Soft kompjuting

Primena Leap motion uređaja za prirodnu interakciju sa objektima iz virtualnog sveta korišćenjem Unity razvojonog okruženja

Nikola Vitanović 761

Mentor:

prof. dr. Aleksandar Milosavljević

**Sadržaj**

[Unity razvojno okruženje 3](#_Toc80481151)

[Leap motion 4](#_Toc80481152)

[Koordinatni sistem 6](#_Toc80481153)

[Podaci o praćenju kretanja ruku 7](#_Toc80481154)

[Vridge 9](#_Toc80481155)

[SteamVR 12](#_Toc80481156)

[Projekat 12](#_Toc80481157)

[Unity SteamVR dodatak 12](#_Toc80481158)

[Unity Gemini Leap Motion dodatak 13](#_Toc80481159)

[Leap motion i VR naočare 14](#_Toc80481160)

[Unity Leap motion dodaci 15](#_Toc80481161)

[Instalacija dodataka 16](#_Toc80481162)

[Osnovna konfiguracija 17](#_Toc80481163)

[Prilagođeni objekat igrača 18](#_Toc80481164)

[Kretanje 20](#_Toc80481165)

[Ruke 21](#_Toc80481166)

[Interaktivni objekti 22](#_Toc80481167)

[Kutija 23](#_Toc80481168)

[Dugme 24](#_Toc80481169)

[Vrata 26](#_Toc80481170)

[Potisna ploča 29](#_Toc80481171)

[Igra 30](#_Toc80481172)

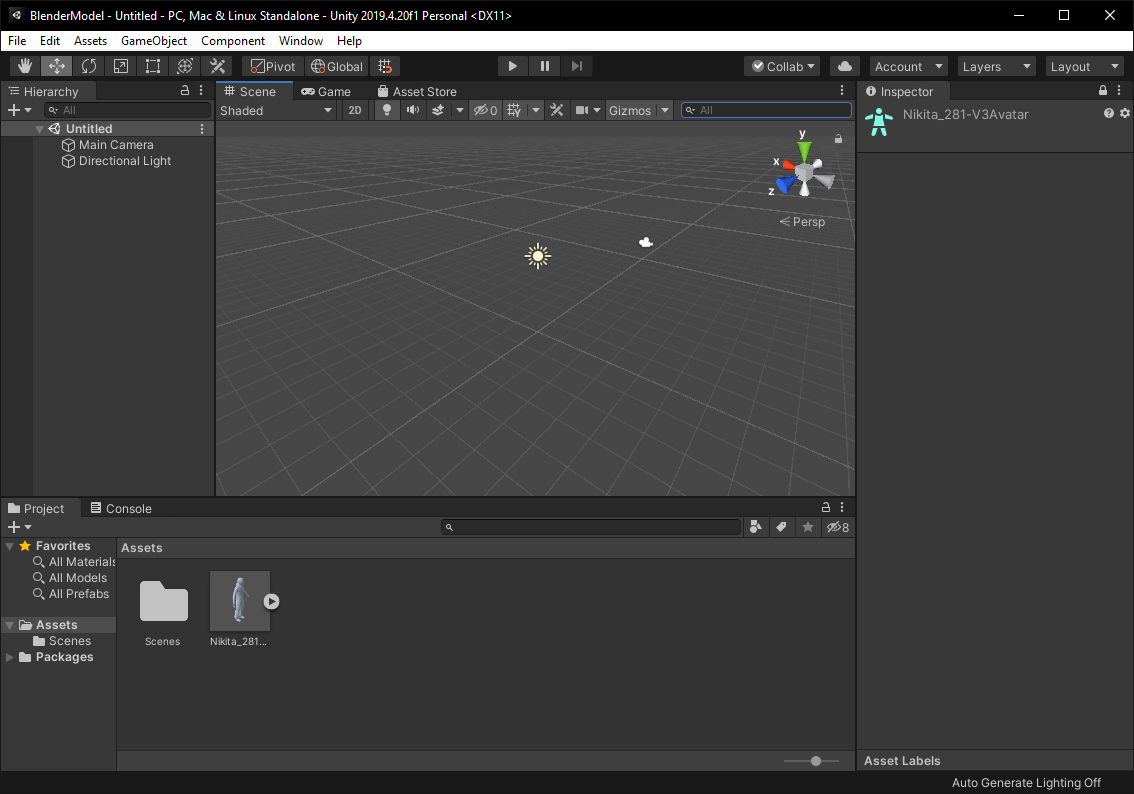
[Zaključak 32](#_Toc80481173)

[Reference 32](#_Toc80481174)

# Unity razvojno okruženje

Firma “Unity Technologies” osnovana je 2004. godine u malom apartmanu u Kopenhagenu [1]. Predstavlja softversku platformu za izradu kompjuterskih igara. Pored primene u izradi video igara, Unity se može iskoristiti za mnogo više. Sylvio Drouin potpredsednik “Untiy Labs R&D” želi da Unity ne bude samo alat za izradu video igara već da bude 3D operativni sistem ovog sveta. Unity se već koristi za izradu više od polovine igara današnjice, pre svega u indie segmentu zbog svoje lakoće, dokumentacije i velike zajednice.

Unity podržava izradu 3D, 2D, VR, AR igara i simulacija [2], dok se najviše koristi za izradu 3D igara. Program izarđen u Unity razvojnom okruženju moguće je izvesti na više od 25 različitih platformi. Za sve popularniji VR i AR sadržaj, Unity predstavlja najbolju platformu za razvoj. Pored izrade igara od 2010. godine Unity počinje da se dosta koristi u drugim industrijama kao što su automobilska, filmska i 3D industrija.



Slika 1 - Izgled Unity razvojnog okruženja.

Unity podržava pisanje skripti u C# i Javascript programskom jeziku, pored samog koda u jednom od ponuđenih jezika moćni vizuelni editor omogućava, animaciju modela, dodavanje tekstura, izradu nivoa, rad sa sistemima čestica, simulaciju fizike, audio i drugo.

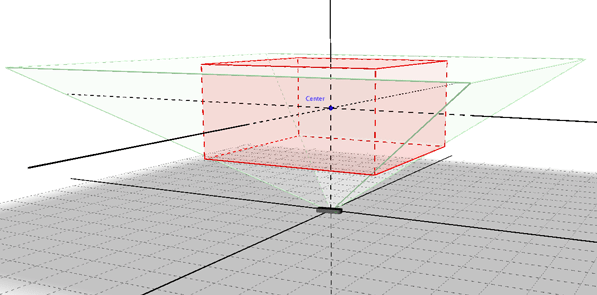
# Leap motion

*Leap motion* kontroler [3] koristi dve infracrvene stereo kamere sa tri infrared le diode koje osvetljuju prostor koji senzor posmatra. Sami detalji kako senzor funkcioniše su patentirani. Senzor ima i neke tehničke nedostatke kao što je ugao vidljivosti kamere od 150 stepeni, zbog koje se javlja distorzija slike. Takođe zbog korišćenja infra crvenog spektra, u zavisnosti od osvetljenja prostorije može doći do pogrešnog očitavanja senzora, čak zbog nečistoće zaštitnog stakla senzora može doći do grešaka. Konačno daljina detektovanja ruka zavisi od jačine osvetljenja od strane tri infracrvene le diode.



Slika 2 – Evolucija Leap motion uređaja.

Senzor je najprecizniji u centralnoj zoni predstavljenom kockom u 3D prostoru, tako da ukoliko se ruka nalazi van ili na ivici zone senzora može doći do grešaka prilikom detekcije ruku. Dubinu predstavlja Y osa i minimalna dubina koju senzor vidi je 82.5mm dok je maksimalna dubina 317.5mm. Z osa je normalna na dužu stranu senzora detektuje objekat od -73.5mm do 73.5mm dok X osa detektuje objekat između -117.5mm do 117.5mm.



Slika 3 - polje detektovanja ruku sa leap motion senzorom.



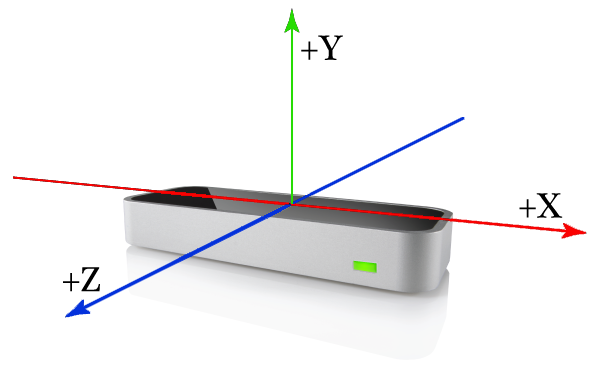
Slika 4 - štampane ploče rasturenog leap motion uređaja sa sočivima na infracrvenim kamerama.

Proizvođač uređaja nudi *Leap Motion SDK* koji omogućava laku integraciju sa raznim programskim jezicima i razvojnim okruženjima kao što je *Unity*. Na nižem nivou moguće je pristupiti senzoru i raditi samom slikom, dok na višem nivou *SDK* omogućava rad sa pokretima ruku, gde je dovoljno definisati određene šablone koji će biti prepoznati. Takođe moguće je dobiti informacije o poziciji i transformaciji ruke, prstiju, zglobovima prstiju. Senzor podržava obe ruke čime se omogućava veća kontrola u okviru virtuelnog okruženja ili aplikaciji koju korisnik pokušava da izradi.

## Koordinatni sistem

*Leap motion* uređaj koristi desni Dekartov koordinatni sistem. Početak koordinatnog sistema se nalazi na vrhu uređaja u centru. X i Z ose su u horizonaltnoj ravni dok Y osa predstavlja vertikalnu ravan. Uređaj koristi sledeće jedinice:

|  |  |
| --- | --- |
| Daljina | milimetri |
| Vreme | mikrosekunde |
| Brzina | milimetri po sekundi |
| Ugaio | radijani |



Slika 5 - koordinatni sistem Leap motion uređaja.

Unity softver koristi levi koordinatni sistem, što znači da Z osa pokazuje suprotno od korisnika, dok je X osa desna a Y predstavlja gore. Zbog toga vrlo je bitno da se kooridnate pomnože sa -1 kako bi se korektno mapirale u Unity prostoru.

