Programiranje Microsoft Hololens uređaja

Naglić, Kristian

Master's thesis / Diplomski rad

2022

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Organization and Informatics / Sveučilište u Zagrebu, Fakultet organizacije i informatike**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:211:511653

Rights / Prava: Attribution-NonCommercial 3.0 Unported/Imenovanje-Nekomercijalno 3.0

Download date / Datum preuzimanja: **2024-03-29**

SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
VARAŽDIN

Repository / Repozitorij:

Faculty of Organization and Informatics - Digital Repository

zir.nsk.hr

DIGITALNI AKADEMSKI ARHIVI I REPOZITORIJI

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Kristian Naglić

Programiranje Microsoft Hololens uređaja

DIPLOMSKI RAD

Varaždin, 2022.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Kristian Naglić

Matični broj: 0016122751

Studij: Informacijsko i programsko inženjerstvo

Programiranje Microsoft Hololens uređaja

DIPLOMSKI RAD

Mentor/Mentorica:

Neven Vrček, prof. dr. sc.

Varaždin, srpanj 2022

Kristian Naglić

Izjava o izvornosti

Izjavljujem da je moj završni/diplomski rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor/Autorica potvrdio/potvrdila prihvaćanjem odredbi u sustavu FOI-radovi

Sažetak

Cilj ovog diplomskog rada je opisati proces programiranja za Microsoft Hololens uređaj. U prvom dijelu rada proći će se kroz teorijski dio vezan uz proširenu stvarnost što uključuje kratku povijest, opis uređaja, primjene, te usporedbu sa virtualnom stvarnošću i mogućoj primjeni u budućnosti. Također će se obraditi teorijska podloga samog Microsoft Hololens uređaja kao i Microsoftove biblioteke MRTK (eng. *Mixed Reality Toolkit*) koja se koristi za izradu aplikacija mješovite stvarnosti. Praktični dio projekta sadržavati će opis izrađenih aplikacija kao i korištenih alata i tehnologija te tijek i izradu aplikacija mješovite stvarnosti za Microsoft Hololens uređaj uz pomoć Unity okvira i MRTK biblioteke putem koje će se prikazati neke od mogućnosti uređaja.

Ključne riječi: Mješovita stvarnost; Microsoft; Hololens; Unity; MRTK; aplikacija

Sadržaj

Sadržaj iii
1. Uvod 1
2. Metode i tehnike rada 2
3. Mješovita stvarnost 3
3.1. Proširena stvarnost 4
3.1.1. Povijest 4
3.2. Proširena virtualnost 7
3.3. Virtualna stvarnost 7
3.4. Vrste uređaja i zaslona 8
3.4.1. Uređaji 10
3.5. Primjene 15
3.5.1. Kolaboracija 15
3.5.2. Proizvodnja, logistika i robotika 16
3.5.3. Edukacija 16
3.5.4. Zabava 17
3.5.5. Medicina 18
3.6. Budućnost i izazovi 19
4. Microsoft Hololens 2 22
4.1. Tehničke specifikacije 22
4.2. Interakcija 23
4.3. Mogućnosti 26
4.3.1. Orijentacija i mapiranje prostora 27
5. Implementacija mješovite stvarnosti 30
5.1. Opis aplikacije 30
5.2. Korišteni alati 32
5.2.1. Unity 32
5.2.2. Visual Studio 33
5.2.3. Mixed Reality Feature Tool 33
5.2.4. MRTK (Mixed Reality Toolkit) 33
5.2.5. Hololens 2 Emulator 35
5.3. Postavljanje razvojne okoline 37
5.4. Razvoj aplikacije 43
5.4.1. Pocket Basketball 46

5.4.2. AirPaint 54
5.4.3. Weather 3D 60
6. Zaključak 69
Popis literature 70
Popis slika 74
Popis tablica 75
Prilozi 76

1. Uvod

Proširena i virtualna, pa tako i mješovita stvarnost prije nekoliko desetina godina je bio koncept o kojemu se samo moglo sanjati. U današnje vrijeme uz pomoć rapidnog razvoja tehnologija i digitalizacije informacija, popularnost mješovite stvarnosti je u porastu. Postoje brojna istraživanja vezana za područje mješovite ili proširene stvarnosti u poljima medicine, edukacije, interesa mladeži, ali i marketinga. Mogućnosti koje se koriste variraju ovisno o mjestu primjene no obuhvaćaju skeniranje i mapiranje prostora, ubacivanje grafike, praćenje oznaka u prostoru, prevođenje teksta kao i zamjenu cijelog prostora za virtualni koji je vidljiv samo preko uređaja.

Najjednostavniji oblik proširene stvarnosti dostupan je gotovo na svim mobilnim uređajima današnjice u obliku aplikacija na Android ili iOS uređajima koje kroz kamere uređaja u prostor prividno postavljaju grafiku. Osim jednostavnijih oblika, dostupne su i naočale za mješovitu stvarnost koje danas imaju mogućnost praćenja ruku korisnika te simulaciju istih koje se koriste za interakciju u virtualnom ili proširenom prostoru bez dodatnih kontrolera ili rukavica. Jedan od takvih uređaja koji se pretežno koristi u korporativne svrhe je Microsoft Hololens. Implementacija mješovite stvarnosti moguća je kroz nekoliko alata ovisno o uređajima za koje se ista radi. Za mobilne uređaje je implementacija proširene stvarnosti moguća uz pomoć jezika Kotlin i raznih okvira za proširenu stvarnost koji koriste Android ARCore. Kako se ovaj rad odnosi na specifičan uređaj, koristiti će se Unity softver za izradu grafike potrebne za mješovitu stvarnost uz pomoć Microsoftove biblioteke MRTK (*eng. Mixed Reality Toolkit*).

Cilj ovog rada je teorijski obraditi koncept mješovite stvarnosti i mogućnosti uređaja Microsoft Hololens te potom na praktičnom primjeru aplikacije opisati postupak programiranja za isti uređaj što će uključivati pregled alata Unity i biblioteke MRTK.

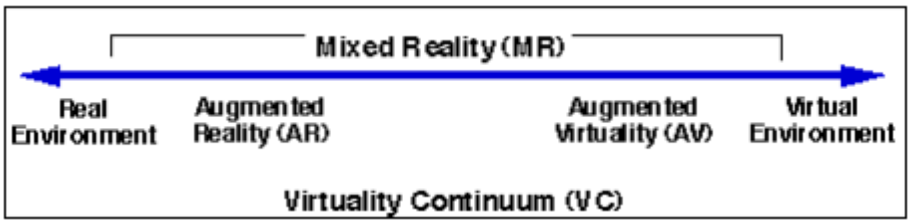
2. Metode i tehnike rada

Teorijski dio rada opširno opisuje mješovitu stvarnost kao i sve ključne pojmove potrebne za razumijevanje daljnjih koncepata programiranja aplikacija mješovite stvarnosti. Posebna pažnja će se posvetiti primjeni kao i uređajima mješovite stvarnosti te njihovim razlikama. Dio će započeti sa općim definiranjem mješovite stvarnosti što uključuje i razliku odnosno sličnost sa virtualnom i proširenom stvarnošću, nakon čega će se proći kroz povijest razvoja i uređaja te prikazati detaljnije primjene u raznim poljima kao i moguće probleme. Iskoristiti će se informacije iz pronađenih članaka i radova sa Google Scholar baze kako bi se pobliže opisala mješovita stvarnost. U nastavku će se opisati sam uređaj Microsoft Hololens te biblioteka za programiranje istog uređaja uz pomoć dostupne dokumentacije na stranicama Microsofta kao i pobliže pregledati alati za izradu računalne grafike.

U praktičnom dijelu rada će se postepeno proći kroz korake i postupak izrade aplikacije mješovite stvarnosti za Microsoft Hololens uređaj. Za izradu aplikacije koristiti će se Unity softver za izradu računalne grafike u jeziku C# kao i MRTK biblioteka za Unity koja će se detaljnije i opisati. Prilikom testiranja aplikacije koristiti će se Unity editor i Hololens emulator koji će omogućavati pokretanje aplikacije bez Hololens uređaja.

