Niilo Lappalainen

Markkinoiden johtavat pelimoottorit

Tietotekniikan kandidaatintutkielma

5. kesäkuuta 2024

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Tekijä: Niilo Lappalainen

Yhteystiedot: niitapla@jyu.fi

Ohjaaja: Tytti Saksa

Työn nimi: Markkinoiden johtavat pelimoottorit

Title in English: Leading game engines in the market

Työ: Kandidaatintutkielma

Sivumäärä: 27+0

Tiivistelmä: Tässä kanditutkielmassa tarkastellaan markkinoiden käytetyimpiä pelimoottoreita. Ensin tutkielmassa selvitetään lähdekirjallisuuden avulla, mitä pelimoottorit ovat. Lopuksi perehdytään siihen, mitkä ovat tämän hetken käytetyimmät pelimoottorit pelinkehityksessä ja millaisia ominaisuuksia näissä pelimoottoreissa on.

Avainsanat: pelimoottorit, pelinkehitys, Unreal Engine, Unity

Abstract: This bachelor's thesis examines the most commonly used game engines on the market. Initially the thesis explores what game engines are, based on the literature. Finally it delves into the most popular game engines in game development and describes their features.

Keywords: Game engines, video game development, Unreal Engine, Unity

Sisällys

1 Johdanto

Tämän tutkielman tavoitteena on tarkastella markkinoiden suosituimpia pelimoottoreita. Tutkin kirjallisuuden avulla sitä, mitä pelimoottorit ovat ja mistä pelimoottorit koostuvat. Kanditutkielmassa tuodaan esiin suosituimpia pelimoottoreita ja niiden ominaisuuksia.

Keskeisiä tutkimuskysymyksiä, johon tämä tutkielma pyrkii etsimään vastauksia ovat: Mikä on pelimoottori? Millaisista komponenteista pelimoottori koostuu? Mitkä ovat tämän hetken käytetyimmät pelimoottorit pelialalla? Mikä on paras pelimoottori kehittäjälle?

Pelialan ja pelimoottoreiden tekninen kehitys kuluneina vuosikymmeninä on ollut huomattavaa. Pelimoottorit ovat olennainen osa pelinkehitystä ja niiden ymmärtäminen tarjoaa arvokasta tietoa tietotekniikasta ja pelialasta.

Pelimoottoreiden teknologian jatkuva kehittyminen luo haasteita ajantasaisen tutkimustiedon löytämiselle. Uudet innovaatiot, kuten tekoälyn hyödyntäminen, sekä yhä edistyksellisemmät grafiikka- ja fysiikkasimulaatiot päivittävät jatkuvasti käsitystämme siitä, mihin pelimoottorit kykenevät. Pelimoottoreiden nopean kehityksen vuoksi tutkimusartikkelit ja akateemiset julkaisut voivat nopeasti vanhentua uusimpien teknologioiden osalta. Tämä asettaa omat haasteensa ajan tasalla pysymiseen uusimmista trendeistä ja ominaisuuksista. Kehityksestä huolimatta pelimoottoreiden perusperiaate on pysynyt ajan saatossa suhteellisen muuttumattomana.

Tutkielmassa perehdytään ensin pelimoottoreihin yleisellä tasolla, jonka jälkeen tarkastellaan, mistä pelimoottorit koostuvat. Kun tiedämme tarkemmin mikä pelimoottori on, tarkastellaan tämän hetken käytetyimpiä pelimoottoreita pelinkehityksessä. Lopuksi vastataan tutkimuskysymyksiin yhteenvetona.

2 Pelimoottorit

Pelimoottori on laajennettavissa ja uudelleen käytettävissä oleva ohjelmisto, jota voidaan käyttää eri pelien kehittämiseen (Gregory 2018). Pelimoottorit ovat kehittyneet vastaamaan kasvavia vaatimuksia suorituskyvystä, realistisesta grafiikasta ja monipuolisista pelimekaniikoista. Pelimoottorit ovat olennainen osa nykyaikaista pelinkehitystä ja niiden tarkoituksena on helpottaa pelinkehittäjää, antamalla tälle valmiit työkalut, jottei pelinkehitystä tarvitse aloittaa aina täysin alusta. Usein modernit pelimoottorit sisältävät erilaisia komponentteja. Näihin komponentteihin kuuluvat muun muassa renderöintimoottori, fysiikkamoottori ja I/O-järjestelmä (Hu, Lu ja Yijin 2012). Kaikki komponentit eivät ole pelimoottorin toiminnan kannalta välttämättömiä, sillä kaikki pelit eivät tarvitse esimerkiksi fysiikanmallinnusta tai audiota toimiakseen.

Joskus pelin ja pelimoottorin raja on häilyvä. Jotkin pelimoottorit tekevät selvän eron, kun taas joidenkin pelien ja pelimoottoreiden välillä eroa on vaikeampi havainnoida. Olennainen tekijä nykyaikaisessa pelinkehityksessä on pelimoottorin uudelleenkäytettävyys. Esimerkiksi 1980-luvun alkupuolella julkaistu Pac-Man hyödynsi pelimoottoria, joka oli suunniteltu kyseiseen peliin, eikä sitä ollut helppo soveltaa muiden pelien kehittämiseen. Vertailun vuoksi taas Epic Gamesin suunnittelema ja valmistama Unreal Engine 4 on esimerkki pelimoottorista, jolla on luotu hyvin erilaisia pelejä. Täysin uudelleenkäytettävä pelimoottori, joka soveltuisi universaalisti mihin tahansa kehitettävissä olevaan peliin, on todennäköisesti skenaariona mahdoton (Gregory 2018). Huomioitavaa on, että Pac-Man julkaistiin yli 30 vuotta Unreal Engine 4:ää aikaisemmin. Peliteollisuudessa käytetty teknologia ja laitteisto ovat kehittyneet tässä ajassa huomattavasti. Vanhoissa tietokonepeleissä pelien ominaisuudet olivat hyvin rajallisia verrattuna nykyaikaisiin peleihin.

Myös valmiin pelimoottorin sijaan oman pelimoottorin valmistaminen on varteenotettava vaihtoehto. Monet pelitalot kehittävät pelejään varten omia pelimoottoreitaan eivätkä turvaudu markkinoilla valmiiksi saataviin pelimoottoreihin. Pelimoottorin kehittäminen mahdollistaa kehittäjälle teknisiä ratkaisuja, jotka eivät välttämättä saatavilla olevilla pelimoottoreilla olisi mahdollisia. Lisäksi pelimoottorin kehittäminen vapauttaa mahdollisilta lisenssimaksuilta, joita markkinoilta löytyvät pelimoottorit saattavat kehittäjiltä periä. Lisenssimaksut

ovatkin merkittävä tekijä pelimoottoreiden valinnassa. Vaikka monet markkinoilla saatavilla olevat pelimoottorit, kuten Unreal Engine tarjoavat ilmaisen pääsyn perustoimintoihin, ne vaativat kehittäjältä rojaltimaksujen maksamista, kun pelin tuotot ylittää tietyn rahallisen rajan.

