Tampereen ammattikorkeakoulu

Kimble-lautapelin digitalisointi Unity-pelimoottorilla

Joa Lamminluoto

OPINNÄYTETYÖ
Joulukuu 2025

Tietojenkäsittelyn tutkinto-ohjelma
Games Production

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietojenkäsittelyn tutkinto-ohjelma
Games Production

LAMMINLUOTO JOA:
Kimble-lautapelin digitalisointi Unity-pelimoottorilla

Opinnäytetyö 29 sivua, joista liitteitä 0 sivua
Joulukuu 2025

Tämän toiminnallisen opinnäytetyön tavoitteena oli toteuttaa digitaalinen versio Kimble-lautapelistä Unity-pelimoottoria käyttäen. Tarkoituksena oli selvittää, miten perinteisen lautapelin mekaniikat voidaan siirtää digitaaliseen ympäristöön siten, että alkuperäinen pelituntuma ja pelin selkeä rakenne säilyvät. Työllä ei ollut ulkopuolista tilaajaa, vaan se toteutettiin itsenäisenä kehitysprojektina.

Työ eteni suunnitteluvaiheesta toteutukseen iteratiivisesti. Suunnittelussa määriteltiin Kimblen keskeiset pelimekaniikat, pelin rakenne sekä tekniset vaatimukset. Toteutuksessa rakennettiin digitaalinen pelilauta, Pop-O-Matic-kupu, pelinappuloiden liikkumislogiikka, pelaajien vuorojärjestelmä sekä käyttöliittymä. Peli toteutettiin kaksidimensionaalisena versiona Unityn 2D-työkaluja hyödyntäen, ja kaikki visuaaliset elementit piirrettiin tätä projektia varten.

Lopputuloksena syntyi toimiva prototyyppi, joka noudattaa Kimblen virallisia sääntöjä ja mahdollistaa 2–4 pelaajan paikallisen moninpelin. Pelin logiikka, liikkumismekanismit, törmäysten käsittely sekä Pop-O-Maticin digitaalinen toteutus muodostavat kokonaisuuden, joka vastaa fyysisen pelin perusperiaatteita.

Johtopäätöksenä voidaan todeta, että lautapelin digitalisointi vaatii sekä sääntöjen ymmärtämistä että digitaaliselle pelille sopivien ratkaisujen suunnittelua. Peli toimii suunnitellulla tavalla, ja sitä voidaan jatkossa kehittää lisäämällä esimerkiksi tekoälyvastustajia tai verkkomoninpelitoiminnallisuus.

Asiasanat: kimble, pelikehitys, unity, digitalisointi

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in Business Information Systems
Games Production

LAMMINLUOTO JOA:
Digitalisation of the Board Game Kimble Using the Unity Game Engine

Bachelor's thesis 29 pages, appendices 0 pages
December 2025

This thesis explores the digital adaptation of the board game Kimble using the Unity game engine. The purpose was to examine how the essential mechanics of a simple physical board game can be transformed into a functional digital prototype while retaining the recognisable structure and flow of the original game.

The work was carried out as an independent development project and followed an iterative approach. The implementation relied on Unity and C#, and a two-dimensional environment was selected to keep the project technically manageable. The core features of the prototype include a digital game board, piece movement rules based on the official instructions, a turn-based structure, and a simulated Pop-O-Matic dice mechanism. All graphics were created specifically for the prototype.

The resulting game functions according to the official rules and supports local multiplayer for two to four players. The project shows that even a simple board game requires careful planning when adapted to digital form. The prototype provides a basis for further development, such as adding AI opponents, online multiplayer features or extended audiovisual elements.

Key words: kimble, unity, game development, digitalisation, prototype

TEKOÄLYN KÄYTTÖ OPINNÄYTTEESSÄ

Opinnäytteessäni on käytetty tekoälysovelluksia:

☐ Ei

☒ Kyllä

Ilmoitukseni mukaan olen käyttänyt opinnäytteessäni opinnäytetyöprosessin aikana seuraavia tekoälysovelluksia: ChatGPT

Tekoälysovellusten nimet ja versiot:
ChatGPT 5.1

Käyttötarkoitus: Käytin tekoälyä bugien paikantamiseen ja korjaamisen pelin kehityksen aikana. Käytin tekoälyä myös lisäämään kommentteja koodiin sen helppolukuisuuden parantamiseksi.

Osiot, joissa tekoälyä on käytetty: Pelin koodi

Olen tietoinen siitä, että olen täysin vastuussa koko opinnäytteeni sisällöstä, mukaan lukien osat, joissa on hyödynnetty tekoälyä, ja hyväksyn vastuun mahdollisista eettisten ohjeiden rikkomuksista.

SISÄLLYS

ERITYISSANASTO

Canvas	Unityn käyttöliittymäkerros.
Coroutine	Unityn ajastettu tai vaiheittainen koodisuoritus.
Debugger	Työkalu koodin virheiden etsimiseen.
GameObject	Unityn perusobjekti pelimaailmassa.
IDE	Ohjelmointiin tarkoitettu kehitysympäristö.
Inspector	Unityn näkymä objektien asetusten muokkaamiseen.
Kotipesä	Pelaajan aloitusalue Kimble-lautapelissä.
Lähtöympyrä	Ruutu, johon nappula siirtyy kotipesästä.
Maaliympyrä	Pelin lopetusruudut, joita kohti nappulat kulkevat.
Olio	Ohjelmoinnin tietorakenne, joka sisältää dataa ja toimintoja.
Pelin ydinsilmukka	Toistuva pelin perusrytmi. Englanniksi core loop.
Skripti	C#:lla kirjoitettu ohjelmakoodi Unityssä.
Sprite	2D-kuva tai -grafiikkaobjekti.
Transform	GameObjectin paikka, koko ja suunta.
UI	Pelin käyttöliittymä.

1 JOHDANTO

Lautapelien digitalisointi on viime vuosina yleistynyt merkittävästi. Yksi syy tähän on lautapelien saavutettavuuden paraneminen, kun niitä voidaan pelata suoraan puhelimella ilman fyysisen pelilaudan kuljettamista. Lisäksi digitaalinen versio mahdollistaa helpomman vastustajien löytämisen esimerkiksi verkkopelaamisen tai automaattisen pelaajahaun avulla.

Tämän opinnäytetyön aiheeksi valittiin Kimble-lautapelin digitalisointi, koska Kimble on Suomessa laajasti tunnettu ja helposti lähestyttävä peli. Unity-pelimoottori valittiin käyttöön, koska Tampereen ammattikorkeakoulun tietojenkäsittelyn pelituotannon opinnoissa Unity on yleisimmin käytetty kehitysalusta ja se soveltuu hyvin tämän tyyppisen projektin toteuttamiseen.

Opinnäytetyön tavoitteena oli toteuttaa digitaalinen versio Kimblestä Unityä käyttäen. Tarkoituksena oli luoda selkeä ja helposti ymmärrettävä malli siitä, miten lautapelin voi siirtää digitaaliseen muotoon, ja tarjota samalla esimerkki aloittaville pelinkehittäjille Unity-projektin rakenteesta ja keskeisistä tekniikoista. Aihe on merkityksellinen työelämälle, sillä lautapelien digitalisointi on hyvin yleistä pelialalla.