3. Mješovita stvarnost

Danas se uglavnom koriste većini poznati pojmovi augmentirane i virtualne stvarnosti, no treći pojam koji se ovdje javlja i nije svima jasan je mješovita stvarnost (*eng. Mixed reality*). Taj pojam prvi put se javlja u znanstvenom članku autora P. Milgram i F. Kishino, objavljenog 1994. godine, kao virtualni kontinuum. Virtualni kontinuum može se opisati kao spektar ili raspon na kojemu se na jednom kraju nalazi stvarnost i svi stvarni objekti koji ju čine, a na drugom kraju kompletno virtualna stvarnost sa digitaliziranim objektima. Autori su tako pojam mješovite stvarnosti poopćili sa virtualnim kontinuumom i definirali kao stvarnost u kojoj su stvarni i virtualni svijet prezentirani zajedno unutar jednog zaslona, odnosno bilo gdje na spektru virtualnog kontinuuma. [1]

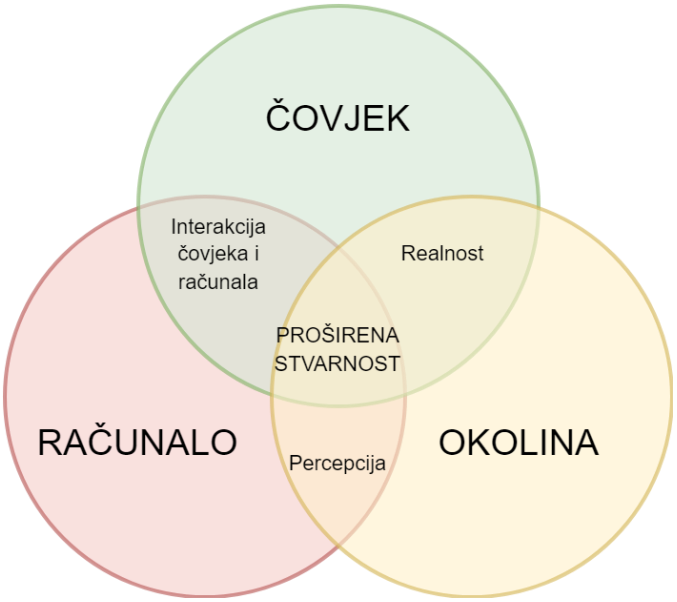


Slika 1. Spektar virtualnog kontinuuma [1]

Jedna definicija koja stoji na stranicama Microsofta definira mješovitu stvarnost kao spektar imerzivnih iskustva koja spaja fizički i digitalni svijet u aplikacijama augmentirane i virtualne stvarnosti [2]. Dakle, prema spektru možemo vidjeti da mješovita stvarnost obuhvaća sve osim prave stvarnosti i kompletno virtualnog svijeta. Glavna karakteristika takve stvarnosti je da čovjek približno gleda na digitalne objekte i fizičke objekte kao jedne pa prema tome mješovita stvarnost uključuje specifične interakcije unikatne za tu stvarnost [2]:

- Praćenje pozicije korisnika u prostoru na temelju podataka o mapiranim površinama.
- Prostorno uređen zvuk kojega korisnik može locirati u prostoru kao i normalne zvukove u stvarnom svijetu
- Digitalni objekti koji, zajedno sa stvarnima, čuvaju informacije o svojoj poziciji i ostaju u prostoru bez obzira na kretanje korisnika.

To čini mješovitu stvarnost specifičnom i omogućuje interakciju između računala i čovjeka na razini percepcije.



Slika 2. Dijagram mješovite stvarnosti, prema: [2]

Čovjek ima mogućnost kretanja u mješovitom svijetu na način da se njegovi koraci direktno preslikavaju. Kretanje, granice prostora i interakcija sa objektima može utjecati na iskustvo korisnika aplikacija što je ključna karakteristika mješovite stvarnosti, jer bez toga, spajanje virtualnog i stvarnog svijeta ne bi bila moguća. [2]

3.1. Proširena stvarnost

Proširena stvarnost (*eng. Augmented reality*) je oblik stvarnosti koji se nalazi na jednom kraju spektra virtualnog kontinuuma i predstavlja izravni ili neizravni pogled u stvarnom vremenu na fizički svijet, poboljšan dodavanjem virtualnih informacija generiranih od strane računala [3]. Možemo reći da je proširena stvarnost zapravo pod skup mješovite stvarnosti sa manje mogućnosti koje su bliže stvarnom svijetu nego virtualnom. Proširena stvarnost se razlikuje od mješovite po tome što označava jednostavniji oblik stvarnosti gdje su digitalni objekti preslikani u stvarni svijet dok mješovita stvarnost odlazi korak dalje umetanjem objekata u stvarni svijet na način da korisnik može s njima rukovati. Primjerice, to je moguće zamjenom ili dopunom fizički stvarnih objekata sa digitalnima ili dodatnim informacijama koje korisnik vidi kroz naočale [4]. S obzirom da je proširena stvarnost vrlo slična današnjoj uporabi mješovite stvarnosti, proći će se kroz kratku povijest iste.

3.1.1. Povijest

Pojam proširene stvarnosti javlja se još davnih 1950-ih godina kada je kinematograf M. Heiligom došao do ideje da proširi iskustvo gledanja filmova u kinu uvođenjem takozvanog

„kina budućnosti“ u kojemu bi filmovi utjecali na osjetila gledaoca. 1962. godine je realizirao ideju izgradnjom uređaja kojega naziva Sensorama. Uređaj je funkcionirao kao simulator za jednu do četiri osobe koji je pružao iluziju stvarnosti za film koji se pušta. Uređaj je imao mogućnosti emulacije 3D slike sa stereo zvukom, mirisom, sjedalima sa vibracijama i vjetrom. [3], [5]. Takav uređaj štoviše podsjeća na današnja 4D kina gdje se slični efekti i dalje koriste uz 3D polarizirane naočale kako bi se dobio osjećaj realnosti filma. Daljnji razvoj proširene stvarnosti može se pripisati I. E. Sutherlandu koji je 1968. godine izumio uređaj pod nazivom Damoklov mač (*eng. The Sword of Dimocles*). Radi se o prvom HMD (*eng. Head mounted display)* uređaju proširene stvarnosti koji je imao mogućnost prikaza 3D objekata u stvarnom svijetu. S obzirom na težinu i kompleksnost uređaja, on je bio zakačen za mehaničku ruku koja ujedno limitira i korištenje uređaja na jedan prostor. Uređaj se sastojao od dva CRT ekrana po jedan za svako oko te je imao vidno polje od 40 stupnjeva. Sensor koji je pratio položaj glave je bio zasebna komponenta koja se nalazila iznad korisnika uređaja [6]. Ovdje se radilo o prvom HMD uređaju te je ovakav izum izrazito bitan za daljnji razvoj virtualne i proširene stvarnosti.



Slika 3. Damoklov mač [6]

Daljnji razvoj proširene stvarnosti obilježio je M. Krueger 1975. godine stvaranjem iskustva koje naziva „The Videoplace“. Riječ je o projektu koji je omogućio korisnicima interaktivno iskustvo sa virtualnim objektima. Pojam proširene stvarnosti kao takav se prvi put javlja od strane T. Caudell i D. Mizell, Boeing inženjera dok su pomagali radnicima složiti kablove za avion. Iste godine krenule su rasprave prednosti i nedostataka proširene i virtualne stvarnosti od kojih je jedan argument bio da proširena stvarnost troši manje energije jer zahtjeva manje piksela za prikaz grafike [3]. Povećanjem interesa za polje proširene stvarnosti i rapidnim razvojem tehnologija, javljaju se novi sustavi proširene stvarnosti, od kojih je jedan razvijen od strane L.B. Rosenberg pod nazivom „Virtual Fixtures“. Radilo se o sustavu tele prisutnosti gdje su radnici obučeni u exo odijelo mogli kontrolirati robotske ruke na udaljenim mjestima kako bi obavljali neke zadatke. Kamera i ruka jednako kao i kontrola za ruke i ekran su bili odvojeni jedno od drugog. Rosenberg je nastojao poboljšati performanse odrađivanja poslova na udaljenosti svojim sustavom te je za isti objavio rezultate istraživanja gdje je zaključeno da „Virtual fixtures“ poboljšava performanse tele operatera za 70% [7]. Još jedan prototip sustava proširene stvarnosti objavili su S. Feiner, B. MacIntyre i D. Seligmann, pod nazivom KARMA [3]. Virtualni kontinuum koji je spomenut ranije u radu je objavljen tek 1994. godine gdje je opisana pozicija stvarne i proširene stvarnosti na spektru u usporedbi sa stvarnom i virtualnom realnošću [1]. 1997. godine, R. Azuma definira pojam proširene stvarnosti, što označava prvi formalni oblik definicije tog pojma. Prema R. Azuma, proširena se stvarnost definira kao kombinaciju stvarne i virtualne okoline, registrirane u 3D prostoru i interaktivne u stvarnom vremenu [3]. Iz ove definicije može se zaključiti koliko je zapravo definicija mješovite stvarnosti slična proširenoj te je dodatna diferencijacija između dva pojma potrebna.