Sekä omien että valmiiden pelimoottoreiden käytössä on omat vahvuutensa ja heikkoutensa. Oman pelimoottorin kehittäminen vaatii huomattavia resursseja ja aikaa. Tämän vuoksi monet pelikehittäjät kääntyvät usein valmiiden pelimoottoreiden, kuten Unityn ja Unreal Enginen puoleen. Kyseiset pelimoottorit tarjoavat ominaisuuksia ja työkaluja, jotka tekevät niistä suosittuja kehittäjien keskuudessa. Tässä tutkielmassa keskitytään erityisesti näihin valmiisiin, markkinoilta ladattaviin pelimoottoreihin.

3 Modernin pelimoottorin ydintoiminnot

Jotta saamme paremman käsityksen pelimoottoreiden toiminnasta, käsittelemme tässä luvussa pelimoottoreiden ydintoimintoja. Pelimoottoreiden ydintoimintoja voidaan jakaa useisiin erillisiin alijärjestelmiin, jotka vuorovaikuttavat keskenään (Gregory 2009). Tyypillisiä alijärjestelmiä pelimoottorissa ovat pääsilmukka, renderöintimoottori, fysiikkamallinnus, audio, tekoäly, verkkoyhteydet ja pelilaitekäsittely (Andrade 2015; Hu, Lu ja Yijin 2012). Kyseisten alijärjestelmien ymmärtäminen tarjoaa syvällisemmän näkemyksen pelimoottoreiden toiminnasta ja pelien teknisestä toteutuksesta.

Pääsilmukan rooli pelimoottorissa on merkittävä, sillä se ylläpitää pelin sisäistä logiikkaa ja varmistaa, että kaikki alijärjestelmät toimivat johdonmukaisesti yhteen (Gregory 2009). Pääsilmukkaa kuvaillaan myös pelimoottorin sisäiseksi rutiiniksi, joka päivittää pelitapahtumat uudelleen jokaisella kuvaruudunpäivityksellä (Andrade 2015). Alijärjestelmien tehokas vuorovaikutus on ratkaisevassa osassa sujuvan pelikokemuksen kannalta. Seuraavissa aliluvuissa syvennymme tarkemmin joihinkin pelimoottoreissa yleisesti esiintyviin alijärjestelmiin.

3.1 Renderöintimoottori

Renderöintimoottorin keskeinen tehtävä on vastata tapahtumien visualisoinnista näyttölaitteelle. Renderöintimoottori on toiminnaltaan yleensä kaikkein monimutkaisin alijärjestelmä (Vohera ym. 2021). Se vaatii jatkuvaa kehittämistä vastaamaan nykyaikaisten pelien yhä kasvaviin graafisiin vaatimuksiin.

Renderöintimoottori pyrkii tuottamaan näyttölaitteelle yksittäisiä kuvia riittävällä nopeudella, jotta pelaaja kokee liikkeen luonnollisena (Gregory 2009). Tätä kuvien päivitysnopeutta kutsutaan kuvataajuudeksi. Renderöintimoottori tuottaa kuvia riittävällä nopeudella, tyypillisesti vähintään 30 tai 60 kuvaa sekunnissa, jotta pelaaja kokee liikkeen luonnollisena. Optimaalisen sujuvuuden varmistamiseksi peleissä voidaan tavoitella korkeampaakin päivitysnopeutta, sillä ihmissilmä kykenee erottamaan huomattavasti suurempiakin kuvataajuuksia (Khoei ym. 2018). Monet suositut pelimoottorit kuten Unity ja Unreal Engine, tukevat huomattavasti korkeampiakin kuvataajuuksia kuin yllämainittu standardi (Unity 2024e; Epic Ga-

mes). Riittävän korkean kuvataajuuden lisäksi myös kuvataajuuden tasainen päivitysnopeus on avainasemassa, jotta liike vaikuttaisi katsojalle sulavalta.

Nykyaikaiset pelimoottorit tukevat kaksiulotteista, että kolmiulotteista grafiikkaa. Grafiikan renderöintiin kuuluu Gregoryn mukaan seuraavia vaiheita (Gregory 2009):

- Virtuaalisen 3D-maailman mallintaminen matemaattisesti.
- Virtuaalikameran sijoittaminen ja suuntaaminen halutun näkymän saavuttamiseksi. Tyypillisesti kamera mallinnetaan idealisoiduksi polttopisteeksi, jossa kuvantamispinta leijuu pienen matkan sen edessä.
- Valonlähteiden määrittely. Valonlähteet tarjoavat valonsäteitä, jotka ovat vuorovaikutuksessa ympäristön esineiden kanssa, heijastuvat niistä ja lopulta löytävät tiensä virtuaalikameran kuvantunnistuspinnalle.
- Pintojen visuaalisten ominaisuuksien kuvantaminen. Ominaisuudet määrittävät, kuinka valon tulee olla vuorovaikutuksessa kunkin pinnan kanssa.

Renderöintitekniikat ovat kehittyneet mahdollistaen entistä realistisemmat ja interaktiivisemmat graafiset esitykset. Useita eri tekniikoita voidaan käyttää edellä kuvattujen renderöinnin perusvaiheiden suorittamiseen. Renderöintimoottoreita hyödynnetään laajasti paitsi pelimoottoreissa myös monenlaisissa muissa reaaliaikaisissa ja interaktiivisissa grafiikkasovelluksissa. Renderöintimoottoreita käytetään pelien lisäksi esimerkiksi tieteellisen datan visualisointijärjestelmissä, tietokoneavusteisessa suunnittelussa (CAD) ja erilaisissa simulaatioissa, kuten lentosimulaattoreissa (Bao ja Hua 2011).

3.2 Fysiikkamoottori ja törmäystunnistus

Nykyään suurimmassa osassa pelimoottoreita on jonkinlainen fyysinen simulointikyky (Gregory 2009). Fysiikkamoottori pyrkii simuloimaan peliobjektien ja pelimaailman mekaanista vuorovaikutusta. Fysiikan mallinnuksessa voidaan pyrkiä realismiin fysiikkamoottorin jäljitellessä oikean maailman fysiikanlakeja, kuten kitkan ja painovoiman toimintaa. Jotkin pelit, kuten lentosimulaattorit, saattavat pyrkiä hyvinkin tarkkaan fysiikan mallintamiseen realistisen pelikokemuksen tarjoamiseksi (Gregory 2009). Toisaalta peleissä voidaan myös leikitellä epärealistisemmalla fysiikan toiminnalla, jotta saadaan vapauksia pelin toimintaan. Grego-

ry listaa kirjassaan seuraavanlaisia esimerkkejä fysiikkamoottorin mahdollistamista asioista videopeleissä (Gregory 2009):

- Törmäysten havaitsemien ja käsittely.
- Painovoiman ja muiden voimien simuloiminen.
- Jousi-massajärjestelmien dynamiikka.
- Rakennusten tuhoutuminen.
- Kiinteiden esineiden poimiminen.
- Monimutkaisten koneiden kuten nostureiden toiminnan mallintamien.
- Ansojen kuten kivivyöryjen toiminta.
- Ajoneuvojen realistinen toiminta.
- Riippuvan rekvisiitan kuten kannujen, kaulakorujen, miekkojen ja vaatteiden liikkeet.
- Kangassimulaatiot.
- Veden pinnan simulaatio ja objektien kelluvuus.

