Interakcija pomoću Leap Motiona u aplikacije razvijene u razvojnom okruženju Unity

Budiša, Juraj

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU

STRUČNI PRIJEDIPLOMSKI STUDIJ RAČUNARSTVO

Juraj Budiša, 0313025599

INTERAKCIJA POMOĆU LEAP MOTIONA U APLIKACIJE RAZVIJENE U RAZVOJNOM OKRUŽENJU UNITY

ZAVRŠNI RAD

Čakovec, rujan 2024.

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU

STRUČNI PRIJEDIPLOMSKI STUDIJ RAČUNARSTVO

Juraj Budiša, 0313025599

INTERAKCIJA POMOĆU LEAP MOTIONA U APLIKACIJE RAZVIJENE U RAZVOJNOM OKRUŽENJU UNITY

INTERACTION WITH LEAP MOTION IN APPLICATIONS DEVELOPED IN THE DEVELOPMENT PLATFORM UNITY

ZAVRŠNI RAD

Mentor**:**

Nenad Breslauer, v. pred.

Čakovec, rujan 2024.

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU

PRIJAVA TEME I OBRANE ZAVRŠNOG/DIPLOMSKOG RADA

Stručni prijediplomski studij:

Računarstvo ☒ Održivi razvoj ☐ Menadžment turizma i sporta ☐

Stručni diplomski studij Menadžment turizma i sporta: ☐

Pristupnik: Juraj Budiša, JMBAG: 0313025599
(ime i prezime)

Kolegij: Razvoj računalnih igara
(na kojem se piše rad)

Mentor: Nenad Breslauer, v.pred
(ime i prezime, zvanje)

Naslov rada: Interakcija pomoću Leap Motiona u aplikacije razvijene u razvojnom okruženju Unity

Naslov rada na engleskom jeziku: INTERACTION WITH LEAP MOTION IN APPLICATIONS DEVELOPED IN THE DEVELOPMENT PLATFORM UNITY

Članovi povjerenstva: 1. Tibor Rodiger, predsjednik
(ime i prezime, zvanje)

2. Ivan Hegeduš, član
(ime i prezime, zvanje)

3. Nenad Breslauer, mentor
(ime i prezime, zvanje)

4. Sanja Brekalo, zamjenski član
(ime i prezime, zvanje)

Broj zadatka: 2022-RAČ-R-9

Kratki opis zadatka: Zadatak je napraviti računalnu igru te iskoristiti mogućnost upravljanja likom pomoću Leap Motiona.

[illegible]

[illegible]

[illegible]

postupcima te se u narednom poglavlju iznose glavni zaključci rada. Rad se završava poglavljima s popisom literature te prilozima.

Datum: 10.9.2024 Potpis mentora: [signature]

Predgovor

Tema ovoga završnog rada odabrana je zbog mojega velikog interesa za tehnologiju *Leap Motiona*, odnosno za ideju upravljanja i interakcije objektima s pomoću takve tehnologije. Pisanje završnoga rada s ovom temom te izrada aplikacije pokazali su se kao edukativno, ali i zabavno iskustvo te prilika za učenje o upotrebi i potencijalu *Leap Motiona*, kao i za stjecanje iskustava u razvojnom okruženju *Unity*, s programskim jezikom C# i s grafičkim softverom *Blender*. Zanimljivo će biti pratiti kako će se ta tehnologija razvijati u budućnosti.

Zahvaljujem svojemu mentoru, mag. prim. educ. Nenadu Breslaueru koji me vodio i pomogao u izradi ovoga završnog rada.

Sažetak

Završni rad o temi izrade aplikacije koja koristi uređaj *Leap Motion* za interakciju s objektima. Opisan je uređaj *Leap Motion* te izrada aplikacije unutar razvojnoga okruženja *Unity*. Korisnik može upravljati aplikacijom s pomoću uređaja *Leap Motion*, samo pokretima ruku. Aplikacija u sebi ima dvije jednostavne igre koje služe kao primjer interakcije virtualnim svijetom jednostavnim pokretima ruku koristeći uređaj *Leap Motion*. U prvoj igri korisnik mora uhvatiti oblike te ih staviti u odgovarajuće otvore u kutiji. Druga igra jednostavna je slagalica sastavljena od devet dijelova, u kojoj korisnik mora uhvatiti dijelove slagalice i staviti ih na odgovarajuće mjesto na drvenoj ploči.

Ključne riječi: *Leap Motion* uređaj, tehnologija, *Unity*, aplikacija, interakcija

Abstract

Final thesis on the topic of creating an application that uses the leap motion device to interact with objects. The thesis describes the device Leap Motion and the development of the application within the Unity development platform. The user can control the application with the help of the leap motion device, using only hand movements. The application contains two simple games, which serve as an example of interacting with the virtual world with simple hand movements using the leap motion device. In the first game, the user must grab the shapes and place them in the corresponding slots in the box. The second game is a simple puzzle game composed of 9 parts, in which the user must grab the puzzle pieces and place them in the corresponding place on the wooden board.

Key Words: Leap Motion device, technology, Unity, application, interaction

SADRŽAJ

1. UVOD

Tema ovoga završnog rada izrada je računalne igre u razvojnom okruženju *Unity* i programskim jezikom C#, u kojem je okolina igrača interaktivna te reagira na akcije igrača s pomoću uređaja *Leap Motion*. Za izradu rada također se koristio besplatan i *open-source* grafički softver *Blender* za modeliranje i teksturiranje interaktivnih objekata.

Cilj je ovog rada bio stvoriti jedinstveno iskustvo igre koje se temelji na gestama igrača, a ne na tradicionalnim metodama kontrole, kao što su, primjerice, tipkovnica i miš ili kontroler za konzolu. Korištenjem revolucionarne tehnologije *Leap Motion* igra se može u potpunosti igrati samo prirodnim pokretima ruku korisnika.

U narednim poglavljima opisani su glavni alati i programi korišteni u izradi rada, način postavljanja i instalacije *Leap Motion* uređaja, integracija *Leap Motion* u razvojno okruženje *Unity*, grafički softver *Blender* i 3D modeliranje i teksturiranje objekata.

2. LEAP MOTION

Leap Motion kontroler mali je periferni USB uređaj tvrtke *Leap Motion*, Inc. namijenjen zamjeni miša kao uređaja za upravljanje računalom. Dizajniran je za postavljanje na radni stol ispred monitora, okrenut prema gore, ali se može pričvrstiti i na uređaj za virtualnu stvarnost – HMD (engl. *Head Mounted Display*). Koristeći dvije monokromatske infracrvene kamere i tri infracrvene LED diode, *Leap Motion* kontroler promatra prostor veličine oko jednoga metra, u kojem korisnik može upravljati računalom koristeći samo svoje ruke. [1]

Za korištenje uređaja potrebno ga je priključit na računalo USB kabelom, staviti ga na vodoravnu površinu ispred ekrana te instalirati potrebne *drivere* sa stranice leapmotion.com.

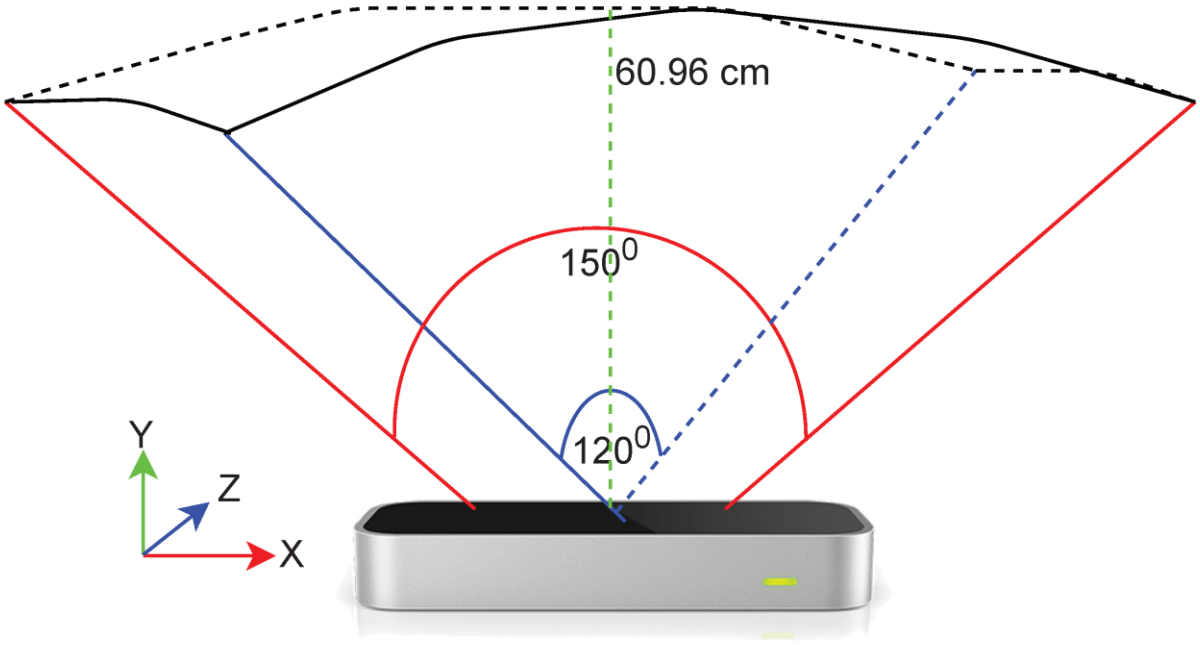
Slika 1. *Leap Motion* uređaj

Izvor: https://www.imore.com/leap-motion-controller-review

2.1 Specifikacije uređaja Leap Motion

Leap Motion je uređaj koji je vrlo kompaktan i prenosiv, s dimenzijama od 80 mm x 30 mm x 11.3 mm i težinom od 32 grama. Za praćenje koristi dvije 640x240-pixel CCD kamere međusobno udaljene 40 mm, u skladu s 3 infracrvenim LED svjetlima. Kamere prate infracrvena svjetla na valnoj duljini od 850 nanometara, izvan spektara vidljive svijetlosti. Raspon je praćenja oko 60 cm x 60 cm (kut pogleda 150 stupnjeva), s brzinom osvježavanja od 120 Hz i preciznošću od 0.01mm. Preporučena je daljina korištenja oko 60 cm, s minimalnom udaljenošću od 2.5 cm i maksimalnom do 80 cm. [1]