Peli rajattiin paikalliseen moninpeliin 2–4 pelaajalle, ja sen toiminnallisuus perustuu Kimblen virallisiin sääntöihin. Säännöt ovat saatavilla Kimblen virallisilla sivuilla osoitteessa: https://www.kimble. fi/wordpress/index.php/saannot/. Digitaaliseen versioon ei toteutettu tekoälyvastustajia, joten yksinpeliä ei ole mahdollista pelata. Myöskään verkkomoninpelitoiminnallisuutta ei toteutettu, vaan peli on suunniteltu pelattavaksi samalta laitteelta.

Opinnäytetyötä aloittaessani minulla oli noin kahden vuoden kokemus Unity-pelimoottorista sekä C#-ohjelmointikielestä. En kuitenkaan ennen ollut suunnitellut pelin käyttöliittymää, tehnyt pelin grafiikoita enkä ollut toteuttanut kokonaista peliä yksin.

Projektissa syntyi prototyyppi versio digitaalisesta Kimble-lautapelistä. Peli toimii Kimblen sääntöjen mukaan ja sitä voi pelata halutessaan 2–4 pelaajan voimin.

Peli toteutettiin iteratiivisesti, rakentaen pelin logiikkaa vaiheittain ja jokainen osa testattiin ennen seuraavaan osaan siirtymistä.

Tämä raportti etenee siten, että ensin käsitellään lautapelien digitalisointia ja Kimblen pelimekaniikkaa. Tämän jälkeen esitellään digitalisoinnissa käytetyt ohjelmat ja työkalut. Luvuissa 4 ja 5 kuvataan suunnitteluprosessi sekä digitaalisen Kimblen toteutus vaiheittain. Lopuksi pohditaan projektin onnistumista ja jatkokehitysmahdollisuuksia.

2 KIMBLE LAUTAPELIN DIGITALISOINTI

Kimble-lautapeli on suomalaisen Aarne Heljakan suunnittelema versio yhdysvaltalaisesta Trouble-pelistä. Pelissä tavoitteena on siirtää omat neljä nappulaa kotipesästä pelilaudan ympäri maaliin ennen muita pelaajia. Kimblen ja Troublen tunnetuin piirre on pelilaudan keskellä sijaitseva Pop-O-Matic-kupu (kuva 1), jonka painaminen pyöräyttää noppaa ja tuottaa tälle ominaisen napsahtavan äänen. Kimblen julkaisi Suomessa vuonna 1967 nykyinen Tactic Games. (Wikipedia 7.11.2025.)

KUVA 1. Aito Kimble-lauta (Kuva: Joa Lamminluoto 2025).
Kuvassa näkyvät kaikki nappulat kotipesissään.

Kimblen perussäännöt ovat seuraavat: Pelaaja painaa Pop-O-Matic-kupua kerran vuorollaan ja siirtää yhtä neljästä nappulastaan nopan silmäluvun verran. Silmäluku 6 oikeuttaa uuteen heittovuoroon ja lisäksi mahdollistaa nappulan siirtämisen kotipesästä lähtöympyrään normaalin laudalla siirtämisen sijaan. Jos pelaajan nappula siirtyy ruutuun, jossa on vastustajan nappula, vastustaja joutuu

palaamaan kotipesään. Kun pelaajan kaikki neljä nappulaa ovat saavuttaneet maaliympyrät, jotka on merkitty numeroin 1–4, pelaaja voittaa pelin.

Kimble soveltuu erinomaisesti digitalisointiin, sillä sen säännöt ovat yksinkertaiset, pelin kulku on selkeää ja Pop-O-Matic tarjoaa ikonisen ja helposti simuloitavan pelimekaniikan. Lisäksi peli on Suomessa hyvin tunnettu, mikä tekee siitä luontevan valinnan digitalisointiprojektiin.

3 DIGITALISOINTIIN KÄYTETTÄVÄT OHJELMAT

Digitaalisen Kimble-pelin toteuttamiseen käytettiin kahta keskeistä työkalua: Unity-pelimoottoria ja Visual Studio -kehitysympäristöä. Unity toimi pelin varsinaisena moottorina, jossa pelilogiikka, animaatiot, käyttöliittymä ja visuaaliset elementit rakennettiin. Visual Studiota käytettiin puolestaan C#-ohjelmointikielen kirjoittamiseen ja bugikorjaukseen. Nämä kaksi ympäristöä muodostavat pelialalla yleisesti käytetyn työparin, erityisesti Unity-projekteissa (Unity Technologies 2025).

Unity on yksi maailman suosituimmista pelimoottoreista, ja sitä käytetään sekä harrastelijaympäristöissä että ammattimaisessa pelituotannossa. Unity tarjoaa valmiin 2D- ja 3D-grafiikkamoottorin, fysiikkamoottorin, animaatiotyökalut sekä laajan dokumentaation ja yhteisön tuen. Lisäksi Unityn toimintaperiaate – komponenttipohjainen arkkitehtuuri – helpottaa pelihahmojen, objektien ja pelilogiikan rakentamista moduuleittain, mikä soveltui hyvin myös Kimble-digitalisointiprojektin tarpeisiin.

Visual Studiota käytettiin C#-koodien kirjoittamiseen ja virhekorjaukseen. C# on Unityn ensisijainen ohjelmointikieli, ja se tarjoaa selkeän syntaksin sekä vahvan tyyppiturvan, jotka helpottavat virheiden välttämistä ja pelilogiikan hallintaa. C#:n ja Unityn integraatio mahdollisti sen, että pelilogiikka (kuten nappuloiden liikkuminen, Pop-O-Matic-mekaniikka ja vuorojärjestys) voitiin toteuttaa selkeästi eriytetyissä koodeissa.

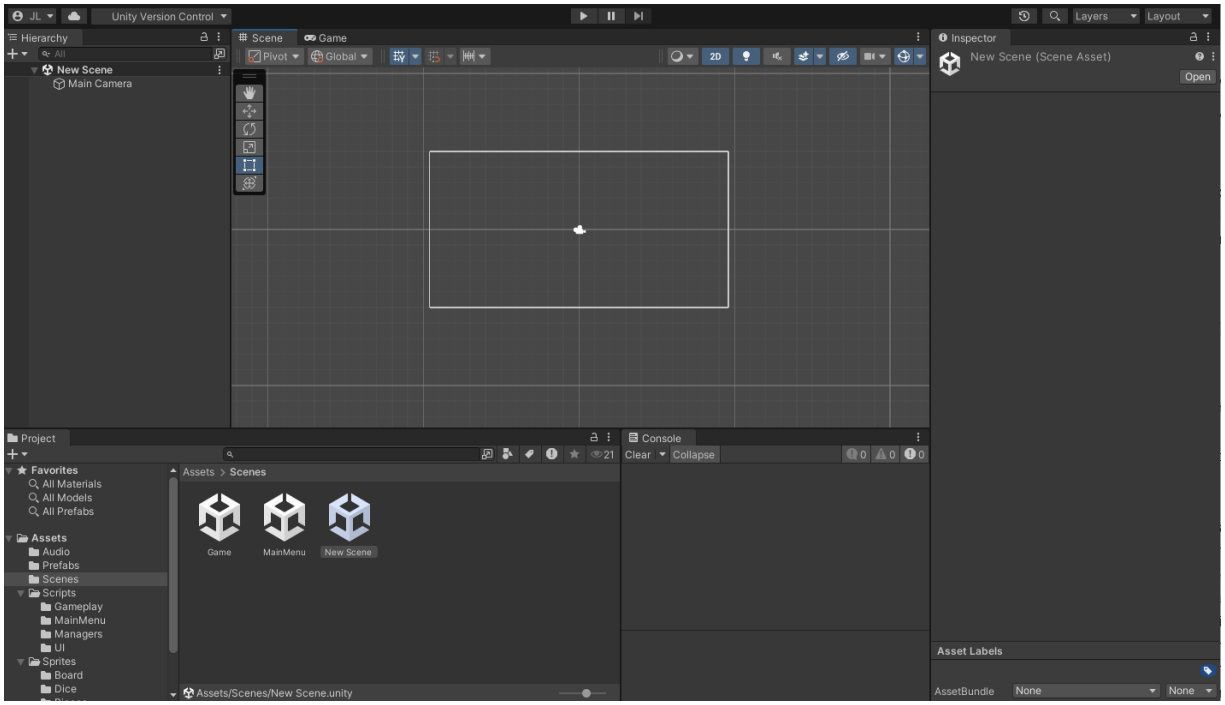
Kokonaisuutena Unity ja Visual Studio muodostivat tehokkaan ja joustavan työkalupaketin pelin digitalisointiin. Niiden avulla pystyttiin rakentamaan visuaalisesti selkeä ja toiminnallisesti eheä lautapeliversio, joka noudattelee uskollisesti Kimblen sääntöjä ja pelimekaniikkaa.