Fysiikkamoottorin lisäksi useat pelimoottorit sisältävät törmäystunnistusjärjestelmän, joka on usein integroitu fysiikkamoottoriin. Törmäystunnistusjärjestelmän pääasiallinen tarkoitus on selvittää, ovatko pelimaailman objektit tulleet kosketuksiin toistensa kanssa (Gregory 2009). Törmäystunnistusjärjestelmä toimii hyödyntäen geometrisia malleja ja koordinaatteja tunnistaakseen, milloin kaksi tai useampi objekti osuu toisiinsa pelimaailmassa (Hu, Lu ja Yijin 2012).

Törmäystunnistus on olennainen osa etenkin niitä pelejä, joissa objektien liike ja keskinäinen vuorovaikutus ovat pelin kannalta merkityksellisiä. Törmäystunnistus on osana jokaista peliä, jossa tapahtuu liikettä joko kaksi- tai kolmiulotteisessa maailmassa. Gregory antaa esimerkkeinä törmäystunnistusjärjestelmän toiminnasta tilanteen, jossa pelihahmo kulkee läpi leijuvan terveyspakkauksen. Tässä tapauksessa törmäyksentunnistin havaitsee pelaajan ja palkitsee tämän elämäpisteillä. Toisena esimerkkinä hän mainitsee tilanteen, jossa ohjus osuu kohteeseensa ja aiheuttaa osuessaan räjähdyksen (Gregory 2009).

Monet pelimoottorit, kuten Unity, tarjoavat sisäänrakennetun fysiikkamoottorin. Markkinoilla on myös yksittäisiä fysiikkamoottoreita, kuten Havok, PhysX ja Bullet (Gregory 2009). Esimerkiksi Unity tarjoaa integraatiomahdollisuuksia kolmansien osapuolien fysiikkamoot-

toreille, joiden hyödyntäminen voi tarjota kehittäjille erikoistuneempia fysiikkasimulaatioita ja parannettua suorituskykyä projekteihinsa (Unity 2024e).

3.3 Audio

Äänimaailman merkitys peleissä korostuu nykyaikaisessa pelisuunnittelussa, sillä se luo olennaisen osan pelikokemuksesta. Taustamusiikin lisäksi peleiltä odotetaan monipuolisia ja realistisia ääniefektejä, kuten askelten kopinaa (Vohera ym. 2021). On myös oletusarvoista, että ääni kantautuu realistisesti etäisyyden ja suunnan mukaisesti. Myös ympäristöllä on merkitystä, sillä esimerkiksi luolassa äänen tulisi kaikua. Monet pelimoottorit pystyvät käsittelemään erilaisia ääniformaatteja. Unity tukee ääniformaatteja, kuten AIFF, WAV, MP3 ja Ogg (Unity 2024b).

3.4 Tekoäly

Tekoälyä kohtaan on kasvavaa kiinnostusta ja sitä käytetään laajasti eri yhteyksissä. Alijärjestelmän ymmärryksen saavuttamiseksi voi olla tarpeen selventää, mitä termi ”tekoäly” tarkalleen ottaen tarkoittaa.

Alan Turing on luonut Turingin testin, jossa kone arvioidaan älykkääksi, mikäli se on keskustelussa niin vakuuttava, ettei ihminen voi erottaa keskusteleeko ihmisen vai koneen kanssa. Nykyään termi ”tekoäly” yleensä viittaa koneiden kykyyn kommunikoida, tehdä päätöksiä ja toimia itsenäisesti sekä tutuissa että uusissa tilanteissa ihmisen kaltaisesti. Joskus termiä ”tekoäly” käytetään myös synonyyminä termille ”koneoppiminen”. Koneoppiminen puolestaan tarkoittaa algoritmeja ja tilastollisia malleja, jotka oppivat tunnistamaan ja päättelemään malleja harjoitusdatan perusteella (Du-Harpur ym. 2020).

Tekoälyllä on keskeinen rooli monissa peleissä sujuvan pelikokemuksen luomisessa. Perinteisesti tekoäly ilmenee peleissä ei-pelaaja-hahmojen (NPC - Non-Playable Character) kautta, joita tekoäly ohjaa. Ohjaus ilmenee esimerkiksi hahmojen liikkeistä ja vuorovaikutuksesta pelimaailmassa. Yksinkertaisimmillaan tekoälyn ohjaama hahmo on vuorovaikutuksessa pelaajan kanssa vain hyvin rajoittuneesti ja vuorovaikutus perustuu vain yksinkertaiseen

logiikkaan. Kuitenkin uskottavan ei-pelaaja-hahmon tulisi omata tunteita ja persoonallisuuden piirteitä. Hahmon tulisi myös kommunikoida sulavasti pelaajan kanssa (Bicalho, Feijó ja Baffa 2020). Tekoälyteknologian kehittyessä sitä hyödynnetään yhä monipuolisemmin ja edistyneemmin. Esimerkiksi tekoälyteknologiaa on alettu hyödyntäminen korvaamaan perinteisiä ääninäyttelijöitä (Pelaaja.fi 2023).

Pelimoottorit tarjoavat entistä tehokkaampia tekoälyä hyödyntäviä kehitystyökaluja. Tämä kehitys on avannut pelinkehittäjille uusia mahdollisuuksia luoda yhä monimutkaisempia ja immersiivisempia pelikokemuksia. Esimerkiksi Unity-pelimoottori sisältää tekoälyominaisuuksia, kuten polunetsinnän ja Unity ML-Agents-laajennuksen.

Pathfinding eli ”polunetsintä” on Unityn NavMesh-työkalua ja polunetsintäalgoritmeja hyödyntävä ominaisuus. Se mahdollistaa NPC:iden älykkään navigoinnin peliympäristössä. Ominaisuus lisää hahmojen liikkumisen ja toiminnan uskottavuutta (Unity 2024c).

Unity ML-Agents on avoimen lähdekoodin projekti ja Unityn laajennus. Laajennus mahdollistaa tekoälyn ja koneoppimisen algoritmien kehittämisen ja testaamisen Unity-ympäristössä. Se on suunniteltu erityisesti pelikehittäjille ja tutkijoille, jotka haluavat kokeilla tekoälyn käyttöä peleissä tai simulaatioissa. ML-Agents tarjoaa alustan, jossa voidaan kouluttaa tekoälyagentteja monenlaisiin tehtäviin käyttäen vahvistusoppimista, imitaatio-oppimista tai muita koneoppimisen menetelmiä (Unity 2024g).

Epic Games mainitsee myös Unreal Enginelle useita tekoälyominaisuuksia, kuten Navigation Mesh ja Behavior Trees. Navigation Mesh auttaa tekoälyohjattuja hahmoja liikkumaan esteiden ympäri ja valitsemaan optimaalisen kulkureitin (Epic Games 2024c). Behavior Trees auttaa luomaan monimutkaisia, päätöksentekoon perustuvia tekoälyjärjestelmiä pelihahmoille ja muille toimijoille. Ominaisuus tarjoaa visuaalisen ja modulaarisen tavan suunnitella tekoälyn käyttäytymistä hierarkkisen puurakenteen avulla. Puurakenne koostuu erilaisista solmuista, kuten tehtävistä, päätöksistä, sekvensseistä ja valitsimista. Puurakenteen suunnittelu tapahtuu Unreal Engine Editorissa. Visuaalinen ja modulaarinen muokkaaminen tarkoittaa, että se mahdollistaa kehittäjille mahdollisuuden luoda ja muokata tekoälyn käyttäytymistä ilman, että kehittäjän tarvitsee kirjoittaa tai muuttaa koodia (Epic Games 2024a).

