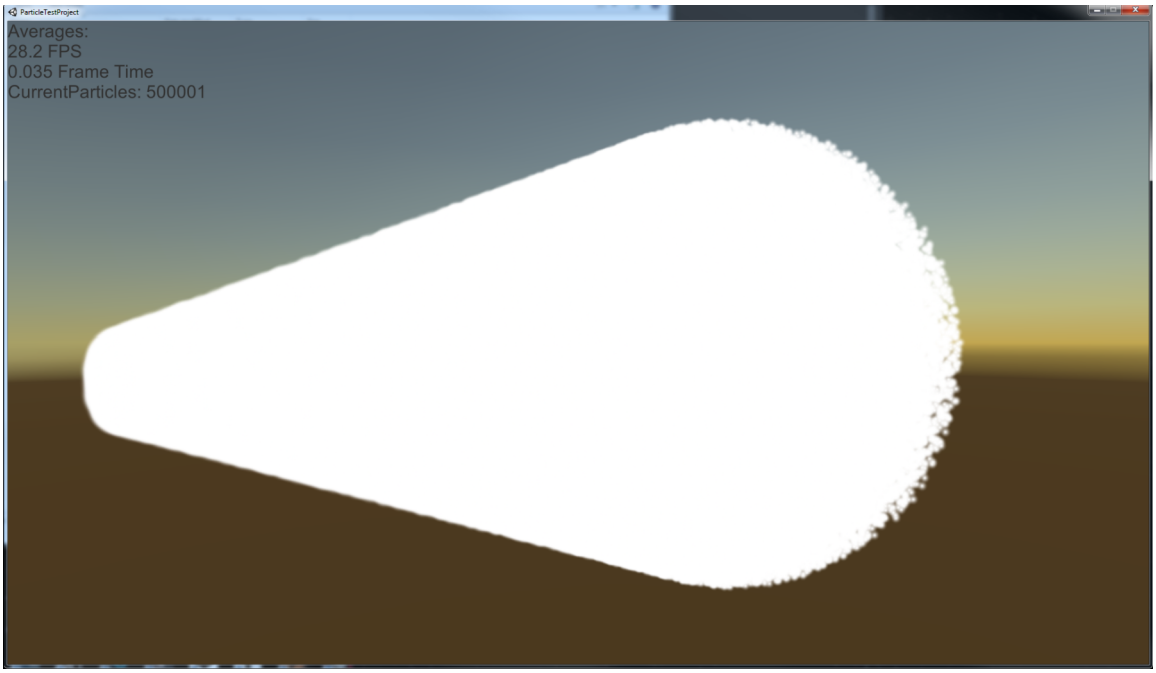
Tutkimuksessa oli tavoitteena selvittää pelimoottorien hiukkasten piirron kyky ilman hidastumista ja resurssien kulutus piirtoon. Tämän avulla selvitettiin, kumpi pelimoottori on tehokkaampi ja kumpi pystyy piirtämään enemmän hiukkasia ilman piirtonopeuden hidastumista.

Tutkimuksessa oli tärkeä erottaa suorittimella mallinnettavat hiukkaset ja näytönohjaimella mallinnettavat hiukkaset. Unreal Engine 4 -pelimoottorissa voi käyttää näytönohjaimella mallinnettavia hiukkasia, mikä lisää mahdollisten hiukkasten määrää huomattavasti isommaksi kuin suorittimella mallinnettaessa olisi, koska näytönohjaimella mallinnettaessa yleinen resurssikulutus on huomattavasti pienempi. Huonona puolena osa mahdollisista hiukkasten lisätehosteista on käytössä vain suorittimella mallinnettaessa. [34]

Unity 5 -pelimoottorissa ei ollut rajoitteita samanaikaisille hiukkasille. Hiukkasten määrän käytännön raja määräytyi kohdealustan pelille saatavien resurssien rajoista. Unity 5 ei myöskään sisältänyt näytönohjaimella mallinnettavia hiukkasia, mikä huomattavasti pienensi mahdollisten hiukkasten määrää Unreal Engine 4 -pelimoottoriin verrattuna. (Liite 1)

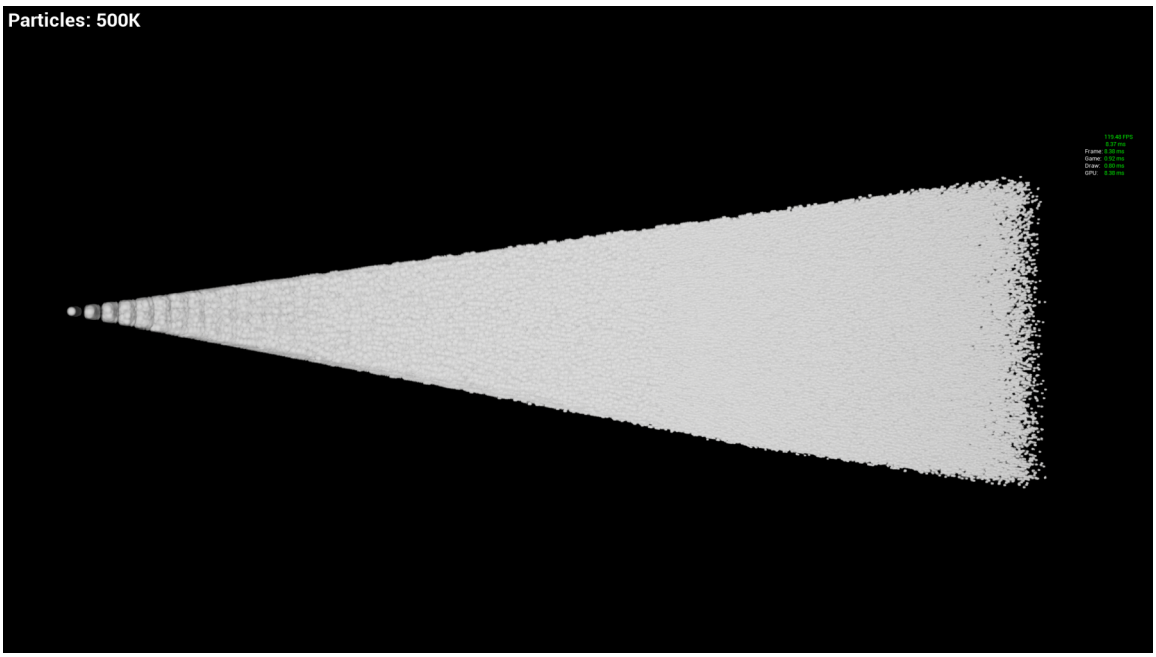
Unreal Engine 4 -pelimoottorissa oli suorittimella mallinnettavien samanaikaisten hiukkasten määrä rajoitettu tuhanteen ja näytönohjaimella mallinnettavien miljoonaan liiallisten resurssikulutusten estämiseksi. Näytönohjaimella mallinnettavien rajoite esti useamman hiukkasen tuottamisen vasta, kun tietokoneen resurssit alkavat hidastamaan kokeiluohjelmaa huomattavasti, joten rajoite ei ollut liian pieni. Sen sijaan suorittimella mallinnetut hiukkaset eivät hidastaneet kokeiluohjelmaa ollenkaan, vaikka tuhannen rajoite saavutettiin. Riippuen kehitettävän pelin suunnitellusta kohdealustasta, suorittimella mallinnettavien hiukkasten rajoite voi estää peliin suunniteltujen hiukkasten kehittämistä. (Liite 1)