3.1 Unity-pelimoottori

Unity on yksi maailman laajimmin käytetyistä pelimoottoreista, ja se soveltuu erityisen hyvin pieniin ja keskikokoisiin projekteihin. Unityssä pelit rakentuvat GameObject-olioista, joihin voidaan liittää erilaisia komponentteja, kuten skriptejä,

fysiikkaelementtejä ja käyttöliittymäosia. Pelimoottorin komponenttipohjainen arkkitehtuuri mahdollistaa pelilogiikan toteuttamisen vaiheittain ja modulaarisesti, mikä helpottaa kokonaisuuden hallintaa ja eri järjestelmien yhteensovittamista. (Unity Technologies 2025.)

Unity on suosittu myös opetuskäytössä, sillä sen editori on visuaalisesti selkeä ja projektin rakenne on helposti hahmotettavissa (kuva 2). Lisäksi Unity tukee nopeaa iterointia: pelin voi käynnistää ja testata suoraan editorissa, mikä nopeuttaa kehitystyötä erityisesti prototyypin rakentamisvaiheessa. Pelimoottori sisältää valmiit työkalut erilaisten 2D- ja 3D-objektien käsittelyyn, animaatioiden luomiseen, äänten toistamiseen, käyttöliittymien rakentamiseen sekä pelilogiikan ohjelmointiin C#-kielellä. Taulukossa 1 on esitetty Unity-komponentit, joita käytettiin pelin keskeisten toiminnallisuuksien rakentamiseen. Lueteltujen ominaisuuksien ansiosta merkittävä osa pelin teknisestä toteutuksesta voidaan rakentaa saman työkalun sisällä ilman tarvetta ulkoisille ohjelmistoille.



KUVA 2. Unityn käyttöliittymä. Unityn käyttöliittymä on selkeä ja helppolukuinen.

Kimblen digitalisoinnissa Unity mahdollisti pelilaudan, nappuloiden liikeratojen ja nopan toiminnallisuuden toteuttamisen selkeästi yhtenäisenä kokonaisuutena.

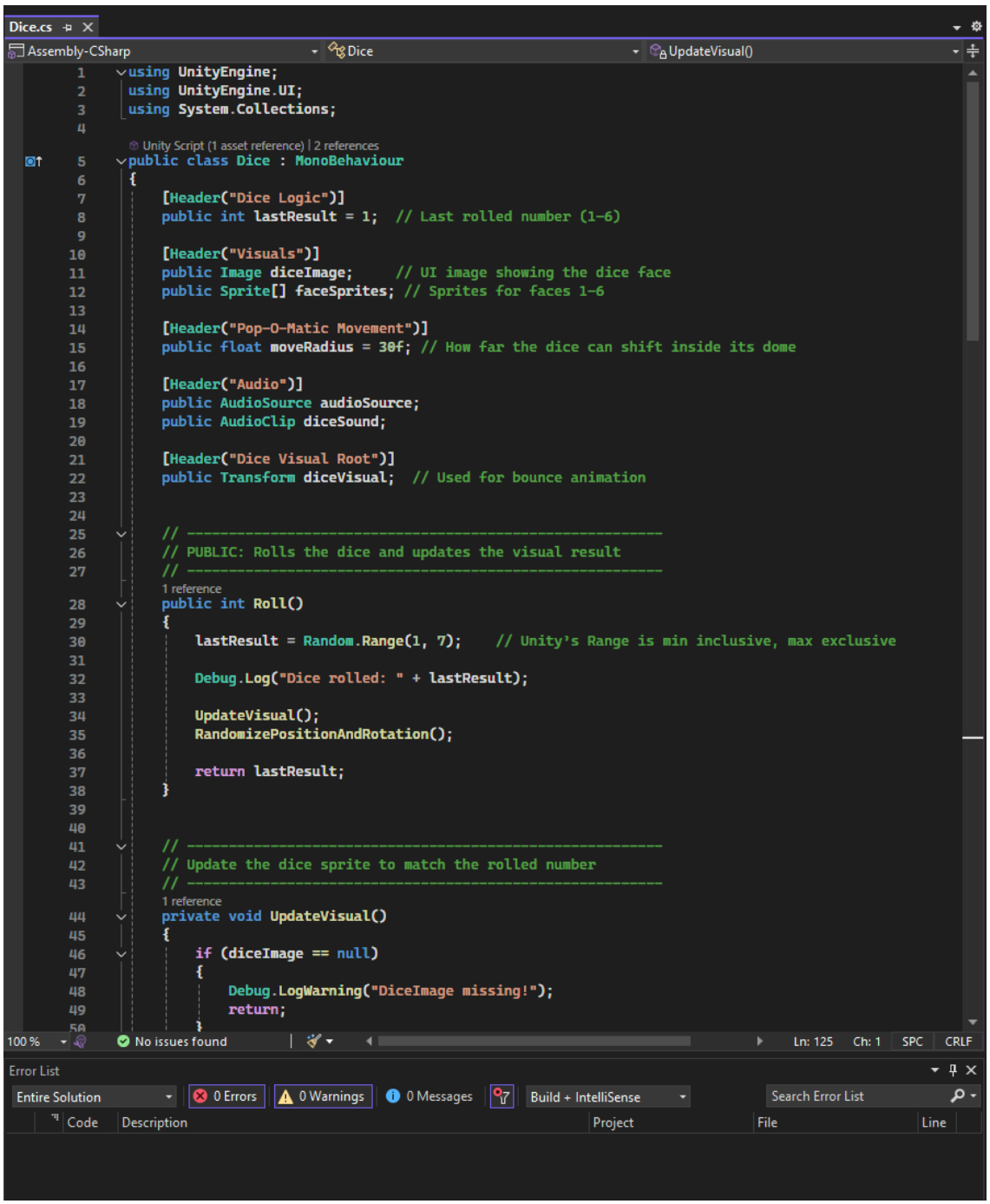
Koska sekä logiikka että visuaaliset elementit rakennettiin samassa ympäristössä, pelin eri osien välinen yhteistyö oli helppo varmistaa ja pelilogiikan testaaminen oli sujuvaa koko kehityksen ajan.

TAULUKKO 1. Unity-pelimoottorin komponentit ja niiden käyttötarkoitus pelissä.