Bitno je takođe da objekti ne prilaze veoma blizu *Leap motion* uređaju, primera radi ako bi stavili desnu ruku tik uz senzore uređaja leva ruka ne bi bila vidljiva, tako da predviđanje lokacije leve ruke ne bi bilo precizno ili moguće.

*Leap motion* uređaj je inicijalno bio namenjen da stoji normalno na stolu korisnika sa rukama iznad uređaja tako da je korisnik sa jedne strane dok sa druge strane se nalazi osoba. Korisnik bi u ovom slučaju bio ispred (+Z) dok bi se monitor nalazio u negativnom delu (-Z) ose. Ukoliko korisnik uključi automatsku orijentaciju, Leap motion softver bi automatski odredio koordinatni sistem u slučaju da je uređaj izokrenut (zelena LED suprotno od korisnika), ali ukoliko korisnik drugačije orijentiše uređaj, napočake ili na stranu Leap motion softver nije u mogućnosti da ovo automatski prepozna i podesi koordinatni sistem. Zbog toga postoji opcija koja specificira kako se *Leap motion* uređaj montira i koristi o kojoj će biti reči u jednom od narednih poglavlja.

## Podaci o praćenju kretanja ruku

Uređaj prati ruke i prste u svom vidnom polju i pruža najnovije informacije u svakom frejmu slike koju vidi. Svaki frejm objekat sadrži ruke i njena svojstva u prostoru u tom trenutku. Frejm objekat je esencijalno osnova *Leap motion* modela podataka.

Model ruku pruža informacije o raznim karakteristikama detektovane ruke, da li je u pitanju desna ili leva ruka i informacije o pozicijama prstiju. Ruka je predstavljena posebnom klasom. Leap motion koristi interni model ljudske ruke da bi predvideo praćenje delova čak i kada ruka nije vidljiva. Model ruke uvek sadrži informaciju o poziciji svih 5 prstiju, ali praćenje je najidealnije kada se silueta ruku i svih prstiju vidi odlično na uređaju. Softver koristi vidljivi deo ruke kao interni model i prethodne pozicije prstiju i ruke da preračuna i predvidi najverovatniju poziciju delova koji nisu vidljivi. Naravno ovaj sistem može da ima nedostatke i da povremeno loše proceni poziciju i orijentaciju u slučaju da neki deo nije vidljiv ili lepo prepoznat. Postoji takozvani rejting sigurnosti koji nam daje procentualnu vrednost koliko je sistem siguran za trenutnu detekciju.

Takođe moguće je da obe ruke budu vidljive u jednom trenutku, to ne predstavlja problem sistemu, ali ukoliko senzor prepozna više od jednog para ruku može doći do neočekivanog praćenja, tako da je savet da senzor vidi samo jedan par ruku u trentuku.

Sistem detektuje i podlaktice i pruža informaciju o orijentaciji, dužini i širini, automatski ih vezuje za ruku. Kada podlaktica nije vidljiva softver pokušava da pretpostavi kako je orijentisana i aprokismira je na osnovu prethodnog stanja.

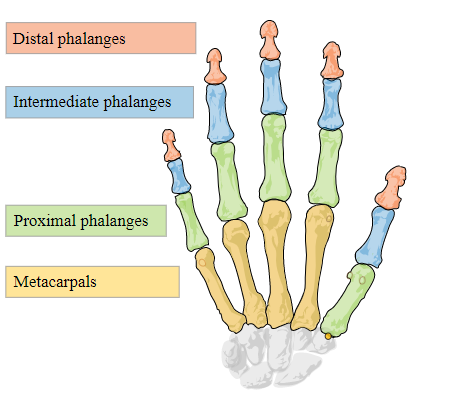
Prsti se detektuju takođe i svaki od njih je indeksiran u softveru po normalnom nazivu na engleskom. Pretpostavka je da ruka sadrži 5 pristiju, kada neki od njih nije u vidnom polju softver pokušava da pretpostavi lokaciju i karakteristiku na osnovu prethodne lokacije i anatomskog modela ruku. Vrhovi prstiju sadrže vektore koji služe da se odredi smer u kome je prst uperen.



Slika 6 - vrhovi prstiju su predstavljeni kao vektori.

Svaki od prstiju sastoji se od manjih delova tj. koski koji se individalno detektuju. Kosti prstiju takođe imaju informaciju o orijentaciji i poziciji. Prst se sastoji iz 4 kosti od osnove šake do vrha prsta.

* *Metacarpal* – kost koja povezuje ručni zglob i prst, ovo se odnosi na sve prste osim palca koji ne sadrži ovu kost.
* *Proximal Phalanx* – kost na sredini ruke, koja vezuje ostatak prsta i šaku.
* *Intermediate Phalanx* – srednja kost prsta između vrha i osnove prsta.
* *Distal Phalanx* – vrh prstiju.



Slika 7 - delovi kostiju od kojih se sastoje prsti.

Izuzetak prestavlja palac koji se ne slaže u potpunosti sa anatomskim standardom za imenovanje. Pravi palac ima jednu manje kost od drugih prstiju. Leap motion zbog lakoće implementacije ipak sadrži i tu nepostojeću kost, ali je ne prikazuje, kako bi omogućio da prsti budu jednako indeksirani u softveru.

# Vridge

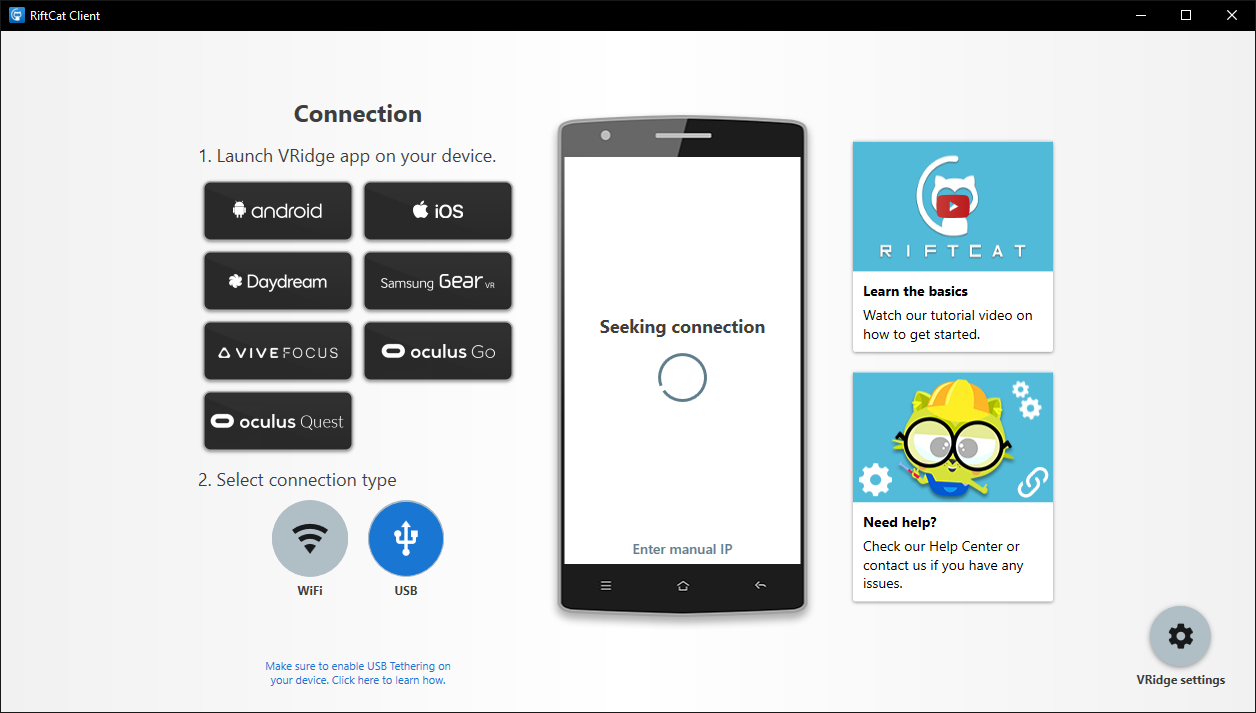
Vridge [4] je softver koji omogućava osobama da bez velikog ulaganja u skupu VR opremu iskuse i igraju poznate VR igre. Korišćenjem mobilnog telefona i VR naočara bez ekrana, gde se umesto ekrana umeće mobilni moguće je iskusiti VR.



Slika 8 – Izgled Google Cardboard naočara za virtuelnu realnost.

Vridge softver razvija firma RiftCat iz Poljske. Postoje dve verzije softvera, besplatna koja omgućava korisniku da igra bilo koju VR igru u sesiji od 10 minuta, kao i plaćena verzija koja omogućava neograničene sesije.