Kuvissa 14 ja 15 näkyvät Unity 5- ja Unreal Engine 4 -pelimoottorien hiukkasmittausohjelmat 500 000 hiukkasessa ja vaikka hiukkasissa on värieroja, kumpikin hiukkaspilvi näyttää yhtä paksulta. Unreal Engine 4 oli kuitenkin vasta näyttämässä hidastumisen ensimerkkejä keskiarvolta yhdellä kuvalla sekunnissa pienemmällä piirtotaajuudella, kun taas Unity 5 oli hidastunut alle kolmenkymmenen kuvan piirtotaajuuteen alkuperäisestä 120 kuvasta sekunnissa.



Kuva 14. Unity 5 -pelimoottorin hiukkaskokeiluohjelma 500 000 hiukkasessa.

Unreal Engine 4 antoi mahdollisuuden huomattavasti suurempaan määrään hiukkasia ilman kokeiluohjelman hidastumista kuin Unity 5. Ensimmäiset hidastumiset kokeiluohjelmissa tapahtuivat Unreal Engine 4 -pelimoottorin kokeiluohjelmassa 500 000 hiukkasen määrällä, kun taas Unity 5:n kokeiluohjelmassa hidastumiset tapahtuivat 200 000 hiukkasen määrällä. Miljoonan hiukkasen samanaikaisessa mallinnuksessa Unreal Engine 4:n kokeiluohjelma oli vielä sulava yli 60 ruudunpäivityksellä sekunnissa, kun taas Unity 5:n kokeiluohjelma oli erittäin hidas hieman yli kymmenellä ruudunpäivityksellä sekunnissa. (Liite 1)



Kuva 15. Unreal Engine 4 -pelimoottorin hiukkaskokeiluohjelma 500 000 hiukkasessa.

Kuten taulukoista 1 ja 2 voi todeta, Unity 5 -pelimoottori käytti huomattavasti vähemmän muistia Unreal Engine 4 -pelimoottoriin verrattuna, 100 000 hiukkasen kohdalla videomuistia 25 % ja keskusmuistia 40 % Unreal Engine 4:ään verrattuna. Näytönohjaimen tehoa Unity 5 sen sijaan käytti 104 % ja suoritintehoa 220 % Unreal Engine 4:n käyttämästä. 100 000 hiukkasen jälkeen Unity 5 alkoi hidastumaan Unreal Engine 4:ään verrattuna, mikä teki suorasta vertailusta 100 000 hiukkasen mittaustulosten jälkeen harhaanjohtavan. (Liite 1)

On tosin huomautettava, että jos mittaus rajoitettaisiin pelkästään suorittimella mallinnettaviin hiukkasiin, Unreal Engine 4 -pelimoottorin tuhannen rajoite oli erittäin matala, vaikka tietokoneen resurssit olisivat riittäneet isompaankin määrään. Unity 5 -pelimoottori taas toimi ilman hidastuksia 100 000:een saakka ja ilman lisätehosterajoituksia. 20 000:kin hiukkasessa Unity 5 käytti vain osan Unreal Engine 4:n resursseista: näytönohjaimen tehoa 33 %, suoritintehoa 70 %, videomuistia 11 % ja keskusmuistia 28 %. (Liite 1)

Taulukko 1. Unity 5:n kokeiluohjelmassa mitattujen hiukkasten mittaustulokset.

Hiukkasten määrä	Näytön-ohjaimen käyttö	Suorittimen käyttö	Videomuistin käyttö	Keskus-muistin käyttö	Piirtotaajuus
20 000	14—15 %	7—8 %	63 Mt	246 Mt	120 k/s
50 000	26—28 %	13—14 %	103 Mt	294 Mt	120 k/s
100 000	50—52 %	21—23 %	151 Mt	343 Mt	120 k/s
200 000	76—79 %	34—38 %	247 Mt	448 Mt	86 k/s
500 000	64—67 %	35—39 %	311 Mt	532 Mt	28 k/s
750 000	51—54 %	33—38 %	311 Mt	546 Mt	15 k/s
1 000 000	51—54 %	41—45 %	311 Mt	565 Mt	11 k/s

Unity 5 -pelimoottori vaikutti myös toimivan paremmin vähemmän muistia omaavissa alustoissa, koska Unity 5:n kokeiluohjelma vei huomattavasti vähemmän muistia kuin Unreal Engine 4:n kokeiluohjelma. Unreal Engine 4 -pelimoottori ei kuitenkaan vaatinut kuin muutaman megatavun verran lisää muistia suurilla hiukkasmäärillä. (Liite 1)

Taulukko 2. Unreal Engine 4:n kokeiluohjelmassa mitattujen hiukkasten mittaustulokset.

