TALLINNA TEHNIKAÜLIKOOL
INSENERITEADUSKOND
Instituudi nimetus

Animatsioooni loomine Unity mängumootoris

Creating animation in Unity's game engine

TELEMAATIKA JA ARUKATE SÜSTEEMIDE ÕPPEKAVA LÕPUTÖÖ

Üliõpilane: Igor Sabenin

Üliõpilaskood: 178483

Juhendaja: Natalja Maksimova lektor

Kohtla-Järve, 2022

AUTORIDEKLARATSIOON

Olen koostanud lõputöö iseseisvalt.

Lõputöö alusel ei ole varem kutse- või teaduskraadi või inseneridiplomit taotletud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

"....." ..... 20…..

Autor: Igor Sabenin

/ allkiri /

Töö vastab bakalaureusetöö/magistritööle esitatud nõuetele

"....." ..... 20…..

Juhendaja: Natalja Maksimova

/ allkiri /

Kaitsmisele lubatud

"....."..... 20… .

Kaitsmiskomisjoni esimees .....

/ nimi ja allkiri /

LIHTLITSENTS LÕPUTÖÖ ÜLDSUSELE KÄTTESAADAVAKS TEGEMISEKS JA REPRODUTSEERIMISEKS

Mina Igor Sabenin

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose Virumaa Kolledži publikatsioonide veebirakenduse loomine, mille juhendaja on Natalja Maksimova,

1.1. reprodutseerimiseks säilitamise ja elektroonilise avaldamise eesmärgil, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. Olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.

3. Kinnitan, et lihtlitsentsi andmisega ei rikuta kolmandate isikute intellektuaalomandi ega isikuandmete kaitse seadusest ja teistest õigusaktidest tulenevaid õigusi.

TalTech Inseneriteaduskond Virumaa kolledž

LÕPUTÖÖ ÜLESANNE

Üliõpilane: Igor Sabenin, 178483

Õppekava, peaeriala: EDTR17/17 - Telemaatika ja arukad süsteemid

Juhendaja(d): Lektor, Natalja Maksimova, natalja.maksimova@taltech.ee

Lõputöö teema:

(eesti keeles) Animatsiooni loomine Unity`s mängumotoris

(inglise keeles) Creating animation in Unity's game engine

Lõputöö põhieesmärgid:

1. 2D mängu "Soul never die" tegelaste animeerimine Unity mängumootoris,
2. luua interaktiivne menüü 2D mängu jaoks.

Lõputöö etapid ja ajakava:

Nr	Ülesande kirjeldus	Tähtaeg
1.	2D animatsioonide loomine mängu tegelastele Unity mängumootoris	15.02.2021
2.	Mängu "Soul never die" mängutegevuste (operatiivsed ja resultatiivsed, mitme objektidega rakendamine) kirjeldamine ja sobiva animatsiooni määramine	01.03.2021
3.	Mängu tegelastele koodi kirjutamine ning nende kontrollimine	15.03.2021
4.	Metoodika kirjeldamine	29.03.2021
5.	Mängu kasutajaliide loomine (stardi menu, dialoogid, ...)	12.04.2021
4.	Lõputöö kirjaliku osa vormistamine	23.05.2022

Töö keel: eesti keel **Lõputöö esitamise tähtaeg:** "03" juuni 20.....a

Üliõpilane: Igor Sabenin. "....."..... 20.....a

/allkiri/

Juhendaja: Natalja Maksimova..... "....."..... 20.....a

/allkiri/

Programmijuht: Zanna Gratsjova..... "....."..... 20.....a

/allkiri/

SISUKORD

EESSÕNA

Bakalaureusetöö on valminud soovist luua mäng „Soul never die". Antud videomäng ei ole ainult autori vaid grupitöö tulemus. Õppejõu Natalja Maksimova algatusel sõnastati antut lõputöö teema. Vajalikud andmed koguti teadusallikatest, artiklitest ja veebilehtedelt.

Antud töö eesmärgiks oli luua interaktiivne menüü ja animeerida tegelasi 2D videomängu jaoks. Lisaks koostas autor õppematerjalid, kus on põhjalik juhendid, kuidas luua animatioon Unity mängumootoris. Antud õppematerjal on loodud aine „Multimeedia" jaoks. Bakalaureusetöö koosneb metoodilisest ja praktilisest osast. Metoodilises osas antakse ülevaade animatsioonist ning Unity mängumootori võimalustest selle loomisel. Praktilises osas kirjeldab autor, mis olid tema ülesanded „Soul never die" mängu loomisel ning kuidas õppematerjalid koostati.

Autor soovib tänada oma juhendajat, Natalja Maksimovat, kes toetas ja abistas teda töö valmimise jooksul.

Märksõnad: Unity, 2D animatsioon, arvutimäng, videomängud, skelet animatsioon, sprite animatsioon

SISSEJUHATUS

Videomängud hõivavad tänapäeval väga kõrge niši. Need on muutunud nii populaarseteks, et on saanud juba spordialaks. Ameerika turu-uuringufirma NPD Groupi uuringu järgi tõusid 2020. aasta seisuga videomängude ja videomängu seadmete maksumus 2019. aastaga võrreldes 27% [1]. Samuti, kui võrrelda mängijate arvu 2015. aastal ja 2021. aastal, on märgatav videomängu mängijate oluline kasv. 2015. aastal on 1,99 miljardit ja juba 2021. aastal 2,81 miljardit mängijat [2]. Nende andmete põhjal võib öelda, et videomängud on meie ajal väga populaarsed.

Paljude inimeste jaoks on mängude mängimine nagu töö, sest nad saavad teenida sama palju kui professionaalsed sportlased ja paljudel juhtudel isegi kaks korda rohkem. Näiteks on üsna tuntud mäng „Dota 2". Selle mänguga toimub suurim rahvusvaheline turniir The International suurima auhinnafondiga. The International toimub igal aastal erinevates riikides ja igal aastal lööb see oma rekordi kogutud auhinnafondi osas. 2019. aastal oli auhinnafond 34 302 501 dollarit ja juba 2020. aastal oli 40 018 195 dollarit [3]. Vaid ühe mängu näitel võib aru saada, et mängud on ammu ületanud piiri, kui need olid lihtsalt meelelahutus või vaba aja veetmise viise.

Seoses videomängude populaarsuse kasvuga on kasvanud ka ettevõtete ja ideede kasv millegi uue loomiseks. Samuti tekivad uued võimalused videomängude loomiseks, täiustatakse mängumootoreid. Mängu loomisel läbib mäng mitmeid etappe, milles arendatakse:

1. süžee;
2. tegelased;
3. tegelaste tegevused vastavalt süžeele;
4. mängumaailm;
5. keskkond ja lokatsioonid;
6. liides;
7. saatemuusika.

Tegelaste elustamise ja tegevuste dünaamika edastamise eest mängu ajal vastutab animatsioon. Sooritatavate liigutuste edastamiseks hästi loodud animatsioon suurendab positiivset suhtumist mängu, kuna mängijat ei sega tegelase halb juhtimine või reaktsiooni puudumine. 2D mängu tegelaste animeerimine sai teemaks, mida autor otsustas käsitleda. Arenduskeskkonnas tegelaste tegevuste animatsioonide loomise lihtsus ja mugavus on arendaja jaoks üks võtmepunkte.