Kyseiset työkalut parantavat ei-pelaaja-hahmojen käyttäytymistä ja mahdollistavat kehittäjien luoda entistä vuorovaikutteisempia pelimaailmoja.

3.5 Verkkoyhteydet

Useat pelimoottorit sisältävät ominaisuuksia, jotka mahdollistavat verkkoyhteyttä hyödyntävien pelien kehityksen. Verkkoyhteys mahdollistaa pelaajien välisen kommunikaation, pelitilan synkronoinnin eri laitteiden välillä ja muita verkkopelien kannalta kriittisiä toimintoja. Verkkoyhteys on ratkaisevassa asemassa moninpelitoimintojen toteuttamisessa. Moninpelejä on erilaisia kuten yhden ruudun moninpeli, jaetun ruudun moninpeli, verkkomoninpeli ja massiivimoninpeli (Vohera ym. 2021).

Massiivi moninpeleissä käytetään yleisesti palvelinpohjaisia arkkitehtuureja, jotka pystyvät käsittelemään tuhansia samanaikaisia käyttäjiä vuorovaikutteisessa pelimaailmassa (Tsipis, Komianos ja Oikonomou 2019). Verkkoyhteydet tuovat haasteita kehittäjille, sillä turvallisuus ja luotettavuus ovat huolenaiheita verkkopeleissä. Keskeistä ovat muun muassa huijausten torjunta ja käyttäjätietojen, kuten kirjautumis- ja maksutietojen suojaaminen. Palvelinten tulee pystyä myös skaalautumaan suuriin määriin samanaikaisia käyttäjiä. Verkkopelit vaativat yleensä myös ylläpitoa ja päivitystä, joidenka tulisi toimia sujuvasti.

3.6 Pelilaitekäsittely

Pelimoottorit tarjoavat kehittäjille tehokkaita työkaluja ja rajapintoja, jotka helpottavat erilaisten laitteiden, kuten tietokoneen näppäimistön, hiiren, peliohjaimen ja puhelimen kosketusnäytön, syötteiden käsittelyä ja integrointia pelin logiikkaan. Tämä mahdollistaa vuorovaikutuksen pelaajan ja pelin välillä (Thorpe, Ma ja Oikonomou 2011).

Alkujaan pelien ohjaus oli yksinkertaista ja syöttölaitteet suunniteltiin vain yksittäisiin peleihin. Useita pelejä tukevien pelilaitteiden yleistyttyä heräsi kiinnostus monikäyttöisemmille ohjausjärjestelmille, jotka soveltuvat useiden eri pelien pelaamiseen. Peliohjaimet ovat kehittyneet pelien ja niitä pyörittävän laitteiston kehittymisen myötä, muutamalla painikkeella varustetuista syöttövälineistä nykyaikaisten pelikonsolien monipuolisempiin ohjaimiin (Thor-

pe, Ma ja Oikonomou 2011).

Esimerkiksi Unity-pelimoottorilla pelilaitekäsittelyyn on suunniteltu Input System. Input Systemin avulla kehittäjät voivat mukauttaa ja laajentaa syöttölaitteiden käsittelyä omien tarpeidensa mukaan. Se tukee monenlaisia syöttölaitteita, kuten peliohjaimia, näppäimistöjä, hiiriä ja liiketunnistimia. Kehittäjät voivat myös määritellä omat syötelaitteensa ja syötteiden käsittelylogiikan (Unity 2023).

Toisena esimerkkinä Unreal Enginen pelilaitekäsittely on idealtaan samankaltaisen kuin edellä kuvailtu Unityn pelilaitekäsittely. Unreal Enginen Input System tarjoaa visuaalisen ja koodipohjaisen lähestymistavan syötteiden käsittelyyn. Syötteet kartoitetaan toimintoihin ja akseleihin visuaalisen Blueprint-skriptausjärjestelmän tai C++ -koodin kautta (Epic Games 2024b). Blueprint-skriptausjärjestelmään perehdytään tarkemmin Unreal Enginea käsittelevässä alaluvussa.

4 Suosituimmat pelimoottorit

Tässä luvussa käsitellään markkinoiden ladatuimpia ja tunnetuimpia pelimoottoreita. Markkinapaikkojen pelidataa ja kehittäjäyhteisöjen aktiivisuutta ja kokoa seuraamalla voidaan huomata joidenkin pelimoottoreiden olevan laajalti kehittäjien suosiossa. Seuraavissa aliluvuissa tarkastellaan näiden pelimoottoreiden taustoja ja ominaisuuksia.

Ensimmäisenä käsitellään Unitya, jonka jälkeen käsitellään Unreal Enginea. Kyseiset pelimoottorit ovat laajalti suosittuja kehittäjien keskuudessa (Vohera ym. 2021). Myös Andrade (Andrade 2015) suosittelee kyseisiä pelimoottoreita, sillä niiden avulla voi kehittää yksinkertaisia projekteja, että AAA-tason pelejä. Sitten käsitellään CryEnginea, GameMakeria ja Godotia. Mainittujen pelimoottoreiden lisäksi on olemassa suuri joukko muitakin pelimoottoreita. Lähdekirjallisuuteen viitaten ne eivät kuitenkaan ole yhtä laajalti käytettyjä ja tehokkaita kuin esimerkiksi Unity ja Unreal Engine (Vohera ym. 2021). Tarkasteltavien pelimoottorien määrä on myös rajattava johonkin pisteeseen.

4.1 Unity

Unity on tanskalaisen Unity Technologiesin luoma pelimoottori. Alun perin Unity-pelimoottori julkistettiin Applen kansainvälisessä kehittäjien konferenssissa vuonna 2005. Unitya on käytetty pieniin indie-projekteihin ja suuriin AAA-tuotantoihin. Unity sisältää visuaalisen editorin ja integroidun kehitysympäristön, jotka mahdollistavat prototyyppien rakentamisen sekä koodimuutosten testaamisen reaaliajassa. Unity integroi useita väliohjelmistoja, kuten DirectX ja OpenGL grafiikan renderöintiin, FMOD-äänentoistoon, sekä PhysX-fysiikkamoottorin, tarjoten näin kehittäjille kattavat työkalut pelien suunnitteluun ja toteutukseen (Andrade 2015). Unitya käytetään videopeliteollisuuden lisäksi myös elokuva-, auto-, insinööri- ja rakennusteollisuudessa. Myös Yhdysvaltain asevoimat hyödyntävät Unitya (Singh ja Kaur 2022).