Slika 2. Raspon praćenja



Izvor: https://www.mdpi.com/2079-9292/9/12/1986

2.2 Uporaba uređaja Leap Motion

Tvrtka *imageHOLDERS* lansirala je 2021. godine prvi kiosk bez dodira, korištenjem *Leap Motion* tehnologije za praćenje ruke kao odgovor na pandemiju bolesti COVID-19. Korisnici mogu tipkati, selektirati, skrolati i navigirati zaslonom kioska koristeći pokrete i geste u zraku (*Swipe*, *Tap*, *Pinch*), bez dodirivanja zaslona. Ovaj pristup bez dodirivanja smanjuje rizik od širenja bakterija koje se nakupljaju na ekranima samoposlužnih kioska, a čega su kupci i krajnji korisnici postali sve svjesniji tijekom pandemije. Takozvani *Touchless* kiosci mogli bi se koristiti u zdravstvenim ustanovama, školama i fakultetima, radnim mjestima i sl. [2]

Slika 3. *Touchless* kiosk

Izvor: https://www.ultraleap.com/company/news/press-release/imageholders-touchless-kiosk/

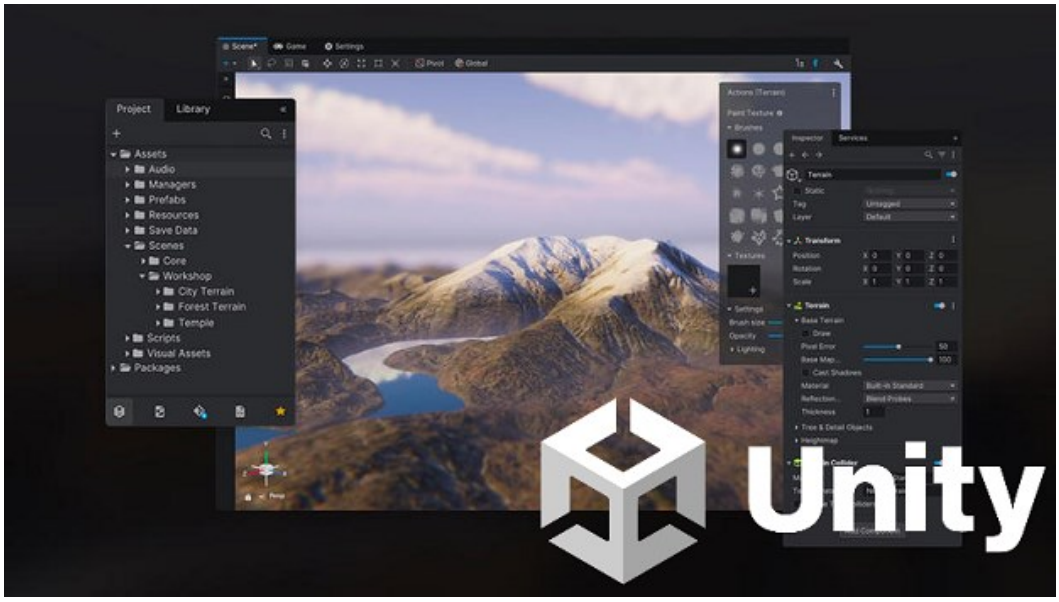
3. KORIŠTENI PROGRAMSKI ALATI

3.1 Unity

Unity je jedan od najpopularnijih softvera za razvijanje igara zbog njegova jednostavnog sučelja za korištenje, pristupačne cijene, bogatih alata za modeliranje, animaciju, fiziku i audio, ali i zbog toga što uključuje biblioteku gotovih skripta i *asseta* što olakšava i ubrzava razvoj igara. *Unity* je višeplatformski softver, što znači da omogućava razvoj igara za razne platforme, uključujući PC, mobilne uređaje, web i VR. [3]

Također, postoji mogućnost razvoja jednostavnih igara koje se mogu igrati samo pokretima ruku korištenjem uređaja *Leap Motion*. Za razvoj igara za Leap Motion u razvojnom okruženju Unity potreban je *plugin* koji je dostupan na stranici developer.leapmotion.com, na kojoj se još mogu pronaći i razni drugi resursi kao što su jednostavne demoaplikacije ili stare verzije *drivera* (engl. *Legacy Releases*): V2, V4 (Orion) i V5 (Gemini).

Slika 4. Razvojno okruženje *Unity*



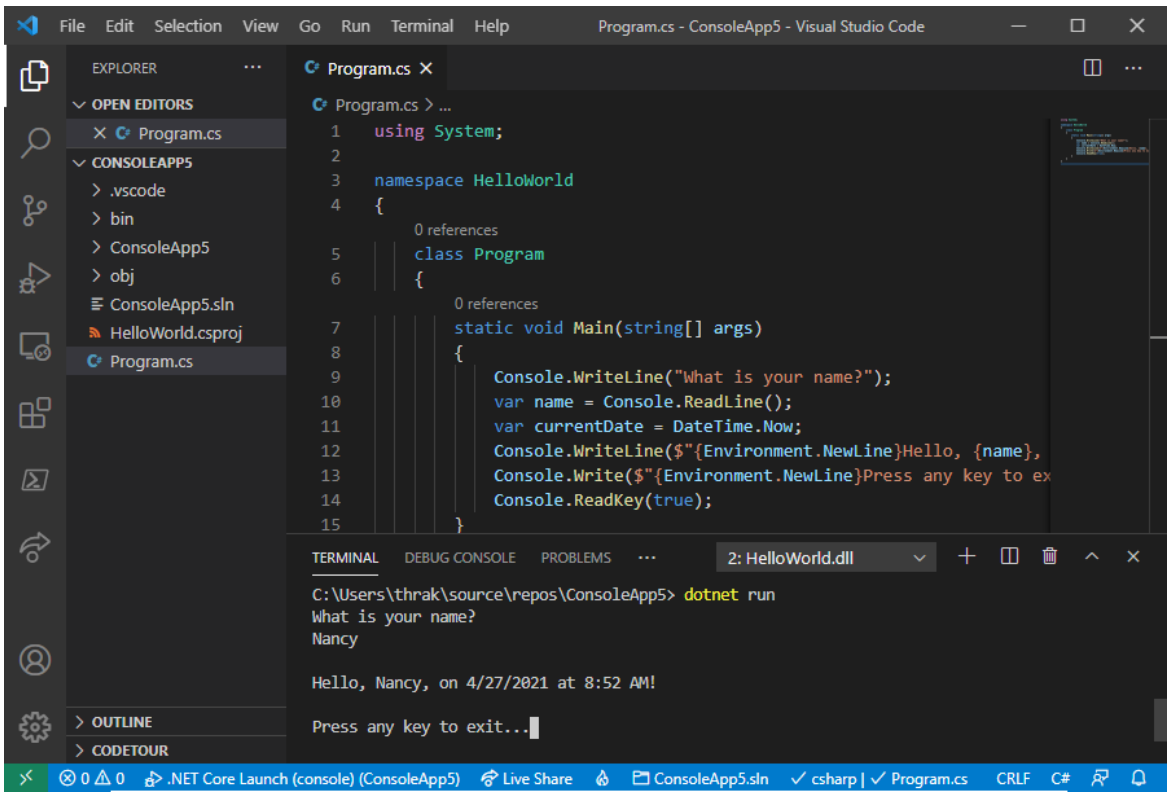
Izvor: https://dotnet.microsoft.com/en-us/apps/games/unity

3.2 Visual Studio

Visual Studio je integrirano razvojno okruženje (IDE) koje je razvila kompanija *Microsoft*. Koristi se za razvoj različitih vrsta softvera, uključujući web aplikacije, mobilne aplikacije, desktop aplikacije, te *cloud* servise. *Visual Studio* podržava više programskih jezika, kao što su *C#, C++, Python, JavaScript*, i mnogi drugi. [4]