Bakalaureusetöö tulemuseks on mäng nimega „Soul never die”. Mäng ei ole autori isiklik looming, vaid meeskonnatöö. Selle töö eesmärk oli uurida põhjalikult 2D animatsiooni loomise ja animatsiooni integreerimise võimalusi Unitys. Teine eesmärk oli interaktiivse menüü loomine ja mängu tegelaste animeerimine mängu jaoks. Samuti luua õppematerjal kursuse “Multimeedia” jaoks.

Selle eesmärgi saavutamiseks püstitati järgmised ülesanded:

1. uurida erinevaid meetodeid animatsiooni loomiseks 2D mängude jaoks ja kui lihtsalt on need realiseeritud erinevates mängumootorites;
2. tegelase juhtimine Unitys ja liikumiste animeerimine eelnevalt ettevalmistatud animatsioonide kaudu;
3. mängumenüü loomine 2D mängus.

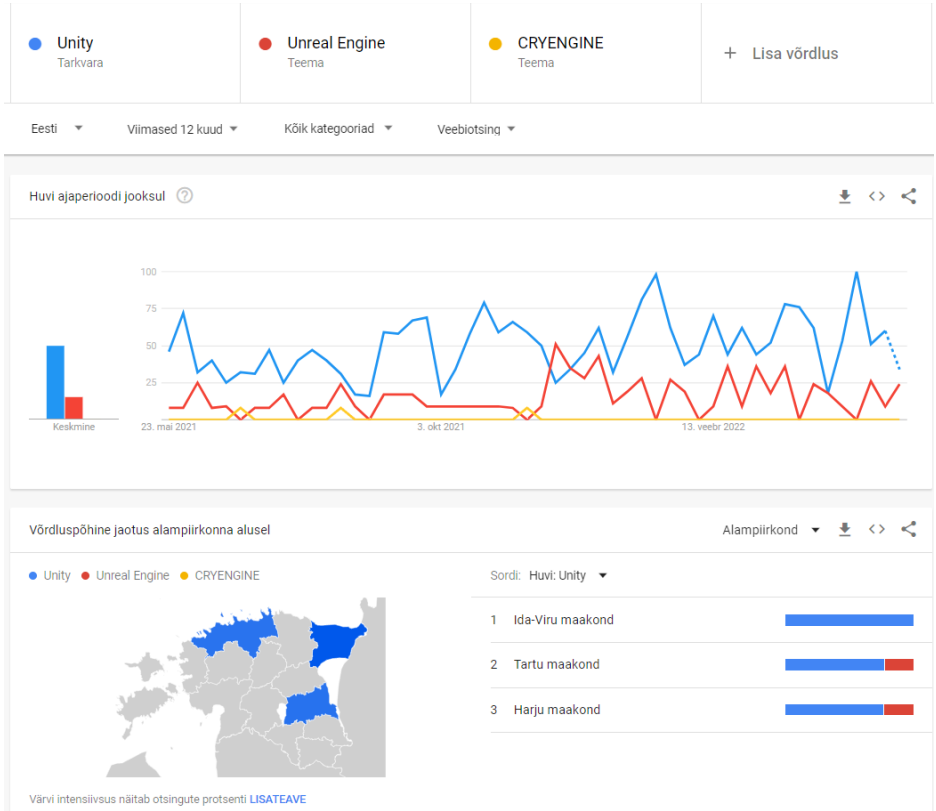
Töö kirjutamise ajal püstitati hüpotees, et teades, kuidas töötada 2D animatsiooniga ühel mootoril, saab rahulikult luua ja töötada teisel. Selle hüpoteesi kontrollimiseks vaadeldakse teist populaarset mootorit, mis on leitud realiseerimise valiku analüütilises osas. Võrdluse mõõdupuuks on tehtavate tegevuste sarnasuse protsent.

1. MÄNGU REALISEERIMISE VALIMINE

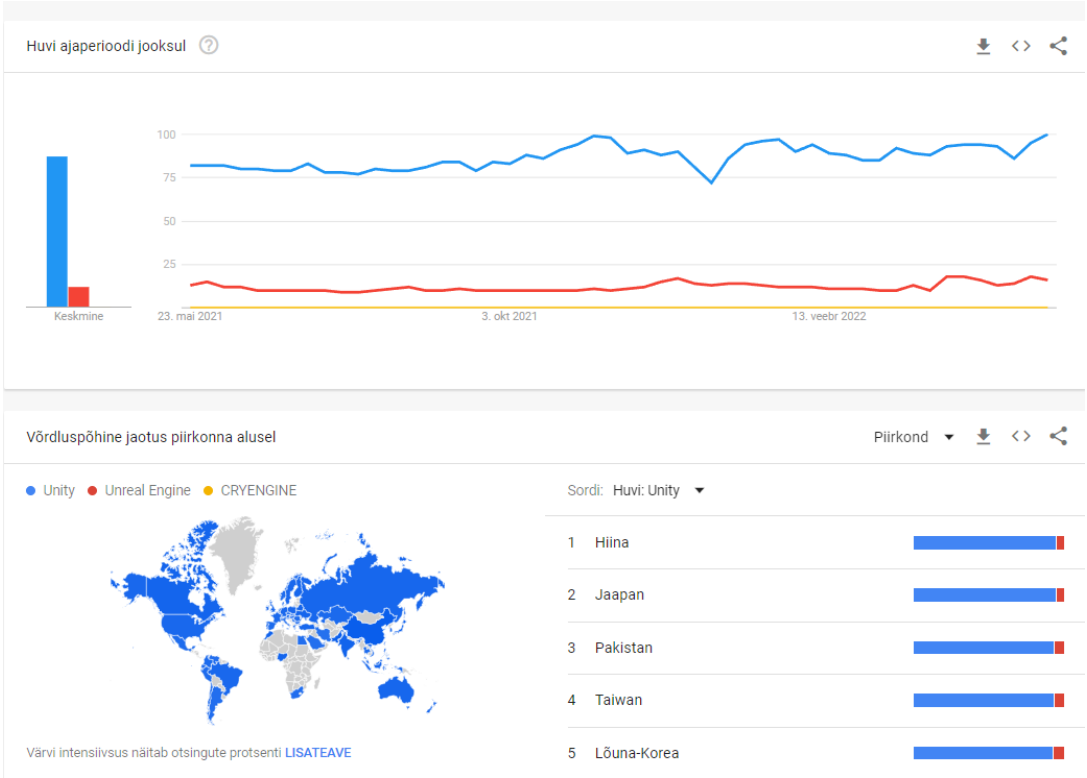
Esmalt tuli meeskonnal mängu idee täielikult läbi mõelda. Arutleti, milline tuleb tegelane, visuaalne mängustiil, žanr, süžee ning millisel mängumootoril mäng tehakse.

Mängu „Soul never die" loomiseks tuli meeskonnal otsustada graafika üle, kas luua 3D või 2D mäng. Samuti kaalus meeskond nn 2.5D loomist. 2.5D on mängugraafika liik, mis püüab imiteerida kolmemõõtmelist mänguruumi [4]. Huvitavam variant oli 2D platvormi mängu loomine, sest 2D videomängud hakkasid taastama oma endist huvi paljudes, nii noortes kui ka vanemates mängijates, kes nautisid sellist kultusmängu nagu „Mario".