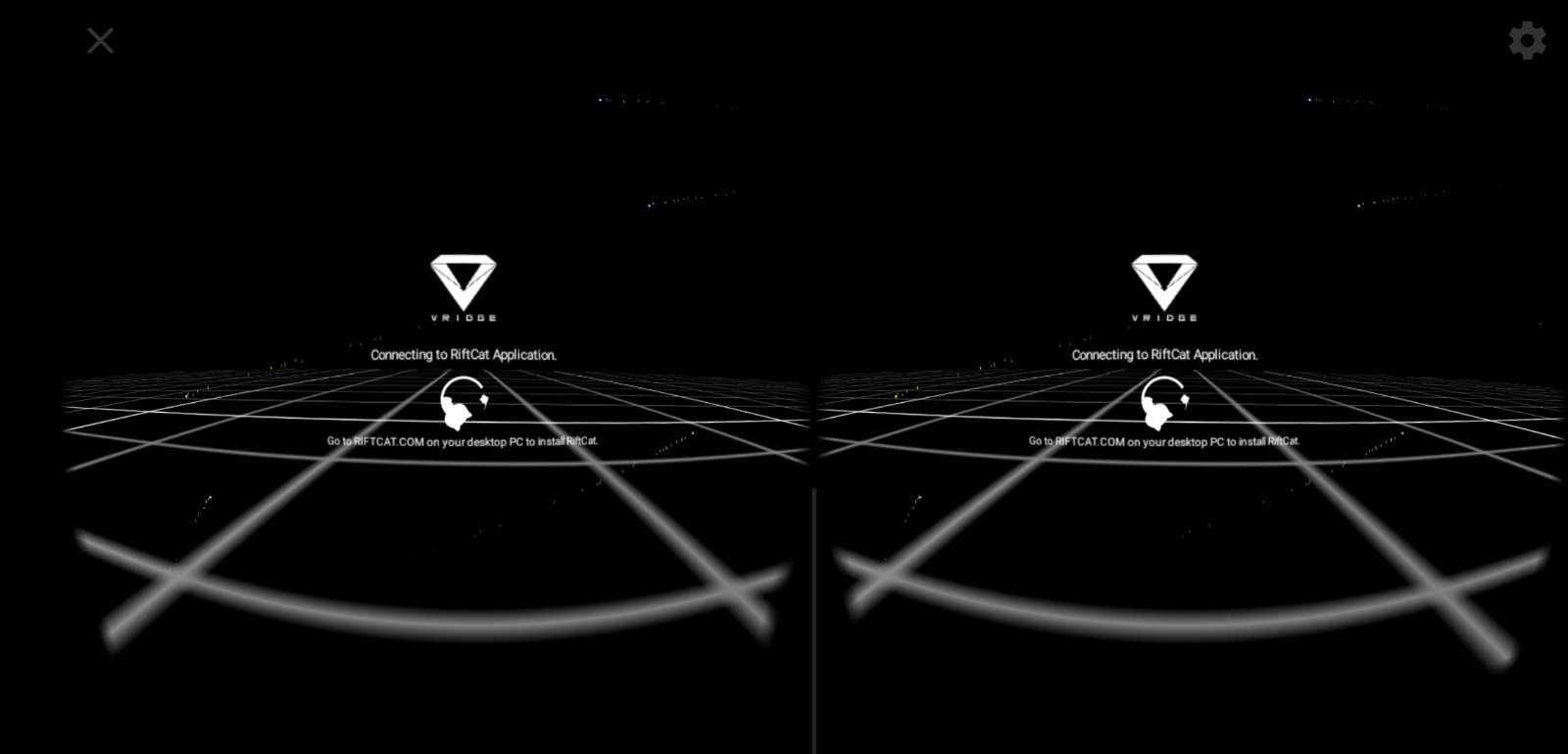
Za potrebe ovog rada kupljena je komercijalna verzija softvera koja omogućava neograničene sesije. Instalacija softvera je veoma jednostavna, potrebno je preuzeti softver sa njihove oficijalne stranice i instalirati. Nakon instalacije softver je spreman i vodi vas kroz čarobnjaka za podešavanje telefona i povezivanje. Na mobilnom telefonu potrebno je takođe preuzeti Vridge aplikaciju sa Google Play prodavnice.



Slika 9 - Izgled RiftCat VRidge aplikacije na računaru.

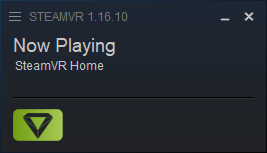
Nakon pokretanja RiftCat Vridge aplikacije potrebno je izabrati način povezivanje aplikacije. Moguće je izabrati bežično ili žičano povezivanje. Preporuka je koristiti žičano USB povezivanje radi kvalitetnije konekcijem, bolje rezolucije i niže latencije. Bežično povezivanje je takođe moguće, ali savetuje se korišćenje WiFi mreže na frekvenciji od 5GHz, ukoliko nemate ruter koji podržava ovu frekvenciju najbolje je povezati putem USB kabla.

Na telefonu je potrebno pokrenuti Vridge. Aplikacija na računaru će automatski prepoznati uređaj i povezati ga sa aplikacijom.

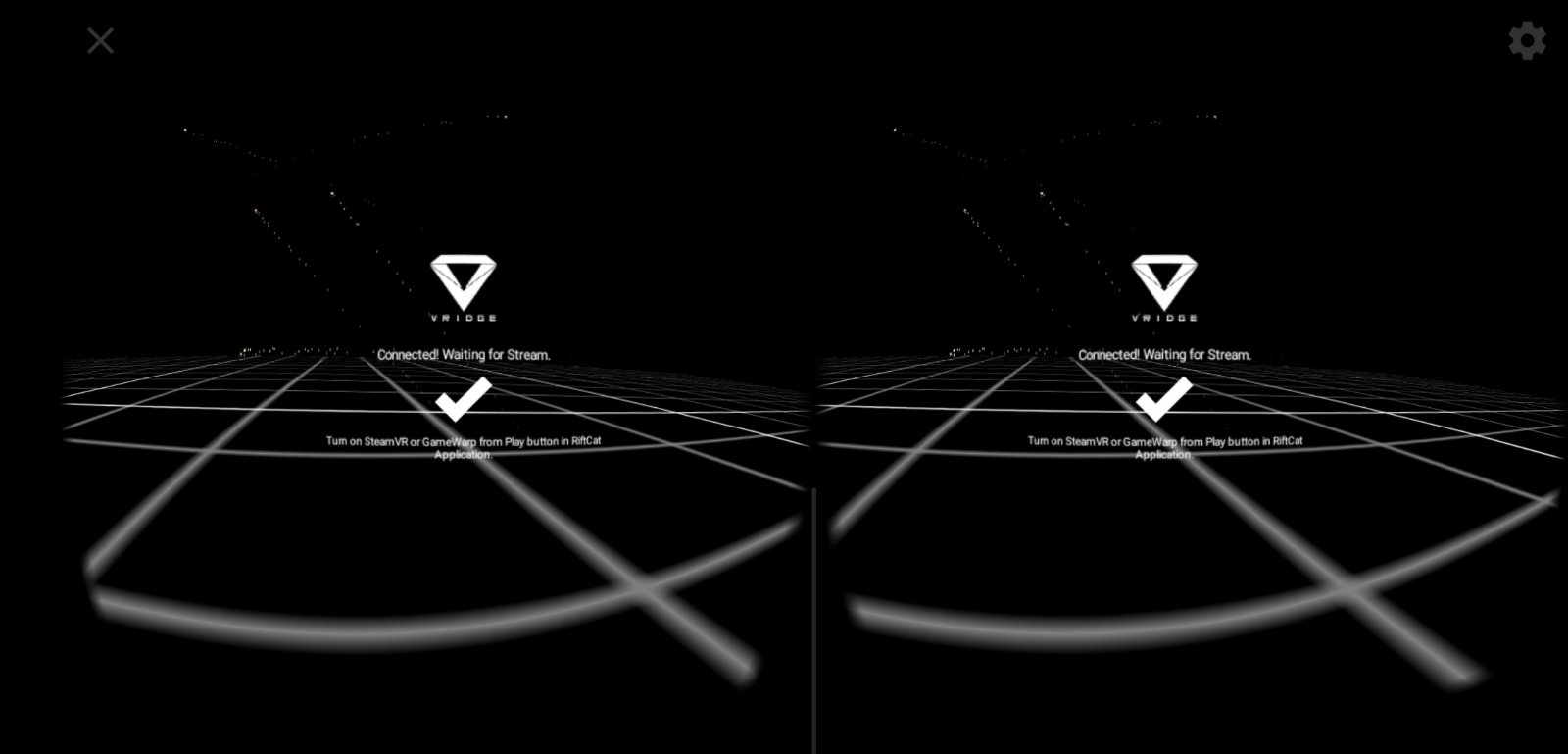


Slika 10 - Izgled VRidge aplikacije na Android uređaju.

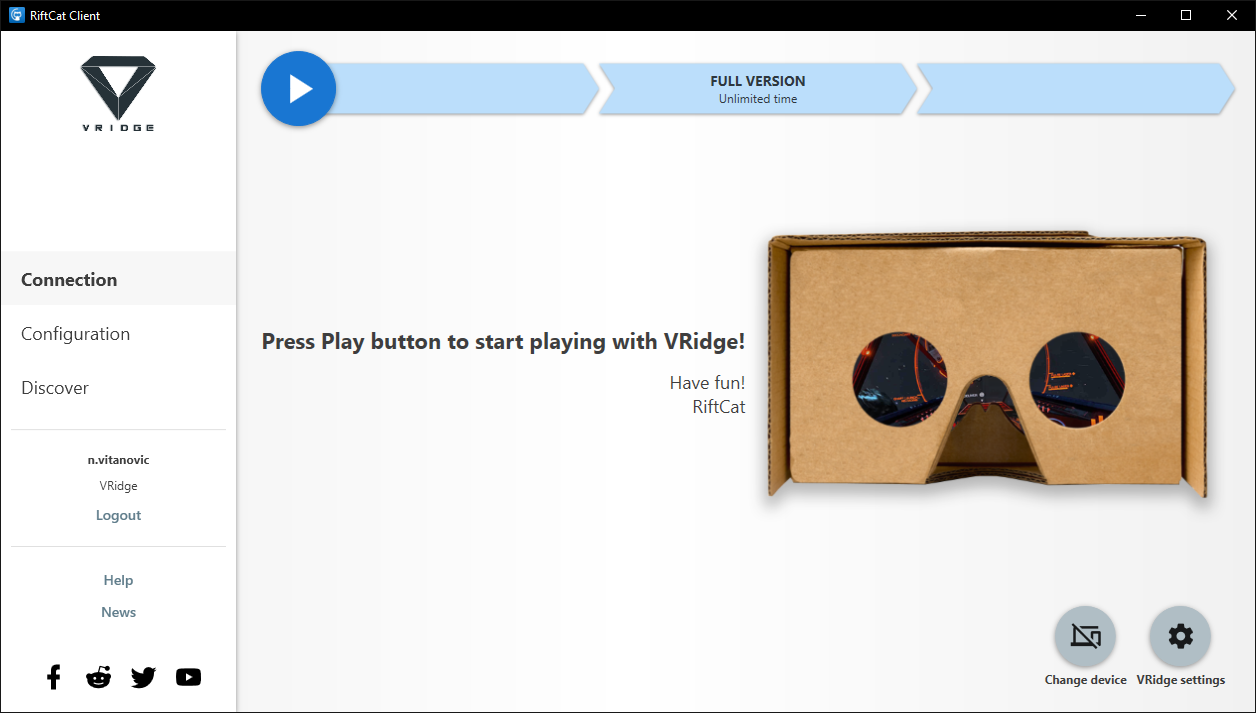
Ukoliko se vrši povezivanje putem USB-a potrebno je na telefonu uključiti opciju *USB debugging* i izabrati opciju *USB tethering*. Nakon prepoznavanja aplikacije na računaru prikazuje se druga forma koja indicira da je telefon uspešno povezan sa Vridge aplikacijom. Na telefonu se takože prikazuje poruka da je veza uspešno uspostavljena. Pritiskom na taster play na aplikaciji pokreće se SteamVR.