Hiukkasten määrä	Näytön-ohjaimen käyttö	Suorittimen käyttö	Videomuistin käyttö	Keskus-muistin käyttö	Piirtotaajuus
20 000	43—44 %	10 %	593 Mt	868 Mt	120 k/s
50 000	45—47 %	10 %	593 Mt	868 Mt	120 k/s
100 000	49—50 %	10 %	593 Mt	868 Mt	120 k/s
200 000	56—58 %	10 %	595 Mt	870 Mt	120 k/s
500 000	83—85 %	10—11 %	597 Mt	875 Mt	119 k/s
750 000	89—91 %	9—10 %	600 Mt	880 Mt	101 k/s
1 000 000	90—92 %	8—9 %	600 Mt	886 Mt	84 k/s

Tutkimustuloksien perusteella Unreal Engine 4 -pelimoottori sopi paremmin hiukkasilla näyttäväksi tehtävään peliin kuin Unity 5 -pelimoottori. Unity 5 kulutti kuitenkin huomattavasti vähemmän muistia ja sopi paremmin alustoille, joissa muisti on päärajoitteena. (Liite 1)

6.2 Fysiikkamoottorin stressitutkimus

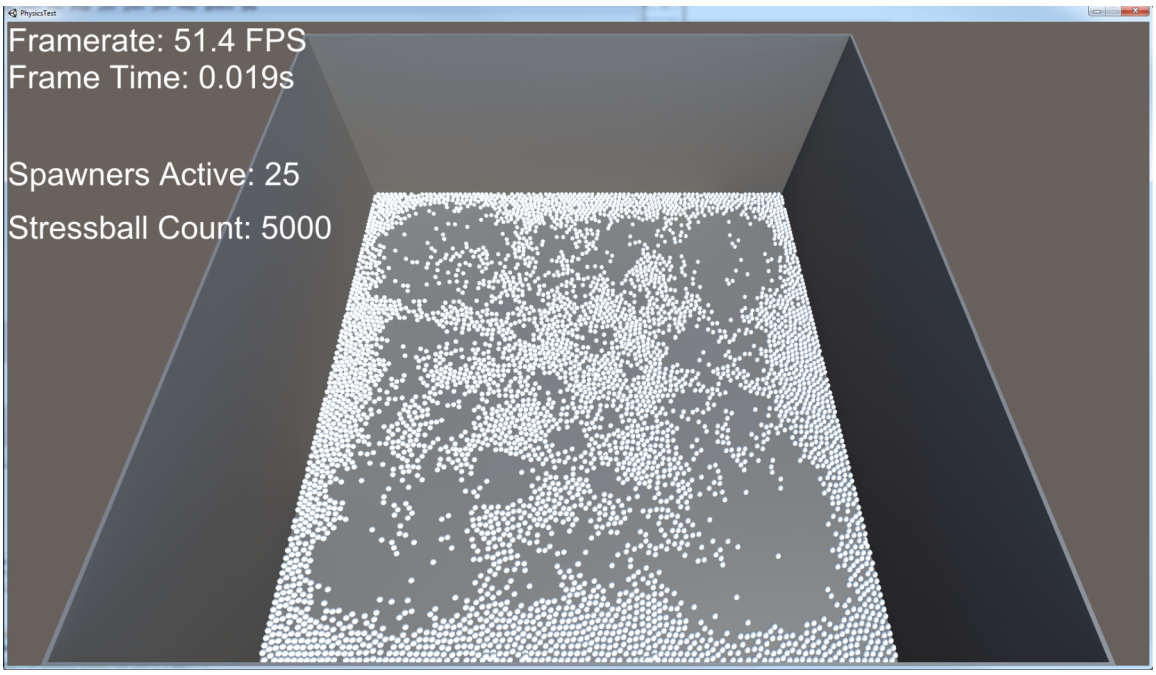
Tutkimuksen tavoitteena oli selvittää pelimoottoreilla laskettavien fysiikkalaskujen määrä ennen hidastumista ja fysiikkalaskuihin käytettyjen resurssien määrä. Näistä voitiin päätellä fysiikanlaskussa tehokkaampi pelimoottori.

Tutkimuksen tulokset oli jaettu kahteen osaan kokeiluohjelman kameran suunnan mukaan. Stressipalloihin suunnattu kamera osoitti suoraan kokeiluohjelman suljettuun alueeseen ja alueen sisällä oleviin fysiikkakappaleisiin, kun taas muualle suunnattu kamera osoitti tyhjään taustaan. Tyhjää taustaa osoittaneen kameran ei tarvinnut piirtää kappaleita, minkä takia ohjelman resurssien käyttö oli huomattavasti pienempi ja vastasi paremmin pelkän fysiikkamoottorin resurssien käyttöä. (Liite 2)

Unity 5 -pelimoottori pystyi käsittelemään mittaustuloksissa 2000 kappaletta ilman hidastuksia suljetun alueen ollessa kamerassa ja 5000 kappaletta suljetun alueen ollessa pois kamerasta. Keskusmuistin käyttö riippui kameran piirtämistä kappaleista ja fysiikkakappaleiden määrästä. Videomuistin käyttö riippui pelkästään piirretyistä kappaleista. (Liite 2)

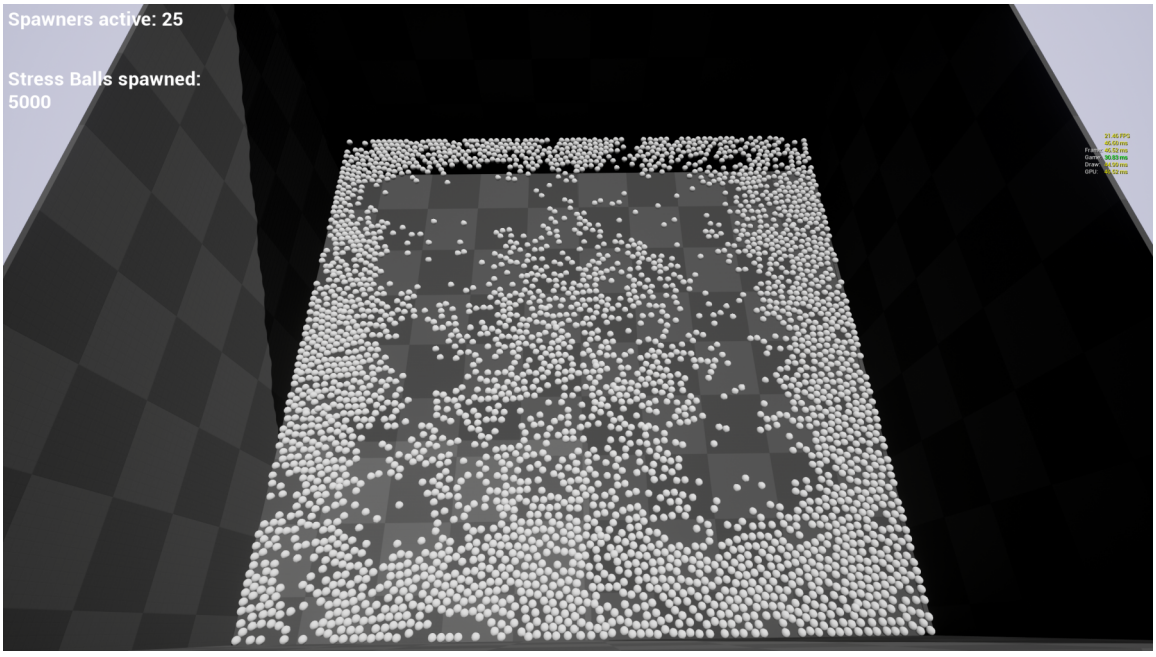
Unity 5 oli huomattavasti tehokkaampi kuin Unreal Engine 4. Unity 5 käytti vähemmän kaikkia mitattuja resursseja ja vähensi resurssien käyttöä kameran ulkopuolella olevien kappaleiden kohdalla. Vaikka kamera olisi osoittanut kappaleita, Unity 5 käytti kuitenkin vähemmän resursseja. Jos kamera ei osoittanut Unity 5 -pelimoottorissa kappaletta, kappale ei vienyt ollenkaan näytönohjaimen laskentatehoa. Unity 5 vaati myös enemmän fysiikkakappaleita hidastumaan kuin Unreal Engine 4 -pelimoottori. (Liite 2)