Unity-pelimoottori tukee C#-ohjelmointikielellä tehtyjä skriptejä. Se on alustariippumaton pelimoottori, jonka editorissa on raahaa ja pudota -toiminto assettien ja objektien käsittelyyn. Unityn editori tukee Linuxia, macOS:ää ja Windowsia, tehden siitä monialustatyöka-

lun. Unity-pelimoottori itse tarjoaa tuen yli 19 eri alustalle. Tähän sisältyvät pöytäkonealustat, Linux, Mac ja Windows, erilaiset pelikonsolit, kuten Nintendo Switch, PlayStation 4 ja Playstation 5 sekä uusin Xbox-sarja. Lisäksi mobiilialustat, kuten tvOS, iOS ja Android, sekä useat VR-alustat, mukaan lukien Google ARCore, PlayStation VR, Windows Hololens, Oculus, Unity XR SDK, Steam VR, Magic Leap ja Google Cardboard, ovat tuettujen alustojen joukossa (Singh ja Kaur 2022). Tuetut alustat luonnollisesti päivittyvät uusien laitteiden markkinoiden myötä.

Unity on suosittu pelimoottori etenkin indie-kehittäjien keskuudessa. Unitylla tehtyjä sovelluksia ladataan keskimäärin 3,6 miljardia kertaa kuukaudessa. Lisäksi 82 prosenttia sadasta suosituimmasta mobiilipelistä on tuotettu Unitylla (Unity 2024f). Unity sisältää merkittävän kokoisen yhteisön ja digitaalisen markkinapaikan: Unity Asset Storen. Markkinapaikka sisältää kehitystyökaluja ja komponentteja, joita voi käyttää pelinkehityksessä tai oppimismateriaalina (Andrade 2015). Laajan yhteisön lisäksi Unity Technologies tukee kehittäjiä tarjoamalla kattavia opetusvideoita, ohjeita ja dokumentaatiota.

Unityn Digitaalinen markkinapaikka eli Asset Store tarjoaa kehittäjille valikoiman ”assetteja” eli pelinkehityksessä käytettäviä elementtejä. Assetteja ovat muun muassa mallit eli 2D-sprite-kuvat ja 3D-mallit, jotka voivat olla esimerkiksi pelissä esiintyviä hahmoja tai esineitä. Muita esimerkkejä asseteista ovat erilaiset tekstuurit, animaatiot, skriptit, sekä pluginit eli lisäosat (Unity 2024a). Asset Storen resurssit vaihtelevat ilmaisista asseteista useiden satojen dollarien hintaisiin tuotteisiin, antaen kehittäjille mahdollisuuden myös myydä omia luomuksiaan digitaalisella markkinapaikalla (Unity 2024a).

Unity tarjoaa eri tasoisia hinnoitteluvaihtoehtoja. Unityn hinnoittelumallit ovat suunnattu eri käyttäjäryhmille projektin kokoon ja budjettiin perustuen. Mahdollistaen Unityn käytön pienille indie-kehittäjille, että suurille studioille.

Unity Personal on ilmainen vaihtoehto, joka on tarkoitettu harrastelijoille ja pienille tiimeille. Unity Personal tarjoaa pääsyn kaikkiin perustyökaluihin ilman aloituskustannuksia (Unity 2024d).

Unity Pro on maksullinen versio, joka tarjoaa lisäominaisuuksia, kuten analytiikkatyökaluja, mukautettavia splash screen -vaihtoehtoja ja tuen konsolikehitykselle. Unity Pro on tarkoi-

tettu keskisuurille ja suurille tiimeille, joiden projektien vaatimukset ovat vaativampia. Unity Pro kustantaa 1877 euroa vuodessa jokaista kehitystyökalua kohti (Unity 2024d).

Unity Enterprise on tarkoitettu suurille organisaatioille ja yrityksille, jotka vaativat kattavimmat työkalut ja suorat tukipalvelut suurten, monimutkaisten projektien hallintaan. Enterprise-sopimus voi sisältää pääsyn Unityn lähdekoodiin, asiakastukea ja mukautettavia ominaisuuksia (Unity 2024d).

Unity Industry on suunniteltu teollisuuden alojen, kuten arkkitehtuurin, insinöörityön ja rakentamisen tarpeisiin. Unity Industry tarjoaa työkaluja ja ominaisuuksia, jotka auttavat luomaan yksityiskohtaisia simulaatioita ja visualisointeja. Paketti kustantaa 4554 euroa vuodessa kehittäjää kohden (Unity 2024d).

Hinnoittelu on viime vuosina muuttunut muutamaan otteeseen. Kehittäjän tai kehittäjien on siirryttävä Unity Personalista Unity Pro -versioon kun Unitylla tuotetun projektin vuositulot tai projektin rahoitus ylittävät sadantuhannen Yhdysvaltain dollarin rajan. Unity ei peri rojalteja sillä tehdyistä tuotteista. Lisäkustannuksia kehittäjille voi kuitenkin syntyä ostetuista asseteista Unity Asset Storesta (Unity 2024d).

Unity koostuu visuaalisesta editorista ja integroidusta kehitysympäristöstä, jotka yhdessä mahdollistavat nopean prototyyppien rakentamisen. Yksinkertaiset kohtaukset ilman valmiita pelimekaniikkoja on mahdollista luoda ilman yhtään koodiriviä, komponenttipohjaisen työnkulun ansiosta. Kyseinen modulaarinen ja joustava lähestymistapa sallii kehittäjien rakentaa monimutkaisia pelimekaniikkoja yhdistelemällä pieniä logiikan osia (Andrade 2015).

Kehityksessä peruselementtiä kutsutaan GameObjectiksi. Scene eli kohtaus toimii näiden GameObjectien säiliönä. Jokaisen GameObjectin spatiaaliset ominaisuudet, graafinen käyttäytyminen ja pelimekaniikat määrittyvät erillisissä komponenteissa. Kehittäjät voivat laajentaa pelin toiminnallisuutta luomalla skriptikomponentteja, jotka perustuvat MonoBehaviour-luokkaan, ja lisätä peliin omia logiikkapalojaan (Andrade 2015). Näitä prototyyppejä voi testata editorissa viiveettä. Eli muutokset koodiin heijastuvat reaaliajassa ja käännösaika on minimaalinen. Tämä ominaisuus tekee kehitysprosessista sujuvan ja mahdollistaa nopean iteraation ja kokeilun eri peli-ideoiden kanssa (Andrade 2015).

4.2 Unreal Engine

Epic Gamesin kehittämä Unreal Engine on alun perin luotu vuonna 1998 julkaistua ensimmäisen persoonan räiskintäpeliä Unrealia varten (Sanders 2017). Lisäksi se on yksi ensimmäisistä merkittävistä pelimoottoreista, joka tuli laajalti kehittäjien käyttöön (Andrade 2015). Pelimoottori on kehittynyt vuosien saatossa ja viidennen version julkaisu tapahtui vuonna 2022. Unreal Engine on ilmaiseksi ladattavissa ja se sisältää pääsyn pelimoottorin lähdekoodiin (Drozina ja Orehovacki 2018). Jos Unreal Enginea käyttävän projektin tulot ylittävät miljoonan dollarin, Epic Games alkaa periä 5 prosentin rojaltimaksua. Muuten käyttö on maksutonta (Epic Games 2024d).