Mängumootori valikul sooviti pöörata tähelepanu mitmele tegurile. Esimeseks teguriks oli mängumootori kättesaadavus. Teine oluline kriteerium oli mängumootori kasutamise lihtsus. On olemas veidi vähem populaarseid mängumootoreid, mis ei kasuta programmeerimiskeeli, mis lihtsustab väga mängu loomist. Need mängumootorid olid ainult kitsasuunalised ja ei sobinud meeskonna ideede elluviimiseks. Kokkuvõttes otsustati pärast pikka kaalumist valida Unity mootor. Google arvates on Unity üks populaarsemaid mängumootoreid mitte ainult Eestis (Joonis 1.1), vaid ka kogu maailmas (Joonis 1.2).



Joonis 1.1 Unity, Unreal Engine, CryEngine populaarsuse graafik ja võrdlusgraafik Eesti piirkondade lõikes 12 kuu jooksul [5]



Joonis 1.2 Unity, Unreal Engine, CryEngine populaarsuse graafik ja võrdlusgraafik kogu maailmas 12 kuu jooksul [6]

Graafikutelt on näha, kui palju populaarsem on Unity teistest tuntud mängumootoritest. Kuigi hetkel on Google'is päringute järgi Unity populaarsus Eestis langenud (Joonis 1.1), jääb ta siiski kõige populaarsemaks. Unity on piisavalt lihtne õppimises ja kasutamises. Kasutajaliides on intuitiivselt arusaadav ega eksita kasutajat. Sellel on palju funktsioone, mis aitavad tööd lihtsustada. Unity kasutab C# programmeerimiskeelt.

Pärast mängumootori valimist jagas meeskond ülesanded liikmete vahel. Lõputöö autor vastutas animatsiooni loomise eest Unitys.

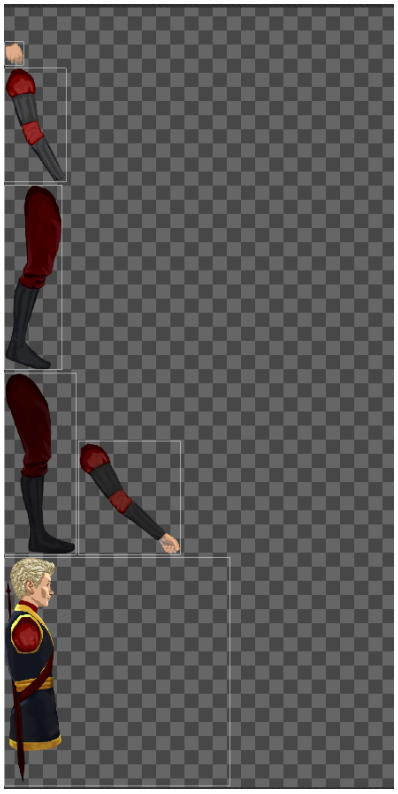
2. ANIMATSIOON UNITYS

Autori ülesandeks oli põhjalikult vaadelda erinevaid meetodeid animatsiooni loomiseks 2D mängu jaoks. Animatsioon on eelnevalt ettevalmistatud liikumiste visuaalsete efektide komplektid [7]. See võib olla mitut tüüpi, kahemõõtmeline (2D) ja kolmemõõtmeline (3D) graafika. 2D graafika on vahelduv pildivahetus. Kiiresti vahetuvad pildid loovad liikumisefekti. Kahemõõtmelises graafikas nimetatakse selliseid pilte spraitideks ja kolmemõõtmelistes tekstuurideks. 3D graafika kasutatakse laialdaselt hariduses, tervishoius, reklaamis ja kinos. laialdase kasutamise tõttu paljudes valdkondades on see kõige populaarsem. 3D graafikas juhitakse tegelast kolmemõõtmelises ruumis. [8]

Animatsiooni loomiseks on Unitys kaks meetodit. Esimene viis on animatsioon spraidi abil. Animeerimiseks on vaja 2D Sprite'i teeki. Seda saab paigaldada otse Unity sees. Uutes Unity versioonides on see teek juba paigaldatud. Spraidi animatsioon on kõige keerulisem ja selle loomine võtab kõige rohekem aega. Selle keerukus seisneb selles, et mänguobjektile peab iga üksiku kaadri jaoks spraidi joonistama (Joonis 2.1). Unitys pole võimalik spraite teha, selleks on vaja kõrvalprogrammi, näiteks Photoshopi. Kui spraidid on valmis, võib alustada animatsiooni loomist Unitys.

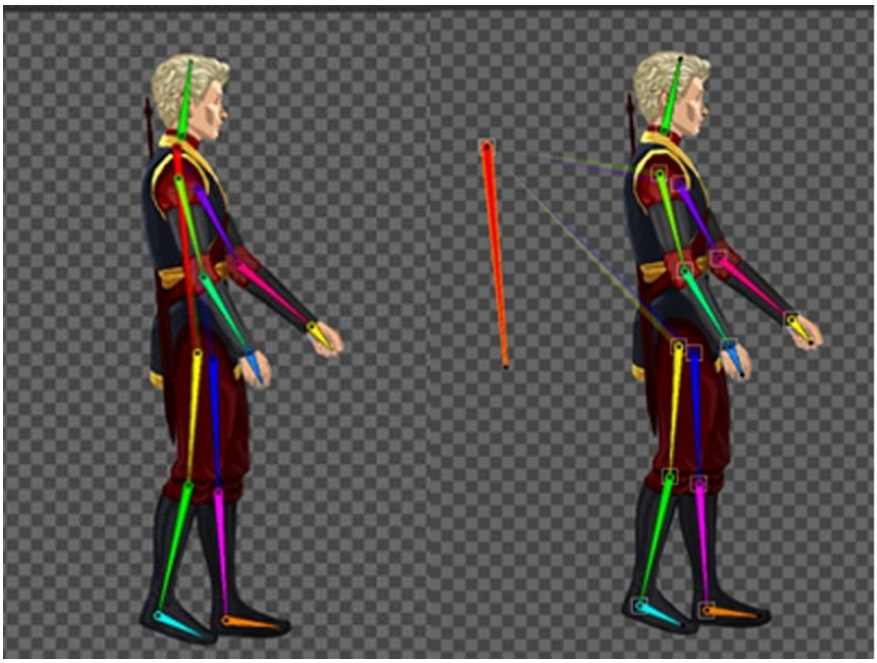
Joonis 2.1 Sprait animatsioon (joonistatud inimese autor Anna Šehovtsova)

Teine animeerimise viis on skelett animatsioon. Skelett animatsiooni loomiseks on vaja kahte teeki: 2D Animation ja 2D PSD Importer (piltide importimiseks fotoshopist). See on spraidi animatsioonist lihtsam. Seal on ka spraiti vaja, aga pole vaja igat kaadrit eraldi välja joonistada. Animatsiooni loomiseks tuleb kõigepealt jagada spraidi eraldi osadeks (Joonis 2.2).