Kuvissa 16 ja 17 näkyvät Unity 5- ja Unreal Engine 4 -pelimoottorien fysiikkamittausohjelmat 5000 fysiikkakappaleessa. Kummassakin kuvassa näkyy laatikon pohjalla selvä kerros palloja samantapaisessa muodostelmassa. Unity 5 onnistui pitämään piirtotaajuuden yli viidessäkymmenessä kuvassa sekunnissa, mutta Unreal Engine 4 oli tippunut alle kolmenkymmenen kuvan sekunnissa piirtotaajuuteen.



Kuva 16. Unity 5 -pelimoottorin fysiikkamoottorin stressikokeiluohjelma 5000 fysiikkakappaleessa.

Unreal Engine 4 -pelimoottori ei vähentänyt resurssien käyttöä yhtä paljon kuin Unity 5 -pelimoottori kameran ollessa käännettynä poispäin kokeiluohjelman suljetusta alueesta. Unreal Engine 4 alkoi myös hidastumaan vähemmillä fysiikkakappaleilla kuin Unity 5 kummassakin mittaustilanteessa. Kumpikin pelimoottori pystyi kuitenkin käsittelemään yli 500 fysiikkakappaletta ilman hidastuksia huolimatta kameran suunnasta. (Liite 2)



Kuva 17. Unreal Engine 4 -pelimoottorin fysiikkamoottorin stressikokeiluohjelma 5000 fysiikkakappaleessa.

Unreal Engine 4 -pelimoottori pystyi käsittelemään mittaustuloksissa 525 kappaleeseen asti ilman hidastuksia kameran osoittaessa fysiikkakappaleisiin ja tuhanteen kappaleeseen kameran osoittaessa muualle. Muistinkäyttö ei riippunut kameran suunnasta ja nousi fysiikkakappaleiden mukana, mutta ei suorassa suhteessa. Unreal Engine 4 -pelimoottori käytti enemmän keskusmuistia kuin videomuistia. (Liite 2)

Resurssien käytössä näkyi selkeä ero 525 kappaleella ja oli viimeinen suoraan verrattava vertailukohta ennen Unreal Engine 4:n hidastumista, kuten taulukosta 3 voi huomata. Unity 5 -pelimoottori käytti paljon vähemmän resursseja kuin Unreal Engine 4: 14 % näytönohjaimen tehosta, 25 % suoritintehosta, 21 % videomuistista ja 55 % keskusmuistista. (Liite 2)

Taulukko 3. Unity 5:n ja Unreal Engine 4:n kokeiluohjelmien tulokset, kun kappaleet olivat kamerassa.

Unity 5 tulokset:					
Kappaleiden määrä	Näytön-ohjaimen käyttö	Suorittimen käyttö	Video-muistin käyttö	Keskus-muistin käyttö	Piirtotaajuus
25	4—5 %	8 %	124 Mt	512 Mt	120 k/s
525	9—10 %	10 %	130 Mt	517 Mt	120 k/s
1000	11—12 %	14 %	130 Mt	518 Mt	120 k/s
2000	16—17 %	24 %	126 Mt	525 Mt	120 k/s
5000	13—14 %	44 %	127 Mt	564 Mt	52 k/s
7500	3—4 %	46—47 %	129 Mt	598 Mt	11 k/s
Unreal Engine 4 tulokset:					
Kappaleiden määrä	Näytön-ohjaimen käyttö	Suorittimen käyttö	Video-muistin käyttö	Keskus-muistin käyttö	Piirtotaajuus
25	61—64 %	12 %	606 Mt	901 Mt	120 k/s
525	69—71 %	39—40 %	632 Mt	936 Mt	120 k/s
1000	69—71 %	43—44 %	638 Mt	961 Mt	80 k/s
2000	40—41 %	49—51 %	644 Mt	1009 Mt	50 k/s
5000	36—38 %	53—55 %	678 Mt	1,1 Gt	21 k/s
7500	25—27 %	55—57 %	706 Mt	1,4 Gt	16 k/s

Unity 5 -pelimoottorin ero Unreal Engine 4 -pelimoottoriin vain kasvoi, jos kamera ei osoittanut fysiikkakappaleita. Taulukosta 4 voi todeta, että 1000 fysiikkakappaleessa ero oli suuri: 6 % näytönohjaimen tehosta, 18 % suorittimen tehoista, 13 % videomuistista ja 49 % keskusmuistista.

Fysiikkakappaleiden määrän kasvaessa tasolle 7500 ja kameran osoittaessa fysiikkakappaleisiin Unreal Engine 4 -pelimoottori toimi nopeammin kuin Unity 5 -peli. Kyseinen tilanne oli kuitenkin käytännössä turha vertailukohta pelimoottorien välillä, sillä kumpikin pelimoottori oli käytännössä pelikelvoton tässä tilanteessa alle 30 kuvaa sekunnissa kuvataajuudella. (Liite 2)

Taulukko 4. Unity 5:n ja Unreal Engine 4:n kokeiluohjelmien tulokset, kun kappaleet eivät olleet kamerassa.