Komponentti	Käyttötarkoitus pelissä
SpriteRenderer	Pelin visuaaliset elementit
Transform	Nappuloiden paikat ja liike
Canvas	Käyttöliittymä
AudioSource	Pop-O-Matic ääni
Coroutine	Animaatiot ja logiikan ajastus

3.2 Visual Studio ja C#-ohjelmointikieli

Visual Studio on Microsoftin kehittämä integroitu ohjelmointiympäristö (IDE), jota käytetään laajasti ohjelmointiin. Unity integroituu Visual Studioon oletuksena, ja Unity-projektiin luodut C#-skriptit avautuvat automaattisesti siinä muokattaviksi. Visual Studio tarjoaa peliohjelmointiin erityisen sopivan työskentely-ympäristön, sillä sen koodieditori on selkeä ja tehokas, virheiden ja varoitusten raportointi tapahtuu reaaliaikaisesti, ja työkalussa on sisäänrakennettu debugger, joka helpottaa virheiden paikantamista ja korjaamista (kuva 3). Lisäksi Visual Studio sisältää Unityä varten suunniteltuja laajennuksia, jotka parantavat editorin ja pelimoottorin välistä yhteistoimintaa.



KUVA 3. Visual Studion koodieditori. Koodin värikorostus ja virheilmoitukset tukevat tehokasta ohjelmointia.

Kehitystyön aikana Visual Studio toimi ensisijaisena ohjelmointiympäristönä, jossa toteutettiin pelin logiikka, kuten nappuloiden liike, pelivuorojen hallinta ja käyttöliittymän toiminnallisuus. Ympäristön tarjoamat työkalut tukivat sujuvaa ja virheet minimoinutta kehitysprosessia, mikä oli tärkeää erityisesti silloin, kun pelin eri järjestelmien tuli toimia keskenään johdonmukaisesti.

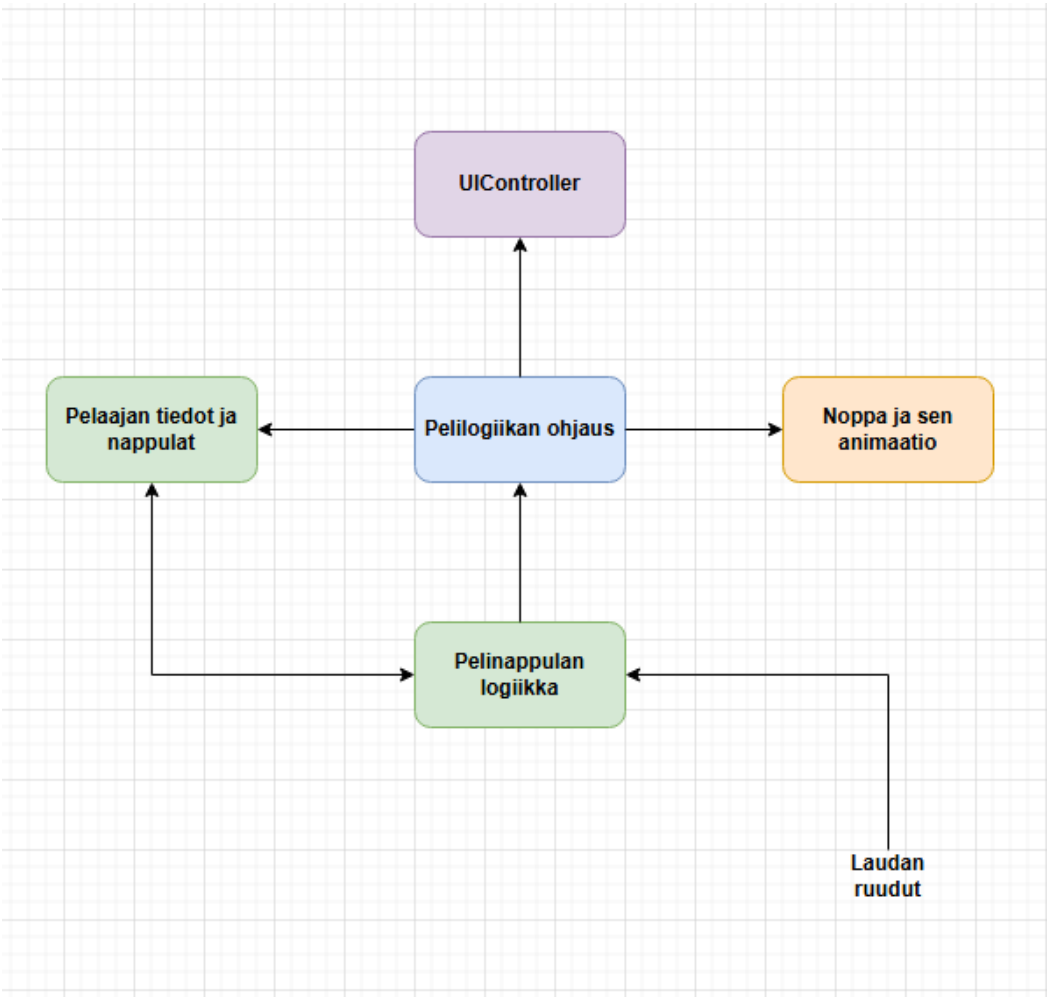
C# on olio-ohjelmointikieli, jota Unity käyttää ensisijaisena skriptikielenään. Sen selkeä syntaksi ja vahva tyyppijärjestelmä tekevät siitä hyvin soveltuvan peliohjelmointiin. Unityssä C#-skriptit liitetään GameObject-olioihin komponenteiksi, ja niiden kautta määritellään pelihahmojen käyttäytyminen, pelilogiikka ja eri järjestelmien vuorovaikutus. Kielen ominaisuudet, kuten luokkien ja olioiden käyttö pelitilojen ylläpitämiseen sekä rajapintojen ja perinnän tarjoamat mahdollisuudet modulaarisen logiikan rakentamiseen, tukevat laajojen ja johdonmukaisten pelijärjestelmien kehittämistä.

Unity hyödyntää C#:n coroutine-toiminnallisuutta, jonka avulla voidaan toteuttaa ajastettuja tapahtumia ja animaatioita ilman erillisiä säikeitä. Lisäksi kieli soveltuu tapahtumapohjaisten rakenteiden toteuttamiseen, mikä on erityisen hyödyllistä esimerkiksi pelaajan syötteiden käsittelyssä ja pelitilan muutoksissa. Kimblen digitalisoinnissa C#-koodilla rakennettiin pelin keskeiset osa-alueet, kuten vuorojen eteneminen, nappuloiden liikkeen laskenta, törmäysten eli “syömisen” käsittely, nopan toiminta sekä käyttöliittymän päivitykset. Näiden järjestelmien yhteistoiminta muodostui pelin tekniseksi rungoksi, jonka varaan koko digitaalinen pelikokemus rakentui.

4 DIGITALISOINNIN SUUNNITELMA

Ennen varsinaista toteutusta oli tärkeää laatia selkeä suunnitelma siitä, miten fyysinen Kimble-lautapeli voidaan siirtää digitaaliseen muotoon siten, että pelin ydinmekaniikat ja pelituntuma säilyvät mahdollisimman uskollisina alkuperäiselle kokemukselle. Suunnitteluvaiheessa keskityttiin hahmottamaan pelin rakenteelliset ja tekniset tarpeet sekä jakamaan kokonaisuus hallittaviin osa-alueisiin. Lautapelien digitalisoinnissa keskeistä on ymmärtää pelin mekaniikat, sääntöjärjestelmä ja pelaajan toiminnan vaikutukset pelimaailmaan, sillä nämä elementit muodostavat pelin logiikan perustan (Salen & Zimmerman 2004, 57–65). Tämän vuoksi Kimblen toimintaketju purettiin suunnittelussa vaiheisiin, jotka voidaan toteuttaa digitaalisesti samalla tavalla kuin fyysisessä pelissä.