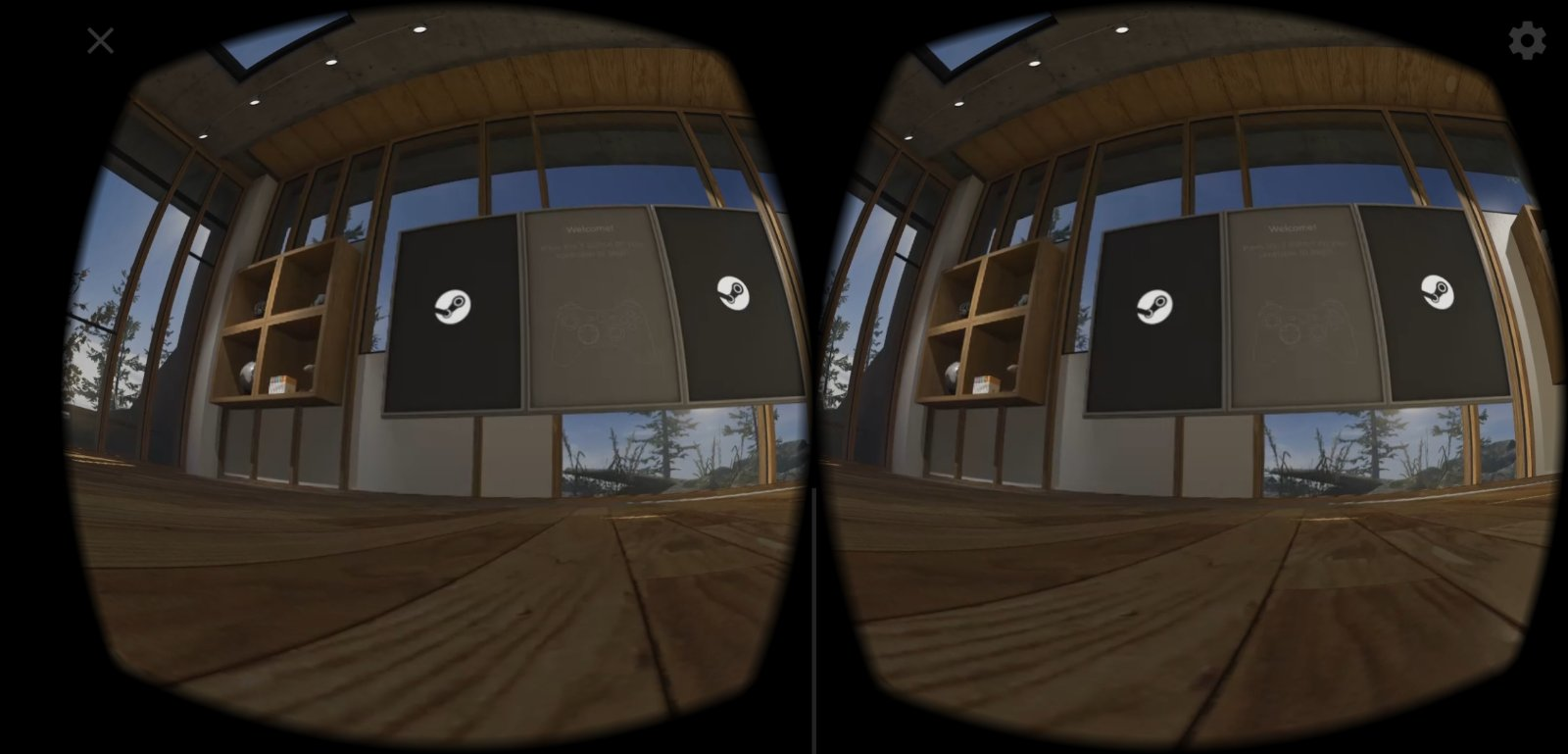
Slika 11 - Izgled SteamVR aplikacije.



Slika 12 - VRidge Android aplikacije je povezana sa računarom.



Slika 13 - RiftCat VRdige Windows aplikacija je povezana sa telefonom.



Slika 14 - Izgled SteamVR početne sobe pre pokretanja VR igre.

# SteamVR

SteamVR [5] predstavlja Valve razvojno okruženje za najpopularniju platofrmu za prodaju igara Steam. SteamVR omogućava programerima lakši razvoj igara koje zahtevaju virtuelnu realnost jer se svi delovi apstrakuju maksimalno moguće. Na primer, čitanje senzora za poziciju glave igrača se automatski obrađuju, bez obzira na tip VR uređaja. SteamVR se bazira na OpenVR inicijativi koja pokušava da standardizuje razvoj VR aplikacija i igara.

Pre igranja bilo koje VR igre potrebno je preuzeti SteamVR ili će se on automatski preuzeti ukoliko pokrećete neku SteamVR igru preko Steam platforme. Nakon instalacije i pokretanja moraćete proći kroz čarobnjaka za kalibraciju, tek nakon kalibracije vašeg VR uređaja pojavljuje se dijalog kao na slici 11., dok na slici 14 možete videti početni meni kroz VR naočare.

# Projekat

Projekat demonstrira mogućnosti prirodne interakcije *Leap motion* uređaja sa virtualnim svetom izrađenim u Unity razvojnom okruženju.

## Unity SteamVR dodatak

Vavle firma održava dodatak za Unity razvojno okruženje koji olakšava intrakciju sa SteamVR [6]. Moderan SteamVR Unity dodatak je zadužen za tri glavne stavke:

* učitavanje 3D modela za VR kontrolere;
* upravljanje VR kontrolerima;
* odlučivanje o izgledu vaših VR ruku u zavisnosti od kontrolera.

Pored ove tri stavke SteamVR Unity dodatak takođe poseduje primere interakcije u okviru Unity igara pomoću SteamVR platoforme, koji mogu da pomognu programeru u bržem razvoju VR igara.

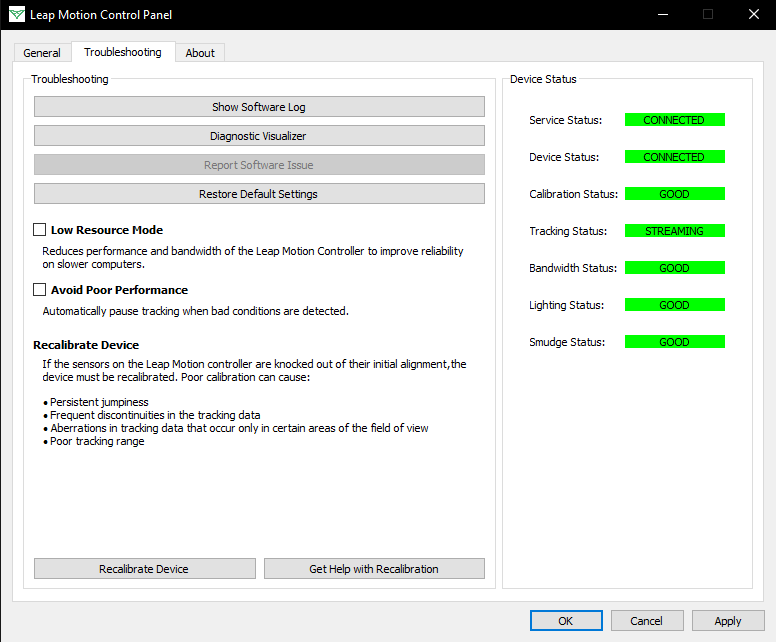
Za korišćenje SteamVR Unity dodatka potrebno je da imate instaliran SteamVR, poželjno Beta verziju zbog novih funkcionalnosti.

Dodatak se može preuzeti sa Unity prodavnice i potpuno je besplatan za korišćenje kako u akademske tako i komercijalne svrhe.

## Unity Gemini Leap Motion dodatak

Pre korišćenja Leap motion uređaja sa VR naočarima u Unity softveru potrebno je preuzeti i instalirati neophodne drajvere i alat za razvoj softvera. Preuzimanje softvera je moguće sa Leap motion sajta, s tim što je bitno napomenuti da je potrebno preuzeti *GEMINI* (V5) verziju drajvera za najbolje performanse:

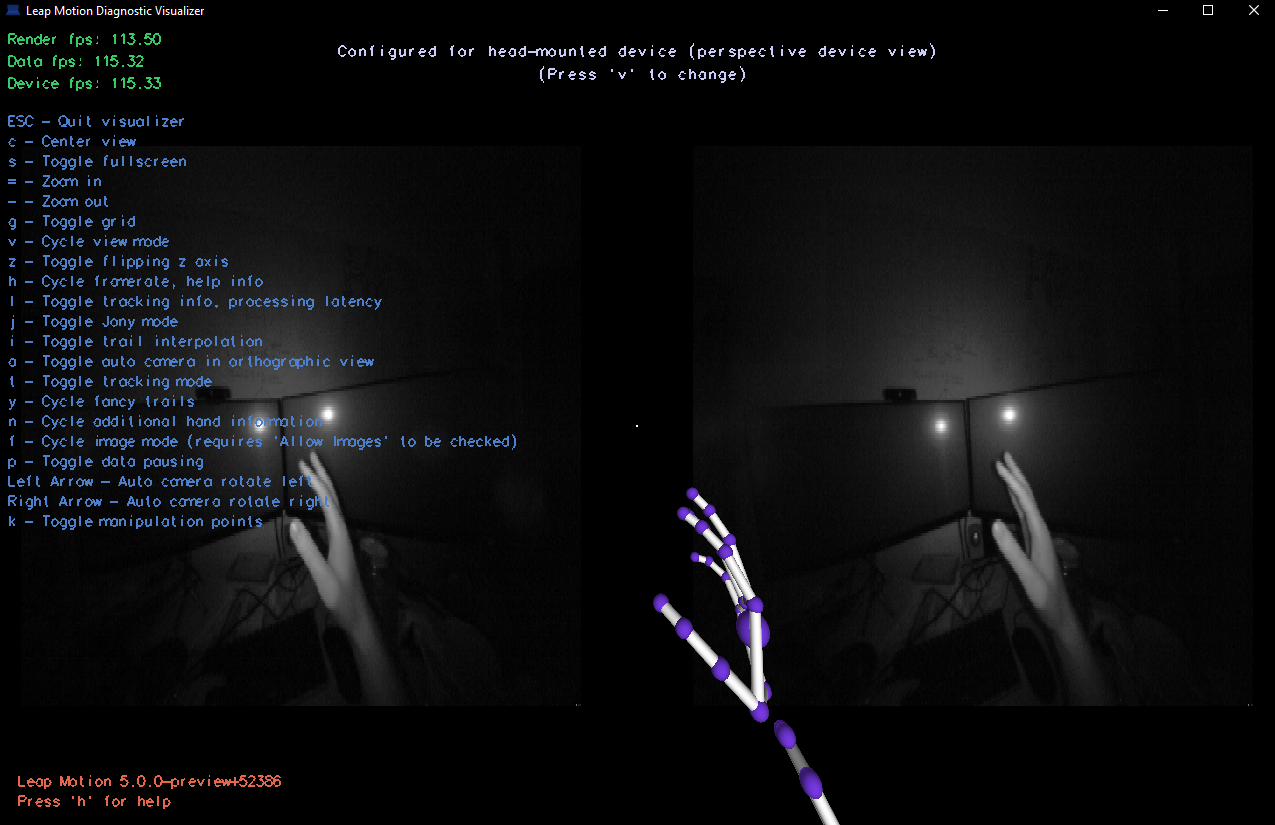
<https://www2.leapmotion.com/downloads/ultraleap-hand-tracking/v5.0.0-preview>



Slika 15 - Kontrolni panel Leap Motion uredaja.

Nakon instaliranja drajvera i priključenja *Leap motion* uređaja na računar potrebno je pristupiti *Leap motion* kontrolnom panelu radi osnovnog podešavanja. U sekciji *Troubleshooting* potrebno je da status uređaja prikazuje sva zelena polja. Takođe bitno je napomenuti da Leap motion uređaj mora da bude čist, kako bi kamere mogle da jasno vide ruku. Radi preciznijeg praćena nije loše odštiklirati opciju *Avoid Poor Performance* kako bi se praćenje ruku radilo i u lošijim manje idealnim uslovima.

Moguće je vizuelizovati trenutnu detekciju ruku kroz *Diagnostic Visualizer* opciju iz kontrolong panela. Ovo omogućava brzu proveru uređaja i podešavanja kako bi bili sigurni da će se praćenje odvijati i u Unity projektu.



Slika 16 - Diagnostic Visualizer omogućava vizuelnu proveru ispravnosti uređaja.

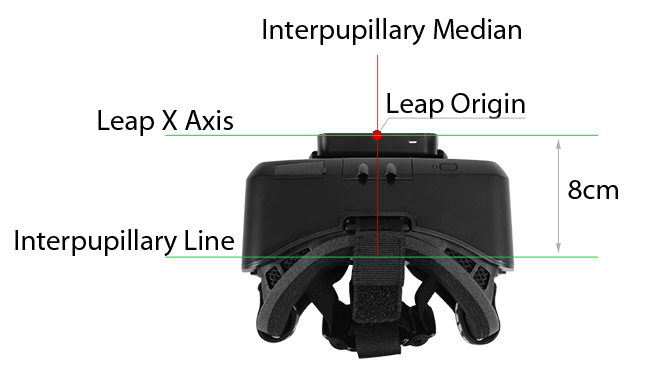
## Leap motion i VR naočare

Leap motion uređaj vraća koordinate u milimetrima u okviru svog referentnog prostora. Centar uređaja ujedno predstavlja i centar prostora u kome se detektuju ruke. Koordinatni sistem je ilustrovan na slici. Leap motion uređaj koristi desni koordinatni sistem i može biti montiran u dve konfiguracije: normlani položaj na stolu korisnika, *HMD* pozicija tj. montiran uređaj na VR naočarima kao na slici 17.



Slika 17 - VR naočare sa montiranim Leap motion uređajem u HMD orijentaciji.

Kako bi uredjaj funkcionisao u HMD režimu, potrebno je postaviti i pričvrstiti ga na pravilan način. Odstojanje izmedju početka koordinatnog sistema Leap motion uređaja i očiju korisnika se savetuje da bude ne više od 8cm. Kamere uređaja treba da gledaju pravo kao oči korisnika i da gledaju ka horizontu.



Slika 18 - rastojanje potrebno za korektnu montažu uređaja u HMD režimu.

## Unity Leap motion dodaci

Leap motion tim je razvio Unity pakete [7] za interakciju sa uređajem. Postoje 3 paketa *Core*, *Hands* i *Interaction engine.* Trenutna najnovija verzija paketa je 4.8.0 i podržava Unity 2019 LTS.

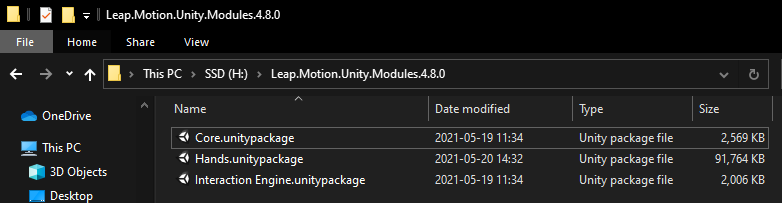
*Core* modul je osnova ostalih modula. Core modul donosi gotove komponente koje su temelj i minimalni interfejs za rad Unity i Leap motion uređaja. Sa ovim modulom je moguće prikazivanje osnovnog seta ruku, kačenje objekata za delove ruku i osnovno prikazivanje i praćenje ruku.

*Interaction engine* modul pruža fizičku reprezentaciju ruku i prepoznavanje gestova kao što su: hvatanje, bacanje, informacije o koliziji i blizini objekata od ruku. Takođe sadrži set primera za izradu 3D grafičkih interfejsa.

*Hands* modul pruža mogućnost korisniku da ubaci svoj 3D model ruku i zameni osnovni model ruku koji pruža Core modul. Takođe sadrži alat za fino podešavanje i pripremu novog modela ruku. Pored toga postoji i nekoliko pripremljenih pari ruku koje mogu da se koriste u projektu bez preteranog podešavanja.

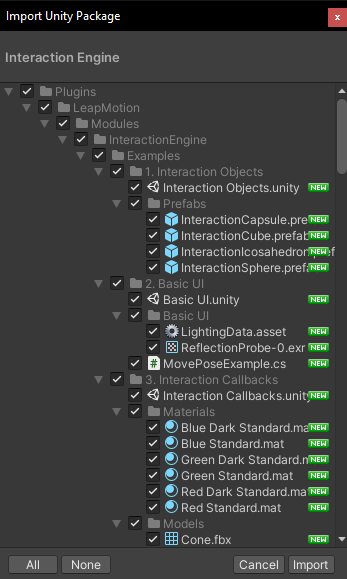
### Instalacija dodataka

Nakon preuzimanja Unity Leap motion dodatka sa [oficijalnog sajta](https://www2.leapmotion.com/downloads/unity-modules/v4.8.0), potrebno je raspakovati ZIP datoteku na računaru.



Slika 19 - Unity leap motion dodaci.

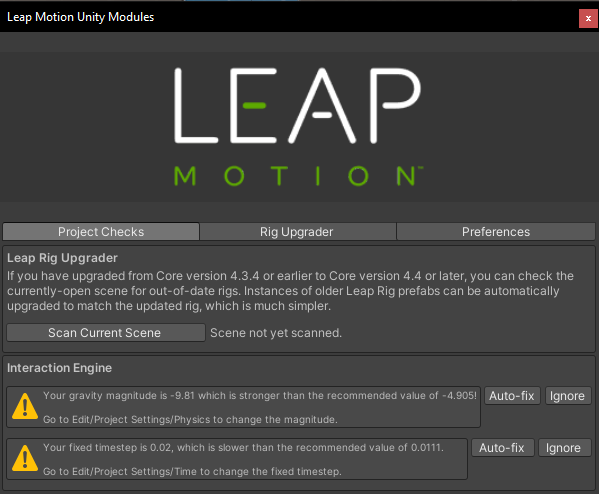
Nakon toga instalirati sve dodatke za potrebe ovog rada. Instalacija se vrši duplim klikom na dodatak, čime se prikazuje prozor u Unity softveru sa listom stvari koje će biti učitane u projekat. Klikom na dugme *Import* započinje uvoz dodatka u projekat koji može da traje par minuta. Redosled instalacije nije bitan, jer su potrebna detaljna podešavanja za potrebe rada.



Slika 20 – primer uvoza Interaction Engine dodatka.

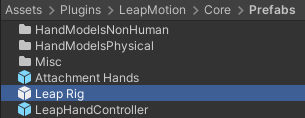
### Osnovna konfiguracija

Na kraju, nakon završenog uvoza dodataka prikazaće se prozor sa preporučenim postavkama. Poželjno je kliknuti dugme *Auto-fix* za svaku od stavki.



Slika 21 - Leap motion prozor sa preporučenim podešavanjima.

U ovom trenutku vaš projekat je spreman za ubacivanje podrške za Leap motion. Najlakši način predstavlja prevlačenje već pripremljenog objekta iz *Core* modula koji se naziva *Leap Rig*. Objekat se nalazi u folderu *Plugins > LeapMotion > Core > Prefabs.*

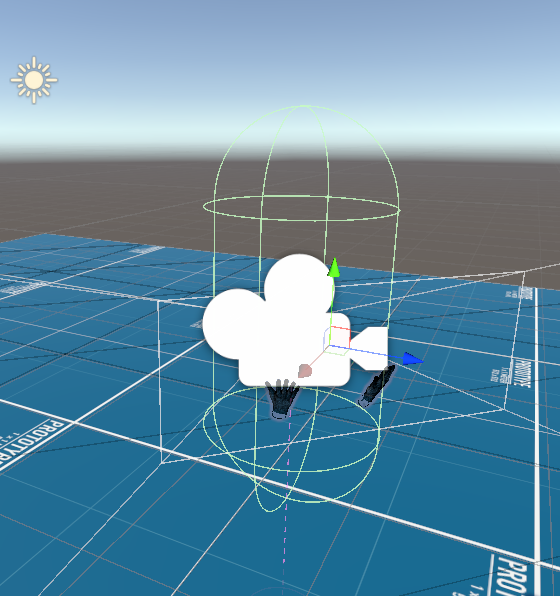
 

Slika 22 - Leap rig objekat.