Za izradu ove aplikacije koristi se isključivo programski jezik *C#.*

Slika 5. Primjer koda u razvojnom okruženju *Visual Studio*



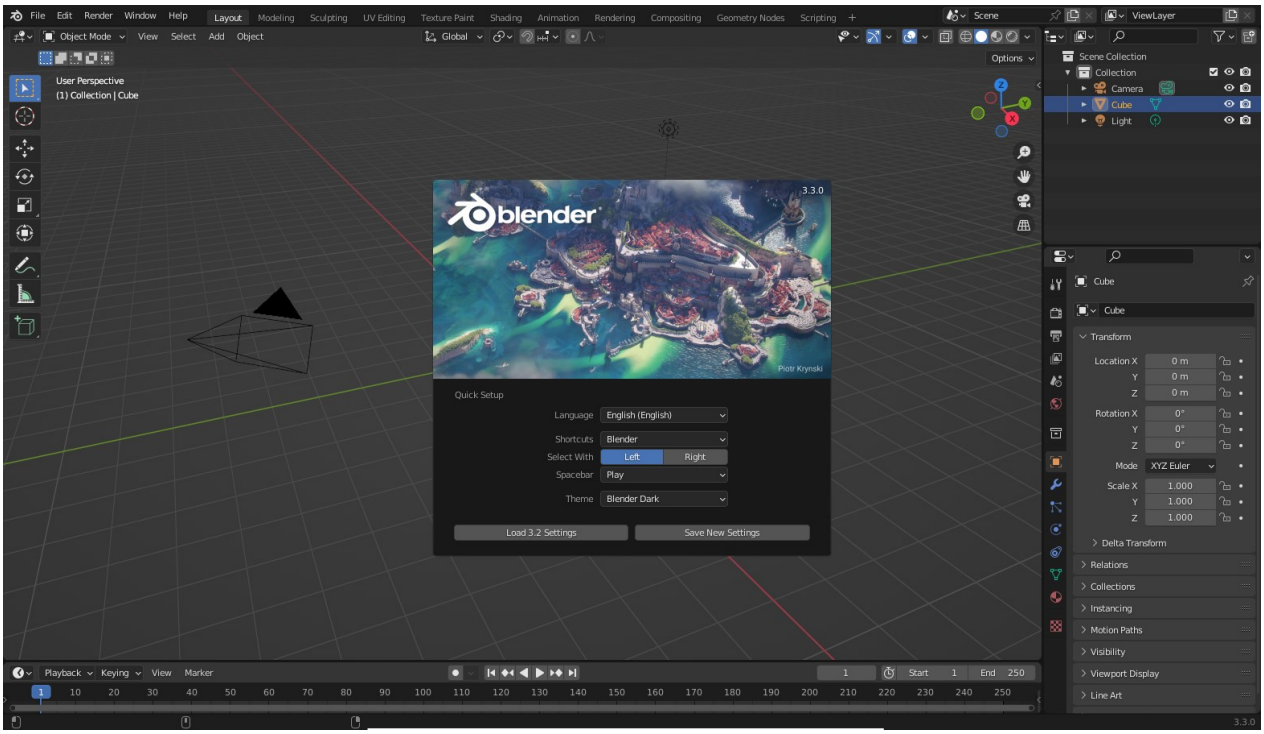
Izvor: https://learn.microsoft.com/en-us/dotnet/core/tutorials/with-visual-studio-code

3.3 Blender

Blender je popularan i besplatan *open-source* softver za 3D modeliranje, animaciju, obradu slika i videosadržaja koji se koristi u mnogim industrijama, uključujući film, televiziju, arhitekturu, igre i produkciju slika. Sadrži mnoštvo alata i funkcija za modeliranje, teksturiranje, animaciju, simulaciju i vizualizaciju, što ga čini jednostavnim za korištenje početnicima, ali i snažnim za profesionalce. U sebi ima alate za 3D modeliranje, UV mapiranje, teksturiranje, raster grafiku, skeletnu animaciju, simulacija fluida i dima, simulaciju čestica, simulaciju mekoga tijela, *sculpting*, animaciju, *rendering*, pokretnu grafiku, *videoediting* i kompoziciju. [5]

Za stvaranje objekata koristit će se alati za 3D modeliranje i teksturiranje.

Slika 6. Softver za 3D računalnu grafiku *Blender*

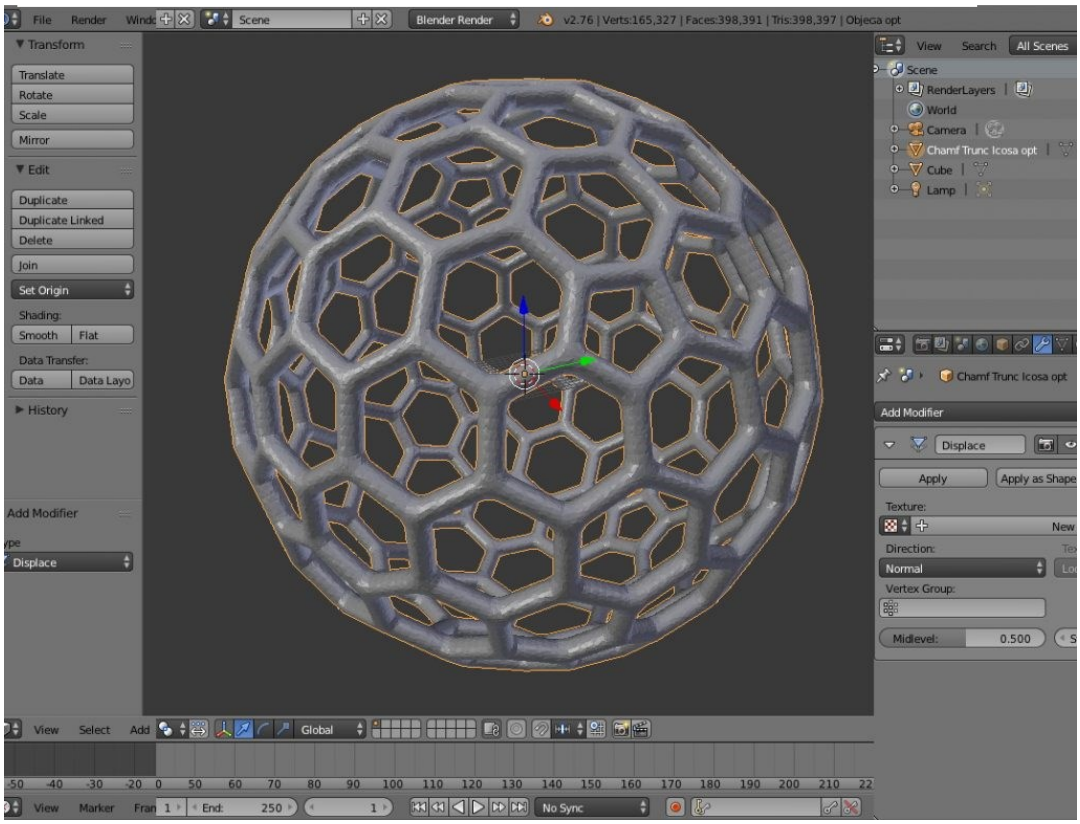


Izvor: https://hr.wikipedia.org/wiki/Blender

3.3.1 3D Modeliranje

3D modeliranje je proces stvaranja virtualne trodimenzionalne reprezentacije objekta ili scena korištenjem računalnoga softvera. Ima širok spektar primjena u mnogim industrijama, uključujući film i televiziju, igre, arhitekturu, proizvodnju i inženjerstvo. U filmu i televiziji, 3D modeliranje omogućuje stvaranje virtualnih svjetova i objekata koji izgledaju realno, što je od ključne važnosti za stvaranje uspješnoga spektakla. U arhitekturi 3D modeliranje omogućuje stručnjacima stvaranje preciznih modela budućih građevina i urbanističkih projekata, što im omogućuje da testiraju različite varijante i donesu najbolje odluke prije nego što počnu s fizičkim gradnjom. U igrama 3D modeliranje omogućuje stvaranje virtualnih svjetova i likova koji su autentični i uvjerljivi, što igračima omogućuje da se uklope u svijet igre. [6]

Slika 7. 3D modeliranje u softveru *Blender*

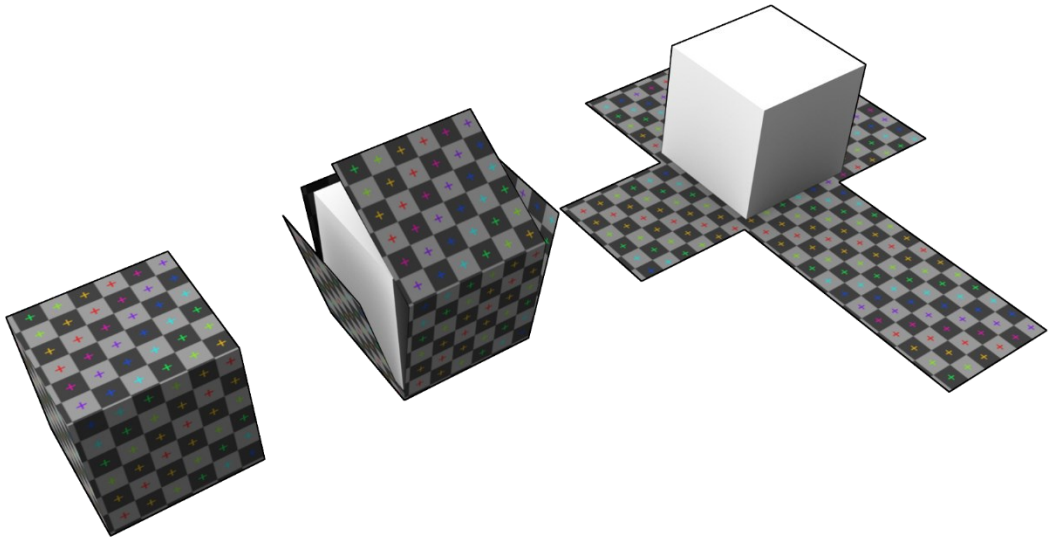


Izvor: https://mathgrrl.com/hacktastic/2017/09/thickening-3d-models-with-blender/

3.3.2 UV odmatanje

UV odmatanje je proces u 3D modeliranju koji omogućava primjenu 2D tekstura na 3D modele. U suštini, UV odmatanje pretvara 3D površinu modela u 2D ravninu, što olakšava mapiranje teksture na taj model. UV odmatanje je ključan korak u 3D modeliranju i animaciji, jer omogućava preciznu kontrolu nad načinom na koji će tekstura biti prikazana na modelu. Bez pravilno postavljenih UV mapa, teksture mogu izgledati iskrivljeno ili neprirodno na 3D objektima. [7]