Pojavom 2000-ih godina, interes za aplikacije proširene stvarnosti raste i javlja se prva AR mobilna igra pod nazivom ARQuake. Igra je predstavljala 3D ekstenziju popularne računalne igre Quake. Aplikacija je bila bazirana na 6DOF sustavu praćenja koja koristi GPS, digitalni kompas i vizualno praćenje putem markera. Igrači su nosili torbu na leđima koja je sadržavala uređaj koji pokreće igru, HMD na glavi i jednostavan kontroler za dodatne kontrole u igri. Igra je omogućavala kretanje u proširenom svijetu jednostavnim hodanjem [8]. Prva pojava ovakvog tipa proširene stvarnosti vrlo brzo se proširila i na većinu mobilnih uređaja današnjice i omogućilo postojanje igara poput „Pokemon GO“.

Ubrzanim razvojem tehnologija, pojavljuje se sve veći interes za aplikacije proširene stvarnosti, o čemu govori i Horizon Report 2005. godine gdje se spominje da će AR tehnologije sve više napredovati u narednih 5 godina. U izvješću se također spominju brojne moguće primjene proširene stvarnosti koje su danas već prisutne, primjerice, u muzejima, botaničkim vrtovima, u treniranju osoblja pogotovo u medicini i slično [9]. Iste godine, javlja se sistem

kamera koje analiziraju fizičko okruženje u stvarnom vremenu i estimiraju pozicije između objekata i okruženja [1]. Kako je Horizon Report predvidio, 2010-ih godina javlja se velik interes ulaganja u virtualne, proširene pa tako i mješovite stvarnosti od strane Microsofta, Facebooka i Google-a. Pojavljuju se uređaji kao što su Google Glass i Microsoft Hololens, uređaji proširene stvarnosti sa brojnim i kompleksnim mogućnostima. Oculus dolazi na tržište sa svojim HMD uređajima pod nazivima CV1, Rift, Rift S i kasnije pod vodstvom Facebook-a odnosno Meta, izbacuje Quest i Quest 2 što predstavlja jedan od najpopularnijih uređaja virtualne stvarnosti današnjice.



Slika 4. Oculus Rift Development kit [8]

Može se reći da Quest uređaji imaju neke elemente mješovite stvarnosti iz razloga što pružaju pogled na stvarnu okolinu kroz kamere na uređaju te omogućuju ubacivanje digitalnih objekata. Bez obzira na to, danas Microsoft Hololens predstavlja jedan od najnaprednijih uređaja mješovite odnosno proširene stvarnosti s brojnim mogućnostima.

3.2. Proširena virtualnost

Na desnom kraju spektra nalazi se proširena virtualnost (*eng. Augmented virtuality*). Takav koncept omogućava potpuno suprotnu funkcionalnost od proširene stvarnosti gdje se stvarni objekti prenose u virtualni svijet [1]. Primjerice, ruke korisnika mogu se projicirati u virtualnom svijetu kako bi korisnik imao veći osjećaj prisutnosti.

3.3. Virtualna stvarnost

Na samom kraju spektra je virtualna stvarnost (*eng. Virtual reality*). Virtualnu stvarnost definiramo kao trodimenzionalnu, kompjuterski generiranu i interaktivnu okolinu koju osoba može istražiti [10]. Na prvi pogled možemo vidjeti kako virtualna stvarnost ima slične karakteristike kao i mješovita stvarnost u smislu da je moguće kretati se kroz okolinu, rukovati digitalnim objektima i slično. Međutim, glavna razlika između virtualne i mješovite stvarnosti je ta da je virtualna stvarnost zapravo kompletno virtualna. Digitalni svijet i objekti su sve što korisnik vidi i nema interakcije sa stvarnim svijetom.

Kada govorimo o virtualnoj stvarnosti, dolazimo do pojmova imerzivnosti i realizma. Ono što virtualna stvarnost želi postići je postojanje čovjeka u digitalno generiranom svijetu uz kompletnu iluziju prisutnosti. Postizanje toga je vrlo teško no razvojem tehnologije je itekako moguće [10]. S obzirom da se moderni sustavi za simuliranje virtualne stvarnosti baziraju na HMD-ima (*eng. Head mounted displays*), prije svega ih je potrebno optimizirati kako bi se postigao osjećaj prisutnosti u digitalnom svijetu.

Današnja ograničenja takvog tipa virtualne stvarnosti i dalje se odnose na stvari kao što je malo vidno polje, slabija implementacija ugrađenog zvuka, manjak preciznog praćenja cijelog tijela, emulacija mirisa, dodira i slično. Međutim, takva ograničenja su manje prisutna u sustavima mješovite stvarnosti kao što je Microsoft Hololens jer se radi o spoju stvarnog i digitalnog svijeta gdje je moguće osjećati prisutnost u stvarnom dijelu mješovite stvarnosti. Bez obzira na to, virtualna stvarnost ima brojne primjene u poljima sporta, medicine, arhitekture, muzeja i sličnog, gdje se također i primjena mješovite stvarnosti pronalazi.

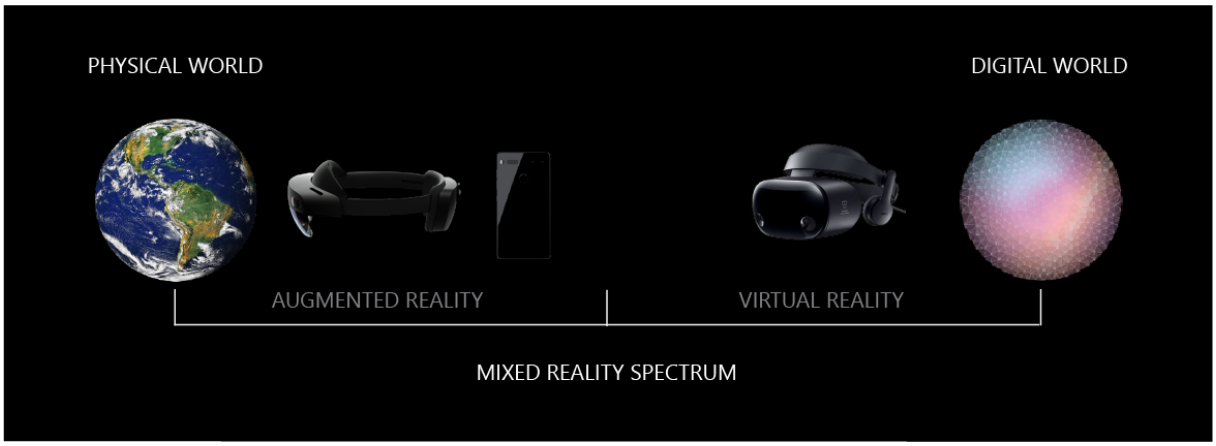
3.4. Vrste uređaja i zaslona

Općenito, uređaje današnjice koji nam pružaju iskustvo mješovite i proširene stvarnosti pronalazimo u HMD obliku. HMD se, ovisno o implementaciji, sastoji od različitih tehnologija ekrana, jednog ili dva ekrana po oku, gustoći piksela, brzini osvježavanja, latenciji, optici i drugih dodataka kao što su ugrađene slušalice. HMD ima tri vrste [11]:

- Samostalni tip – Operira na bazi baterije i samostalnog sustava. Kompletno mobilni tip sa ugrađenim potrebnim komponentama.
- Eksterni tip – Nema vlastiti operativni sustav niti komponente te služi kao eksterni ekran kojega je potrebno spojiti na osobno računalo.
- Hibridni tip – Ima mogućnost samostalnog rada ali i spajanja sa računalom.

S druge strane imamo mobilne uređaje koji danas imaju ugrađenu AR mogućnost pomoću praćenja prostora putem kamera. Prema [12], postoje dvije vrste uređaja ovisno o tipu zaslona kojega sadrže:

- Holografski uređaji – prikazuju digitalne objekte u stvarnom svijetu na način da su što sličniji stvarnim objektima
- Imerzivni uređaji – Blokiraju vanjske podražaje i pružaju što veći osjećaj prisustva u digitalno generiranom svijetu.