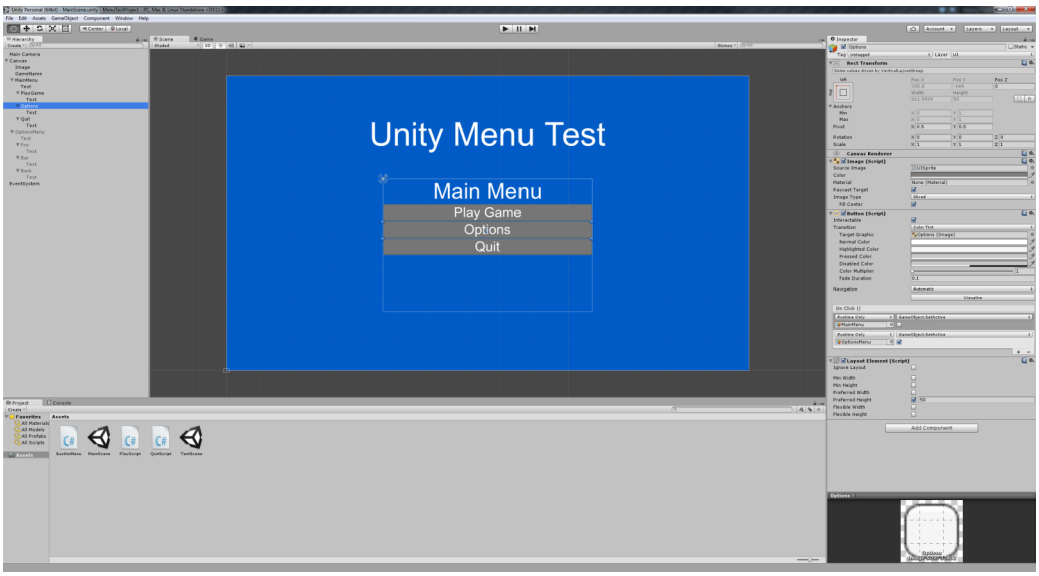
Unity 5 tulokset:					
Kappaleiden määrä	Näytön-ohjaimen käyttö	Suorittimen käyttö	Video-muistin käyttö	Keskus-muistin käyttö	Piirtotaajuus
25	3—4 %	3—4 %	79 Mt	464 Mt	120 k/s
525	3—4 %	5—6 %	80 Mt	469 Mt	120 k/s
1000	3—4 %	6—7 %	80 Mt	470 Mt	120 k/s
2000	3—4 %	8—9 %	80 Mt	480 Mt	120 k/s
5000	3—4 %	24—25 %	80 Mt	519 Mt	120 k/s
7500	3 %	40—41 %	80 Mt	556 Mt	90 k/s
Unreal Engine 4 tulokset:					
Kappaleiden määrä	Näytön-ohjaimen käyttö	Suorittimen käyttö	Video-muistin käyttö	Keskus-muistin käyttö	Piirtotaajuus
25	60—61 %	11 %	606 Mt	901 Mt	120 k/s
525	60—61 %	20—21 %	632 Mt	936 Mt	120 k/s
1000	60—61 %	36—37 %	638 Mt	961 Mt	120 k/s
2000	57—59 %	51—53 %	644 Mt	1004 Mt	92 k/s
5000	27—29 %	46—48 %	678 Mt	1,1 Gt	40 k/s
7500	15—17 %	44—46 %	706 Mt	1,4 Gt	23 k/s

Tutkimustulosten pohjalta vaikuttaisi, että Unity 5 -pelimoottori sopii paremmin enemmän fysiikkaa vaativiin peleihin kuin Unreal Engine 4 -pelimoottori. (Liite 2)

6.3 Valikon rakennus

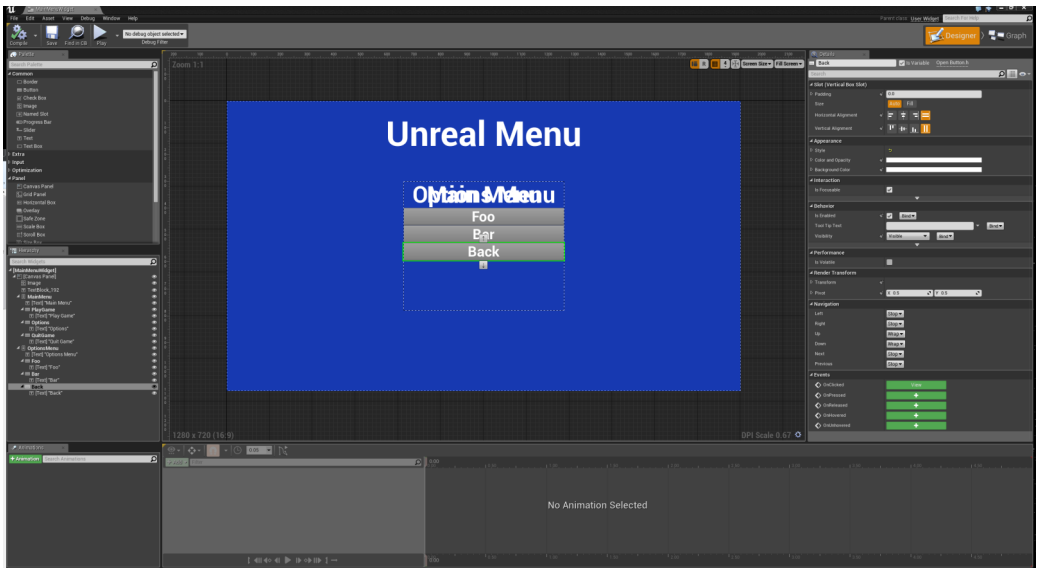
Tutkimuksen tavoitteena oli selventää eroavaisuuksia pelimoottorien käyttöliittymätyökalujen välillä ja selvittää mahdollisia eroja käyttöliittymien kehittämiseen.

Käyttöliittymän rakentaminen oli samanlaista kummallakin pelimoottorilla. Rakenneosien määrä oli tarkalleen sama ja kummankin kokeiluohjelman valikon rakenne kangasrakenneosan sisällä oli miltei täysin identtinen. Tämä voidaan huomioida kuvan 18 vasemmalla ja kuvan 19 vasemmalla alhaalla olevista kappalehierarkioista. Kumpikin pelimoottori myös käytti rakenneosiin pohjautuvia kappaleita, kuten kuvien 18 ja 19 oikealla olevista kappalerakenteista voidaan todeta. (Liite 3)



Kuva 18. Unity 5 -pelimoottorin muokkain valikko-ohjelmassa.