Joonis 2.2 Sprait

Pärast seda võib asetada luud jagatud kehaosadele, et neid seejärel animeerida. Kui mõnel osal luu puudub, on seda võimatu animeerida. Luude loomise käigus tuleb need omavahel siduda, välja arvatud üksikud objektid, mis ei sõltu teistest. Näiteks tegelase keha on peamine osa ja sellel on luu, millest moodustuvad ülejäänud: käed, jalad ja pea (Joonis 2.3).



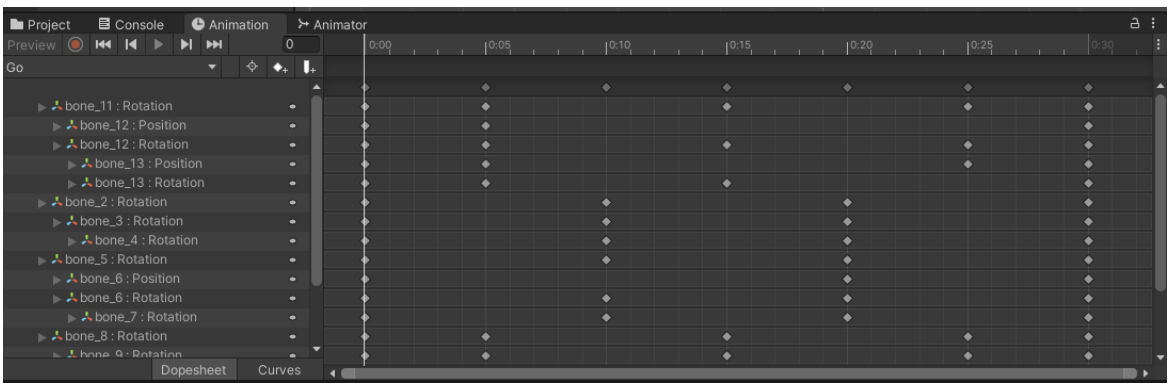
Joonis 2.3 Skelett mängutegelase jaoks

Kui luud on loodud, saab spraiti animeerida. Animeerimine toimub luude liikumise põhimõttel eraldi kaadrites.

2.1 Animation

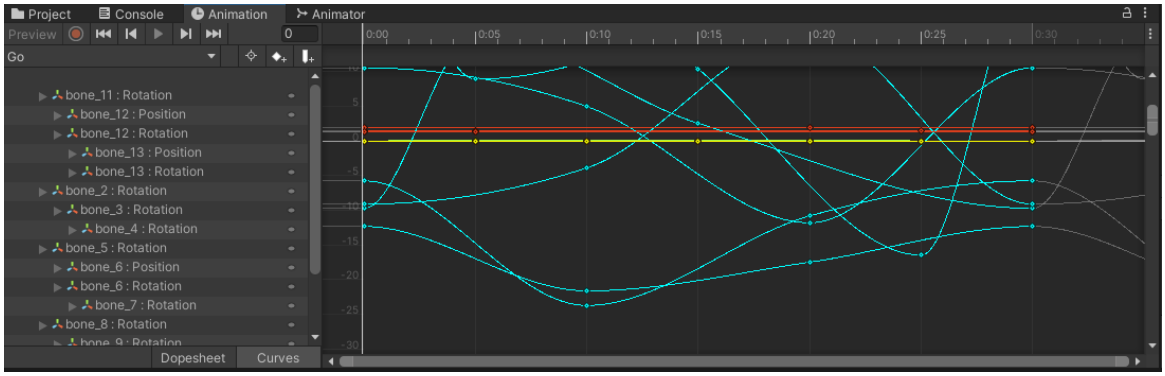
Unityl on tööriistade komplekt, mida kasutatakse animatsiooni loomiseks ja muutmiseks. Üks neist tööriistadest on "Animation". See on ala, kus asub "Timeline" koos võtmekaadritega. "Animation" võimaldab liigutada, pöörata, muuta objekti suurust igas eraldi kaadris. Kahemõõtmelise graafika jaoks on "Animation" asendamatu tööriistakomplekt, mis võimaldab muuta spraite eraldi animatsiooni kaadrites.

"Animation Timeline" aken võimaldab redigeerida animatsiooni võtmekaadreid. Sellel on kaks režiimi: "Dope Sheet" ja kõverate redigeerimiseks "Curves". "Dope Sheet" (Joonis 2.4) on kompaktne võtmekaadrite vaade, mis võimaldab vaadata nende omadusi üksikutes horisontaalsetes ribades.



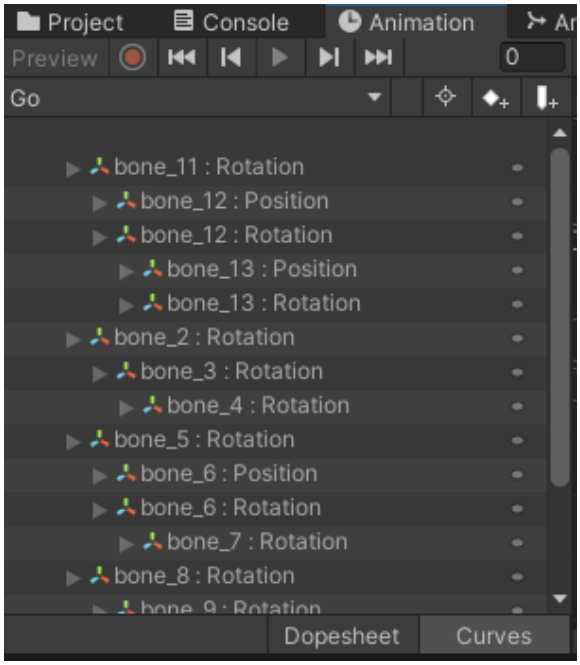
Joonis 2.4 Animatsiooni režiim „Dope Sheet"

Kõverate redigeerimise aken (Joonis 2.5) võimaldab vaadata, kuidas omadused aja jooksul mängitakse ja pakub seadistuste muutmiseks tööriistu.



Joonis 2.5 Animatsiooni režiim "Curves"

"Animation View" aken annab võimaluse luua ja parandada animatsiooniklippe Unity sees. Võimaldab laiendada klippe, kasutades animatsiooni sündmusi "Animation Events". Animatsiooni ajal kutsutakse neid funktsioone esile õigel ajal. Animeeritud omaduste nimekirja aken "Animated Properties List" (Joonis 2.6) võimaldab lisada või eemaldada animatsiooni omadusi. [9]



Joonis 2.6 "Animated Properties List"

Lisaks näitab "Animated Properties List", millised objekti omadused on animeeritud ja milline on selle tähendus valitud kaadris. [9]