Unreal Enginea käytetään laajasti eri alustoilla, kuten Windowsilla, uuden sukupolven pelikonsoleilla, Androidilla ja iOS:lla. Kehitystyökalut ovat yhteensopivia Windowsin, macOS:n ja Linuxin kanssa. Ohjelmointi tapahtuu C++-ohjelmointikielellä. Unreal Engine mahdollistaa paitsi pelien, myös esimerkiksi demojen, trailereiden, elokuvien ja erilaisten visualisointien tuottamisen. Lisäksi Unrealilla on laaja yhteisö ja omat kauppapaikat, jotka tukevat sen käyttöä (Epic Games 2024d).

Unreal Engine tarjoaa perinteisen ohjelmoinnin lisäksi visuaalisen ohjelmointijärjestelmän, Blueprintin. Blueprint on solmupohjainen käyttöliittymä, jonka avulla voidaan luoda pelielementtejä suoraan editorissa. Tämä on suunniteltu erityisesti helpottamaan suunnittelijoiden työskentelyä, mahdollistaen prototyyppien rakentamisen ja testaamisen konsepteilla, jotka tavallisesti ovat vain ohjelmoijien hallinnassa. Blueprint siis mahdollistaa pelien suunnittelun ja toteutuksen ilman perinteistä koodaamista (Andrade 2015).

Vaikka Unreal Editoria luonnehditaan vähemmän aloittelijaystävällisemmäksi kuin sen montaa kilpailijaa, Unrealin kuvataan kykenevän tuottamaan visuaalisesti vaikuttavampaa jälkeä verrattuna moneen muuhun pelimoottoriin. Lisäksi Unreal Enginen erottuva piirre on sen avoin lähdekoodi tilaajille. Tämä mahdollistaa yhteisön tekemien parannukset ja tarjoaa oppimismahdollisuuksia kehittäjille (Andrade 2015). Tunnettuja pelejä, jotka on tuotettu Unreal Enginen avulla ovat muun muassa PUBG: BATTLEGROUNDS, Hogwarts Legacy, Fortnite, Borderlands 3, XCOM: Enemy Unknown, Tony Hawk's Pro Skater HD, Mass Effect 2 ja Gears of War 3 (Andrade 2015; *SteamDB* 2024).

4.3 CryEngine

CryEngine on saksalaisen Crytekin kehittämä pelimoottori, joka luotiin alkujaan tukemaan vuonna 2004 julkaistua Far Cry -peliä. Nykyisin pelimoottori on ladattavissa ilmaiseksi heidän verkkosivuiltaan, ja se mahdollistaa julkaisun sekä tietokoneilla että konsoleilla. Kun CryEnginea verrataan Unityyn ja Unreal Engineen, CryEnginen oppimiskäyrä vaikuttaa jyrkimmältä, editorin käyttöliittymä vähemmän käyttäjäystävälliseltä, yhteisö pienemmältä, eikä siinä ole yhtä laajaa markkinapaikkaa kuin Unitylla ja Unreal Enginella. Siitä huolimatta CryEngine tunnetaan kyvystään tuottaa huipputason grafiikkaa ja suorituskykyä. CryEnginen ohjelmointi toteutetaan C++- tai Lua-ohjelmointikielillä (Andrade 2015). Crytek perii viiden prosentin rojaltimaksun peliprojektien tuotoista, mutta ensimmäiset 5000 Yhdysvaltain dollaria vuodessa per peli ovat rojalteista vapaita. Tiettyjä poikkeuksia lukuun ottamatta, kuten tuloja Crytekin CRYENGINE markkinapaikalta rojaltimaksuja ei peritä (Crytek 2024).

4.4 GameMaker

GameMaker on kaupallinen pelinluontijärjestelmä ja pelimoottori, joka tukee monia genrejä ja on suunnattu pöytäkoneille, mobiililaitteille sekä uusimmille pelikonsoleille. GameMaker on kehitetty ajatellen kehittäjiä, jotka eivät osaa ohjelmoida. GameMaker sisältää raahaa ja pudota-editorin, joka mahdollistaa pelien luomisen piilottamalla taustalla olevan logiikan. Tätä kutsutaan yleisesti visuaaliseksi ohjelmoinniksi, ja sen esikuva on MIT:n Scratch-ohjelmointioppimisympäristö. Siitä huolimatta GameMaker tarjoaa myös oman ohjelmointikielensä monimutkaisempien toimintojen toteuttamiseen: Game Maker Languagen. GameMakerin editori on pääasiassa suunnattu 2D-grafiikalle ja vaikka se tukee 3D:tä, kaikki 3D-toiminnallisuus on mukautettava ohjelmoimalla. Näin ollen GameMakerin etuna on sen helpompi oppimiskynnys verrattuna moneen muuhun pelimoottoriin (Andrade 2015).

GameMaker on ilmaiseksi ladattavissa ja käytettävissä, mutta tietyin ehdoin. Mikäli kehittäjä julkaisee pelin kaupalliseen käyttöön, täytyy kehittäjän ostaa sadan Yhdysvaltain dollarin hintainen lisenssi (YoYo Games 2024).

4.5 Godot

Godot on avoimen lähdekoodin pelimoottori, joka mahdollistaa pelien kehittämisen useille eri alustoille sekä kaksi- että kolmiulotteisina. Pelimoottori sisältää mahdollisuuden ohjelmoida useammalla eri ohjelmointikielellä, suorituskyvyn optimoinnin, verkkotoiminnot sekä solmupohjaisen järjestelmän taso- ja ympäristösuunnitteluun. Godot tarjoaa resurssit pelien optimointiin ja julkaisuun useilla alustoilla, myös verkossa. Godot on täysin ilmainen ladata ja käyttää (Ranaweera ja Mahmoud 2024; Linietsky ja avustajat 2024).

Godot tukee useita käyttöjärjestelmiä, kuten Windowsia, macOS:ia, Linuxia, Androidia ja HTML5:ä. Godotin MIT-lisenssin avoimuus mahdollistaa kehittäjien muokata ja hyödyntää pelimoottoria ilman juridisia esteitä. Godot sisältää useita ohjelmointikieliä, kuten GDScript, joka muistuttaa Pythonia, sekä tuen C#, C++ ja Visual Script -kielille. Lisäksi pelimoottori sisältää sekä kaksi- että kolmiulotteisia fysiikkamoottoreita. Godot käyttää oletuksena Bullet-fysiikkamoottoria ja sen omaa, yksinkertaisempaa GodotPhysics-fysiikkamoottoria (Ranaweera ja Mahmoud 2024).

4.6 Muut pelimoottorit

Aikaisemmissa aliluvuissa mainittujen pelimoottoreiden lisäksi on olemassa muitakin vapaasti ladattavissa olevia pelimoottoreita. Sen lisäksi useat pelistudiot hyödyntävät itse kehittämiään pelimoottoreita, joihin ei muilla kehittäjillä ole pääsyä. Etenkin suuret pelistudiot, kuten Ubisoft, Sony Interactive Entertainment ja Electronic Arts hyödyntävät omistamiaan pelimoottoreita. Vaikka nämä pelimoottorit eivät ole saatavilla yleisölle, niiden kehitys ja käyttö antavat arvokasta tietoa peliteknologian kehityssuunnista ja innovaatioista peliteollisuudessa.