Muiden rakenneosien komentojen kutsumistapa oli suurin ero pelimoottorien käyttöliittymätyökaluissa. Unreal Engine 4 -pelimoottorin käyttöliittymätyökalut oli yhdistetty Blueprint-järjestelmään, joten käyttöliittymän vaatima ohjelmointi voitiin tehdä suoraan käyttöliittymätyökaluissa. Unity 5 -pelimoottorin työkalut voivat kutsua muiden rakenneosien komentoja suoraan, minkä avulla käyttöliittymän ohjelmointi voitiin yksinkertaisissa tapauksissa suorittaa pelkästään kutsumalla toisen rakenneosan komentoa tai ohjelmoimalla ohjelmointikielellä omia komentoja. (Liite 3)

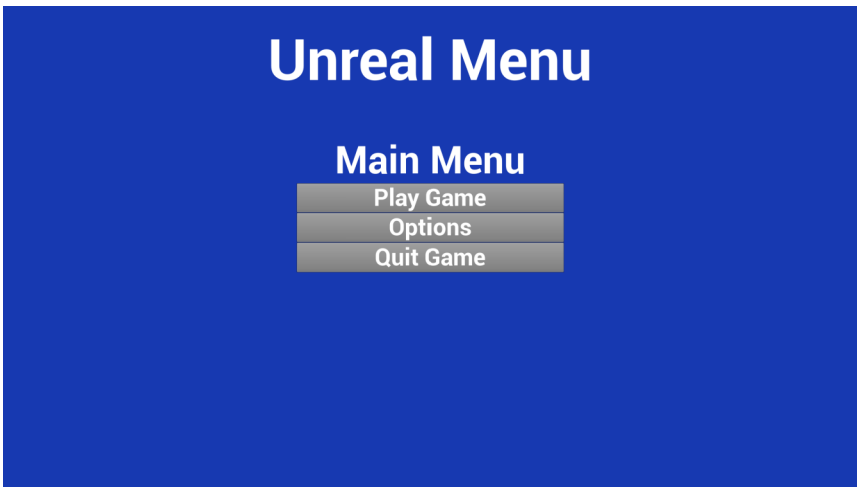


Kuva 19. Unreal Engine 4 -pelimoottorin muokkain valikko-ohjelmassa.

Lopputuloksissa ei ollut toiminnallisesti mitään eroa. Kummassakin kokeiluohjelmassa Play Game -näppäin avasi tyhjän kentän, josta näppäimistön escape-näppäin vei takaisin päävalikkoon. Options-näppäin vaihtoi valikkoa toiminnallisesti tyhjään valikkoon, jossa toinen näppäin vaihtoi päävalikon takaisin esiin. Quit-näppäin sulki pelin. Lopputulokset olivat näöltään myös lähellä toisiaan, kuten kuvista 20 ja 21 voidaan todeta. (Liite 3)

Kuva 20. Unity 5 -pelimoottorin kokeiluvalikko.

Tutkimuksen tuloksien perusteella pelimoottorien käyttöliittymätyökalujen välillä ei ollut pelimoottorien omia ohjelmointijärjestelmiä lukuun ottamatta eroavaisuuksia. Kumpikin pelimoottori pohjasi työkalunsa samanlaisiin periaatteisiin ja käyttöliittymien rakentaminen oli käytännössä samanlaista kummallakin pelimoottorilla. (Liite 3)



Kuva 21. Unreal Engine 4 -pelimoottorin kokeiluvalikko.

7 JOHTOPÄÄTÖKSET

Opinnäytetyössä pyrittiin selvittämään Unity 5 ja Unreal Engine 4 -pelimoottorien hyvät ja huonot puolet sekä käyttötarkoitukset. Pelimoottorien ominaisuudet ja työkalut selvitettiin pelimoottorien omista ohjeistuksista. Pelimoottorien tehokkuutta ja käytettävyyttä tutkittiin käytännön tutkimuksilla hiukkasten piirrossa, fysiikkamoottorin stressitestauksessa ja käyttöliittymätyökalujen käytettävyydessä.

Ominaisuuksien puolesta Unity 5 pystyy käsittelemään useampaa ohjelmointikieltä, mutta Unreal Engine 4 sisältää enemmän erikoistuneita työkaluja ja yleiskäyttöisen visuaalisen ohjelmointijärjestelmän. Näitä lukuun ottamatta kummankin pelimoottorin ominaisuudet ja työkalut ovat karkeasti samanveroisia.

Käytännön tutkimuksissa selvisi Unity 5:n käyttävän huomattavasti vähemmän resursseja yleisellä tasolla ja muistia jokaisessa mittauksessa. Unity 5:n fysiikkamoottori pystyy myös käsittelemään useampaa kappaletta hidastumatta. Sen sijaan Unreal Engine 4 pystyy piirtämään enemmän hiukkasia vähemmällä resurssien kulutuksella, muistia lukuun ottamatta. Käyttöliittymätyökalut paljastuivat olevan lähes samanlaiset ilman suuria eroja työkalujen toimintojen periaatteissa ja rakenteissa.

Unity 5 -pelimoottori on parempi valinta pelimoottoriksi tilanteissa, joissa pelin käyttämät resurssit täytyy pitää mahdollisimman pieninä. Unity 5 tukee myös paremmin raskaampia fysiikkaan pohjautuvia pelejä. Unity 5 -pelimoottorin kehittyneempi käyttö vaatii kuitenkin jonkin Unity 5:n tukeman ohjelmointikielen tuntemusta, sillä Unity 5:ssä ei ole yleisesti käytettävää visuaalista ohjelmointijärjestelmää.

Unreal Engine 4 -pelimoottori sopii paremmin voimakkaammille alustoille suunnattuihin peleihin, missä voi käyttää resursseja vähemmillä rajoituksilla. Varsinkin näyttävämpään grafiikkaan Unreal Engine 4 sopii paremmin suuremmalla määrällä hiukkasia ja lievästi kattavammilla graafisilla ominaisuuksilla. Unreal Engine 4:n Blueprint-ohjelmointijärjestelmän ja muiden

sisäänrakennettujen työkalusarjojen ansiosta Unreal Engine 4 sopii paremmin projekteihin, joissa ohjelmoijia on vähemmän käytössä.