Slika 5. Uređaji na spektru mješovite stvarnosti [12]

Ova podjela u dvije vrste uređaja je najčešća. Holografski uređaji iskorištavaju prozirni zaslon kroz kojega je korisnik u mogućnosti promatrati stvarni svijet oko sebe. Preko tih zaslona se zatim umeću digitalni predmeti u svijet korisnika te se time dobiva iluzija da su stvarni. Time se postiže istovremena interakcija sa stvarnim i virtualnim svijetom što bi prema definicija bila mješovita stvarnost [13]. Svi uređaji koji nude ovakav tip mješovite stvarnosti ujedno i koriste takve zaslone radi što boljeg osjećaja prisutnosti. Primjer takvih uređaja su Microsoft Hololens i Magic Leap One. Takav tip zaslona se može podijeliti na [13]:

- Monokularne – Pruža mogućnost pregleda augmentiranog sadržaja kroz jedan ekran, odnosno kroz jedno oko dok se drugim okom promatra stvarna okolina.
- Biokularne – Sastoji se od jednog ekrana koji se pregledava kroz oba oka kroz refleksije.
- Binokularne – Vrsta koja se najviše koristi danas i sastoji se od dva zasebna ekrana za svako oko, stvarajući najveći doživljaj dubine i realizma prikazanog sadržaja. Ovakav tip zaslona koristi Hololens.

S druge strane imamo imerzivne uređaje koji nemaju prozirne zaslone i nastoje „blokirati“ što više vanjskog svijeta u korist digitalnog kako bi se postigao digitalni realizam. Većina takvih uređaja se ne bi klasificirala kao uređaji mješovite stvarnosti jer korisnik nije

u mogućnosti vidjeti stvarnu okolinu i na taj način ne može postavljati digitalne objekte u svoj prostor. Međutim, Microsoft je pokrenuo platformu pod nazivom Windows Mixed Reality u koju spadaju uređaji koji ne moraju biti povezani za računalo već su samostalni i koriste montirane kamere koje skeniraju i mapiraju prostor oko korisnika te prate njegov položaj. Takvo praćenje implementirano je kao praćenje sa 6 stupnjeva slobode (*eng. Six degrees of freedom, 6DoF*) pod nazivom „inside-out“ [13][14]. Time se postiže prirodno kretanje korisnika u digitalnom prostoru kao i pogled na stvarnu okolinu i same WMR uređaje se pomiče bliže sredini spektra mješovite stvarnosti. Najpopularniji primjeri ovakvih uređaja danas su Meta (prijašnje Oculus) Quest 2, Samsung Odyssey, Pico Neo 3 i drugi.

U nastavku će se opisati neki od postojećih uređaja mješovite stvarnosti.

3.4.1. Uređaji

Danas pronalazimo brojne uređaje dostupne javnosti koje omogućuju iskustvo mješovite stvarnosti. No kako se neki uređaji smatraju samo Windows Mixed Reality kompatibilnim, tako postoje i velike razlike između cijena istih, ovisno o tome da li su namijenjeni običnom korisniku ili u korporativne svrhe.

Samsung HMD Odyssey+

HMD Odyssey+ je nastavak na običan Odyssey uređaj koji, uz poboljšane specifikacije, uvodi i WMR (*eng. Windows Mixed Reality*) podršku. Samsung je ovaj uređaj pustio u prodaju krajem 2018. godine uz cijenu od 499$. U samom paketu je došao HMD i dva kontrolera potrebna za upravljanje uređajem. Od glavnih specifikacija najbitnije je spomenuti da se radi o 2 AMOLED binokularna ekrana sa rezolucijom od 1440x1600 piksela po oku, svaki sa osvježenjem od 90Hz. Vidno polje uređaja iznosilo je 101 stupanj po horizontali te je imao ugrađene dvije kamere na prednjem dijelu koje su, uz kontrolere, služile za praćenje kretanja korisnika u stvarnosti te repliciranjem iste pozicije u virtualnom svijetu, čime se ovaj uređaj smatrao WMR uređajem. Odyssey+ nije bio samostalan te je zahtijevao vezu sa računalom kako bi funkcionirao. S obzirom da se ovaj uređaj više ne proizvodi, danas ga je moguće pronaći sa polovnim cijenama od 200$. [15]



Slika 6. Samsung Odyssey+ [15]

HP Reverb G2

Reverb G2 je HP-ov odgovor na WMR uređaje koji je izašao 2020. godine sa cijenom od 600$. Radi se o primarno VR uređaju sa WMR podrškom koji, kao i Odyssey, ima dvije prednje kamere i još po jednu kameru sa svake strane sa mogućnostima analize i praćenja prostora u kojemu se korisnik kreće. Također, valja napomenuti da je ovaj uređaj izašao na tržište u suradnji sa partnerima Microsoft i Valve. Radilo se o dva LCD ekrana sa razlučivosti od 2160x2160 piksela po oku te osvježenjem od 90Hz. Ovaj uređaj bio je nešto lakši od ostalih i sadržavao je veoma ugodnu soluciju za pričvršćivanje uređaja na glavu korisnika kao i ugrađene zvučnike sa svake strane. Za razliku od Odyssey uređaja, Reverb je omogućavao korisnicima namještanje udaljenosti između leća kako bi se omogućilo lagodno korištenje od strane svakog korisnika. [16]



Slika 7. HP Reverb G2 [16]

Gotovo svi WMR kompatibilni uređaji današnjice su zapravo VR uređaji sa ugrađenim kamerama koji omogućuju analizu prostora, praćenje korisnika u prostoru te neki i pregled prostora kroz leće uređaja. Takvi uređaji izuzetno dobivaju na popularnosti i dostupni su u prodaji uz privlačne cijene. Razvoju u ovom polju uvelike su pridonijeli Facebook i Oculus koji trenutno drže veliki udio na tržištu sa svojim Meta Quest 2 VR uređajima koji trenutno implementiraju neke sposobnosti uređaja mješovite stvarnosti kao što je praćenje ruku korisnika, orijentacija u prostoru, dodavanje stvarnih objekata u virtualni prostor, i slično. Valja spomenuti i druge WMR kompatibilne i popularne VR uređaje kao što su Valve Index, HTC Vive Pro 2 i Oculus Rift S.

Do sad su opisani samo WMR kompatibilni uređaji koji se nalaze bliže virtualnom dijelu spektra, no sada će se pogledati neki od popularnijih uređaja mješovite, odnosno proširene stvarnosti.

Google Glass

Google Glass je uređaj proširene stvarnosti koji je prvi puta predstavljen javnosti 2014. godine u obliku pametnih naočala proširene stvarnosti. Naočale su razvijene od strane Google X, divizije unutar Google-a zadužene za razvijanje naprednijih tehnologija. Takav prototip ubrzo se prestao prodavati dok 2017. godine nije izašla Enterprise verzija naočala sa poboljšanjima, a ubrzo nakon toga i 2019. trenutna verzija Enterprise Edition 2. Google Glass je poseban po tome što nalikuje običnim naočalama pritom imajući mogućnosti raznih uređaja mješovite stvarnosti. Glass ima područje osjetljivo na dodir sa jedne strane koje omogućuje korisnicima kontrolu nad sučeljem koje se prikazuje na naočalama. Također, ima ugrađenu kameru s mogućnošću snimanja videa i slikanja. Ekran koji preslikava digitalne objekte u stvarnost se nalazi na jednoj strani naočala i ima mogućnosti puštanja videa, integraciju sa Google kartama, translacija teksta, otvaranje mailova i poruka, dijeljenja sadržaja na društvenim mrežama i slično [17]. Ovdje se više radi o AR naočalama s obzirom da se digitalni objekti samo preslikavaju, te Glass nema mogućnosti drugih uređaja specifičnih za mješovitu stvarnost.