Suunnittelussa määriteltiin ensin pelin ydinsilmukka: pelaaja heittää noppaa, liikuttaa nappulaansa sääntöjen mukaisesti ja vuoro siirtyy seuraavalle. Tätä seuraa säännöllisesti toistuva logiikka, joka on digitaalisen pelin toiminnan kannalta keskeinen. Ydinsilmukan tunnistaminen auttoi varmistamaan, että pelin tekninen rakenne pysyy koherenttina ja että kaikki toiminnot tukevat samaa kokonaisuutta. (Schell 2020, 121–123.) Samalla päätettiin, että peli toteutetaan täysin Kimblen virallisten sääntöjen mukaisesti, jotta pelaajille säilyy tuttu pelikokemus.



KUVIO 1. Pelin arkkitehtuurikaavio. Kuviossa näkyy digitaalisen Kimble-pelin keskeisten komponenttien suhde toisiinsa. GameManager koordinoi pelin logiikkaa ja kommunikoi sekä UI-järjestelmien, pelaajien, heidän nappuloidensa, että nopan kanssa.

Pelilaudan ja nappuloiden rakenteen suunnittelu keskittyi siihen, miten fyysisen laudan eri osat voidaan kuvata digitaalisina elementteinä. Jokaiselle pelaajalle määriteltiin kotipesä, lähtöruutu, pelirata sekä maali, jotka muodostivat nappuloiden liikeradan alusta loppuun. Tällä tavalla luotiin selkeä ja yksiselitteinen reitti jokaiselle pelinappulalle, mikä helpotti myöhemmin siirtologiikan ja syöntilogiikan ohjelmointia. Suunnittelussa huomioitiin myös pelilaudan pyörähtäminen vuorossa olevan pelaajan näkökulmaan sopivaksi, mikä tukee pelin visuaalista selkeyttä ja parantaa käyttäjäkokemusta.

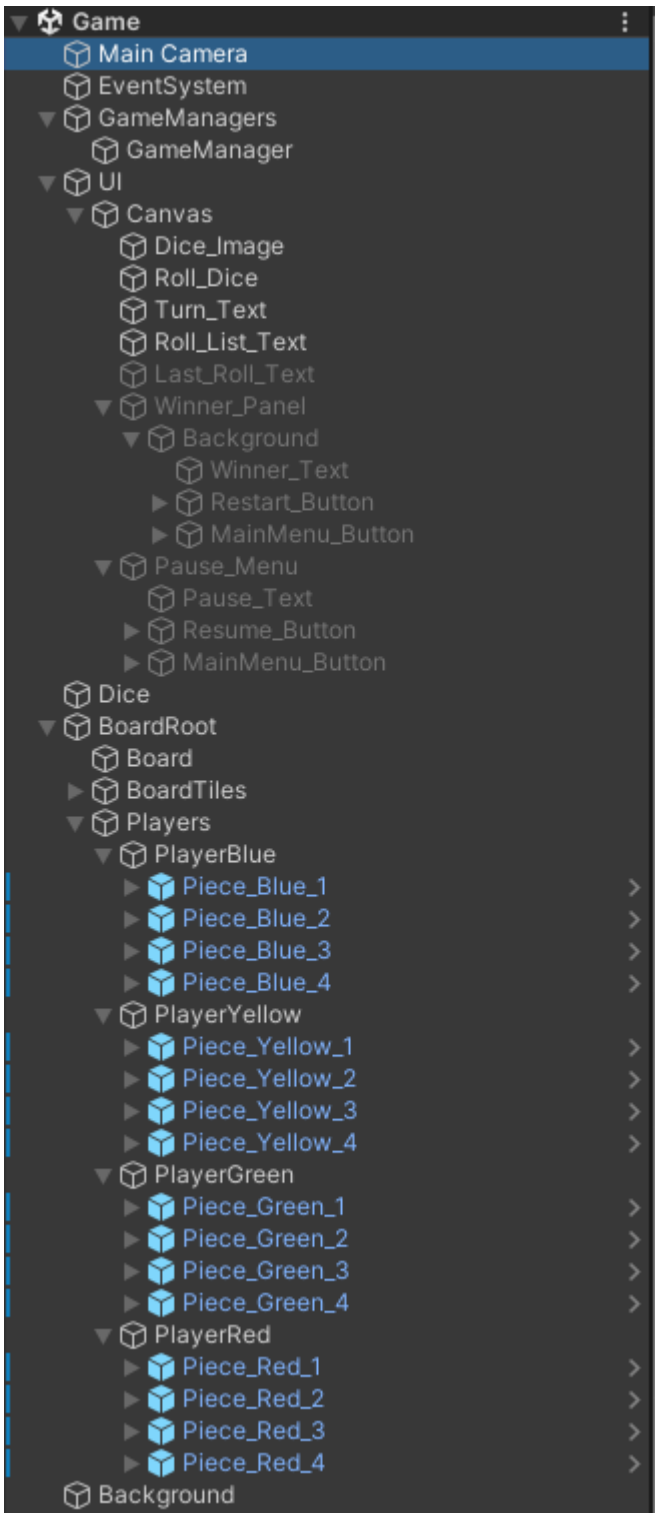
Pop-O-Matic-mekanismin suunnittelu oli yksi digitaalisen version erityispiirteistä. Fyysisessä Kimble-pelissä kupua painamalla syntyy tunnistettava ääni ja nopan

liike. Digitaalisessa versiossa tavoitteena oli luoda vastaava visuaalinen ja auditiivinen vaikutelma hyödyntämällä animaatiota, satunnaislukugeneraattoria ja äänitehosteita. Tämä oli tärkeää, jotta pelin ikoninen tuntuma säilyisi ja pelaajalle muodostuisi sama selkeä syy–seuraussuhde kuin fyysistä Pop-O-Maticia käytettäessä.

Käyttöliittymä suunniteltiin tukemaan pelin kulkua selkeästi ja intuitiivisesti. Käyttäjälle näytetään vuorossa oleva pelaaja, nopan tulos, liikutettavat nappulat sekä pelin lopputulokset. UI-elementit sijoitettiin siten, että ne eivät peitä pelialuetta mutta ovat jatkuvasti nähtävissä ilman, että pelaajan täytyy erikseen etsiä niitä. Suunnittelun lähtökohtana oli yksinkertainen ja johdonmukainen ulkoasu, joka prototyyppitasolla korostaa pelin toiminnallisuuksia grafiikan sijaan.

Projektin laajuuden määrittely oli tärkeä osa suunnittelua. Pelistä päätettiin tehdä paikallinen moninpeli 2–4 pelaajalle, mikä supisti toteutusta merkittävästi ja teki projektista hallittavan. Tekoälyvastustajien tekeminen sekä verkossa toimiva moninpeli jätettiin suunnitelman ulkopuolelle, koska ne olisivat vaatineet huomattavasti laajemman teknisen toteutuksen ja haastaneet projektin aikataulua. Rajaukset auttoivat pitämään projektin realistisena, sillä yhden kehittäjän resurssit riittivät hyvin prototyypin toteuttamiseen.

Kaiken suunnittelun perustaksi valittiin iteratiivinen kehitysmalli, jota käytetään laajasti pelialalla. Sen avulla peliä voitiin rakentaa pienissä osissa siten, että jokainen osa testattiin ja korjattiin ennen seuraavaan vaiheeseen siirtymistä. Iteratiivinen tapa edisti pelin vakauden varmistamista ja teki testaamisen sekä virheiden paikantamisen helpoksi, mikä oli tärkeää erityisesti teknisesti monivaiheisen logiikan, kuten nappuloiden liikkumisen ja syöntilogiikan, kehityksessä.