LÄHTEET

[1] J. Gregory, *Game engine architecture*, Second edition. Boca Raton: CRC Press, 2014.
[2] ”What is a Game Engine? - GameCareerGuide.com”. [Verkossa]. Saatavissa: http://www.gamecareerguide.com/features/529/what_is_a_game_.php. [Viitattu: 02-loka-2016].
[3] J. Riccitiello, ”Unity 5 Launch – Unity Blog”, *Unity Technologies Blog*, 03-maalis-2015. [Verkossa]. Saatavissa: http://blogs.unity3d.com/2015/03/03/unity-5-launch/. [Viitattu: 29-loka-2015].
[4] Unity Technologies, ”Unity - Multiplatform - Publish your game to over 10 platforms”. [Verkossa]. Saatavissa: https://unity3d.com/unity/multiplatform. [Viitattu: 29-loka-2015].
[5] Unity Technologies, ”Unity - Manual: Editor Version Release Dates”. [Verkossa]. Saatavissa: http://docs.unity3d.com/460/Documentation/Manual/ReleaseDates.html. [Viitattu: 29-loka-2015].
[6] Unity Technologies, ”Unity - Release Archive”. [Verkossa]. Saatavissa: http://unity3d.com/unity/whats-new/archive. [Viitattu: 29-loka-2015].
[7] Unity Technologies, ”Unity 1.0 Core Features”, 09-maalis-2005. [Verkossa]. Saatavissa: https://web.archive.org/web/20050309172942/http://otee.dk/site/Tech/5.html. [Viitattu: 29-loka-2015].
[8] ”Unity - Unity 4.0”. [Verkossa]. Saatavissa: https://unity3d.com/unity/whats-new/unity-4.0. [Viitattu: 17-helmi-2016].
[9] ”Unity - What’s new in Unity 4.6”. [Verkossa]. Saatavissa: https://unity3d.com/unity/whats-new/unity-4.6. [Viitattu: 17-helmi-2016].
[10] ”Documentation, Unity scripting languages and you – Unity Blog”. [Verkossa]. Saatavissa: http://blogs.unity3d.com/2014/09/03/documentation-unity-scripting-languages-and-you/. [Viitattu: 11-huhti-2016].
[11]”Unity - Engine Features”. [Verkossa]. Saatavissa: https://unity3d.com/unity/engine-features. [Viitattu: 11-huhti-2016].
[12] ”Unity - Manual: Animation System Overview”. [Verkossa]. Saatavissa: https://docs.unity3d.com/Manual/AnimationOverview.html. [Viitattu: 02-syys-2016].
[13] ”Unity - Manual: 2D Physics Reference”. [Verkossa]. Saatavissa: http://docs.unity3d.com/Manual/Physics2DReference.html. [Viitattu: 19-touko-2016].
[14] ”Made with Unity - Kerbal Space Program”. [Verkossa]. Saatavissa: https://madewith.unity.com/games/kerbal-space-program. [Viitattu: 07-touko-2016].
[15] ”Kerbal Space Program Blog, KSP 0.18.4 is out!” [Verkossa]. Saatavissa: http://kerbaldevteam.tumblr.com/post/43110343228/ksp-0184-is-out. [Viitattu: 07-touko-2016].
[16] ”Kerbal Space Program Blog, Kerbal Space Program update 1.1 “Turbo Charged” is...” [Verkossa]. Saatavissa: http://kerbaldevteam.tumblr.com/post/143068450419/kerbal-space-program-update-11-turbo-charged-is. [Viitattu: 07-touko-2016].

[17] ”Kerbal Space Program on Steam”. [Verkossa]. Saatavissa: http://store.steampowered.com/app/220200/. [Viitattu: 02-syys-2016].
[18] ”Made with Unity - Blackbird’s Epic Road Trip”. [Verkossa]. Saatavissa: http://madewith.unity.com/stories/blackbirds-epic-road-trip. [Viitattu: 25-touko-2016].
[19] ”Homeworld: Deserts of Kharak on Steam”. [Verkossa]. Saatavissa: http://store.steampowered.com/app/281610/. [Viitattu: 02-syys-2016].
[20] ”Welcome to Unreal Engine 4”. [Verkossa]. Saatavissa: https://www.unrealengine.com/blog/welcome-to-unreal-engine-4. [Viitattu: 02-maalis-2016].
[21] M. Thomsen, ”History of the Unreal Engine”, *IGN*. [Verkossa]. Saatavissa: http://www.ign.com/articles/2010/02/23/history-of-the-unreal-engine. [Viitattu: 02-maalis-2016].
[22] ”America’s Army (PC)”, *IGN*. [Verkossa]. Saatavissa: http://www.ign.com/games/americas-army/pc-482289. [Viitattu: 07-maalis-2016].
[23] ”Gears of War - Xbox 360 - IGN”. [Verkossa]. Saatavissa: http://www.ign.com/games/gears-of-war/xbox-360-747891. [Viitattu: 23-maalis-2016].
[24] ”About Unreal Engine 4”. [Verkossa]. Saatavissa: https://www.unrealengine.com/unreal-engine-4. [Viitattu: 07-touko-2016].
[25] ”Rendering Overview | Unreal Engine”. [Verkossa]. Saatavissa: https://docs.unrealengine.com/latest/INT/Engine/Rendering/Overview/index.html. [Viitattu: 12-touko-2016].
[26] ”Persona Animation Editor Reference | Unreal Engine”. [Verkossa]. Saatavissa: https://docs.unrealengine.com/latest/INT/Engine/Animation/Persona/. [Viitattu: 02-syys-2016].
[27] ”Matinee | Unreal Engine”. [Verkossa]. Saatavissa: https://docs.unrealengine.com/latest/INT/Resources/Showcases/MatineeFightScene/. [Viitattu: 03-syys-2016].
[28] ”Street Fighter V | Rise Up!” [Verkossa]. Saatavissa: http://streetfighter.com/. [Viitattu: 14-touko-2016].
[29] ”Street Fighter V on Steam”. [Verkossa]. Saatavissa: http://store.steampowered.com/app/310950/. [Viitattu: 14-touko-2016].
[30] ”Media | Street Fighter V”. [Verkossa]. Saatavissa: http://streetfighter.com/media/. [Viitattu: 02-syys-2016].
[31] ”Battlefleet Gothic: Armada - Focus Home Interactive”. [Verkossa]. Saatavissa: http://focus-home.com/games/battlefleet-gothic-armada. [Viitattu: 25-touko-2016].
[32] ”Media - Battlefleet Gothic: Armada”. [Verkossa]. Saatavissa: http://www.battlefleetgothic-armada.com/en/media. [Viitattu: 02-syys-2016].
[33] ”Process Explorer”. [Verkossa]. Saatavissa: https://technet.microsoft.com/en-us/sysinternals/processexplorer.aspx. [Viitattu: 31-elo-2016].
[34] ”1.1 - CPU and GPU Sprite Particles Comparison | Unreal Engine”. [Verkossa]. Saatavissa: https://docs.unrealengine.com/latest/INT/Resources/ContentExamples/EffectsGallery/1_A/. [Viitattu: 01-heinä-2016].