Slika 8. Google Glass sa okvirima s dioptrijom [17]

Magic Leap 1

Magic Leap 1 je prvi uređaj proširene stvarnosti firme Magic Leap koji je izašao na tržište javnosti 2018. godine nakon dugogodišnjeg tajnog razvoja. Uređaj nije nimalo jeftin sa cijenom od preko 2000$. Nalik naočala kao što je to slučaj i kod Google Glass, Magic Leap 1 teži svega 316 grama te je samostalan uređaj koji se vrti sustav Lumin OS. Ima ugrađene zvučnike, 1.3 milijuna piksela po oku te osvježenje od 120Hz. Mana naspram ostalih uređaja je izuzetno niskim vidnim poljem od samo 50 stupnjeva. S obzirom da je uređaj pogonjen baterijom koja traje nekoliko sati, moguće je slobodno kretanje po prostoru. Uređaj koristi jedan kontroler nalik klasičnim VR i AR kontrolerima kojim se upravlja digitalnim objektima i menijima. Magic Leap reklamira 1 kao uređaj za konferencijske i poslovne svrhe ali i za zabavne svrhe kao što je igranje igara. Jedna od igara koja je dostupna za ovaj uređaj je Angry Birds, napravljena specifično za Magic Leap 1 i njihovu trgovinu. Trenutno je ovakav uređaj veoma teško preporučiti javnosti radi cijene, ali i dostupnosti sadržaja, no Magic Leap je osigurao dodatna ulaganja kako bi u bližoj budućnosti razvio novu verziju uređaja koji će, moguće, biti pristupačniji. [18][19]



Slika 9. Magic Leap One [18]

Microsoft Hololens 2

Microsoft Hololens 2 je poboljšana verzija originalnog Hololensa, koja je izašla 2019. godine kao Microsoftov odgovor na mješovitu stvarnost. Za razliku od prijašnje spomenutih uređaja, ovaj uređaje je izuzetno skup sa cijenom od preko 3000$ te je namijenjen u edukacijske, medicinske i korporativne svrhe. Uređaj radi samostalno i koristi Microsoftov sustav sa brojnim aplikacijama dostupnim u Microsoft trgovini. Uređaj ima nekoliko kamera za praćenje prostora, očiju korisnika te dubine prostora. Koristi se prozirni holografski ekran kako bi korisnik vidio digitalne objekte u stvarnom prostoru. Microsoft na svojim stranicama navodi kako je uz pomoć Hololens uređaja povećana efikasnost u proizvodnji za 90%, 4 milijune dolara estimirane vrijednosti uštedevine u gradnji, brže vrijeme obuke medicinskog osoblja i brojne druge pogodnosti [20]. Detalji uređaja razraditi će se u zasebnom poglavlju u nastavku.

3.5. Primjene

Neupitno je da je mješovita stvarnost tehnologija koja je korisna u brojnim poljima. Uređaji koji implementiraju mješovitu stvarnost postaju sve pristupačniji, lakši i udobniji za nošenje. Trenutno dostupni uređaji kao što su Hololens, Magic Leap 1, pa tako i nadolazeći Apple MR, većinom se i reklamiraju na način da se njihovim uvođenjem u radnu okolinu povećava produktivnost, efikasnost, olakšavaju zadaci i slično. Prije svega, valja napomenuti nekoliko uobičajenih slučajeva uporabe ove tehnologije [2]:

- Dizajn i izrada prototipa – Omogućuje suradničku interakciju fizičkih i virtualnih modela u stvarnom vremenu među višefunkcionalnim timovima.
- Razvoj i obuka – Olakšava instruktorima posao davanja obuka, kao i pruža zanimljivo i olakšano iskustvo učenja polaznicima.
- Geoprostorno planiranje – Omogućuje planiranje unutrašnjih i vanjskih prostora uz 3D prikaze pomoću kojih se mogu planirati buduće trgovine, interijeri i slično.
- Prodaja – Uvođenje 3D kataloga i virtualnih iskustava u trgovine čime se poboljšava iskustvo kupaca prilikom kupovine.
- Terenska usluga i podrška – Omogućuje virtualni izlazak stručnog tima na teren kako bi se riješili problemi kupca bez fizičke prisutnosti. Također može služiti i kao platforma za ciljane oglase kako bi se povećala prodaja.
- Produktivnost i suradnja – Pretvara fizički prostor u virtualni koji služi kao prošireno radno mjesto gdje udaljeni korisnici mogu surađivati, pretraživati i dijeliti sadržaj sa fizičkima kao da su u istoj prostoriji.

3.5.1. Kolaboracija

Možda najzanimljivija točka od gornje navedenih je zajednički kolaborativni prostor bez potrebe fizičke prisutnosti, ali istovremeno pregled i fizičkog i virtualnog svijeta. Microsoftova implementacija takvog prostora naziva se Microsoft Mesh. Mesh predstavlja upravo to, zajednički virtualni prostor za kolaboraciju. Unutar Mesh prostora, ljudi koji nisu fizički prisutni su predstavljeni kao virtualni avatari koji su vidljivi na točnom mjestu gdje se korisnik i fizički nalazi. Sustav prostornog zvuka dodatno poboljšava osjećaj prisutnosti. Microsoft Mesh se primarno može koristiti za kolaboraciju, pogotovo s osobama koje su u drugim vremenskim zonama, ali i za preglede 3D dizajna, pomoć drugima bez fizičke prisutnosti, kolaboraciju na istim projektima koji se spremaju u oblaku, učenje i zajedničku obuku, virtualne sastanke i slično. [2]

3.5.2. Proizvodnja, logistika i robotika

Prilikom proizvodnje, sustavi mješovite stvarnosti mogu se koristiti kako bi precizno mjerili dijelove ili proizvode prilikom procesa njihove proizvodnje i sastavljanja. Također bi mogli provoditi radnika kroz proces sastavljanja proizvoda pritom informirajući ga o najboljim praksama i metodama. Slično, Hololens se može koristiti kao zamjena za sustav upravljanja robota odnosno robotskih ruku. Korisnik bi na udaljenom mjestu kontrolirao robotsku ruku, obavljao zadatke i pritom putem kamere u stvarnom vremenu pratio svoj rad. Koncept mješovite stvarnosti može se primijeniti i u kompliciranijim slučajevima gdje je potrebno predvidjeti putanju kretanja robota putem algoritma planiranja puta. Još jedna zanimljiva primjena uključuje stvaranje sekvenci rada za robote koji se pohranjuju u sustavu i zatim pokreću. Korisnik bi snimio sekvencu kontroliranja robota putem MR aplikacije, gestura i glasovnih komandi te interakcijom sa objektima. Kasnije bi se snimljena sekvenca mogla koristiti sa pravim robotima. U logistici bi se mješovita stvarnost mogla koristiti prilikom praćenja paleta i montaže kompleksnijih proizvoda. Aplikacija bi prikazivala koji dijelovi su potrebni za određeni proizvod te u prostoru označila koja paleta sadrži određeni dio za montažu proizvoda. [21]

3.5.3. Edukacija

Također korisna primjena VR i AR tehnologije se nalazi u polju obuke i edukacije. Prema [22], provedeno je istraživanje o utjecaju aplikacija proširene stvarnosti kod podučavanja učenika u polju znanosti. Cilj je bio identificirati stav 42 učenika osnovne škole prema aplikacijama proširene stvarnosti kao i njihov pogled na takvu vrstu tehnologije. Kroz trajanje istraživanja od 8 tjedana, studenti su podijeljeni u dvije grupe sa sličnim razinama motivacije bez korištenja tehnologija mješovite stvarnosti, te je jedna grupa bila zadužena za učenje uz pomoć MR aplikacija. Polja učenja bila su primarno biologija i solarni sustav. Krajem istraživanja zaključeno je kako su učenici koji su koristili MR aplikacije imali viši postotak motivacije za učenjem, manje razine anksioznosti, veću želju za kolaboracijom s ostalim učenicima kao i pozitivan stav prema tehnologiji mješovite stvarnosti. S obzirom da se kroz godine radilo o različitim implementacijama ove tehnologije, prema [23] je provedeno starije istraživanje iz 2005. godine, gdje su uvedene dvije aplikacije za edukaciju i zabavu u izložbenu dvoranu znanstvenog centra i u vojne svrhe. Prva aplikacija implementirana je u Orlando znanstvenom centru gdje su bile postavljeni fosili raznih vodenih životinja. Radilo se o spoju VR i AR tehnologija uz pomoć sferičnog ekrana kroz kojega su gosti gledali u drugi dio prostora gdje su se fizički i digitalni prostor spajali prikazom vode, virtualnog vodiča i kretanja životinja koje su finalno završile kretanje na mjestu gdje se fizički nalaze njihovi fosili. Ispitivanjem su došli do zaključka da gotovo 98% gostiju smatra ovu tehnologiju pozitivnom te ih potiče na

ponovnu posjetu. Drugi dio istraživanja odnosio se na edukaciju vojne prirode u obliku MR MOUT simulacije. Radilo se o HMD-u koji je kroz kamere korisniku, uz fizički prostor i pušku, dodao digitalne objekte zgrada i protivnika. Korisnik se mogao kretati kroz prostor, skrivati iza zidova, pucati po protivnicima i uz pomoć prostornog zvuka, orijentirati i otkriti gdje se neprijatelj nalazi. Računalo je pratilo simulaciju i temeljem lokacije korisnika je otvaralo prozore i vrata u fizičkom svijetu. Ovim istraživanjem pokazalo se koliko je bitno trenirati svjesnost situacije vojnika i operativaca te od kolike je važnosti oslanjanje na zvukovne podražaje u bitkama koje se odvijaju na urbanim područjima.