Slika 8. UV odmatanje kocke

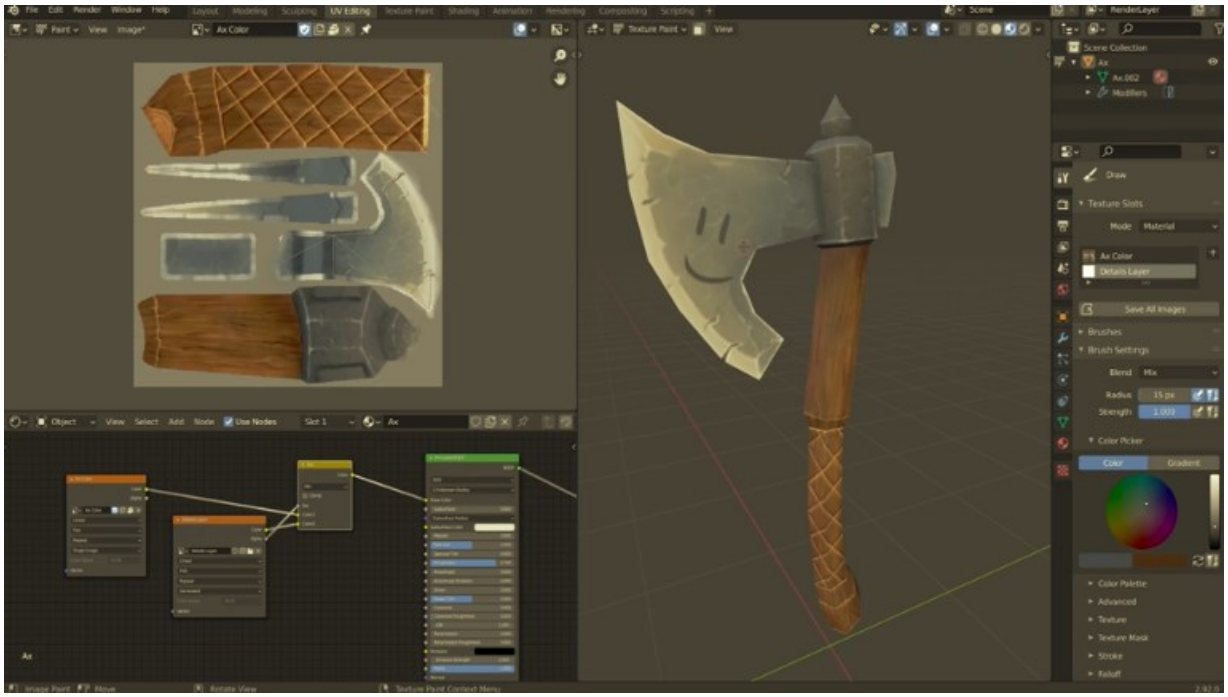


Izvor: https://en.wikipedia.org/wiki/UV_mapping

3.3.3 Teksturiranje

Teksturiranje (engl. *Texturing*) proces je dodavanja teksture, boje i detalja na trodimenzionalni model/objekt. Ova tehnika omogućuje da se modelu dodaju dodatne karakteristike kako bi izgledao autentičniji i realniji. Teksturiranje se obično koristi za dodavanje različitih materijala na model, kao što su drvo, kamen, metal i tkanine. Također, omogućuje dodavanje različitih tekstura, poput pukotina i ogrebotina kako bi se modelu dodao dodatni život. Bez teksturiranja model bi izgledao dosadno i bezlično. [8]

Slika 9. 3D modeliranje u softveru *Blender*



Izvor: https://cgcookie.com/courses/fundamentals-of-texturing-in-blender

4. APLIKACIJA

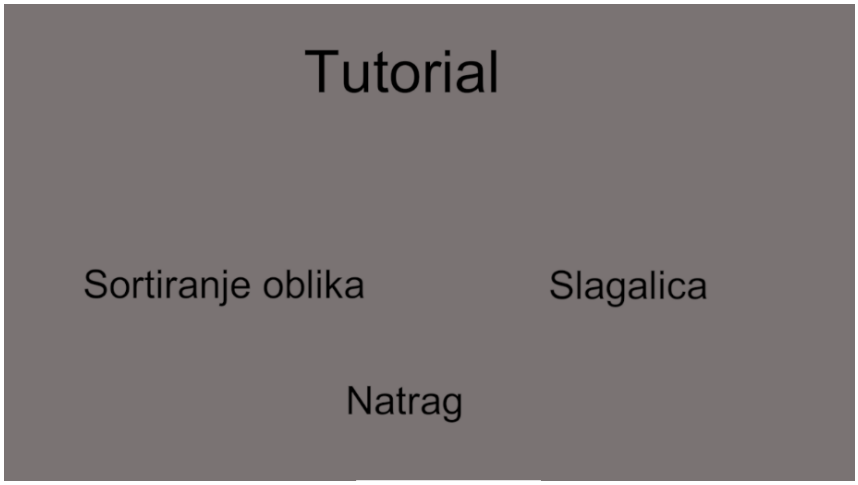
Kada korisnik pokrene aplikaciju, prvo što vidi je meni kojim može pokrenuti igru, vidjeti objašnjenja za postojeće dvije igre u aplikaciji ili ugasiti aplikaciju.

Slika 10. Početni zaslon

Izvor: Autor

Klikom miša na tekst *Tutorial* scena se promijeni i dobije se opcija između dvaju kratkih video objašnjenja za postojeće dvije igre u aplikaciji.

Slika 11. Odabir objašnjenja



Izvor: Autor

Klikom miša na lijevu opciju otvori se video objašnjenje za igru *Sortiranje oblika*, a klikom desne opcije otvara se video objašnjenje za igru *Slagalica*.

Slika 12. Objašnjenje igre sortiranje oblika

Izvor: Autor

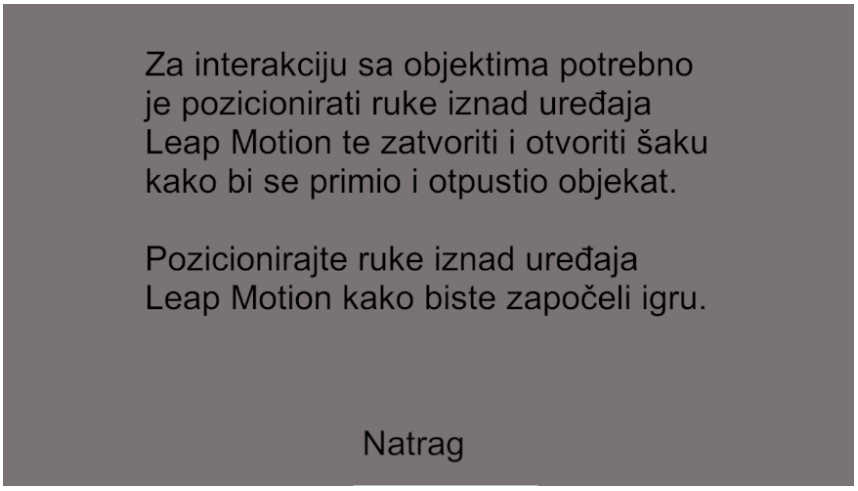
Slika 13. Objašnjenje igre slagalica

Izvor: Autor

Klikom na tekst *Natrag* igrača se prebaci na prethodnu stranicu.

Nakon što korisnik klikne na tekst *Početak*, pojavi se kratko objašnjenje kako upravljati uređajem *Leap Motion*. Igrač započinje igru tako da pozicionira ruke iznad uređaja i čeka pet sekundi.

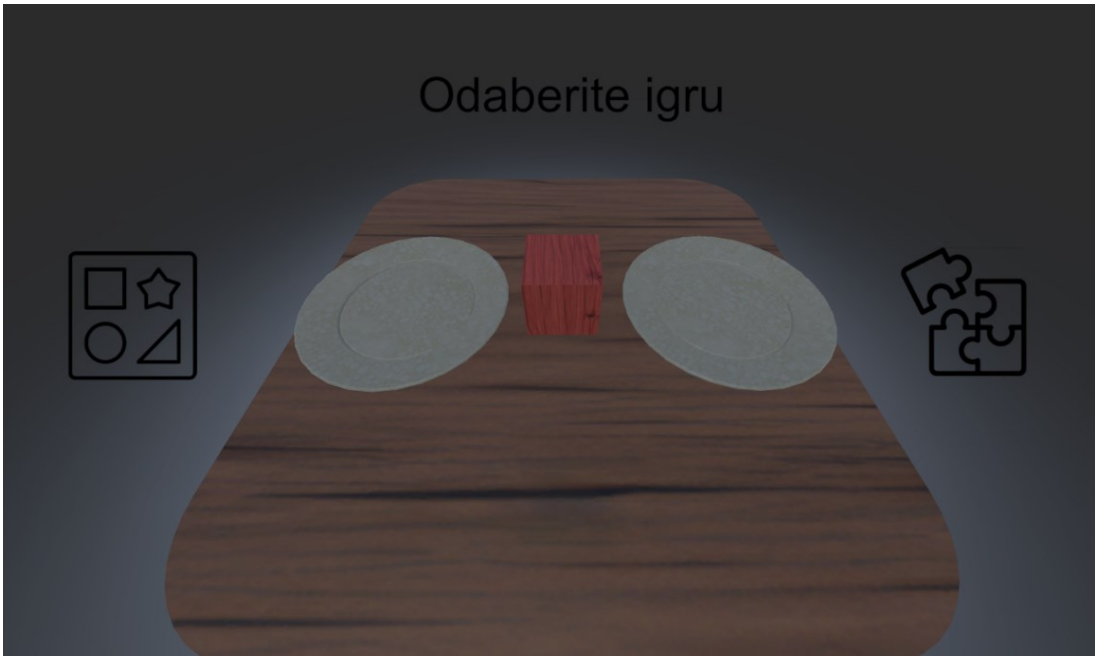
Slika 14. Objašnjenje upravljanja *Leap Motion* uređajem



Izvor: Autor

U sljedećoj sceni igrač može izabrati jednu od dviju igri. Korisnik odabere igru tako da uhvati crvenu kocku na sredini scene i stavi ga na lijevi ili desni tanjur. Nakon toga pojavljuje se brojač od pet sekundi za potvrdu selekcije igre nakon čega se učitava odabrana igra.

Slika 15. Odabir igre



Izvor: Autor

Ako korisnik odabere lijevu igru, učita se igra u kojoj korisnik mora staviti oblike u odgovarajuće otvore u kutiji.