LIITTEET

Liite 1: Hiukkasten mittaus

Testikoneessa:
Emolevy: ASRock Z97 Extreme4
Prosessori: Intel® Core™ i5-4690K Processor (6M Cache, up to 3.90 GHz)
(Ylikellotettu 4.20 GHz)
Näytönohjain: MSI GeForce® GTX 980 Ti GAMING 6G

Unity 5, suoritinmallinnetut hiukkaset:

Hiukkasten määrä	Näytön-ohjaimen käyttö	Suorittimen käyttö	Video-muistin käyttö	Keskus-muistin käyttö	Piirtotaajuus
20 000	14—15 %	7—8 %	63 Mt	246 Mt	120 k/s
50 000	26—28 %	13—14 %	103 Mt	294 Mt	120 k/s
100 000	50—52 %	21—23 %	151 Mt	343 Mt	120 k/s
200 000	76—79 %	34—38 %	247 Mt	448 Mt	86 k/s
500 000	64—67 %	35—39 %	311 Mt	532 Mt	28 k/s
750 000	51—54 %	33—38 %	311 Mt	546 Mt	15 k/s
1 000 000	51—54 %	41—45 %	311 Mt	565 Mt	11 k/s

Unreal Engine 4, näytönohjainmallinnetut hiukkaset:

Hiukkasten määrä	Näytön-ohjaimen käyttö	Suorittimen käyttö	Video-muistin käyttö	Keskus-muistin käyttö	Piirtotaajuus
20 000	43—44 %	10 %	593 Mt	868 Mt	120 k/s
50 000	45—47 %	10 %	593 Mt	868 Mt	120 k/s
100 000	49—50 %	10 %	593 Mt	868 Mt	120 k/s
200 000	56—58 %	10 %	595 Mt	870 Mt	120 k/s
500 000	83—85 %	10—11 %	597 Mt	875 Mt	119 k/s
750 000	89—91 %	9—10 %	600 Mt	880 Mt	101 k/s
1 000 000	90—92 %	8—9 %	600 Mt	886 Mt	84 k/s

Mt = Megatavu, muistin yksikkö
k/s = Kuvaa sekunnissa, piirtotaajuuden yksikkö

Liite 2: Fysiikkamoottorin stressitutkimuksen mittaus

Testikoneessa:
Emolevy: ASRock Z97 Extreme4
Prosessori: Intel® Core™ i5-4690K Processor (6M Cache, up to 3.90 GHz)
(Ylikellotettu 4.20 GHz)
Näytönohjain: MSI GeForce® GTX 980 Ti GAMING 6G

Unity 5: kappaleet kamerassa:

Kappaleiden määrä	Näytön-ohjaimen käyttö	Suorittimen käyttö	Video-muistin käyttö	Keskus-muistin käyttö	Piirtotaajuus
25	4—5 %	8 %	124 Mt	512 Mt	120 k/s
525	9—10 %	10 %	130 Mt	517 Mt	120 k/s
1000	11—12 %	14 %	130 Mt	518 Mt	120 k/s
2000	16—17 %	24 %	126 Mt	525 Mt	120 k/s
5000	13—14 %	44 %	127 Mt	564 Mt	52 k/s
7500	3—4 %	46—47 %	129 Mt	598 Mt	11 k/s

Unity 5: kappaleet kameran ulkopuolella:

Kappaleiden määrä	Näytön-ohjaimen käyttö	Suorittimen käyttö	Video-muistin käyttö	Keskus-muistin käyttö	Piirtotaajuus
25	3—4 %	3—4 %	79 Mt	464 Mt	120 k/s
525	3—4 %	5—6 %	80 Mt	469 Mt	120 k/s
1000	3—4 %	6—7 %	80 Mt	470 Mt	120 k/s
2000	3—4 %	8—9 %	80 Mt	480 Mt	120 k/s
5000	3—4 %	24—25 %	80 Mt	519 Mt	120 k/s
7500	3 %	40—41 %	80 Mt	556 Mt	90 k/s

Unreal Engine 4: kappaleet kamerassa:

Kappaleiden määrä	Näytönohjaimen käyttö	Suorittimen käyttö	Videomuistin käyttö	Keskusmuistin käyttö	Piirtotaajuus
25	61—64 %	12 %	606 Mt	901 Mt	120 k/s
525	69—71 %	39—40 %	632 Mt	936 Mt	120 k/s
1000	69—71 %	43—44 %	638 Mt	961 Mt	80 k/s
2000	40—41 %	49—51 %	644 Mt	1009 Mt	50 k/s
5000	36—38 %	53—55 %	678 Mt	1,1 Gt	21 k/s
7500	25—27 %	55—57 %	706 Mt	1,4 Gt	16 k/s

Unreal Engine 4: kappaleet kameran ulkopuolella:

Kappaleiden määrä	Näytönohjaimen käyttö	Suorittimen käyttö	Videomuistin käyttö	Keskusmuistin käyttö	Piirtotaajuus
25	60—61 %	11 %	606 Mt	901 Mt	120 k/s
525	60—61 %	20—21 %	632 Mt	936 Mt	120 k/s
1000	60—61 %	36—37 %	638 Mt	961 Mt	120 k/s
2000	57—59 %	51—53 %	644 Mt	1004 Mt	92 k/s
5000	27—29 %	46—48 %	678 Mt	1,1 Gt	40 k/s
7500	15—17 %	44—46 %	706 Mt	1,4 Gt	23 k/s

Mt = Megatavu, muistin yksikkö
Gt = Gigatavu, muistin yksikkö
k/s = Kuvaa sekunnissa, piirtotaajuuden yksikkö

Liite 3: Valikon rakennusosien mittaus

UE4:
Valikko koostuu 19 rakenneosasta.
Ohjelmointi muodostuu 22 ohjelmointikappaleesta Blueprint-järjestelmässä. Näiden lisäksi Blueprint-järjestelmässä on neljä kytkemätöntä ohjelmointikappaletta, jotka voi jättää huomioimatta.

U5:
Valikko koostuu 19 rakenneosasta.
Ohjelmointi koostuu kahdesta eri C#-ohjelmointikielitiedostosta, kummassakin yksi luokka. Yhteensä tiedostoissa on 22 riviä ohjelmointia ja kaksi ajossa toimivaa komentokutsua. Kolmas kutsu on huomautus Unityn omaan muokkaimeen ja voidaan jättää huomioimatta.
Unity 5:n valikoista on yhdistetty yhteensä seitsemän komentokutsua.