Slika 10. MR tehnologije u Orlando znanstvenom centru [23]

3.5.4. Zabava

Aplikacije mješovite stvarnosti ne moraju biti limitirane na edukacijske i korporativne svrhe, već se mogu koristiti i u svrhu zabave. Kroz godine su izašle razne implementacije takve tehnologije za zabavu, no s porastom popularnosti mobilne tehnologije, možda najpopularnija implementacija koja je većini poznata je mobilna igra Pokemon Go. Pokemon Go je mobilna igra koju je na tržište pustio Niantic 2016. godine koja je omogućila korisnicima virtualno hvatanje Pokemon stvorenja preko mobitela, ali i fizičko kretanje. Korisnici se moraju fizički kretati kako bi otkrili i došli do lokacija gdje se nalaze Pokemoni kako bi ih uhvatili. Možda najzanimljivija značajka igre bila je mogućnost korištenja kamere kako bi se Pokemoni koji se hvataju postavili u fizički prostor korisnika. Time se omogućio aspekt mješovite stvarnosti i dodatna iluzija prisutnosti virtualnih stvorenja u stvarnom svijetu [24]. Ubrzo nakon što je igra dobila na popularnosti, počele su se razvijati slične mobilne igre kao što su Mario Kart Live:

Home Circuit gdje je moguće voziti karting utrke u vlastitom domu spojem fizičkog i virtualnog prostora, i Harry Potter: Wizards Unite, igra slične prirode kao i Pokemon Go.

3.5.5. Medicina

Primjena MR tehnologije u medicini je i dan danas aktivno istraživačko područje koje ima brojne primjene. Može se koristiti u svrhe edukacije osoblja prikazom 3D anatomskih modela koje osoblje može pobliže proučavati ali i kontrolirati u prostoru. Također, moguće je i stvoriti biblioteku slučajeva koju osoblje može pregledavati. Pacijenti bi mogli pregledavati vlastite skenove ili procedure kako bi im kirurg ili doktor pobliže objasnio njihov slučaj, odnosno proceduru te time smanjio anksioznost pacijenata. Dodatno bi se ista tehnologija mogla ukomponirati u salu za kirurgiju projiciranjem 3D slike tijekom obavljanja zahvata. Time bi kirurg dobio virtualnu asistenciju ali i mogućnost direktnog asistiranja osoblja koje nisu fizički prisutne [25]. Zanimljiva primjena Microsoft Hololens uređaja pronalazi se upravo u odjelu medicine za vrijeme tretiranja COVID-19 pacijenata. Prema [26], korištenjem Hololens uređaja prilikom rada sa COVID-19 pacijentima rezultiralo je padom boravka u područjima visokog rizika zaraze za 83% kao i izuzetno smanjilo korištenje osobne zaštitne opreme jer jedino doktor koji nosi Hololens mora biti prisutan uz pacijenta dok su drugi doktori i osoblje prisutni virtualno. Osoblje koje nosi uređaj na taj način može istovremeno biti uz pacijenta i biti u virtualnom Teams pozivu sa ostatkom osoblja ali i stručnjacima diljem svijeta koji mogu tako davati savijete.

Inovacije tehnologija mješovite stvarnosti u računalno potpomognutoj kirurgiji nastoji povećati preciznost izvođenja zahvata smanjenjem komplikacija. Takvi razvitci su prisutni i danas se koriste u raznim poljima medicine, naročito korištenjem MR tehnologija jer zahtijevaju manje planiranja prije operacija, pozicioniraju se u prostor uz fizičke objekte i moguće ih je, za razliku od AR tehnologije, kontrolirati. Ovdje se također istaknuo Hololens kao najlakši i najugodniji za nošenje naspram drugih uređaja. Kirurzi su izjavili kako je sučelje Hololens uređaja intuitivno, može se namještati po volji te je izuzetno bitna interakcija putem gestura i glasa prilikom izvođenja zahvata. Specifično u ortopedske svrhe, Hololens se koristio pri izvođenju zahvata na više načina od kojih je jedan izvođenje operacije uz virtualni video poziv sa četiri druga stručnjaka, čime je operacija završila u optimalno vrijeme i bez komplikacija [27]. Također, prezentiran je sustav mješovite stvarnosti koji prema kirurškom alatu pruža smjernice i podršku za brzu lokalizaciju ulazne točke. Korisnik odabire kirurški alat i uspostavlja planiranu putanju koristeći CT podatke pacijenta. Time korisnik dobiva na vremenu i može s lakoćom prostorno pozicionirati kirurški alat. Još jedna zanimljiva aplikacija je u ortognatskoj kirurgiji gdje se prati gibanje čeljusti i u stvarnom vremenu preko fizičkog kretanja stavlja

virtualni model koji prati isto kretanje čeljusti. Taj model se može prikazati na zasebnom ekranu kako bi isti zahvat pratio i ostatak osoblja [27].

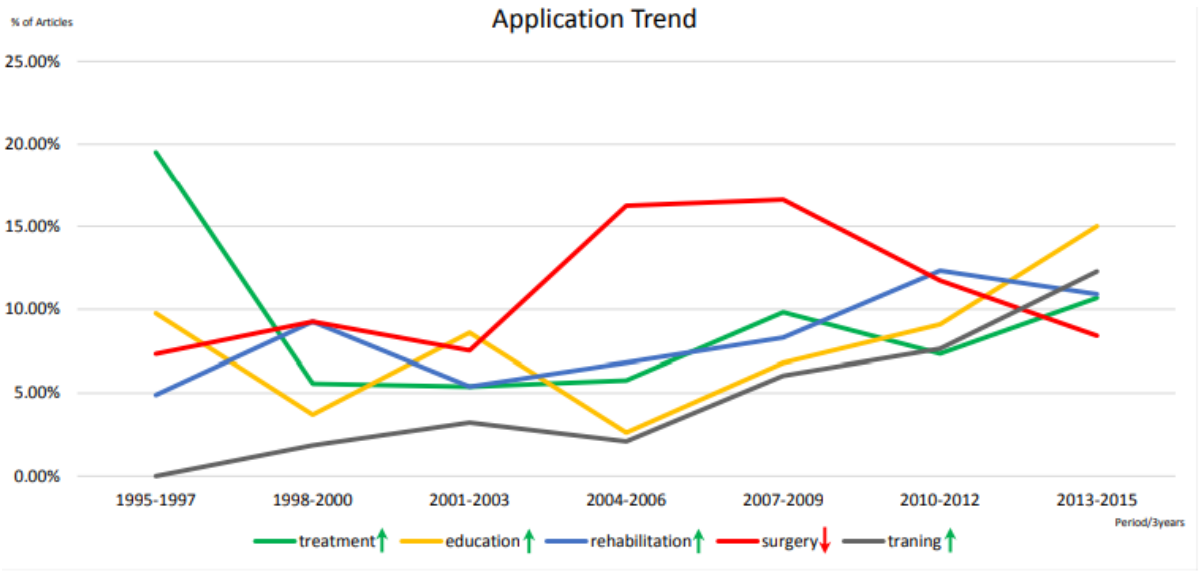
3.6. Budućnost i izazovi

San da se stvori virtualna soba u kojoj prisustvuju stvarni objekti i ljudi sa virtualnima postoji još od Sutherlandove ideje trodimenzionalnog uređaja. Hologrami i razni lebdeći UI (*eng. User interface*) elementi uvijek su bili prisutni samo u filmovima naučne fantastike, no zahvaljujući današnjem razvoju u polju proširene i mješovite stvarnosti, taj san polako prelazi u stvarnost.