2.2 Animator

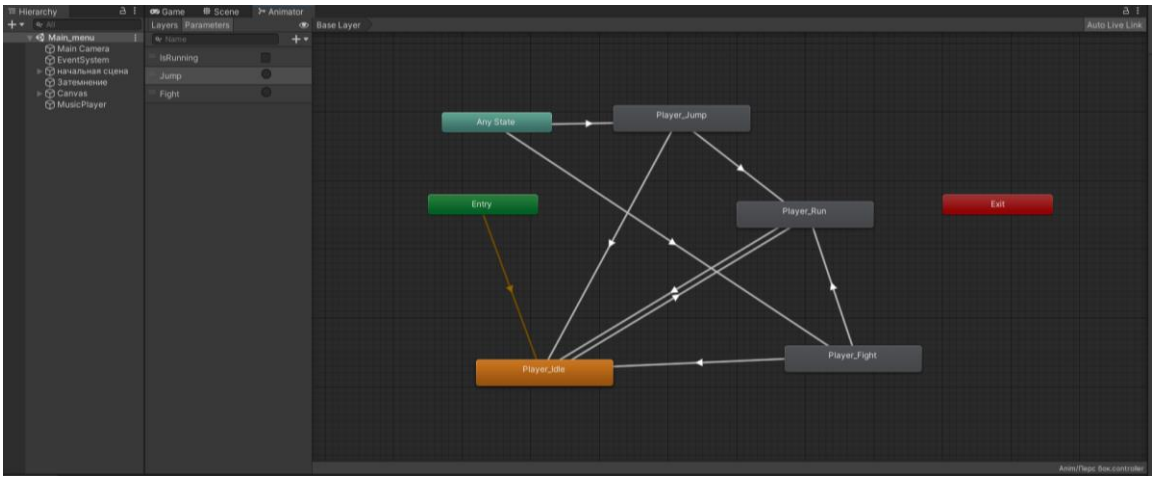
"Animator" on suur ja keeruline süsteem, mis juhib animatsiooni. Komponent "Animator" on mõeldud ühe või teise animatsiooni esitamiseks ja omavaheliseks seostamiseks. Seoste loomiseks tuleb valmis animatsioonid viia aknasse "Animator". Selles etapis tuleb läbi mõelda, milliseid animatsioone omavahel siduda ja millistest olekutest animatsioon algab. [9]

Näiteks võib vaadata loodud animatsioonide seoseid mängutegelasele mängus „Soul never die" (Joonis 2.7). Staatus, oskused ja eesmärgid on toodud allolevas tabelis (Tabel 2.1)

Tabel 2.1 Mängutegelase staatus, oskused ja eesmärgid

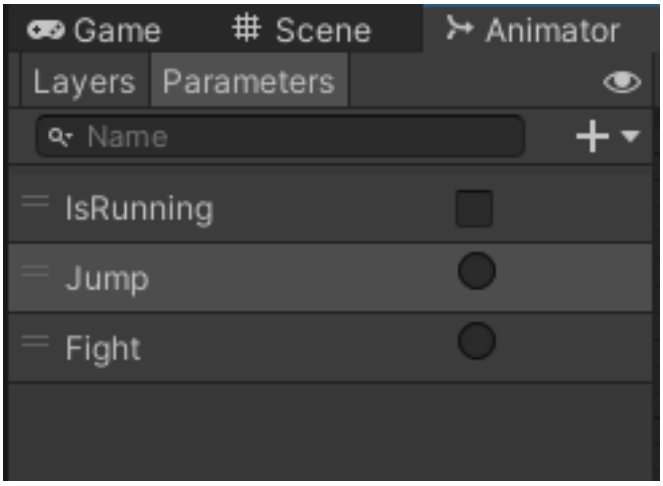
Tegelane	Staatus	Oskused	Eesmärk
Vaim Kangelane, kes on kutsutud maa peale, et päästa püha puu koda.	Peategelane Mängija	• Jookseb • Hüppab • Võitleb	• Puhastada lokatsioonid vaenlase mobidest • Koguda boonuseid ja parandusi • Hävitada lokatsioonide ülemused/bosse

Selles mängus on kaks olekut: rahu (Idle) ja mis tahes (Any state). Rahuolek on autori loodud animatsioon, mis käivitub mängu alustamisel. Kui vaadata näidet, saab näha, et rahuolekust toimub üleminek jooksu animatsioonile ja tagasi. Igast olekust on loodud üleminekud hüppe ja võitluse animatsioonile. See tähendab, et neid animatsioone käivitatakse teistest animatsioonidest sõltumatult, kui nende käivitustingimused on täidetud. Samuti võib märgata, et hüppe ja võitluse animatsioonidest on tehtud rahuolekusse ja jooksu animatsiooni ülemineku seosed. Teoreetiliselt võiks näiteks näidatud animatsioone omavahel siduda, kuid suure tõenäosusega need käivitu, vaid hanguvad ega liigu ühest teise. Näiteks oli algselt autoril kõik animatsioonid omavahel seotud. Võitluse animatsioon töötas, kuid siis see hangus ega läinud jooksu animatsioonile üle, kuigi tingimused olid täidetud. Seetõttu on väga oluline läbi mõelda üleminekute algoritm ja mitte unustada ka selle algoritmi toimimist kontrollida.



Joonis 2.7 "Animator" aken mängus „Soul never die"

"Animator" võimaldab luua animatsioonipuu, et kontrollida üksikute animatsioonide vahelisi suhteid. Näiteks kui mängutegelase iga kehaosa on animeeritud eraldi. Sel juhul saab luua tingimuse, mille korral üks animatsioon vaheldub teisega ja vastupidi. Samuti on võimalik korraga käivitada üksikuid animatsioone. "The Animator Window" võimaldab luua, vaadata ja modifitseerida "Animator Controller". Animaatori aken (Joonis 2.7) koosneb kahest osast: olekute ja üleminekute redigeerimine, kihtide ja parameetrite redigeerimine (Joonis 2.8). Vahekaardil "Parameters" määratakse parameetrid, mida edaspidi programmeeritakse koodi abil. [9]



Joonis 2.8 Kihid ja parameetrid

"Animator Controllers" animaatori kontroller annab võimaluse seadistada ja juhtida tegelase või mänguobjekti animatsiooni, ümber lülitades määratud tingimustel mitme animatsiooni vahel. Näiteks saab kõndimise animatsioonilt üle minna hüppele, klõpsates programmeeritud nupule. Kontrolleril on lingid animatsiooniklippidele, mida ta kasutab, juhib erinevaid olekuid ja nendevahelisi üleminekuid. [10]

2.3 Kood

Koodi kaudu saab kirjutada 2D ruumis tegelase koordinaatide muutmise ja kui on tehtud liikumise animatsioon, siis saab neid seostada. Näiteks lõi autor koodi tegelase animatsioonide käivitamiseks (Joonis 2.9).