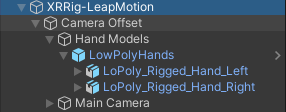
Leap Rig objekat se sastoji od glavne kamere i osnovog seta ruku koje se nazivaju *Capsule Hands* koji su prikazani na slici 16*.* Ovaj set ruku nije predstavljen 3D modelom već se generiše kroz skriptu koja dolazi sa *Core* modulom.

## Prilagođeni objekat igrača

Osnovni *Leap Rig* objekat opisan u prethodnoj sekciji biće zamenjen naprednijom varijatnom. Potrebno je konstruisati objekat ispočetka tj. od nule. Željeni cilj je objekat igrača koji može da se kreće, čuči i interaguje sa objektima iz okoline rukama.

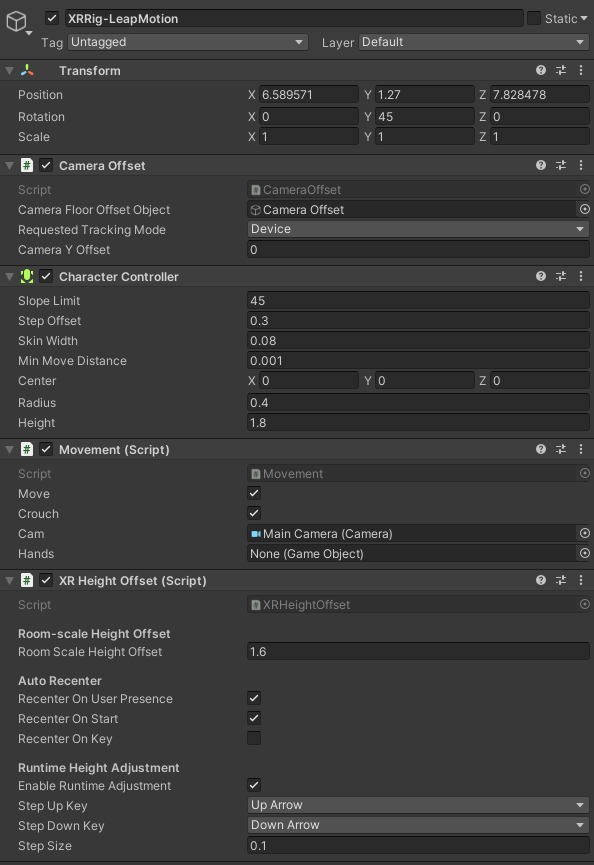


Slika 23 – izgled prilagođenog objekta igrača.



Slika 24 - Struktura prilagodjenog objekta igrača.

*XRRig-LeapMotion* objekat se sastoji od tri glavne komponente: glavne kamere, ruku i odstojanja kamere od poda. Osnovni objekat koristi 4 komponente koje su prikazane na slici 25. Najbitnija komponenta je *Movement i XR Height Offset. Movement* komponenta reguliše kretanje igrača u VR svetu. Kretanje se obavlja gledanjem kamerom na dole, rotacijom glave menja se smer kretanja. Gledanjem skroz na dole igrač prelazi položaj čučanja. *XR Height Offset* reguliše visinu na kojoj se nalaze vrituelne ruke igrača koje se kontrolišu putem *Leap Motion* uređaja. Podešavanja sa slike su podrazumevana i najbolje ih je ostaviti na tim vrednostima.



Slika 25 - komponente objekta koji predstavlja igrača.

### Kretanje

Kretanje u VR svetu prestavlja izazov. Radi postizanja najrealnijeg doživljaja VR igara, potrebno je translirati fizičko kretanje u stvarnom svetu u virtuelno kretanje. Da bi tako nešto bilo moguće morali bismo da imamo neogračien prostor po kome bi se kretali ili neki od naprednih VR postolja. Naravno to nije jednostavno, iz tog razloga razni proizvođači igara su smislili razne načine kretanja u virtuelnom svetu.

Kretanje u ovoj igri je jednostavno, gledanjem blago na dole u realnom svetu translira se na kretanje igrača u virtuelnom svetu u smeru gledanja. Gledanjem pod većim uglom prema podu igrač prelazi u stanje čučanja i sporije se kreće u smeru gledanja. To je omogućeno kodom iz skripte *Movement.*cs koja je zakačena za *XRRig-LeapMotion* objekat i omogućava mu pomeranje.

public class Movement : MonoBehaviour

{

private CharacterController controller;

private float playerSpeed = 1.0f;

public bool move = true;

public bool crouch = true;

public Camera cam;

public GameObject hands;

private float originalHeight;

private float crouchHeight;

void Start()

{

controller = GetComponent<CharacterController>();

originalHeight = controller.height;

crouchHeight = originalHeight / 4;

}

void Update()

{

if(cam.transform.localEulerAngles.x > 40 && cam.transform.localEulerAngles.x < 60)

{

move = true;

crouch = false;

}

else if(cam.transform.localEulerAngles.x > 75 && cam.transform.localEulerAngles.x < 85)

{

crouch = true;

}

else

{

move = false;

crouch = false;

}

if (move)

{

Vector3 moveDistance = Camera.main.transform.forward;

controller.SimpleMove(moveDistance \* playerSpeed);

}

else if(crouch)

{

controller.height = crouchHeight;

controller.SimpleMove(Camera.main.transform.forward);

}

else

{

controller.height = originalHeight;

}

}

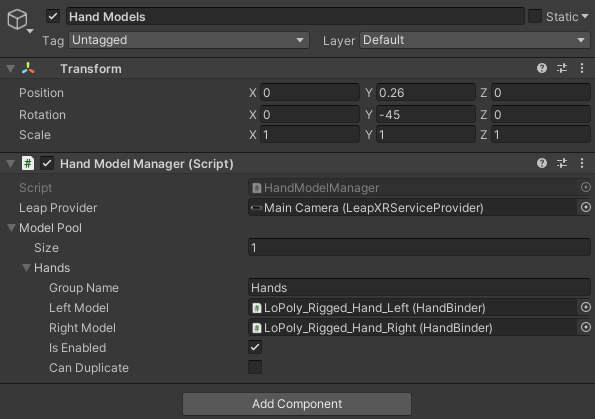
}

Deo koda 1 - Skripta za kretanje igrača Movement.cs.

### Ruke

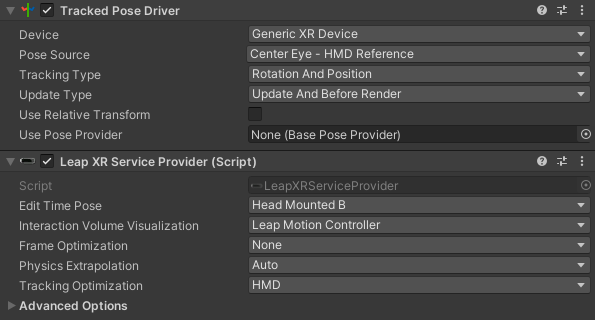
*SteamVR* pruža podršku za prikaz ruku korišćenjem eksternih kontrolera, koji su najčešće deo VR kompleta. Kontroleri pružaju igraču virtuelne ruke, ali je potreban period navikavanja korišćenjem tih kontrolera. Interakcija između čoveka i računara je manje prirodna, Leap Motion uređaj stremi ka tome da interakcija postane prirodnija. Ruke i pozicije prstiju u realnom svetu se transliraju na virtualni svet.

Iskorišćen je *LowPolyHands* koji je deo *Hands* paketa. U okviru *Hand Models* objekta potrebno je dodati željene modele ruku. Nakon toga kao na slici 26 potrebno je podesiti koji objekti sa scene predstavljaju desnu i levu ruku. Bitno je napomenuti da nije moguće iskoristiti bilo koj model za izgled ruku, potrebno je da model sadrži kosti koje se mapiraju kroz *Hand Binder* alat koji nudi Leap Motion.



Slika 26 - podešavanje modela ruku.

Glavna kamera takođe mora biti podešena na ispravan način kako bi se ruke prikazale korektno. Korektno prikazivanje i mapiranje ruku zavisi od pozicije Leap Motion uređaja. S obzirom da je Leap motion uređaj zakačen za same VR naočare sa prednje strane, kabli je na levoj strani, bitno je postaviti komponente *Tracked Pose Driver* i *Leap XR Service Provider* iz objekta *Main Camera* kao na slici 27.



Slika 27 - Korektno podešavanje kamere za prikaz ruku.

## Interaktivni objekti

Kako bi interakcija ruku sa objektima funkcionisala potrebno je da u sceni postoji *Interaction Manager* objekat. Objekat je idealno postaviti iznad glavne kamere u okviru objekta igrača kao što se može videti na slici 28. *Interaction Manager* je zadužen za svu internu logiku koja omogućava interakcije.

Interaktivni objekti moraju da imaju par komponenti:

* *InteractionBehaviour –* omogućava akcije kao što su hvatanje, guranje, udaranje objekta od strane virtaulnih ruku.
* *RigidBody* – dodavanjem ove komponente objekat biva kontrolisan od strane Unity Physics koda. Ovo mu omogućava da se ponaša kao realan objekat između ostalog vuče ga gravitacija ka zemlji, naravno potrebna je i *Collider* komponenta kako bi objekat reagovao sa ostalim objektima i ne bi propao kroz zemlju.
* *Collider* – služi za detekciju kolizije između objekata ili u ovom slučaju detekciju interakcije ruku i objekta.