U današnje doba, sve je veća potreba za online komunikacijama i online nastavom na fakultetima i raznim učilištima. Dovođenje mješovite stvarnosti u takve online edukacije dovodi sa sobom brojne benefite ali i izazove. Prostori za edukaciju mogli bi se prikazati kao 3D modeli unutar kojih bi studenti prisustvovali nastavi bilo kroz 2D ekrane na računalima ili u potpunosti u virtualnom prostoru kroz uređaje mješovite stvarnosti. Ne samo da bi to donijelo veći osjećaj prisutnosti nego bi studenti bili više zainteresirani za praćenje nastave, što je i prikazano u istraživanju učenika osnovne škole pri korištenju aplikacija mješovite stvarnosti u edukaciji [22]. Profesorima bi bili dostupni brojni alati kao što je virtualna ploča za pisanje s mogućnostima virtualnog ali i stvarnog crtanja od strane studenata koji virtualno prisustvuju te 3D modeli putem kojih bi profesori mogli preciznije studentima prenijeti znanje što bi ujedno bilo vrlo korisno u podučavanju medicine, fizike, biologije i sličnih polja. Putem web kamera bi se mogli pratiti studenti te njihove kamere prikazivati na ploči, pa tako i web sadržaji i video zapisi. Kako bi se što više poboljšala prisutnost, studenti koji prisustvuju virtualno bi mogli biti prikazani kao hologrami projicirani na stakla, ali i profesori koji bi mogli podučavati istovremeno u stvarnošću i virtualnom svijetu [28]. Međutim, koliko takva implementacija mješovite stvarnosti zvučala privlačno, ona dolazi sa brojnim izazovima. Prvenstveno, takvo virtualno praćenje i projiciranje studenata zahtjeva stabilnu i dobro implementiranu umreženost. U istraživanju [28], takva virtualna učionica napravljena je u Unity 3D alatu te je učionica modelirana po jednoj IT sobi na učilištu računalne znanosti u Dublinu. Povezanost su pokušali riješiti korištenjem ugrađenog alata za umrežavanje UNET. Korištenjem tog alata moguće je napraviti jednostavne P2P (*eng. Peer-to-peer*) veze između studenata. Međutim, alat nije bio savršen i nije omogućavao streaming video poziva između više studenata bez greške. Korištenjem softvera FlexiHub je taj problem popravljen, no došlo je do problema glatkog prijenosa video poziva, gdje je FlexiHub znatno usporio video pozive. U sažetku navode kako će se dalje koristiti UNET no za gladak prijenos video poziva i umrežavanje studenata će biti potrebno razviti dodatne C# skripte, čime se dobiva uvid da umrežavanje nekoliko stotina

studenata i profesora kroz nekoliko dvorana istovremeno nije jednostavno. Drugi izazov koji se ovdje javlja je hardver. Prvenstveno, virtualne dvorane je potrebno dizajnirati ili skenirati što zahtjeva znanje u Unity ili sličnom alatu. Metode koje su istražene u istraživanju [28] su skeniranje dvorane i objekata sa Kinect alatom te dizajn dvorane i objekata kroz alat Unity. Naravno, skeniranje objekata sa Kinect alatom nije najpreciznije pa je bolja opcija njihov dizajn u alatu Unity, no valja voditi računa o broju poligona od kojih se objekt sastoji. Ako su objekti pre kompleksni, oni zahtijevaju bolji hardver i time ruše performanse uređaja mješovite stvarnosti. Slično se javlja i kod korištenja dinamičnih izvora svjetlosti koji zahtijevaju više procesorske snage od statičnih. Naravno, ni sama nabava uređaja kojima bi se ovakva učionica izvela nije jednostavna ni jeftina.

Medicina je polje od velike važnosti za razvoj tehnologije mješovite stvarnosti. Kao što se pokazalo u ranijim poglavljima, takva tehnologija može uvelike ubrzati vrijeme izvođenja operacija i zahvata te poboljšati edukaciju medicinskog osoblja. U znanstvenom članku [29], objavljeno je istraživanje raznih trendova korištenja tehnologija proširene i mješovite stvarnosti od 1995. godine do danas. Analizirano je nekoliko tisuća objavljenih članaka vezanih za mješovitu stvarnost u medicini. Prema istraživanju, vidljivo je kako je porast broja objavljenih članaka na tu temu u znatnom eksponencijalnom porastu u skladu sa razvitkom MR tehnologija, čime je sa 41 publikacija u vremenu od 1995. do 1997. godine, broj objava porastao na 440 do 2015. godine. Predviđanjem je zaključeno da bi moglo biti i preko 700 publikacija do 2022. godine. Još jedna zanimljiva analiza odnosi se na trendove u aplikacijama koja je provedena u istom članku, a prikazuje kretanje trendova od 1995. do 2015. godine kod korištenja MR u medicini. [29]



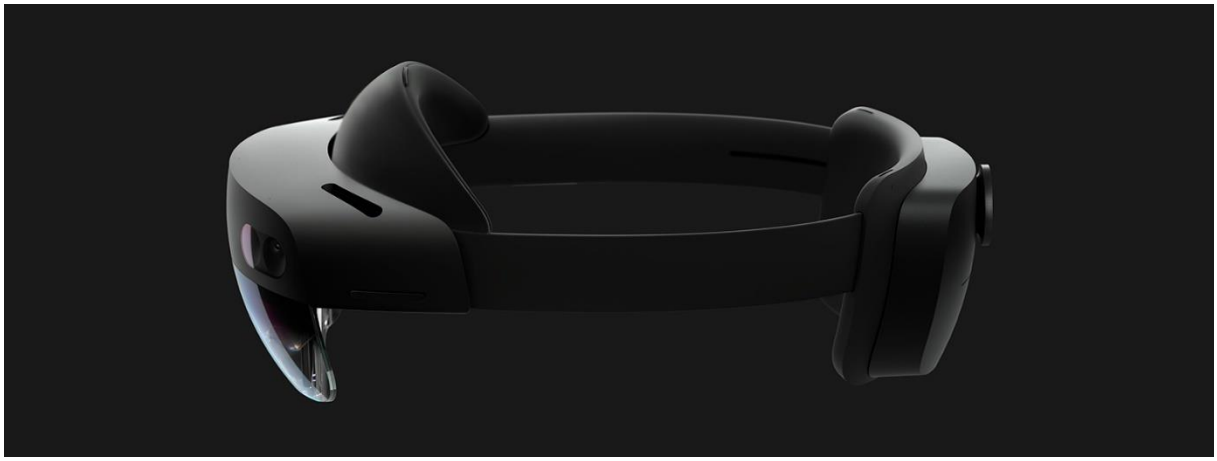
Slika 11. Kretanje MR trendova kod aplikacija u medicini [29]

Prema grafikonu se može zaključiti da je ujedno najistraživija tema vezana za liječenje pacijenata u ranim godinama, ali i ostaje aktivna danas sa 10%-nim udjelom objavljenih članaka. U tom pogledu, tehnologije mješovite stvarnosti mogu se koristiti kao pomoć pri liječenju pacijenata prikazivanjem njihove povijesti bolesti, skeniranja tijela, prikaz vena pri uzimanju krvi i sl. Ono što je odmah uočljivo je također i nagli porast u istraživanjima korištenje takvih tehnologija kod izvođenja operacija što korelira sa naglim razvojem tehnologija mješovite stvarnosti u to doba. Kod operacija, MR tehnologije omogućuju minimalno invazivne operacije i navigaciju kroz tijelo pacijenta što bi teoretski smanjilo broj komplikacija nakon i tijekom zahvata. Aplikacije takvih načina izvođena operacija su i dalje u istraživačkoj fazi te mali broj ih je trenutno u primjeni, no taj broj će brzo rasti. Najmanje istražena tehnologija pripada treniranju osoblja, no njezin postepeni rast sugerira da bi mogla, zajedno sa edukacijom, biti najbitnije područje primjene AR i MR tehnologija u budućnosti, ne samo medicine, nego i ostalih polja. [29]

Također aktivno područje istraživanja proširene i mješovite stvarnosti je u marketingu i zabavi. Danas, pogotovo u većim gradovima u, primjerice, Americi i Japanu je moguće vidjeti brojne oglase, reklame kao i znakove za trgovine te obrte koji se nalaze po zgradama, zidovima, ekranima u javnim prijevozima i slično. Takav način reklamiranja uz pomoć mješovite stvarnosti bi se mogao prikazivati direktno korisniku kroz naočale koje nosi. Reklame i oglasi bi se prilagođavale navikama korisnika, te ne bi bile vidljive izvan proširene stvarnosti. Ekrani, mobilni uređaji i konzole bi se mogle sastojati od samo ekrana na koji se projiciraju sadržaji u obliku holograma. Statusi poput poruka, trenutnog vremena, informacija o zdravlju korisnika bi se kroz naočale kao što je Hololens mogle prikazivati direktno na ruci korisnika bez ikakvih drugih pomagala. Ono što je najviše nalik tome, no i dalje se radi o potpuno virtualnom umjesto proširenom iskustvu je Metaverse, trodimenzionalni svijet unutar kojeg su ljudi prikazani putem vlastitih avatara. Metaverse je dobio na popularnosti u ranom dijelu 2020. godine te je od tada u razvitku od strane brojnih tvrtki sa vlastitim implementacijama kao što su Nvidia, Facebook (trenutno Meta) i Microsoft. Takvi svjetovi bi bili nalik stvarnima te bi imali vlastitu ekonomiju, posjede, interakciju s ljudima pa i marketing. No tu se navodi pitanje što Metaverse zapravo predstavlja. Mnogi autori ga definiraju kao Web 3.0 te da je on već skoro tu u primitivnoj formi. Također se javlja problem privatnosti u takvom svijetu te kako će se ista čuvati u vidu korisnika, s obzirom da je velik dio takvog svijeta osmišljen oko marketinga i druženja. Trenutna integracija Metaverse-a sa proširenom stvarnošću kako bi se dobila mješovita stvarnost nije poznata, no samo će vrijeme reći. [30]