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CharecterAnimation : MonoBehaviour
{
    public Animator anim;

    void Start()
    {
        anim = GetComponent<Animator>()

    }


    void Update()
    {
        if (Input.GetKey(KeyCode.LeftArrow) || Input.GetKey(KeyCode.RightArrow) || Input.GetKey(KeyCode.A) || Input.GetKey(KeyCode.D))
        {
            anim.SetBool("IsRunning", true);
        }
        else
        {
            anim.SetBool("IsRunning", false);
        }

        if(Input.GetKey(KeyCode.Space))
        {
            anim.SetTrigger("Jump");
        }


    }
}
```

Joonis 2.9 Kood mängutegelase animatsiooni käivitamise tingimustega

Koodis võib näha pöördumist animaatori poole. Koodi abil toimub animatsioonide lugemine ja neile tingimuste lisamine. Näites real 18 võib täheldada üht tingimustest, et klõpsates ühele nuppudest (vasak või parem nool, nupp A või D) käivitub jooksu animatsioon. Sel põhimõttel loodi koodid mängutegelasele, vaenlasele, stardimenüüle ja pausimenüüle.

3. TÖÖ TULEMUSED

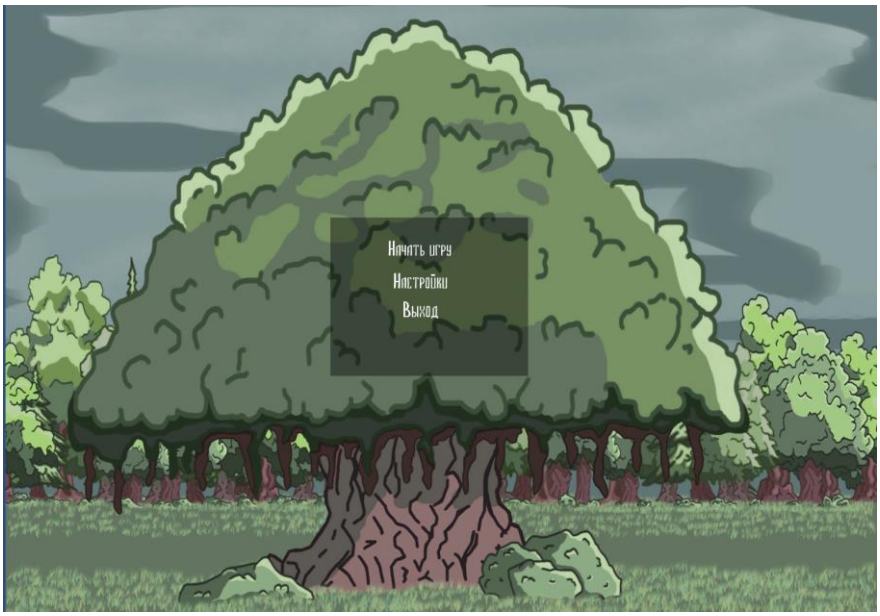
Selles peatükis kirjeldab autor kogu tehtud töö tulemusi mängu „Soul never die” ja “Multimeedia” kursuse jaoks animatsiooni teemaliste õppematerjalide loomist ning püstitatud hüpoteesi kontrollimist.

3.1 Menüü mängu „Soul never die“ jaoks

Mängumenüü on oluline osa mängus, sest see on esimene, mida mängija näeb. Mängija esmamulje on üks olulisemaid aspekte, sest esmamuljest sõltub, kas mängija hakkab antud mängu mängima. Kui menüü on keeruline, arusaamatu või näeb halb välja, võib see mängija eemale tõugata. See annab ka esimese kommunikatsiooni mänguga. Mängija saab mitte ainult alustada mängu, vaid ka seadistada videot, heli ja juhtimist. Lisaks põhilistele käivitus-, seadistus- ja väljumisfunktsioonidele saab mängijale anda ka võimaluse vaadata meediateeki või mängu autoreid.

Mängus „Soul never die” tehti üsna lihtne ja huvitav mängumenüü. Menüü on loodud kasutades sprait animatsiooni. Autori eesmärk oli vaadata üksikasjalikult menüü tööpõhimõtet ja luua dünaamiline, kuid samas intuitiivselt arusaadav menüü. Seda eesmärki võis saavutada lihtsa animatsiooniga.

Põhiekraanile (Joonis 3.1) on valitud puu kujutis, mida mängija peaks mängu loo järgi kaitsma. Puu jaoks tehti väike animatsioon lehtede liikumisest. Mängule on lisatud saatemuusika. Menüü sisaldab kolme nuppu: alustada mängu, seaded ja väljumine. Seadistustes saab muuta muusika helitugevust. Samuti loodi iga menüüpunkti jaoks väike animatsioon, et luua dünaamilisem pilt.



Joonis 3.1 Peamenüü

Lisaks stardimenüüle on mängul pausimenüü (Joonis 3.2).

Joonis 3.2 Pausimenüü

See võimaldab peatada mänguprotsessi igal hetkel või teha täiendavaid seadistusi otse mängu ajal. Samuti saab pausimenüüst minna peamenüüle ja väljuda mängust.

3.2 Mängu „Soul never die" tegelased

Mängu „Soul never die" jaoks tegi autor mängutegelastele animatsioonid. Mängu tegelased on loodud kasutades sprait animatsiooni. Vaenlase jaoks loodi tehisintellekti taoline intellekt.

Autor tegi mängutegelasele mitu liikumise animatsiooni: jooksmine, hüppamine (Joonis 3.3) ja löömine (Joonis 3.4).

Joonis 3.3 Hüppe animatsiooni sprait

Joonis 3.4 Rünnaku animatsiooni sprait

Autor lõi ja animeeris vaenlase tegelase. Koostas koodi (Joonis 3.5), kus on kirjutatud vaenlase olekud. Tal on kolm olekut: rahu, agressiivsus ja patrullpiirkonda naasmine. Vaenlase tööpõhimõte seisneb selles, et ta patrullib talle määratud piirkonda ja kui ta on mängu tegelase läheduses, hakkab ta teda jälitama. Kui tegelane lahkub vaenlasest piisavalt kaugele, naaseb ta patrullpiirkonda.