Slika 16. Igra *Sortiranje oblika*

Izvor: Autor

Selekcijom desne igre učitava se slagalica od devet dijelova. Korisnik rješava slagalicu tako da uhvati dijelove slagalice i stavi ih na odgovarajuća mjesta na drvenoj ploči.

Slika 17. Igra *Slagalica*

Izvor: Autor

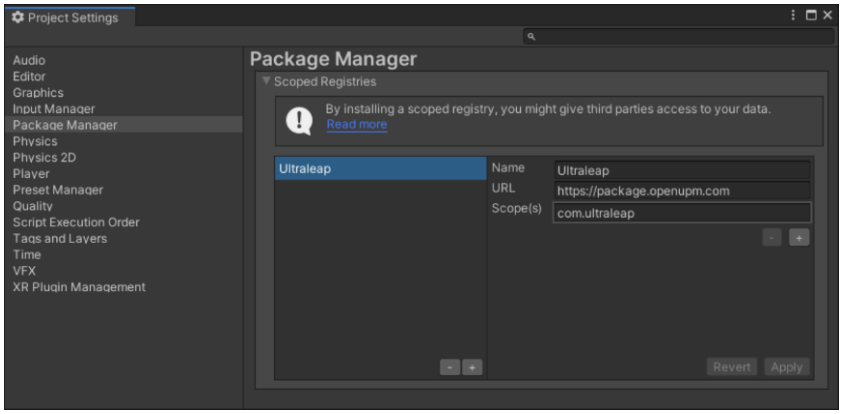
Nakon što korisnik pobijedi u igri, aplikacija ga šalje natrag na scenu za odabir igre. Želi li korisnik otići natrag na početnu scenu, samo treba udaljiti ruke od uređaja *Leap Motion*.

4.1 Spajanje s uređajem

Za početak rada uređajem *Leap Motion* u razvojnom okruženju *Unity* potrebno je integrirati uređaj sa službenim paketom koji se preuzima s pomoću instrukcija sa stranice „docs.ultraleap.com“.

Prvi je korak za instalaciju paketa otvoriti *Project Settings* i u *Package Manager* dodati registar imenom *Ultraleap*, u polje URL staviti „https://package.openupm.com“ te u polje *Scope* umetnuti *com.ultraleap*.

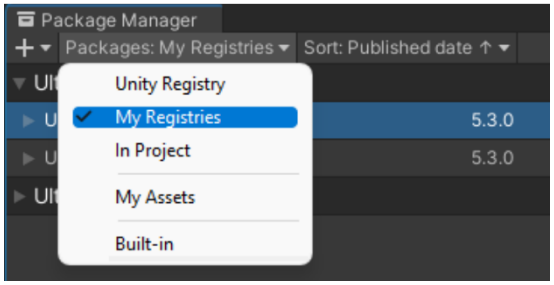
Slika 18. *Package Manager*



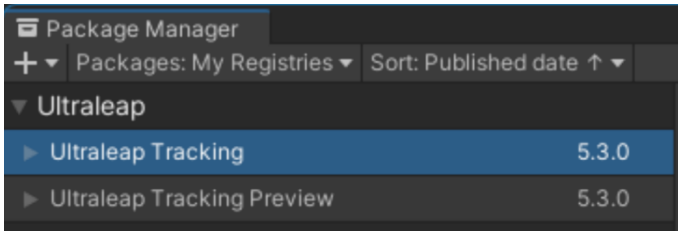
Izvor: docs.ultraleap.com/xr-and-tabletop/xr/unity/getting-started/

Tada je potrebno otvoriti prozor *Package Manager*, u padajućem izborniku izabrati *My Registries* te unijeti u projekt *Ultraleap Tracking*.

Slika 19. *My Registries* u padajućem izborniku



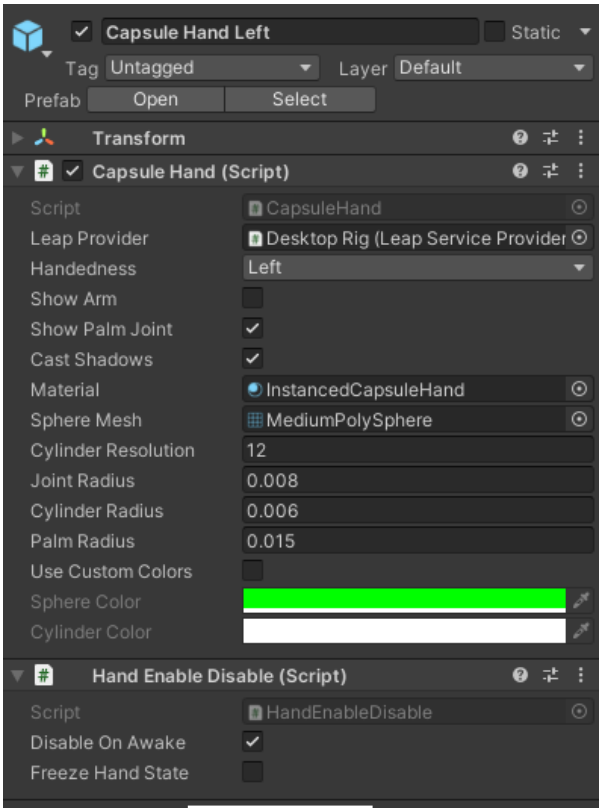
Slika 20. Paket *Ultraleap Tracking*



Izvor: docs.ultraleap.com/xr-and-tabletop/xr/unity/getting-started/

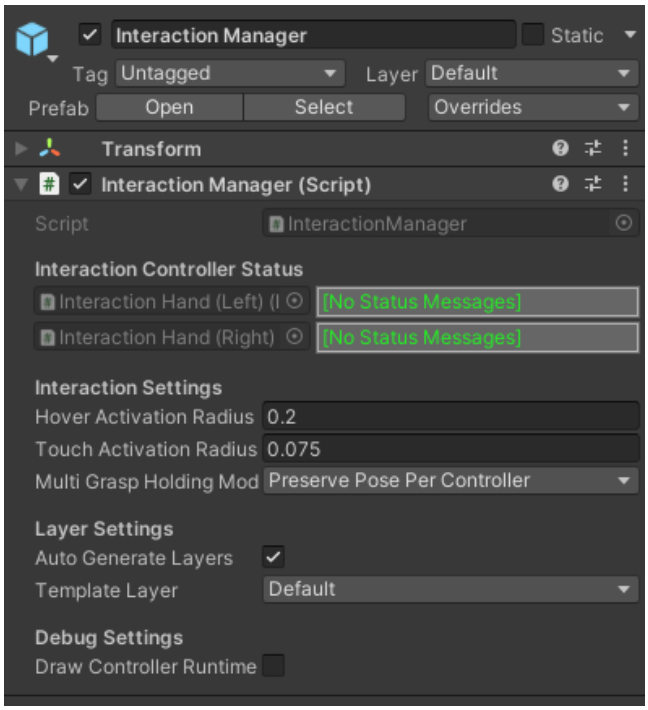
Paket sadrži prefabe *Capsule Hands*, što su virtualne ruke korisnika, i *Interaction Manager* koji je potreban da aplikacija prepoznaje uređaja *Leap Motion* i ruke korisnika. Također sadrži skriptu *InteractionBehaviour* koju je potrebno staviti na objekte koje bi korisnik morao moći primiti.

Slika 21. *Capsule Hand* prefab



Izvor: Autor

Slika 22. *Interaction Manager* prefab

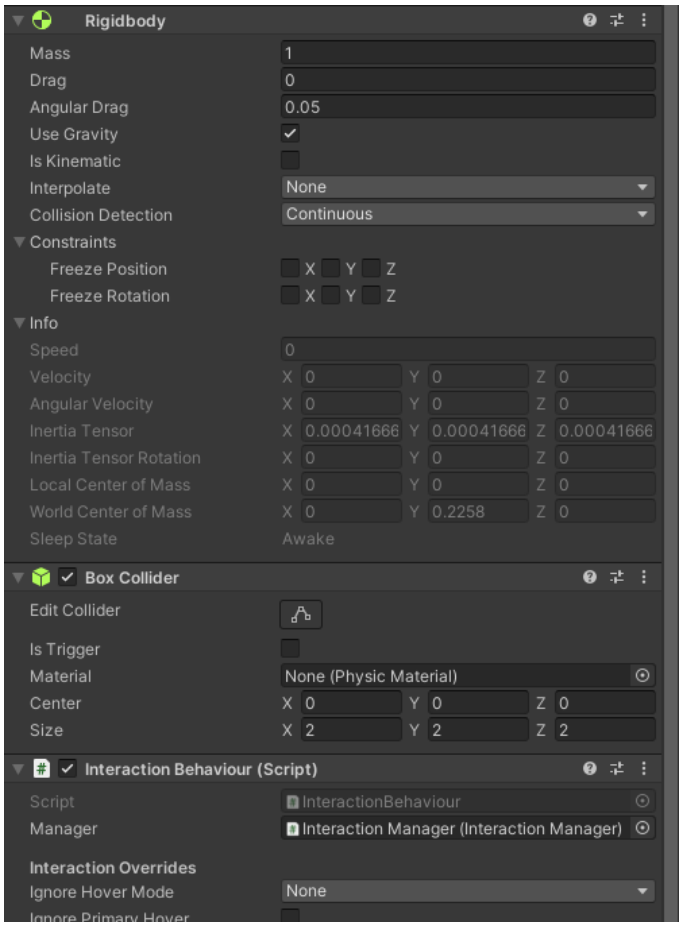


Izvor: Autor

4.2 Objekti aplikacije

Svaki objekt koji igrač može uhvatiti u sebi ima komponente *Rigidbody*, *Mesh Collider* i skriptu *Interaction Behaviour*. *Rigidbody* daje objektu fiziku. Bez njega bi objekt stajao na jednom mjestu i bilo bi nemoguće pomaknuti ga, bio bi zamrznut na jednom mjestu. *Mesh Collider* omogućuje objektu da fizički reagira na druge objekte u sceni. Bez njega bi pao kroz pod i ne bi ga bilo moguće uhvatiti. Također, bilo bi nemoguće utjecati na objekt skriptama jer sve skripte koriste funkciju *OnColliderEnter* koja zahtijeva da objekt u sebi ima komponentu *Collider*. Zadnja je komponenta skripta *InteractBehaviour* koja omogućuje virtualnim rukama igrača fizičku interakciju s objektom.