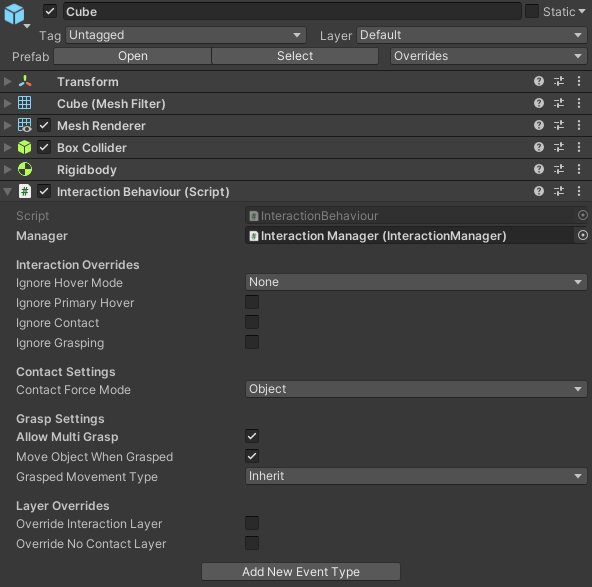


Slika 28 - unapređeni objekat igrača, sa interakcionim mogućnostima.

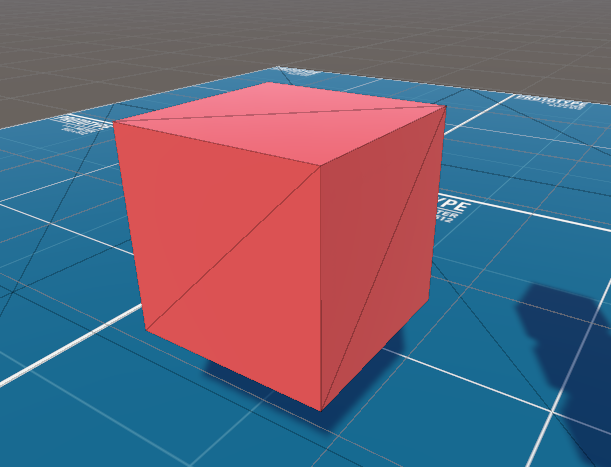
### Kutija

Kutija je najprostiji oblik interaktivnog objekta. Sadrži *InteractionBehaviour* i ostale prethodno navedene komponente. *Simple Interaction Glow* komponenta olakšava korisniku da zna kada je moguće interagovati sa objektom, kako su ruke bliže objektu tako objekat menja boju. Osnovne fukncije kao što su guranje objekta, hvatanje i bacanje su moguće.

Svaki objekat sa kojim se interaguje mora da ima povezan *Interaction Manager* koji vodi brigu o tome kako se objekat ponaša. Na slici 29 vide se podešavanja koja su potrebna kako bi objekat kutija bio funkcionalan.



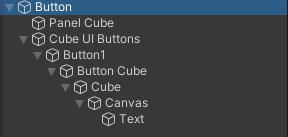
Slika 29 - komponenta InteractionBehaviour sa osnovnim podešavanjima za objekat kutije.



Slika 30 – izgled objekta.

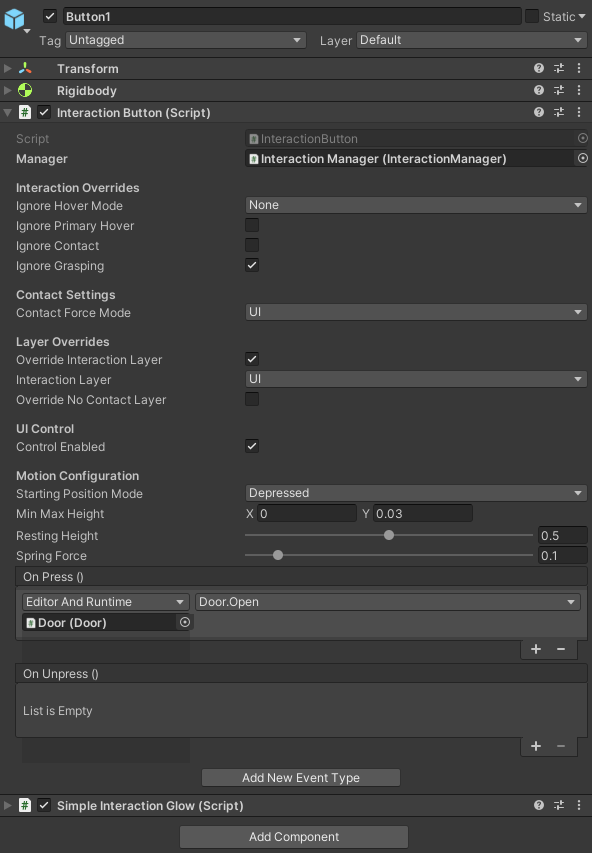
### Dugme

Dugme spada u specijalnu vrstu objekta jer može da zove funkcije ostalih objekata. Na ovaj način svako dugme u sceni može da ima različitu namenu.



Slika 31 - struktura Button objekta.

*Button* je složeni objekat. Najbitniji deo je *Button1* objekat koji koristi *Interaction Button* komponentu.

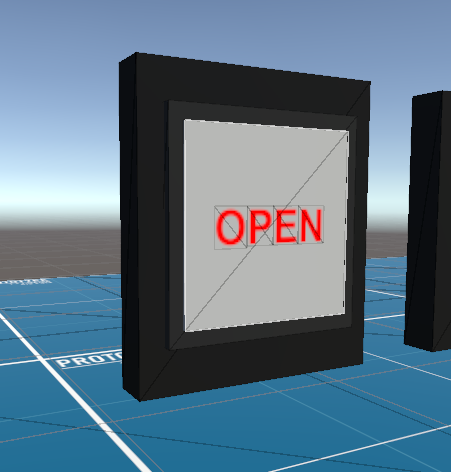


Slika 32 - Interaction Button skripta i njena osnovna podešavanja.

Za razliku od normalnog interaktivnog objekta kao što je kutija, ovaj objekat mora biti stacionaran, igrač ne može da pomeri objekat, ali može da ga pritisne. Postoji više tipa događaja na koje objekat može da pozove druge funkcije, najkorisnije su:

* *OnPress* – objekat zove funkciju na pritisak dugmeta.
* *OnUnpress* – objekat zove funkciju nakon pustanja dugmeta.

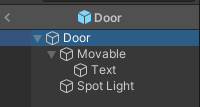
Moguće je pozvati bilo koju funkciju objekta koja je javna. Promena teksta objekta se vrši u UI podkomponenti *Text.*



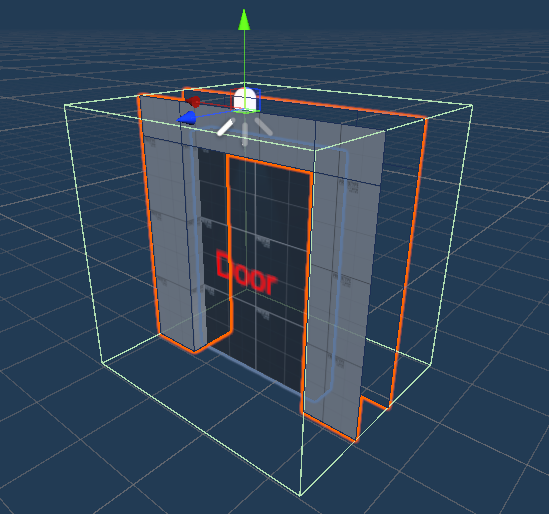
Slika 33 - izgled dugmeta u virtualnom svetu.

### Vrata

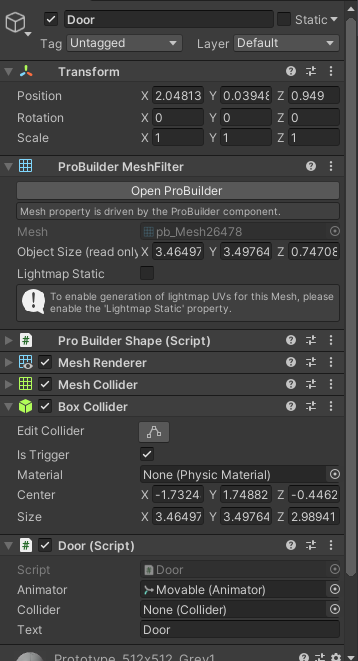
Vrata u svakoj igri predstavljaju osnovnu mehaniku. Vrata su složen objekat koji se sastoji iz nekoliko komponenti. Fiksna komponenta su sam ram vrata *Door* dok sama vrata prestavlja objekat pod nazivom *Movable*.



Slika 34 - Struktura objekta vrata.



Slika 35 - Izgled vrata u Unity editoru.



Slika 36 - Komponente od kojih su vrata sačinjena.

Komponenta BoxCollider je služi za detektovanja prisustva objekta blizu vrata. U slučaju da se bilo koji objekat uđe u zonu za otvaranje vrata, pokreće se animacija za otvaranje vrata, podešavanjem stanja *Closed* na *false* u okviru animacionog kontrolera. Animacioni kontroler je postavljen na deo vrata koji se pomera tj. na *Moveable* objekat.

Na početku vrata se nalazue u zatvorenom položaju tj. stanju *DoorClosed* animacija tog stanja su zatvorena vrata. Iz tog stanja je moguće preći u stanje *DoorOpen* koje predstavlja otvaranje vrata i reprodukuje animaciju za otvaranje vrata.



Slika 37 - Animacioni graf koji omogućava otvaranje i zatvaranje vrata.