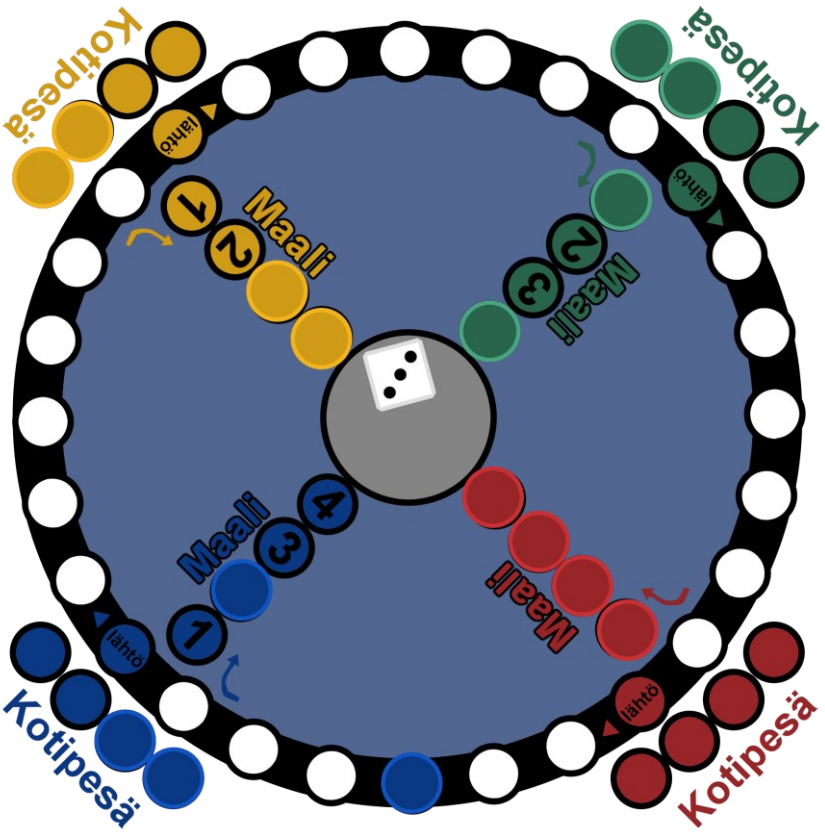
KUVA 4. Unityn inspectorin hierarkia. Kuvassa näkyvät komponentit, joita käytettiin digitaalisen Kimblen luomiseen.

5 DIGITALISOINNIN TOTEUTUS

5.1 Pelilaudan luominen ja Pop-O-Matic

Pelilaudan rakentaminen aloitettiin määrittelemällä digitaalisen ympäristön perusrakenne Unity-pelimoottorissa. Fyysisessä Kimble-laudassa pelialue koostuu kehästä, lähtöruuduista, kotipesistä ja maaliympyröistä, jotka kaikki täytyi mallintaa selkeästi myös digitaalisessa versiossa. Projektin alkuvaiheessa pelilaudasta oli suunnitteilla kolmiulotteinen malli, mutta kehityksen edetessä huomattiin, että kolmiulotteisen pelilaudan, nappuloiden ja Pop-O-Maticin mallintaminen olisi liian monimutkaista. Kaksiulotteinen toteutus myös tukee paremmin pelin selkeää luettavuutta ja vähentää teknistä monimutkaisuutta. 2D-toteutuksen etu oli myös se, että pelaajien pelinappulat ja niiden reitit voitiin esittää täsmällisesti numeroituna polkuna, mikä vähensi virhemahdollisuuksia pelilogiikan toteutuksessa.

Kaikki pelin spritet ja muut grafiikkaelementit piirrettiin itse (kuva 5). Tämä mahdollisti visuaalisen tyylin yhdenmukaisuuden sekä sen, että kaikki elementit voitiin rakentaa erityisesti digitaalista versiota varten.



KUVA 5. Itse piirretty digitaalinen Kimble-lauta (Joa Lamminluoto 2025). 2D-versio pelilaudasta, jossa punainen pelaaja on voittanut pelin.

Pelilaudan visuaaliset elementit rakennettiin Unityn sprite-pohjaisella 2D-grafiikalla. Jokainen nappulan kulkema askel määriteltiin ruutuna, ja ruudut asetettiin Unityn editorissa oikeille paikoille pelilaudan kuvan päälle. Tämän jälkeen jokaiselle pelinappulalle määritettiin yksilöllinen reitti, joka kulki täsmälleen niiden ruutujen läpi, joita vastaava oikea Kimble-pelinappula käyttää. Lisäksi kullekin värille asetettiin kotipesäalue, joka toimi lähtöpisteenä ja johon nappula palautettiin, mikäli vastustaja "söi" sen pelilaudalla.

Laudan rakentamiseen sisältyi myös kameran ja pelin rakenteen suunnittelu siten, että pelaajan vuoron alussa pelilauta kääntyy kohti aktiivista pelaajaa. Tämä visuaalinen ratkaisu paransi pelin selkeyttä, koska vuorossa oleva pelaaja näkee oman lähtöalueensa ja nappulansa aina ruudun vasemmassa alareunassa. Kierron toteuttaminen vaati sen huomioimista, että animaatiot ja nappuloiden liikkeet eivät saa häiriintyä laudan liikkeen aikana, joten laudan käännöt ajoitettiin aina pelin logiikan kannalta turvalliseen hetkeen.

Kimblen ehkä ikonisin osa on pelilaudan keskellä sijaitseva Pop-O-Matic: muovikupu, jonka sisällä oleva noppamekanismi tuottaa tunnistettavan “kutunk”-äänen (kuva 6). Digitaalisessa versiossa tavoitteena ei ollut mallintaa tätä rakennetta fyysisesti, vaan tuottaa sen toiminnallinen ja auditiivinen kokemus mahdollisimman uskollisesti.

KUVA 6. Oikean Kimble-laudan Pop-O-Matic-kupu. Kupu, jonka sisällä oleva mekanismi pompauttaa noppaa äänen saattelemana.

Pop-O-Maticin digitaalinen vastine toteutettiin sprite-kuvakkeita sekä pientä koonvaihtoon perustuvaa animaatiota hyödyntämällä. Kun pelaaja painaa heittopainiketta laudan keskellä, noppaa vastaava UI-elementti suorittaa nopean “pomppuanimaation”, joka simuloi kupua painettaessa syntyvää liike-energiaa. Lisäksi noppakuvake siirtyy satunnaiseen kohtaan kuvun sisällä ja pyörähtää satunnaisen määrän asteita, jotta heitto saadaan näyttämään elävämmältä. Tämän visuaalisen vaikutelman tueksi lisättiin myös ääniefekti, joka vastaa Pop-O-Maticin tunnusomaista ääntä. Ääni on otettu aidosta Kimble-pelilaudalta löytyvästä Pop-O-Maticista. Koska peli käyttää vain tätä yhtä ääniefektiä, ei ollut tarpeen toteuttaa monimutkaista äänijärjestelmää.