Pelistudiot voivat saavuttaa merkittäviä etuja kehittämällä oman pelimoottorinsa sen sijaan, että turvautuisivat markkinoilta saataviin pelimoottoreihin. Pelistudiot voivat näin vaikuttaa pelimoottorin ominaisuuksiin, mikäli markkinoilta saatavat pelimoottorit eivät täytä pelistudion kriteerejä. Tällöin myös pelimoottorin ylläpito ja päivittäminen ei ole kolmansista osapuolista riippuvaista. Näin pelistudio myös välttää toisten studioiden pelimoottorin hyödyntämisestä koituvat lisenssimaksut. Se tarjoaa useita muitakin etuja kuten mahdollisen kil-

pailuedun kilpaileviin pelistudioihin, mikäli muilla pelistudioilla ei ole saatavilla yhtä edistyksellistä teknologiaa pelien kehittämiseen.

Kilpailukykyisen pelimoottorin kehittäminen voi kuitenkin olla haasteellista ja kallista. Myös pelimoottorin kehitysprosessi on usein aikaa vievä, joka johtaa useita kehittäjiä valitsemaan valmiin pelimoottorin nopeamman kehitystahdin vuoksi.

5 Pelimoottorin valinta

Se, mitä pelimoottoria olisi järkevintä käyttää, tulee ottaa huomioon eri pelimoottoreiden ominaisuudet, kehittäjän osaaminen ja käytössä olevat resurssit (Andrade 2015).

Andraden ja Voheran mukaan ei ole olemassa ”parasta pelimoottoria” (Andrade 2015; Vohera ym. 2021). Valittaessa pelimoottoria kehittäjän tulisi Andraden (Andrade 2015) mukaan ottaa huomioon muutama asia: Millaista peliä kehittäjä on luomassa ja mille alustalle? Mikä on kehittäjän aikaisempi kokemus pelialasta ja ohjelmoinnista? Kuinka paljon vaivaa kehittäjä on valmis näkemään alustan oppimiseksi? Kuinka paljon vaivaa kehittäjä on valmis näkemään pelin luomiseksi? Kuinka pitkä kehitysvaihe tulee olemaan ja mikä on pelimoottorin hinnoittelumalli (Andrade 2015)?

Pidemmän aikavälin pelikehitykseen tähtäävälle Unity tai Unreal Engine ovat suositeltavia vaihtoehtoja, mahdollistaen kehittämisen yksinkertaisista projekteista AAA-tason peleihin. Jos tavoitteena on saada toimiva prototyyppi mahdollisimman nopeasti tai jos ei ole niin perehtynyt logiikkaohjelmointiin, jokin toinen alusta voisi olla parempi vaihtoehto (Andrade 2015). Vohera mainitsee (Vohera ym. 2021), että käyttöönottokriteerien suhteen Unity ja Unreal Engine ovat parhaita pelimoottoreita. Mikäli kyse on grafiikasta ja animaatiosta, Unreal Engine ja CryEngine nousevat kärkeen.

Uusien kehittäjien tulisi myös käyttää aikaa kokeillakseen muutamaa eri vaihtoehtoa ja saadakseen tuntumaa erilaisista editoriliittymistä ja pelin abstraktioista. Ennen lopullisen valinnan tekemistä tulisi myös arvioida, kuinka aktiivinen kyseisen pelimoottorin yhteisö on, kuinka paljon tukea sieltä voi odottaa ja mikä on pelimoottorin dokumentaation ja esimerkkien laatu (Andrade 2015). Vohera (Vohera ym. 2021) mainitsee, että pelimoottoreiden erilaisista ominaisuuksista huolimatta pelimoottorit ovat vain työkaluja. Lopulta kehittäjien taitotaso määrittää pelin laadun.

6 Yhteenveto

Tutkielma on antanut katsauksen pelimoottoreihin, jotka ovat keskeisessä osassa peliteollisuutta ja pelinkehitystä. Ensin tutkielmassa perehdyttiin pelimoottoreihin yleisellä tasolla. Selvitettiin pelimoottorin olevan pohjimmiltaan ohjelmistokehys, jonka tarkoituksena on helpottaa pelinkehitystä. Pelimoottori tarjoaa valmiin testatun ja optimoidun pohjan, jonka päälle peliä voi rakentaa, jottei kaikkea tarvitse jokaisen pelin kohdalla luoda alusta. Näin ollen kehittäjän ei tarvitse käyttää resursseja perusteknologioiden rakentamiseen alusta asti, vaan voi keskittyä pelin luovaan prosessiin ja pelikokemuksen hiomiseen. Pelimoottorit ovat siis keskeisessä roolissa nykyaikaisten pelien kehityksessä.

Perehtyessä aiheeseen huomattiin, että pelimoottoreiden ominaisuuksia voitiin jakaa alijärjestelmiin. Alijärjestelmät tarjoavat kehittäjille työkaluja muun muassa fysiikan, visuaalisuuden, äänimaailman ja monien muiden pelin kannalta keskeisten osien toteuttamiseen. Lähdekirjallisuuden avulla huomattiin, että pelkästään yksittäiset komponentit, kuten fysiikkamoottori ovat toiminnaltaan monipuolisia ja monimutkaisia järjestelmiä. Komponenttien parempi tekninen ymmärtäminen ja tarkempi kuvaus tarjoavat mahdollisuuksia myös jatkotutkimuksille.

Pelimoottoreiden yleisemmän käsittelyn ja komponenttien kuvailun jälkeen tarkasteltiin erityisesti markkinoiden johtavia pelimoottoreita. Markkinoiden ladatuimpia ja käytetyimpiä pelimoottoreita ovat muun muassa Unreal Engine ja Unity. Yksityiskohtaista tilastotietoa pelimoottoreiden käyttöasteesta on saatavilla vain rajoitetusti. Pelimoottoreiden suosio voidaan kuitenkin havaita useista lähteistä, kuten yritysten omista raporteista, käyttäjäyhteisöjen toiminnasta sekä markkinapaikkojen pelidatasta.

Tutkittaessa eri pelimoottoreita havaittiin, että vaikka kaikilla pelimoottoreilla on sama perusperiaate, on niiden ominaisuuksissa ja kehittäjille asettamissa vaatimuksissa eroja. Esimerkiksi Unreal Engine tarjoaa kehittyneitä grafiikkaominaisuuksia. Kyseisen pelimoottorin käyttö kuitenkin edellyttää kehittäjältä perehtymistä lisenssiehtoihin ja mahdollisiin rojaltimaksuihin. Toisaalta Godot-pelimoottori ei tarjoa yhtä vaikuttavia grafiikkaominaisuuksia, mutta sen käyttö ei vaadi kehittäjältä lisenssisitoumuksia. Markkinoilla on siis useita ladatta-

vissa olevia pelimoottoreita ja niiden käyttöön liittyvät ehdot vaihtelevat. Unity tarjoaa runsaasti valmiita skriptejä ja ohjeita, kun taas vähemmän suosituille pelimoottoreille tällainen tuki on rajallisempaa. Lisäksi eri pelimoottorit tukevat eri alustoja.