Slika 23. Komponente objekta

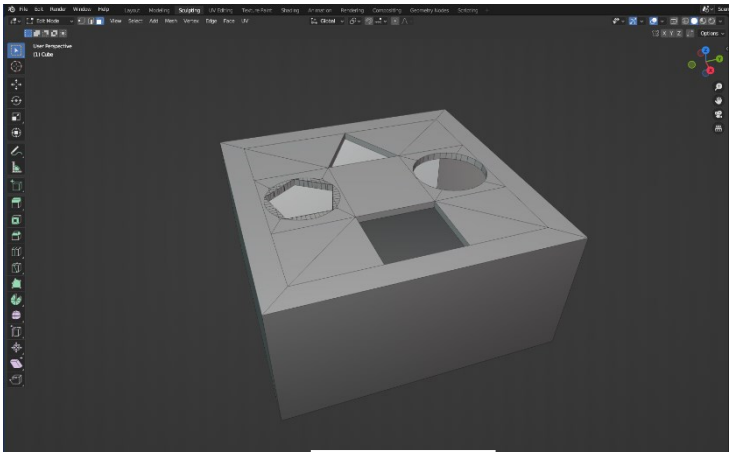


Izvor: Autor

4.2.1 Rad u blenderu

Za kreiranje kutije za scenu *Sortiranje oblika* potreban je softver za 3D modeliranje *Blender*. Za početak je potrebno stvoriti jednostavnu kocku prečicom Shift+A i klikom na *Cube* te u načinu rada *Edit Mode* iz gornje strane kocke izrezati trokut, kvadrat, peterokut i krug.

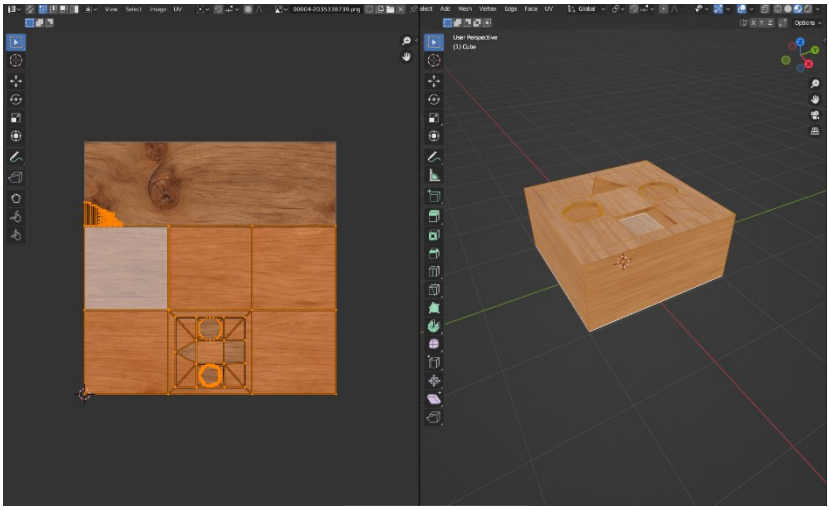
Slika 24. *Blender* model kutije za sortiranje oblika



Izvor: Autor

Nakon modeliranja kocke potrebno ju je teksturirati. Za teksturiranje se koristi *UV Editing*.

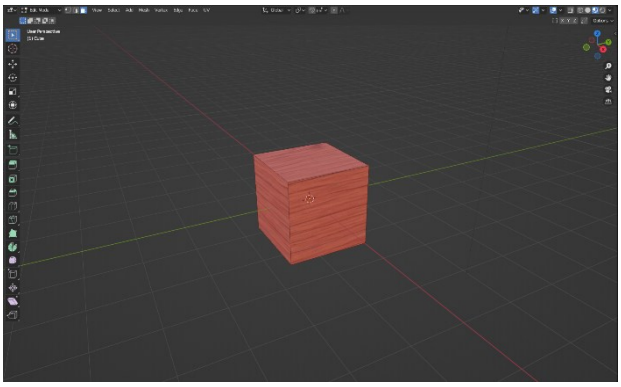
Slika 25. UV odmatanje kutije za sortiranje oblika



Izvor: Autor

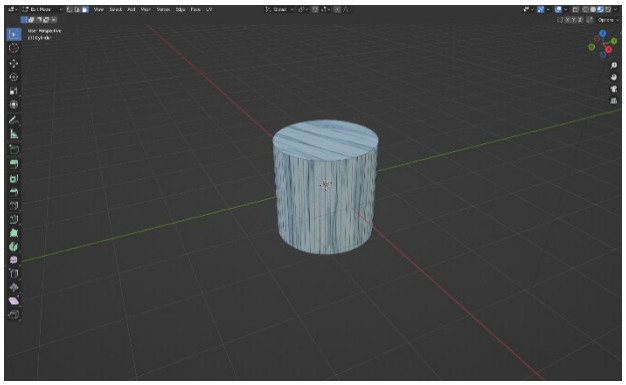
Za kreiranje objekata koje igrač može uhvatiti također se koristi *Blender*. Objekti cilindar i kocka ugrađeni su u *Blender* pa ih nije potrebno posebno kreirati, nego samo umetnuti prečicom Shift+A. Jedino je potrebno modelirati objekte peterokutna prizma i trokutna prizma.