Digitaalisen Pop-O-Maticin suunnittelussa keskeistä oli ajastusten ja estotilojen hallinta. Pelaajan ei tule voida painaa heittopainiketta uudelleen ennen kuin animaatio on kokonaan päättynyt, jotta vältetään virheelliset tuplaklikkaukset ja pelin

tilan sekoittuminen. Tämä toteutettiin rajaamalla vuorovaikutus mahdolliseksi vasta, kun animaatio ja nopan tuloksen laskenta oli valmis. Yhdessä nämä visuaaliset ja tekniset ratkaisut loivat uskottavan ja selkeän digitaalisen vastineen alkuperäisen pelin Pop-O-Matic-mekanismille. Kokonaisuudessaan Pop-O-Maticin digitaalisessa toteutuksessa pyrittiin pitämään se mahdollisimman lähellä alkuperäistä pelituntumaa, jotta pelaajien kokemus vastaisi mahdollisimman paljon fyysisen pelin luomaa odotusarvoa.

5.2 Laudalla liikkuminen

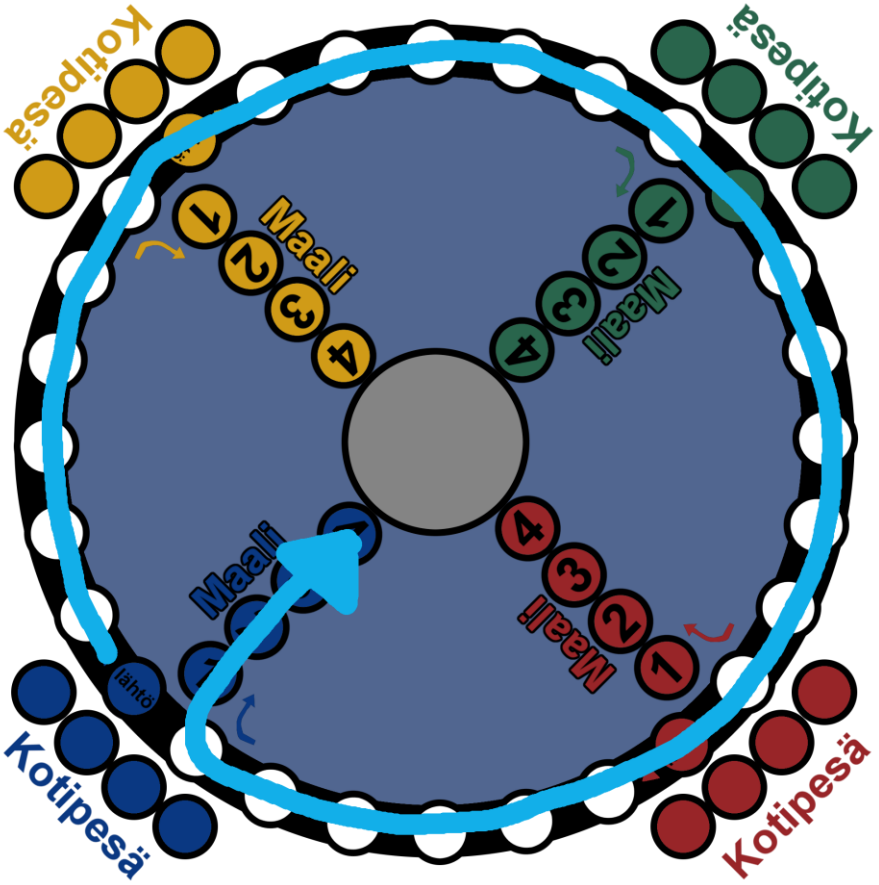
Pelissä toteutettiin Kimblen virallisten sääntöjen perusteella liikkumislogiikka, jonka mukaan pelaaja voi siirtää nappulaansa nopan silmäluvun verran eteenpäin ja siirtää nappulan kotipesästä lähtöruutuun ainoastaan heittämällä kuutosen. Liikkuminen toteutettiin tarkasti sääntöjen mukaan siten, että nappula ei voi ohittaa maaliaan eikä se voi siirtyä ruutuun, jossa on jo pelaajan oma nappula. Vastustajan nappulan päälle sen sijaan voi siirtyä, jolloin vastustajan nappula palautetaan kotipesäänsä. Digitaalisen pelin kannalta oli olennaista rakentaa logiikka, joka seuraa tarkasti jokaisen nappulan sijaintia ja varmistaa, että kaikki mahdolliset siirrot ovat sääntöjen mukaisia.

Liikkuminen toteutettiin yhdistämällä reittipohjainen järjestelmä ja animaatiot. Jokaiselle nappulalle määriteltiin ennalta määrätty polku, jota pitkin se voi edetä kohti maaliaan (kuva 7). Kun pelaaja valitsi siirrettävän nappulan, peli tarkisti, voiko nappula liikkua annettua silmälukua vastaavan määrän askeleita eteenpäin. Siirtojen aikana nappuloille toteutettiin yksinkertainen animaatio, jossa nappula liikkui ruudulta toiselle sulavasti.

TAULUKKO 2. Kimblen sääntöjen digitaalinen toteutus Unityssä.

Kimblen sääntö	Digitaalinen toteutus
6 → uusi heitto	GameManager asettaa ”waitingForRoll = true” uudestaan
Vastustajan syönti	Piece.cs → SendToHomeAnimated()

Ei siirrettäviä nappuloita	Pelivuoro vaihtuu tauon jälkeen
Nappula ei voi mennä samanvärisen nappulan päälle	GameManager.IsTileOccupiedByOwnPiece = true
Kotipesästä lähtöön vaaditaan 6	Piece.CanMove() -metodi



KUVA 7. Liikerata Kimble-laudalla. Kuvassa esitetään nappulan kulkema reitti.

Erityistä huomiota vaati Pop-O-Matic-efektin ja napin painamisen yhdistäminen liikkumiseen. Pelissä estettiin noppaa painamasta uudelleen ennen kuin nappulan liikkumisanimaatio oli täysin valmis, mikä ehkäisi odottamattomia logiikkavirheitä. Lisäksi toteutettiin tarkistus siihen, että vastustajan nappula lähetetään takaisin omaan kotipesäänsä vasta nappulan liikkeen päätyttyä, mikä sai peliin tuntua siitä, että vastustajan nappula oli oikeasti ”syöty”. Jotta laudankääntöefekti ei häirinnyt kotipesäänpalautuslogiikkaa, palautus animaatio odotettiin loppuun ennen vuoron vaihtoa. Kokonaisuutena liikkumisen toteutus yhdistää pelilogiikan, sääntöjen tulkinnan ja animaatiot saumattomaksi osaksi pelin kulkua.

5.3 Pelaajien syötteen lukeminen

Pelaajien syötteen lukeminen toteutettiin hiirtä käyttäen, mikä oli tarkoituksenmukaista pelin ollessa suunniteltu samalta laitteelta pelattavaksi. Pelissä kaikki vuoroon liittyvät toiminnot perustuvat siihen, että pelaaja klikkaa hiirellä noppaa heittääkseensä sitä tai valitsee nappulan, jota siirtää. Syötteiden käsittelyssä keskeistä oli varmistaa, että peli hyväksyy syötteen vain oikeaan aikaan: noppaa voi painaa vain, kun vuoro on aktiivinen, ja nappulaa voi klikata vain, kun liikkuminen on sääntöjen mukaan sallittua.

Syötteenhallinta rajattiin niin, että peli ei rekisteröi useita nopanpainalluksia peräkkäin, mikä olisi voinut aiheuttaa odottamattomia muutoksia pelitilaan. Samoin toteutettiin valintalogiikka, joka sallii nappulan klikkauksen vain, jos se on siirrettävissä kyseisellä heittotuloksella. Tämä selkeytti pelin käyttöä ja esti virheitä, kuten tilanteita, joissa pelaaja yrittäisi siirtää nappulaa, joka ei voi liikkua.