Otvaranje i zatvaranje vrata ne bi bilo moguće bez skripte *Door.cs* koja se stavlja na glavni objekat *Door* i uz njega se povezuje deo vrata koji se pomera. Pored animatora moguće je podesiti i tekst koji se prikazuje na vratima.

public class Door : MonoBehaviour, IInteractable

{

public Animator animator;

public bool open;

public bool triggerEnabled = true;

public string text;

private Text textMesh;

void Start()

{

textMesh = GetComponentInChildren<Text>();

textMesh.text = text;

}

void Update()

{

if(open == true)

{

animator.SetBool("Closed", false);

}

else

{

animator.SetBool("Closed", true);

}

}

private void OnTriggerEnter(Collider other)

{

if(triggerEnabled)

open = true;

}

private void OnTriggerExit(Collider other)

{

if(triggerEnabled)

open = false;

}

public bool Interact()

{

return open = !open;

}

public void Open()

{

open = true;

}

public void Close()

{

open = false;

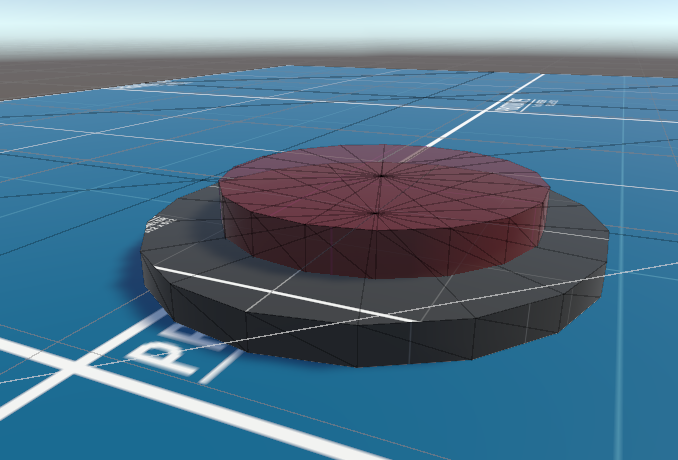
}

}

Deo koda 2 - koji omogućuje otvaranje i zatvaranje vrata.

### Potisna ploča

Potisna ploča je dodatni objekat koji pridaje interaktivnost igri. Kada igrač stane na potisnu ploču ona vrši interakciju sa povezanim objektom, najčešće će to biti objekat vrata. Naravno nekada je potrebno staviti kutiju na potisni objekat kako bi igrač uspeo da prođe kroz vrata. Ovo pruža veoma jednostavnu mehaniku igri. Objekat ne sadrži specijalne skripte i komponente kao objekat dugme jer nije deo *Interaction Engine* dodatka.



Slika 38 - potisna ploča.

public class PressurePlate : MonoBehaviour

{

public GameObject objectToTrigger = null;

public bool pressed = false;

public bool triggerEnabled = true;

private Animator animator;

void Start()

{

animator = GetComponentInChildren<Animator>();

}

void Update()

{

}

private void OnTriggerExit(Collider other)

{

if (triggerEnabled)

{

animator.SetBool("pressed", false);

if (objectToTrigger != null)

objectToTrigger.GetComponent<IInteractable>().Interact();

pressed = false;

}

}

private void OnTriggerEnter(Collider other)

{

if (triggerEnabled)

{

animator.SetBool("pressed", true);

if (objectToTrigger != null)

objectToTrigger.GetComponent<IInteractable>().Interact();

pressed = true;

}

}

}

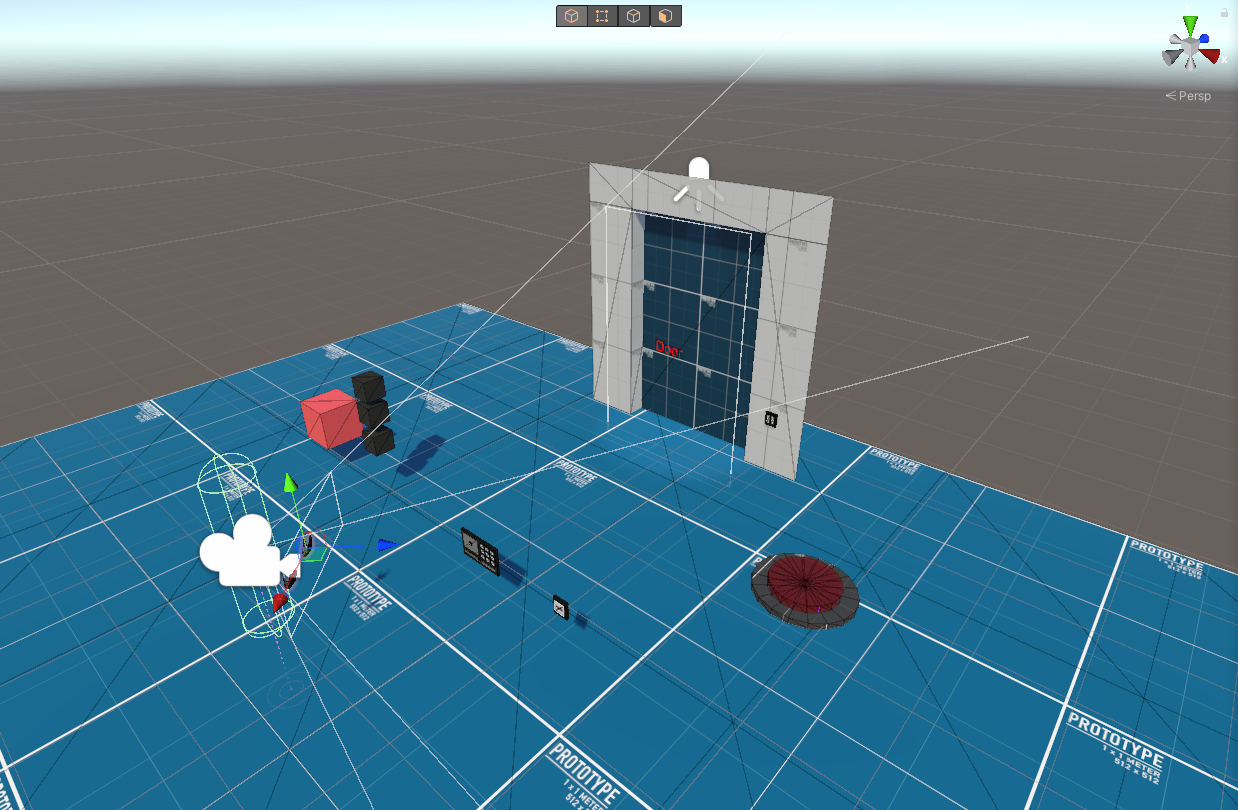
Deo koda 3 – koji omogućuje funkciju interakcije sa objektom.

Potisna ploča radi u *toggle* modu. Kada je nešto na njoj ona se nalazi u stanju pritisnuta i poziva funkciju za interakciju, pri izlazu sa potisne ploče ista funkcija se poziva. Objekat koji potisna ploča aktivira mora da ima svoju logiku za funkciju interakcije. U slučaju vrata, kada se pozove funkcija za interakciju, vrata se otvaraju, ako su već bila otvorena pri pozivu se zatvaraju, tako da početno stanje vrata ima uticaja na način rada potisne ploče.

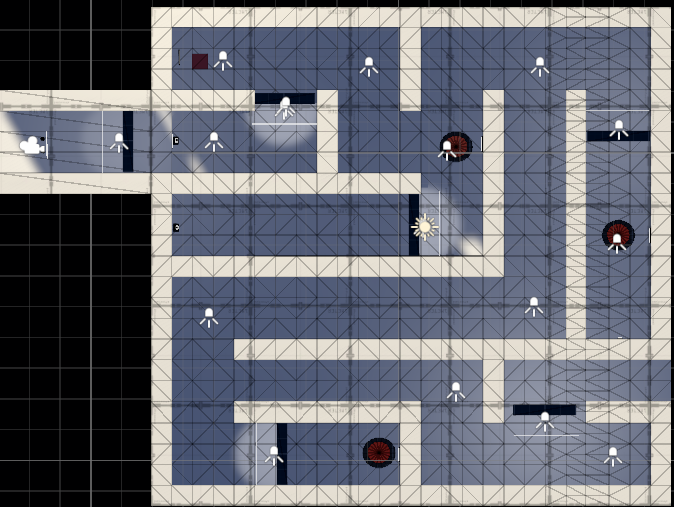
## Igra

Igra se sastoji od jednog glavnog lavirinta prikazanog na slici 40, koji korisnik treba da pređe korišćenjem prethodno opisanih komponenti. Kreiranje samog nivoa je jednostavno i ne zahteva programiranje ukoliko se koristi mehanika i komponente koje su implementirane u radu.

Osnovne komponente su testirane na *Test* nivou koji nije deo igre. Nivo je služio kao mesto za brzo prototipovanje komponenti i probu njihove funkcionalnosti. Neke od komponenti kao duplo dugme ili panel sa slajderom nisu deo igre zbog svoje kompleksnosti, ali bi u budućnosti mogle da bude deo, jer mogu da pruže interesantan način interakcije sa virtuelnim svetom korišćenjem Leap Motion uređaja.



Slika 39 - Test nivo.



Slika 40 – primer jednog nivoa u igri.

# Zaključak

Prirodni pristup interakciji u virtuelnom svetu je svakako budućnost, iako trenutno deluje kao da tehnologija nije spremna, na ovom polju se dosta radi. Leap Motion uređaj u radu nije u potupunosti stabilan, bez obzira na dobre uslove osvetljenja i najnovijih drajvera. Uz upotrebu naprednih neuronskih mreža moguće je poboljšati detekciju ruku i predviđanje pozicije zakrivenih objekata. Leap Motion ne pruža previše informacija o tome kako se podaci o rukama detektuju i obrađuju, ali s obzirom na korišćenje infracrvenih kamera, možemo zaključiti da se vrši obrada slike. Ipak tehnologija mapiranja ruku i pristiju u virtuelnom svetu sa kombinacijom VR naočara može dosta pomoći u prirodnijoj interakciji sa računarom.

# Reference

|  |  |
| --- | --- |
| [1] | „Unity (game engine),“ Wikipedia, 2021. [Na mreži]. Available: https://en.wikipedia.org/wiki/Unity\_(game\_engine). |
| [2] | U. Technologies, „Unity User Manual (2019.4 LTS),“ 2021. [Na mreži]. Available: https://docs.unity3d.com/Manual/index.html. |
| [3] | „Leap motion history,“ Wikipedia, [Na mreži]. Available: https://en.wikipedia.org/wiki/Leap\_Motion. |
| [4] | RiftCat, „Vridge,“ [Na mreži]. Available: https://riftcat.com/vridge. |
| [5] | Valve, „SteamVR documentation,“ [Na mreži]. Available: https://partner.steamgames.com/doc/features/steamvr/info. |
| [6] | Valve, „SteamVR Unity Plugin documentation,“ [Na mreži]. Available: https://valvesoftware.github.io/steamvr\_unity\_plugin/. |
| [7] | „Leap motion documentation for Unity,“ [Na mreži]. Available: https://leapmotion.github.io/UnityModules/index.html. |