Slika 26. Kocka u *Blenderu*



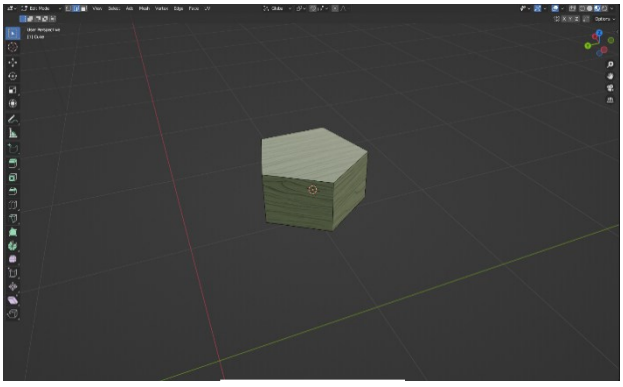
Izvor: Autor

Slika 27. Cilindar u *Blenderu*



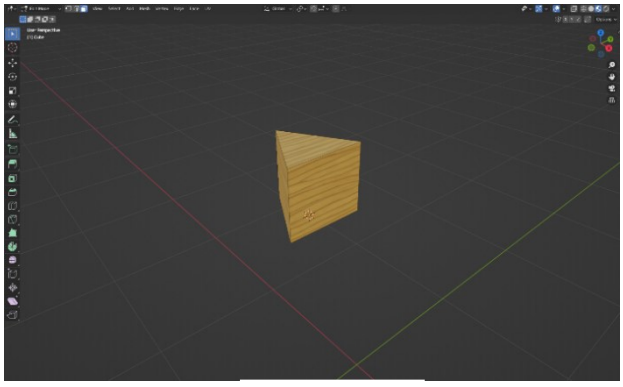
Izvor: Autor

Slika 28. Peterokutna prizma u *Blenderu*



Izvor: Autor

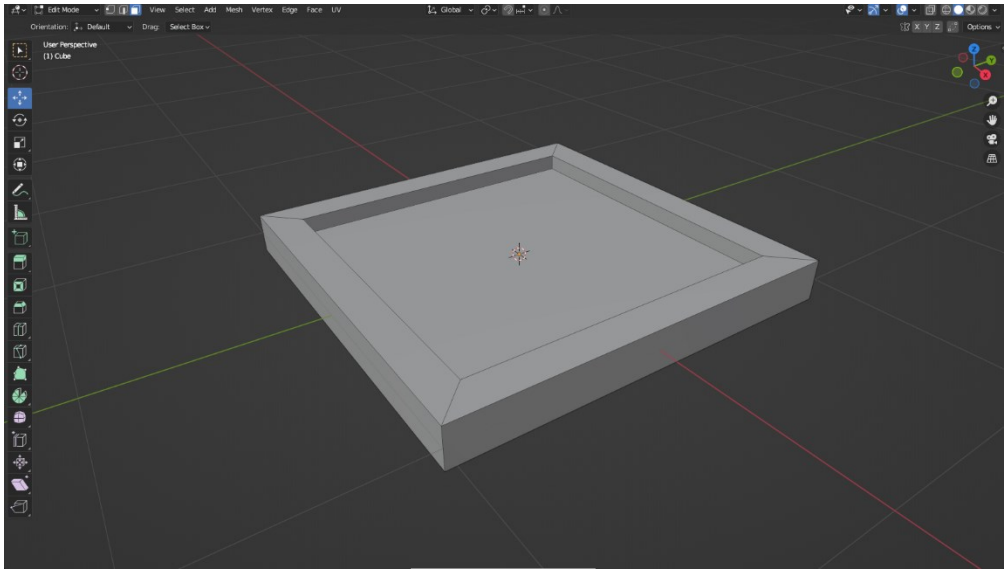
Slika 29. Trokutna prizma u *Blenderu*



Izvor: Autor

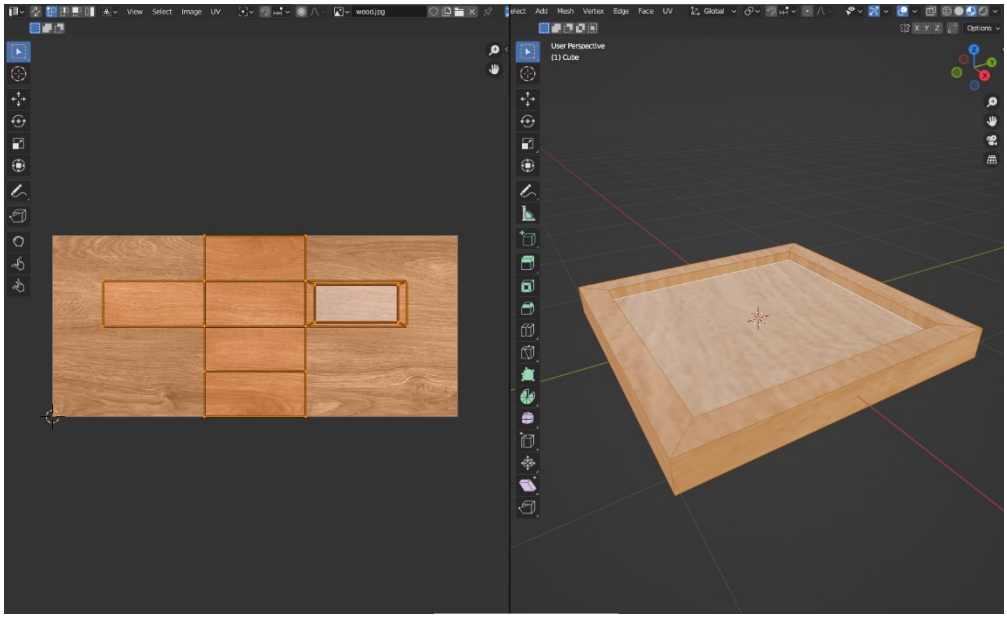
Drvena ploča u igri *Slagalica* je jednostavan oblik. Potrebno je samo stvoriti kocku, spljoštiti ju, te sa alatima *Inset* i *Extrude* napraviti udubinu.

Slika 30. *Blender* model drvene ploče



Izvor: Autor

Slika 31. UV odmatanje drvene ploče



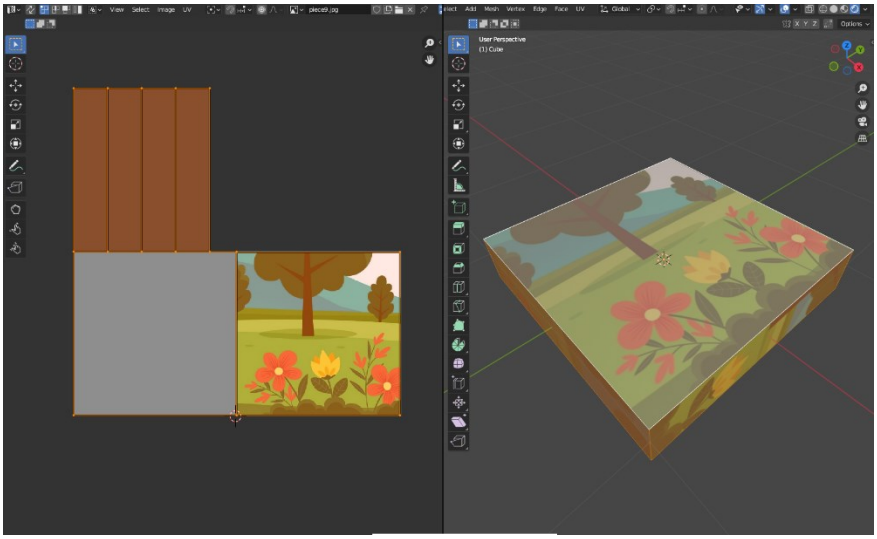
Izvor: Autor

Dijelovi slagalice napravljeni su izrezom jednostavne crtane slike u devet dijelova. Izrezani dijelovi slike se tada primjenjuju na objekte pomoću UV odmatanja.

Slika 32. Crtana slika

Izvor: Autor

Slika 33. UV odmatanje dijela slagalice



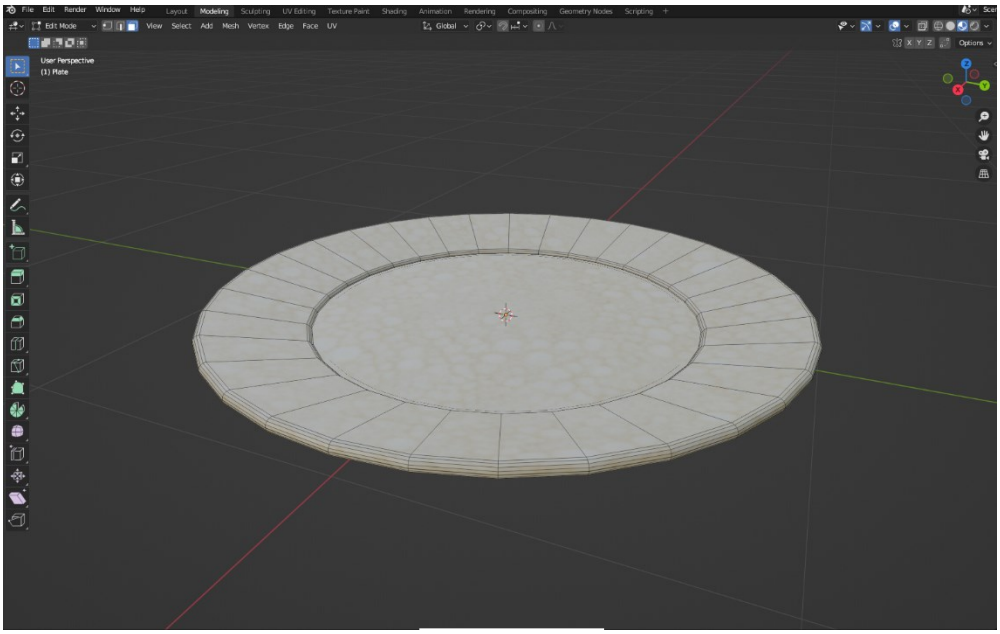
Izvor: Autor

Stol i tanjuri u sceni za odabir igre također su izrađeni u *Blenderu*.

Slika 34. Stol u *Blenderu*

Izvor: Autor

Slika 35. Tanjur u *Blenderu*



Izvor: Autor

4.3 Scene aplikacije

4.3.1 Početna scena

Početna scena sadrži naslov aplikacije i tri teksta koja korisnik može kliknuti. Svaki od tih tekstova u sebi ima komponentu *Button* koja omogućuje korisniku da ih klikne i pozove funkciju. Tekst *Početak* poziva funkciju *LoadLevel()* koja otvara sljedeću scenu, a tekst *Kraj* poziva funkciju *QuitRequest()* koja zatvara aplikaciju.

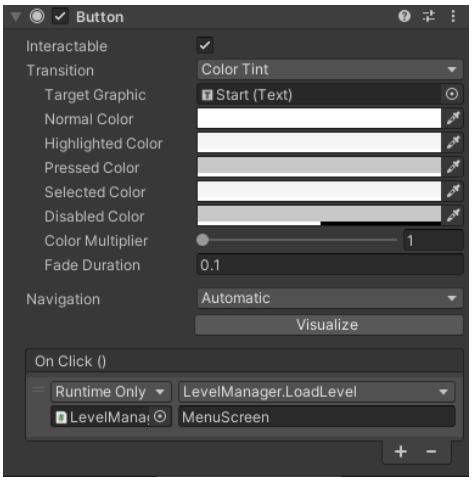
Kod 1. Funkcije *LoadLevel()* i *QuitRequest()*

```
0 references
public void LoadLevel(string name)
{
    Debug.Log("Loading level:" + name);
    SceneManager.LoadScene(name);
}

0 references
public void QuitRequest()
{
    Debug.Log("Quitting the game.");
    Application.Quit();
}
```

Izvor: Autor

Slika 36. Komponenta *Button*



Izvor: Autor

Tekst *Objašnjenje* poziva funkciju *Tutorial()* koja mijenja elemente scene i daje korisniku opciju objašnjenja za dvije igre. Klikom na *Sortiranje oblika* ili *Slagalica* poziva funkcije *ShapeTutorial()* ili *PuzzleTutorial()* koje vode do njihovih objašnjenja. Klikom teksta *Natrag* poziva funkcije *Back()* ili *Back2()*, ovisno o trenutačnoj stranici.

Kod 2. *Tutorial* funkcije

```
0 references
public void Tutorial()
{
    page1.SetActive(false);
    tutorialPage.SetActive(true);
}

0 references
public void Back()
{
    page1.SetActive(true);
    tutorialPage.SetActive(false);
}

0 references
public void ShapeTutorial()
{
    tutorialPage.SetActive(false);
    shapeGameTutorial.SetActive(true);
}

0 references
public void PuzzleTutorial()
{
    tutorialPage.SetActive(false);
    puzzleGameTutorial.SetActive(true);
}
0 references
public void Back2()
{
    tutorialPage.SetActive(true);
    shapeGameTutorial.SetActive(false);
    puzzleGameTutorial.SetActive(false);
}
```

Izvor: Autor

4.3.2 Scena za objašnjenje korištenja uređaja

Scena sadrži kratak tekst koji objašnjava igraču kako koristiti uređaj *Leap Motion*. Nakon što korisnik pročita tekst i stavi ruke iznad uređaja, skripta *LevelManager* započinje brojač. Kada brojač istekne, skripta šalje korisnika na sljedeću scenu. Skripta radi tako da u sceni traži objekte virtualne ruke igrača imenom *Capsule Hand Left* i *Capsule Hand Right*. Nakon što igrač stavi ruke iznad uređaja, stvore se virtualne ruke i skripta započinje brojač s funkcijom *StartTimer()*. Ako korisnik prestane držati ruke iznad uređaja *Leap Motion*, virtualne ruke nestanu i skripta zaustavlja brojač funkcijom *CancelInvoke()*.