```
65
66      void Chill() // состояние покоя
67      {
68          if (transform.position.x > point.position.x + positionOfPatrol)
69          {
70              moveingRight = false;
71
72          }
73          else if (transform.position.x < point.position.x - positionOfPatrol)
74          {
75              moveingRight = true;
76
77          }
78
79          if (moveingRight)
80          {
81              transform.position = new Vector2(transform.position.x + speed * Time.deltaTime, transform.position.y);
82              transform.localScale = new Vector3(-1,1,1);
83          }
84          else
85          {
86              transform.position = new Vector2(transform.position.x - speed * Time.deltaTime, transform.position.y);
87              transform.localScale = new Vector3(1, 1, 1);
88          }
89
90      }
91      void Angry() // метод в котором враг гонится за игроком
92      {
93
94              transform.position = Vector2.MoveTowards(transform.position, player.position, speed * Time.deltaTime);
95
96              //speed = 20;  пример увеличения перемнной для смены состояния
97
98
99      }
100     void GoBack() //метот возвращения
101     {
102         transform.position = Vector2.MoveTowards(transform.position, point.position, speed * Time.deltaTime);
103
104     }
105
106 }
```

Joonis 3.5 Vaenlase olek

Vaenlasele loodi ka kõndimise animatsioon (Joonis 3.6).

Joonis 3.6 Vaenlase kõndimise animatsioon

Lisaks kõndimise animatsioonile on võimaluik luua surma ja rünnaku animatsioon.

3.3 Õppematerjalid

Autor on loonud õppematerjali "Multimeedia" kursuse jaoks. Kõik materjalid on vene keeles. Õppematerjalid aitavad kiiremini ja näitlikumalt uurida mõningaid funktsioone Unitys animatsiooni jaoks. Materjalid kirjeldavad animatsiooni järk-järgulist loomist.

Loodud on kaks õppematerjali. Üks materjal sisaldab mängumenüü järk-järgulist loomist. Teises kirjeldatakse alternatiivset viisi animatsiooni loomiseks Unitys, kasutades skeletti. Selgema ülevaate saamiseks lisas autor õppematerjalidesse pildid. Kõik õppematerjalides kasutatud näited on mängust „Soul never die".

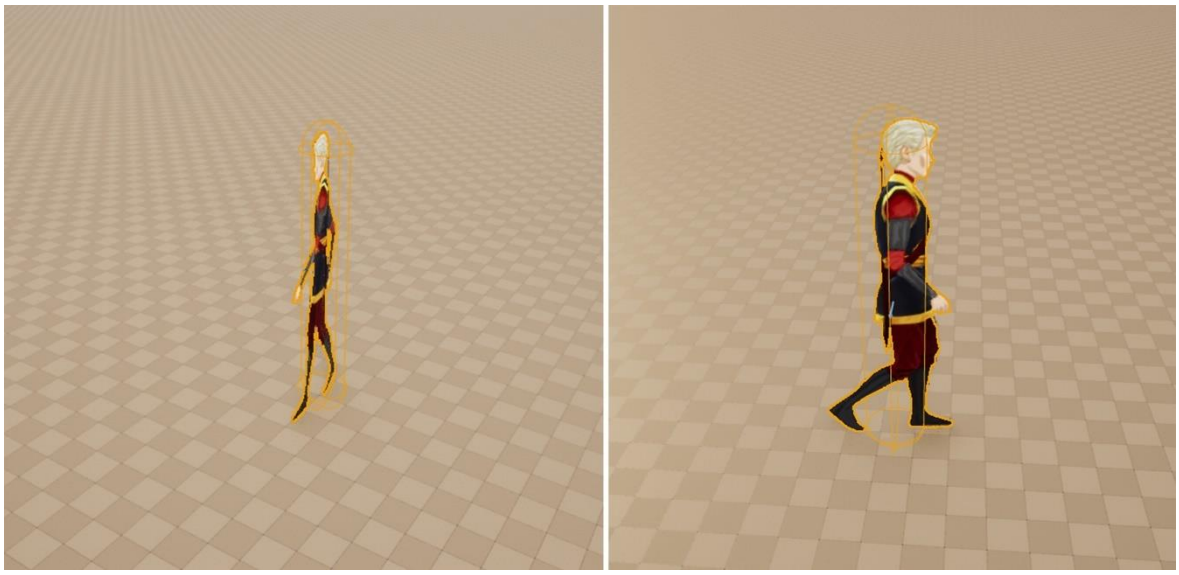
Õppematerialid on leitavad järgmisel lingil:

https://github.com/klin32/Oppematerjalid-Multimeedia

3.4 Hüpoteesi kontrollimine

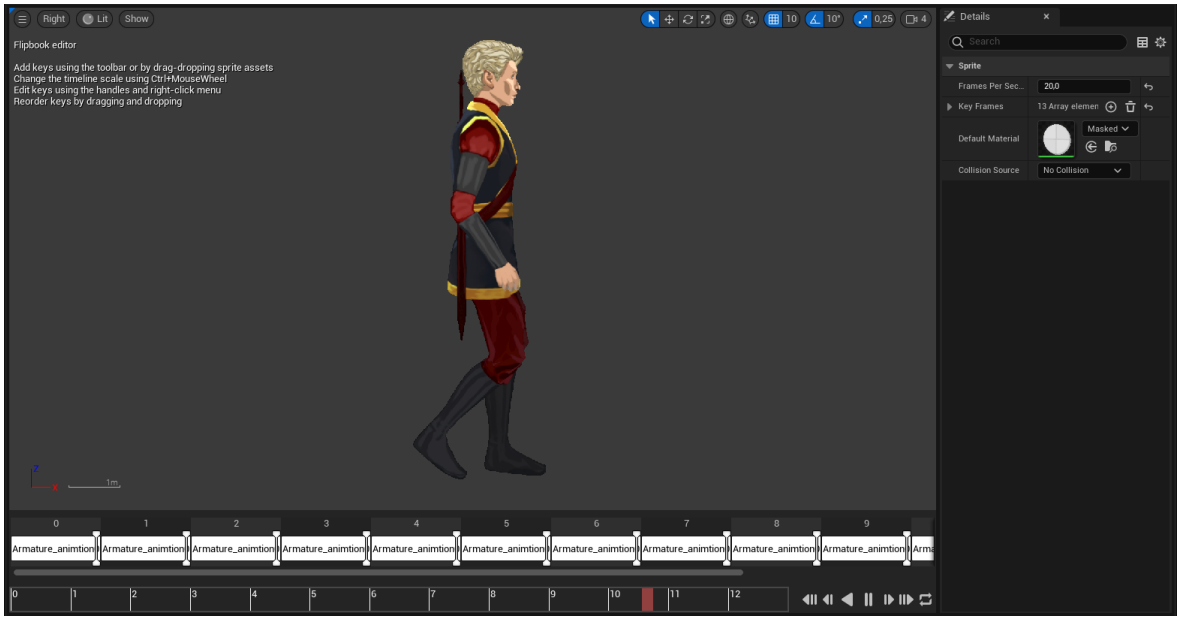
Selle töö püstitatud hüpotees seisneb selles, et teades, kuidas töötada animatsiooniga ühel mootoril, saab hõlpsasti luua ja töötada teisel. Selle hüpoteesi kontrollimiseks vaadeldi teist populaarset mootorit Unreal Engine.

Peale seda, kui autor oli uurinud 2D animatsiooni loomise võimalusi Unitys, otsustas ta proovida animeerida tegelast mootoril Unreal Engine. Unreal Engine on orienteerunud rohkem kolmemõõtmelise graafika loomisele. See on peamine erinevus Unity ja Unreal Engine'i vahel. Liides on piisavalt keeruline ja ebaselge, kuna kahemõõtmelise graafikaga töötamine kolmemõõtmelises ruumis ei ole mugav (Joonis 3.1).



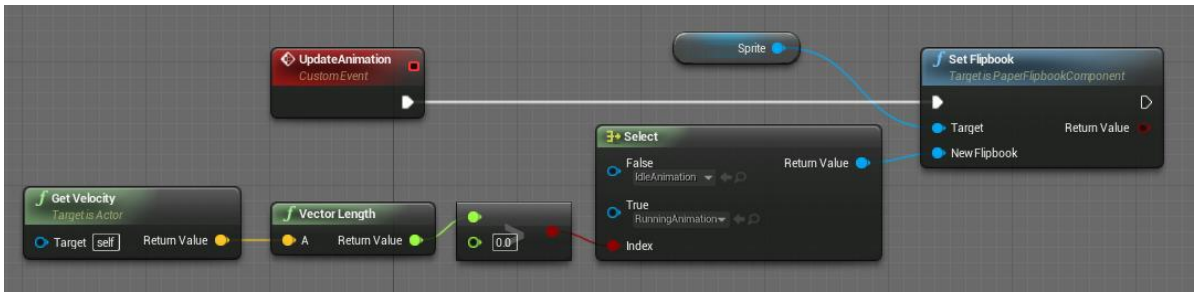
Joonis 3.7 2D tegelane kolmemõõtmelises graafikas

Animatsioon luuakse, lisades kõik spraidid funktsioonile Flipbook (Joonis 3.2). See koosneb võtmekaadrite seeriast, millest igaüks sisaldab spraiti ja selle kuvamise kestust. Parameeter Frames Per Second määrab, kui kiiresti kaadrid kuvatakse. Samuti tuleb määrata, kui kaua kaadrid sekundis kestavad. Võtmekaadreid saab redigeerida paneelil Details või ajaskaala abil. [11]



Joonis 3.8 Flipbook tegelase animatsiooniga

Kuid 3D graafika ja animatsiooniga töötamiseks on Unreal Engine'il rohkem võimalusi kui Unityl. Autor arvab, et üks mugavamaid funktsioone on graafid ehk visuaalne programmeerimine (Joonis 3.3).



Joonis 3.9 Graafide töö näide [12]

Graafide abil saab hõlpsasti luua animatsioonide lülitamise seoseid, lisada täiendavaid funktsioone ja sündmusi. See kõik lihtsustab animatsiooniga tööd, kuid samal ajal sunnib animatsiooni jaoks tegevuste järjestust hoolikalt läbi mõtlema.

KOKKUVÕTE

Selle töö eesmärk oli luua 2D animatsioon „Soul never die” videomängu jaoks. Kõik püstitatud ülesanded selle eesmärgi saavutamiseks täideti. Autor tutvus põhjalikult 2D animatsiooni loomise võimalustega Unitys, tegi mängu tegelasele animatsioonid, lõi mängu vaenlase ja 2D mängu menüü. Lisaks koostas autor ka kaks õppematerjali, kus kirjeldatakse, kuidas teha skelett animatsiooni ja mängumenüü Unitys.

Oli püstitatud hüpotees, et teades, kuidas töötada 2D animatsiooniga ühel mootoril, saab rahulikult luua ja töötada teisel. Selle hüpoteesi kontrollimiseks võrdles autor mootorite Unity ja Unreal Engine võimalusi. Autori hüpotees on ümber lükatud, sest iga mootor on ainulaadne ja animatsioonide loomise viisid on erinevad, kuigi need kasutavad ühtseid sprait animatsioone.