Pelissä toteutettiin myös yksinkertainen vuoronvaihtologiikka, joka reagoi automaattisesti tilanteisiin, joissa pelaaja ei voi siirtää yhtään nappulaansa. Tällöin peli ilmoittaa, ettei siirrettäviä nappuloita ole, ja siirtyy seuraavan pelaajan vuoroon joko odotusajan jälkeen tai heti, jos pelaaja klikkaa hiirellä ohittaakseen viestin. Tämä paransi pelin sujuvuutta ja teki pelikokemuksesta selkeämmän ilman, että pelaajan tarvitsisi itse tehdä ylimääräisiä valintoja.

5.4 Käyttöliittymä

Käyttöliittymä suunniteltiin selkeäksi ja mahdollistamaan perinteisen Kimblen pelitilanteen kulun seurantaa. Pelin keskeiset visuaaliset elementit, kuten pelaajien vuoron näyttö sekä viimeisin heittotulos, sijoitettiin siten, että ne näkyvät selkeästi riippumatta pelin laudan pyörähtämisestä. Käyttöliittymä toteutettiin Unityn omalla UI-järjestelmällä, ja sen elementit määriteltiin toimimaan riippumatta kameran liikkeistä, jotta pelaaja ei kokisi, että UI liikkuu pelilaudan mukana.

Käyttöliittymässä näkyvät viestit liittyvät erityisesti vuoroon ja heittoihin. Pelaajalle kerrotaan, kuka on vuorossa, sekä nopan heiton tulos. Viesti ”Ei siirrettäviä nappuloita” pysyy ruudulla hetken ja voidaan haluttaessa ohittaa hiirtä klikkaamalla. Tämä paransi pelin käytettävyyttä, koska pelaaja pystyy vaikuttamaan pelin etenemisnopeuteen. Lisäksi voittoikkuna suunniteltiin yksinkertaiseksi: pelin päätyttyä näkyviin tulee ilmoitus voittajasta sekä painikeet, joilla voidaan palata päävalikkoon tai aloittaa uusi peli.

Pelin käyttöliittymässä pyrittiin siihen, että se ei häiritse pelin kulkua, vaan tukee sitä. Kaikki tekstielementit pidettiin lyhyinä ja selkeinä. Suurin osa UI-elementeistä, kuten fontit ja taustakuviot, suunniteltiin yksinkertaisiksi, jotta ne eivät veisi huomiota itse pelimekaniikoilta.

6 POHDINTA

Opinnäytetyön tavoitteena oli toteuttaa digitaalinen versio Kimble-lautapelistä Unity-pelimoottoria käyttäen ja samalla tutkia, miten perinteisen lautapelin mekaniikat voidaan siirtää toimivaksi digitaaliseksi kokonaisuudeksi. Kokonaisuutta tarkastellessa voidaan todeta, että projekti saavutti sille asetetut tavoitteet. Valmis peli toimii Kimblen virallisten sääntöjen mukaisesti, ja se tarjoaa 2–4 pelaajalle mahdollisuuden pelata peliä samalta laitteelta. Pelimekaniikat, käyttöliittymä ja pelin vuorologiikka muodostavat yhdessä selkeän ja toimivan kokonaisuuden.

Projektin aikana keskeiseksi havaittiin selkeä rakennesuunnitelma. Koska peli koostui useista erillisistä, mutta toisiinsa vaikuttavista järjestelmistä – kuten nopan heittämisestä, nappuloiden liikkumisesta, törmäysten käsittelystä ja käyttöliittymästä – oli olennaista jakaa projekti vaiheisiin ja testata jokainen osa ennen seuraavaan siirtymistä. Iteratiivinen kehitystapa osoittautui toimivaksi ratkaisuksi, sillä se auttoi havaitsemaan ja korjaamaan virheitä jo varhaisessa vaiheessa.

Teknisesti haastavimmaksi osoittautui pelin liikkumis- ja syöntilogiikan toteuttaminen animaatioiden kanssa yhteensopivalla tavalla. Pelissä tuli varmistaa, että nappulat eivät päätyneet väärille paikoille, vaikka kamera tai pelilauta olisi samaan aikaan liikkunut. Samoin tuli huolehtia siitä, että pelaaja ei voinut painaa noppaa kesken liikkeen, mikä olisi voinut rikkoa pelitilan. Näiden ongelmien ratkaiseminen lisäsi ymmärrystä siitä, kuinka tärkeää pelinkehityksessä on aikajärjestyksen ja tilamuutosten huolellinen hallinta.

Projekti oli hyödyllinen kokemus myös pelisuunnittelun näkökulmasta. Lautapelin digitalisointi ei ole pelkkää sääntöjen siirtämistä ohjelmakoodiin, vaan siihen sisältyy päätöksiä pelin selkeydestä, visuaalisesta ilmeestä ja käyttäjäkokemuksesta. Esimerkiksi animaatioiden lisääminen ei ollut sääntöjen kannalta pakollista, mutta ne paransivat pelin ymmärrettävyyttä ja tekivät siitä miellyttävämmän pelata. Myös UI:n selkeys vaikutti suoraan pelin sujuvuuteen.

Projektin rajaukset osoittautuivat toimiviksi. 3D-toteutusta ei lopulta voitu toteuttaa, mikä olisi pitänyt tunnistaa jo suunnitteluvaiheessa, mutta valittu 2D-toteutus

osoittautui toimivaksi ratkaisuksi. Peli toteutettiin ilman tekoälyvastustajia ja ilman verkkomoninpeli tukea, mikä piti työn laajuuden hallittavana. Nämä ominaisuudet voisivat olla luontevia jatkokehityskohteita, jos peliä haluttaisiin laajentaa tulevaisuudessa. Muita mahdollisia laajennuksia olisivat äänimaailman kehittäminen, käyttäjälle tarjottavat erilliset teemat tai visuaaliset tyylit sekä valmiiden pelien tallentaminen ja jatkaminen myöhemmin.

Kokonaisuutena projekti vahvisti ymmärrystäni Unity-pelinkehityksestä sekä erityisesti siitä, miten pelilogiikka ja käyttöliittymä tulee suunnitella toimimaan johdonmukaisesti keskenään. Työ osoitti, että jopa suhteellisen yksinkertaiselta vaikuttava lautapeli sisältää monimutkaisia vuorovaikutustilanteita, jotka vaativat huolellista toteuttamista digitaalisessa ympäristössä. Lopputuotos vastaa asetettuja tavoitteita ja toimii hyvänä esimerkkinä yksinkertaisen lautapelin digitalisoinnista.

Tässä linkki GitHub-repositorioon, josta voitte itse ladata ja pelata tekemääni digitaalista Kimbleä: KimbleInUnity.

LÄHTEET

Kimblen viralliset säännöt. Verkkosivu. Viitattu 5.11.2025. https://www.kimble.fi/wordpress/index.php/saannot/

Microsoft. 2023. C# documentation. Viitattu 8.11.2025. https://learn.microsoft.com/fi-fi/dotnet/csharp/

Salen, K. & Zimmerman, E. 2004. Rules of Play: Game Design Fundamentals. MIT Press.

Schell, J. 2020. The Art of Game Design: A Book of Lenses. CRC Press.

Unity Technologies. 2025. Unity Manual & Documentation. Viitattu 7.11.2025. https://docs.unity3d.com/Manual/UnityManual.html

Wikipedia. 2025. ”Kimble.” Luettu 7.11.2025. https://fi.wikipedia.org/wiki/Kimble