Kod 3. Skripta *LevelManager*

```
Unity Message | 0 references
void Update()
{
    if(GameObject.Find("Capsule Hand Left") || GameObject.Find("Capsule Hand Right")) {
        if(!timerBool) {
            timerBool = true;
            StartTimer();
        }
    }
    else {
        CancelInvoke();
        timer.text = "";
        timerValue = 5;
        timerBool = false;
    }
}

1 reference
void StartTimer() {
    Invoke("changeScene", 5);
    InvokeRepeating("Timer", 1, 1);
    timer.text = timerValue.ToString();
}

0 references
void Timer() {
    timerValue -= 1;
    timer.text = timerValue.ToString();
}

0 references
void changeScene() {
    SceneManager.LoadScene("PickGame");
}
```

Izvor: Autor

4.3.3 Scena za odabir igre

Oba tanjura sadrže skriptu koja koristeći funkcije *OnCollisionEnter()* poziva funkciju za promjenu scene samo ako ih kocka dodiruje. Ako kocka dodiruje lijevi tanjur, skripta lijevoga tanjura poziva funkciju *changeSceneShapeGame()*, a ako dodiruje lijevog tanjura, skripta desnog tanjura poziva funkciju *changeScenePuzzleGame()*. Funkcije za promjenu scene pozivaju se pet sekundi nakon dodira funkcijom *Invoke()*. Ako kocka prestane dodirivati tanjur, poziva se funkcija *OnCollisionExit()* koja resetira brojač i funkcijom *CancelInvoke()* otkazuje poziv funkcije za promjenu scene. Nakon što brojač istekne, skripta šalje korisnika na scenu koju je odabrao.

Kod 4. Skripta za odabir igre

```
Unity Message | 0 references
void OnCollisionEnter(Collision collision)
{
    if (collision.gameObject.name == "Cube")
    {
        if (this.gameObject.name == "PickShapeGame")
        {
            Invoke("changeSceneShapeGame", 5);
            shapeGame.enabled = true;
        }
        else if (this.gameObject.name == "PickPuzzleGame")
        {
            Invoke("changeScenePuzzleGame", 5);
            puzzleGame.enabled = true;
        }
        InvokeRepeating("Timer", 1, 1);
        timer.text = timerValue.ToString();
    }
}

Unity Message | 0 references
void OnCollisionExit(Collision collision)
{
    CancelInvoke();
    timer.text = "";
    timerValue = 5;
    shapeGame.enabled = false;
    puzzleGame.enabled = false;
}

0 references
void Timer() {
    timerValue -= 1;
    timer.text = timerValue.ToString();
}

0 references
void changeSceneShapeGame() {
    SceneManager.LoadScene("ShapeGame");
}

0 references
void changeScenePuzzleGame() {
    SceneManager.LoadScene("PuzzleGame");
}
```

Izvor: Autor

4.3.4 Sortiranje oblika i slagalica

Scena za sortiranje oblika sadrži četiri objekta, kutiju s odgovarajućim otvorima i brojač bodova. Svaki otvor u sebi ima nevidljiv objekt sa komponentama *Collider i ColliderScript* koji prepoznaje odgovarajući objekt koristeći funkcije *OnTriggerEnter()*. Kada korisnik stavi objekt u točan otvor i dodirne nevidljivi objekt, skripta uništava objekt funkcijom *Destroy()*, daje igraču jedan bod i napravi zvuk koristeći funkciju *PlayClipAtPoint()*. Scena slagalice radi na istom principu te koristi istu skriptu. Jedina je razlika što nevidljiv objekt, kada se dodirne, postane vidljiv te ima isti oblik kao i dio slagalice kojim ga igrač mora dodirnuti.

Kod 5. Skripta *ColliderScript*

```
Unity Message | 0 references
void OnTriggerEnter(Collider collider)
{
    if(collider.gameObject == assignedObject) {
        if (gameObject.name.Contains("Puzzle"))
            GetComponent<Renderer>().enabled = true;

        Destroy(collider.gameObject);
        LevelManagerScript.counterValue += 1;
        LevelManagerScript.counter.text = LevelManagerScript.counterValue.ToString();
        AudioSource.PlayClipAtPoint(scoreSound, Camera.main.transform.position);
    }
}
```

Izvor: Autor

Oba dvije igre imaju u sebi nevidljiv objekt imenom *Catcher* koji se nalazi na dnu scene. On u sebi ima komponentu *Collider* i skriptu *CatcherScript*. Kada objekt padne iz dosega igrača i dodirne taj nevidljiv objekt, on ga tada šalje na njegovu početnu poziciju. Svi objekti scene su uneseni u listu *objects*. Početne pozicije svakog objekta su uneseni u listu *startPosition*, a početne rotacije u listi *startRotation*. Kada objekt dodirne nevidljiv objekt *Catcher*, poziva se funkcija *OnTriggerEnter*, koja tada stavlja objekt na njegovu početnu poziciju sa njegovom početnom rotacijom.

Kod 6. Skripta *CatcherScript*

```
public List<GameObject> objects;
public List<Vector3> startPositions;
public List<Quaternion> startRotation;

public GameObject currentObject;

Unity Message | 0 references
private void Start()
{
    foreach (GameObject obj in objects)
    {
        startPositions.Add(obj.transform.position);
        startRotation.Add(obj.transform.rotation);
    }
}

Unity Message | 0 references
void OnTriggerEnter(Collider collider)
{
    for (int i = 0; i < objects.Count; i++)
    {
        if (objects[i] != null && collider.gameObject.name == objects[i].name)
        {
            currentObject = objects[i];
            currentObject.transform.position = startPositions[i];
            currentObject.transform.rotation = startRotation[i];

            currentObject.GetComponent<Rigidbody>().constraints = RigidbodyConstraints.FreezeAll;
            Invoke("Unfreeze", 0.5f);
        }
    }
}

0 references
void Unfreeze()
{
    currentObject.GetComponent<Rigidbody>().constraints = RigidbodyConstraints.None;
}
}
```

Izvor: Autor

Nakon što igrač osvoji sve bodove, skripta *LevelManagerGame* proizvodi zvuk pobjede i stvara tekst *Uspjeh!* Skripta tada funkcijom *changeScene()* šalje igrača natrag na scenu za odabir igre.

Kod 7. Skripta *LevelManager*Game

```
Unity Message | 0 references
void Update()
{
    if(GameObject.Find("Capsule Hand Left") == null && GameObject.Find("Capsule Hand Right") == null) {
        if(!timerBool && !uspjehBool) {
            timerBool = true;
            StartTimer();
            exit.enabled = true;
        }
    }
    else {
        if(!uspjehBool) {
            CancelInvoke();
            timer.text = "";
            timerValue = 5;
            timerBool = false;
            exit.enabled = false;
        }
    }

    if(counterValue == requiredPoints) {
        uspjehBool = true;
        uspjeh.text = "Uspjeh!";
        Invoke("changeScene", 5);
        if (!playClipOnce)
        {
            AudioSource.PlayClipAtPoint(winSound, Camera.main.transform.position);
            playClipOnce = true;
        }
    }
    else uspjeh.text = "";
}

1 reference
void StartTimer() {
    Invoke("changeScene", 5);
    InvokeRepeating("Timer", 1, 1);
    timer.text = timerValue.ToString();
}

0 references
void Timer() {
    timerValue -= 1;
    timer.text = timerValue.ToString();
}

0 references
void changeScene() {
    SceneManager.LoadScene("StartMenu");
}
```

Izvor: Autor

5. ZAKLJUČAK

Može se zaključiti da razvoj igara za *Leap Motion* tehnologiju predstavlja izazov, ali i priliku za unapređenje interakcije s igračima i proširivanje horizonta videoigara. *Leap Motion* tehnologija daje korisniku mogućnost upravljanja računalom i igranje računalnih igara na potpuno nov i imerzivan način. Daje korisniku mogućnost interakcije s objektom kao i u stvarnom životu te otvara vrata za nova i jedinstvena iskustva u svijetu razvoja videoigara.

Također, postoji mogućnost razvijanja *Leap Motion* aplikacija za druge svrhe osim zabave, kao što su, na primjer, edukacije ili manipuliranje objektom u softverima za izradu 3D modela. *Leap Motion* također pruža mogućnost pomaganja ljudima s otežanom pokretljivošću te bi se mogao koristiti i za rehabilitaciju ljudi s artritisom ili cerebralnom paralizom.

Literatura

[1] https://www.ultraleap.com/datasheets/Leap_Motion_Controller_Datasheet.pdf, (28.8.2024)

[2] https://www.ultraleap.com/company/news/press-release/imageholders-touchless-kiosk, (28.8.2024)

[3] http://unity3d.com/unity, (28.8.2024)

[4] https://visualstudio.microsoft.com/, (28.8.2024)

[5] https://www.blender.org/about/, (28.8.2024)

[6] https://www.futurelearn.com/info/blog/general/what-is-3d-modelling, (28.8.2024)

[7] https://conceptartempire.com/uv-mapping-unwrapping/, (28.8.2024)

[8] https://www.a23d.co/blog/what-is-3d-texturing, (28.8.2024)

Prilozi

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU

Bana Josipa Jelačića 22/a, Čakovec

IZJAVA O AUTORSTVU

Završni/diplomski rad isključivo je autorsko djelo studenta te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, internetskih i drugih izvora) bez pravilnog citiranja. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom i nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, Juraj Budiša (ime i prezime studenta) pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog rada pod naslovom

INTERAKCIJA POMOĆU LEAP MOTIONA U APLIKACIJE RAZVIJENE U RAZVOJNOM OKRUŽENJU UNITY

te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:

Juraj Budiša

(vlastoručni potpis)