Autoril õnnestus luua tegelase ja vaenlase animatsioone. Tehti stardimenüü ja pausimenüü. Vaenlase jaoks loodi tehisintellekti taoline intellekt. Samuti sai autor kirjeldada üksikasjalikult mängumenüü loomist ja skeleti animatsiooni meetodit õppematerjalides.

Meeskond ei suutnud kõiki ideid ellu viia. Sel põhjusel ei ole mäng täielikult valmis ja sellel on mitmeid puudusi. Pärast vaenlase ülekandmist põhimängu ja Unity versiooni muutmist, erines tema animatsioon algsest. Autoril ei õnnestunud muudatuse põhjust tuvastada. Mäng ei ole avalikult kättesaadav.

Autor on tehtud tööga rahul. Ainus asi, mida ta muudaks, on spraitide animeerimise meetodit. Selles töös rakendati sprait animatsiooni, kuid uurimisprotsessis mõistis autor, et objektide animeerimine skelett animatsiooni abil on palju mugavam.

SUMMARY

Creating animation in Unity's game engine

Igor Sabenin

The aim of this work was to create a 2D animation for the "Soul never die" video game. The author thoroughly got acquainted with the possibilities of creating 2D animation in Unity, made animations for the game character, created an enemy of the game and a 2D game menu. In addition, the author compiled two tutorials describing how to make a skeleton animation and a game menu Unity.

To achieve this goal, the following tasks were set:

1. to study different methods of creating animation for 2D games and how easily they are implemented in different game engines;

2. controlling the character in Unity and animating the movements through pre-prepared animations;

3. create a game menu in a 2D game.

It was hypothesized that knowing how to work with 2D animation on one engine would allow you to create and work on another. To test this hypothesis, the author compared the capabilities of the Unity and Unreal Engine engines. The author's hypothesis has been refuted because each engine is unique and the ways in which the animations are created are different, although they use same sprites.

The author managed to create animations for the character and the enemy. The start menu and pause menu were made. An intellect like artificial intelligence was created for the enemy. The author was also able to describe in detail the creation of a game menu and the method of skeletal animation in the study materials.

The team was not able to implement all the ideas. For this reason, the game is not completely ready and has several drawbacks. After transferring the enemy to the main game and changing the Unity version, his animation was different from the original. The author could not determine the reason for the change. The game is not publicly available.

The author is satisfied with the work done. The only thing he would change is the method of animating the sprites. Sprite animation was used in this work, but in the research process the author realized that animating objects with skeletal animation is much more convenient.

KASUTATUD KIRJANDUS

1. Porter, J. US consumers spent record amounts on video games in 2020, NPD reports. [*Online*] https://www.theverge.com/2021/1/15/22233003/us-npd-group-video-game-spending-2020-record-nintendo-switch-call-of-duty-animal-crossing-ps5-ps4 (20.03.2022).
2. FinancesOnline. Number of Gamers Worldwide 2022/2023: Demographics, Statistics, and Predictions. [*Online*] https://financesonline.com/number-of-gamers-worldwide/ (20.03.2022).
3. Ametlik veebileht "Dota 2". [*Online*] https://www.dota2.com/international/battlepass/ (22.03.2022)
4. Картаслов „Псевдотрёхмерность". [*Online*] https://kartaslov.ru/карта-знаний/Псевдотрёхмерность (22.03.2022).
5. Google Trends. Unity, Unreal Engine, CryEngine mängumootorite populaarsus Eestis. [Online] https://trends.google.ee/trends/explore?geo=EE&q=%2Fm%2F0dmyvh,%2Fm%2F025wnp,%2Fm%2F04lh6j (15.04.2022).
6. Google Trends. Unity, Unreal Engine, CryEngine mängumootorite populaarsus maailmas. [Online] https://trends.google.ee/trends/explore?q=%2Fm%2F0dmyvh,%2Fm%2F025wnp,%2Fm%2F04lh6j (15.04.2022).
7. Geig, M. (2018). *Unity 2018 Game Development in 24 Hours*. Sams Publishing
8. Всё о 3D-анимации: советы, софт и многое другое. [*Online*] https://www.renderforest.com/ru/blog/3d-animation (05.05.2022)
9. Косагова, О. Работа с анимациями в Unity3D. [*Online*] https://docplayer.com/39299582-Rabota-s-animaciyami-v-unity3d-16-noyabrya-2016.html (05.05.2022).
10. Unity Technologies. Animator Controller. [*Online*] https://docs.unity3d.com/ru/current/Manual/class-AnimatorController.html (05.05.2022).
11. Epic Games. Paper 2D Flipbooks. [*Online*] https://docs.unrealengine.com/5.0/en-US/paper-2d-flipbooks-in-unreal-engine/ (22.05.2022).
12. Epic Games. Flipbook Components in Blueprints. [*Online*] https://docs.unrealengine.com/4.27/en-US/AnimatingObjects/Paper2D/Flipbooks/Components/Blueprints/ (22.05.2022).