Näin ollen pelimoottoria valittaessa on tärkeää ottaa huomioon kehittäjän tavoitteet, projektin vaatimukset ja käytössä olevat resurssit. Ymmärrys pelimoottoreiden erilaisista ominaisuuksista ja lisenssiehdoista auttaa kehittäjiä tekemään valintoja, jotka tukevat parhaiten heidän projektejaan.

Lähteet

Andrade, Antonio. 2015. “Game engines: a survey”. *EAI Endorsed Transactions on Serious Games Research,* https://doi.org/10.4108/eai.5-11-2015.150615.

Bao, Hujun, ja Wei Hua. 2011. *Real-Time Graphics Rendering Engine.* 1. painos. Springer Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-18342-3.

Bicalho, Luís Fernando, Bruno Feijó ja Augusto Baffa. 2020. “A Culture Model for Non-Player Characters’ Behaviors in Role-Playing Games”. Teoksessa *2020 19th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames).* Rio de Janeiro, Brazil: IEEE. https://doi.org/10.1109/SBGames51465.2020.00013.

Crytek. 2024. *CryENGINE Licensing.* https://www.cryengine.com/support/view/licensing. Viitattu: 24.4.2024.

Drozina, Adrian, ja Tihomir Orehovacki. 2018. “Creating a Tabletop Game Prototype in Unreal Engine 4”. Teoksessa *MIPRO 2018.* Pula, Croatia: Juraj Dobrila University of Pula, Faculty of Informatics. https://doi.org/10.23919/MIPRO.2018.8400282.

Epic Games, Inc. 2024a. “Behavior Tree in Unreal Engine - Overview”. Viitattu: 24.4.2024. https://dev.epicgames.com/documentation/en-us/unreal-engine/behavior-tree-in-unreal-engine---overview?application_version=5.0.

———. 2024b. “Input”. Viitattu: 24.4.2024. https://docs.unrealengine.com/4.26/en-US/InteractiveExperiences/Input/.

———. 2024c. “Modifying the Navigation System”. Viitattu: 24.4.2024. https://docs.unrealengine.com/4.27/en-US/InteractiveExperiences/ArtificialIntelligence/NavigationSystem/ModifyingTheNavigationMesh/ModifyingtheNavigationSystem/.

———. 2024d. “Unreal Engine 5”. Viitattu: 24.4.2024. https://www.unrealengine.com/en-US/unreal-engine-5.

———. *Smooth Frame Rate.* https://docs.unrealengine.com/4.27/en-US/TestingAndOptimization/PerformanceAndProfiling/SmoothFrameRate/. Viitattu: 12.4.2024.

Gregory, Jason. 2009. *Game Engine Architecture.* 1. painos. Boca Raton, FL: A K Peters/CRC Press.

———. 2018. *Game Engine Architecture.* 3. painos. Boca Raton, FL: Taylor & Francis, CRC Press.

Du-Harpur, X., F.M. Watt, N.M. Luscombe ja M.D. Lynch. 2020. "What is AI? Applications of artificial intelligence to dermatology". *British Journal of Dermatology* 183. https://doi.org/10.1111/bjd.18880.

Hu, Ying, Huiqiang Lu ja Wang Yijin. 2012. "Design and Implementation of Three-Dimensional Game Engine". Teoksessa *World Automation Congress.* Puerto Vallarta, Mexico: IEEE. ISBN: 978-1-889334-47-9.

Khoei, Mina A., Francesco Galluppi, Quentin Sabatier, Pierre Pouget, Benoit R. Cottereau ja Ryad Benosman. 2018. "Faster is better: Visual responses to motion are stronger for higher refresh rates" (joulukuu). https://doi.org/10.1101/505354.

Linietsky, Juan, ja Ariel Manzur ja muut avustajat. 2024. *Godot Engine Features.* https://godotengine.org/features/. Viitattu: 23.4.2024.

Pelaaja.fi. 2023. "Hittiräiskintä The Finals herätti kohun – korvasi ääninäyttelijät tekoälyllä". Viitattu: 4.4.2024, *Pelaaja* (lokakuu). https://www.pelaaja.fi/uutiset/hittiraiskinta-the-finals-heratti-kohun-korvasi-aaninayttelijat-tekoalylla/.

Ranaweera, Mahesh, ja Qusay H. Mahmoud. 2024. "Deep Reinforcement Learning with Godot Game Engine". Julkaistu: 5.3.2024, *Electronics,* https://doi.org/10.3390/electronics13050985.

Sanders, Andrew. 2017. *An Introduction to Unreal Engine 4.* CRC Press, Taylor & Francis Group. ISBN: 978-1-4987-6509-1.

Singh, Swati, ja Amanpreet Kaur. 2022. "Game Development using Unity Game Engine". Teoksessa *2022 3rd International Conference on Computing, Analytics and Networks (ICAN).* Chitkara University, Punjab, India: IEEE. https://doi.org/10.1109/ICAN56228.2022.10007155.

SteamDB. 2024. Viitattu: 24.4.2024. https://steamdb.info/tech/Engine/Unreal/?min_reviews=500.

Thorpe, Andrew, Minhua Ma ja Andreas Oikonomou. 2011. “The 16th International Conference on Computer Games History and Alternative Game Input Methods”. University of Derby, UK ja Glasgow School of Art, UK: IEEE. https://doi.org/10.1109/CGAMES.2011.6000321.

Tsipis, Athanasios, Vasileios Komianos ja Konstantinos Oikonomou. 2019. “A Cloud Gaming Architecture Leveraging Fog for Dynamic Load Balancing in Cluster-Based MMOs”. Teoksessa *2019 4th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM)*. Piraeus, Greece: IEEE. https://doi.org/10.1109/SEEDA-CECNSM.2019.8908282.

Unity. 2023. “Actions”. Viitattu: 24.4.2024. https://docs.unity3d.com/Packages/com.unity.inputsystem@1.7/manual/Actions.html.

———. 2024a. “AssetStore”. Viitattu: 24.4.2024. https://assetstore.unity.com/.

———. 2024b. “Audio Overview”. Viitattu: 9.4.2024. https://docs.unity3d.com/Manual/AudioOverview.html.

———. 2024c. “Beginner AI Pathfinding”. Viitattu: 24.4.2024. https://learn.unity.com/project/beginner-ai-pathfinding.

———. 2024d. “Plans and pricing”. Viitattu: 24.4.2024. https://unity.com/pricing.

———. 2024e. *Unity Documentation, Application.targetFrameRate*. https://docs.unity3d.com/ScriptReference/Application-targetFrameRate.html. Viitattu: 9.4.2024.

———. 2024f. “Unity etusivu”. Viitattu: 24.4.2024. https://unity.com/.

———. 2024g. “Unity Machine Learning Agents”. Viitattu: 24.4.2024. https://unity.com/products/machine-learning-agents.

Vohera, Chaitya, Ayush Desai, Heet Chheda, Vijal Jain ja Dhruveel Chouhan. 2021. “Game Engine Architecture and Comparative Study of Different Game Engines”. Kharagpur, India, heinäkuu. https://doi.org/10.1109/ICCCNT51525.2021.9579618.

YoYo Games. 2024. *GameMaker Pricing*. https://gamemaker.io/en/get. Viitattu: 24.4.2024.