4. Microsoft Hololens 2

Hololens 2 je uređaj mješovite stvarnosti koji je na tržište pustio Microsoft 2019. godine. Kao nastavak na originalni Hololens, ovaj uređaj druge generacije pridonosi brojna poboljšanja. Uređaj radi na Windows Holographic operacijskom sustavu koji je vrlo sličan Windows 10 operacijskom sustavu te Microsoft konstantno izbacuje ažuriranja za isti. Trenutno se ovaj uređaj prodaje u više varijanti, i to Hololens 2, Hololens 2 Industrial Edition i Development Edition čija se cijena kreće od 3,500$ pa na više [31]. I dan danas, Hololens se smatra de facto standardom za mješovitu stvarnost.



Slika 12. Microsoft Hololens 2 [31]

4.1. Tehničke specifikacije

Hololens 2 se sastoji od nekoliko komponenti [31]:

- Vizor – Sadrži senzore i ekran te ga je moguće podignuti prema gore
- Traka za glavu – Nalazi se na zadnjem dijelu uređaja i služi za učvršćivanje uređaja na glavu korisnika. Uključuje i kotačić čijim okretanjem se sužava ili olabavi traka.
- Gumbi – Hololens ima nekoliko gumbi koji se nalaze na strani uređaja a služe za povećavanje odnosno smanjivanje zvuka i svjetline ekrana te gumb za gašenje i paljenje uređaja.
- USB-C port – Nalazi se na strani uređaja i služi za punjenje ili prebacivanje sadržaja sa ili na uređaj.

Uređaj koristi prozirni ekran s mogućnošću prikazivanja holograma te praćenja točne pozicije očiju korisnika. Od ostalih senzora, Hololens sadrži četiri svjetlosne kamere pozicionirane na lijevoj i desnoj strani uređaja kako bi se korisnik pozicionirao u prostoru te dvije infracrvene kamere za praćenje očiju korisnika. Dodatno, uređaj ima i jednu kameru za praćenje dubine prostora te prednju kameru od 8 megapiksela za snimanje videa. Microsoft navodi kako uređaj ima ugrađene zvučnike sa mogućnošću reprodukcije pozicijskog zvuka te ugrađen mikrofon [31]. Niže je navedena tablica sa specifikacijama hardvera:

Tablica 1. Tehničke specifikacije uređaja Microsoft Hololens 2

Procesor	Qualcomm Snapdragon 850
Holografski procesor	Vlastiti holografski procesor druge generacije
Memorija	4GB LPDDR4x
Prostor	64GB UFS 2.1
Wi-Fi	802.11ac
Bluetooth	5.0
USB	USB Type-C DRP

(Izvor: [31])

S obzirom da je uređaj kompletno mobilan, radi na baterijama koje drže 2-3 sata na jednom punjenju. Hlađenje uređaja odrađeno je pasivno bez ventilatora te je Hololens moguće koristiti i prilikom punjenja. Microsoft na svojim stranicama ima grafički prikaz svake komponente od koje se uređaj sastoji te detaljnu dokumentaciju dostupnu javnosti.

4.2. Interakcija

Hololens je u prvom pogledu uređaj namijenjen za hands-free kontrolu pa se kao takav može kontrolirati putem gestura, glasovnih komandi, rukama i očima.

Najbitnija interakcija je putem očiju korisnika. Ovdje se radi o indirektnoj interakciji jer Hololens koristi podatke o praćenju očiju korisnika kako bi pravilno i u što boljoj kvaliteti pozicionirao i prikazao holograme u prostoru. U tom pogledu, kalibracija se dešava automatski i specifična je za svakog korisnika. Također, kako se radi o praćenju očiju na razini softvera, nije potrebno namještati inter pupilarnu udaljenost, odnosno udaljenost između oba oka, kao što je to potrebno na većini komercijalno dostupnih uređaja. S obzirom da uređaj automatski prati u koji dio ekrana korisnik gleda, ovo omogućava developerima daljnju razinu realnosti

kod razvoja aplikacija za Hololens. Praćenje očiju na Hololens 2 uređaju nije moguće ugasiti, no isto je moguće napraviti na Hololens 1 uređaju. Kako se u većini uređaja koji koriste ručno podešavanje inter pupilarne udaljenosti javlja nezadovoljnost korisnika koji nisu u rangu IPD (*eng. Interpupillary distance*) uređaja, u slučaju Hololensa to se neće javljati. Uobičajeno, kalibracija praćenja očiju dešava se čim novi korisnik stavi Hololens na glavu. Korisnik će morati proći kroz kratku kalibraciju gdje je potrebno pogledati u nekoliko heksagona koji se prikažu na ekranu te potvrditi kalibraciju. Međutim, takva kalibracija se izvodi čak i za aplikacije koje ne koriste tu mogućnost pri čemu je korisna mogućnost automatske pozicije očiju (AEP, *eng. Auto Eye Position*). AEP je mogućnost koju je moguće uključiti i omogućava sustavu da automatski namješta holograme i objekte ovisno o podacima pogleda korisnika koji se skupljaju pasivno kako korisnik koristi uređaj. Na taj način, korisnik neće morati raditi kalibraciju ukoliko upali aplikacije koje ne koriste tu mogućnost, međutim Microsoft i dalje preporučuje kalibraciju ako se koriste druge mogućnosti koje zahtijevaju praćenje očiju [31].

Drugi glavni način interakcije je putem gestura i ruku. Prilikom korištenja Hololens uređaja, kako bi se kontrolirali objekti rukama, potrebno ih je držati ispred sebe kako bi senzori mogli pratiti ruke. Holograme je moguće selektirati jednostavnim dodirivanjem čime će se na prstu s kojim se hologram dodiruje prikazati kursor. Prozori se mogu micati ali i skrolati jednostavno pomakom prsta gore ili dolje. Holografi se mogu nositi i micati po prostoru selektiranjem i dodirivanjem palca i kažiprsta odnosno „štipanjem“. Na isti način se mogu i rotirati ili povećavati odnosno smanjivati hvatanjem na dvije suprotne strane te rotacijom ili odvajanjem odnosno približavanjem ruku. Ako objekti nisu blizu, Hololens će automatski prikazati isprekidanu liniju koja vodi do objekta prema kojemu prikazujemo, čime se i on sam može kontrolirati. Takvu interakciju Microsoft naziva Air Tap i ona se može koristiti kako bi se [31]:

- Skrolali prikazani prozori prstohvatom i micanjem gore ili dolje.
- Hvatali razni objekti prstohvatom i držanjem kao i mijenjala pozicija prikazanim prozorima.
- Otvarali kontekstualni izbornici

Startni gumb nije prikazan na prozorima, nego je dostupan putem gesture na način da se prstom dodirne područje ispod zgloba ruke gdje će Hololens prikazati Start gumb. Na sličan način, moguće je otvoriti start tako da se ruka gdje se on projicira okrene prema sebi i napravi prstohvat te pogleda u ikonu na ruci. Pomoć pri rotiranju ili mijenjanju veličine hologrami omogućuju kutni indikatori na hologramima koje je moguće uhvatiti te izvršavati navedene radnje. Za pisanje se koristi virtualna tastatura po kojoj je moguće tipkati kao da se koristi

fizička tastatura. Također, slično mobilnim uređajima, postoji i opcija prelaska prstiju po tipkama i povlačenjem linije kako bi se pisala riječ po riječ. [31]

Zadnji način kontrole je putem glasovnih komandi. Glasovne komande na uređaju rade na principu sličnom kao i na Windows operacijskim sustavima. Neke glasovne komande su ugrađene i specifične za Hololens uređaj, no moguće je koristiti i virtualnog asistenta Cortana za dodatne glasovne kontrole. Glasovne komande prvotno su isključene na uređaju te ih je potrebno uključiti odlaskom u opcije uređaja. Kako Hololens koristi isti način prepoznavanja glasa kao i Windows, glasovne komande su dostupne u više jezika. Neke od dostupnih komadni uključuju [31]:

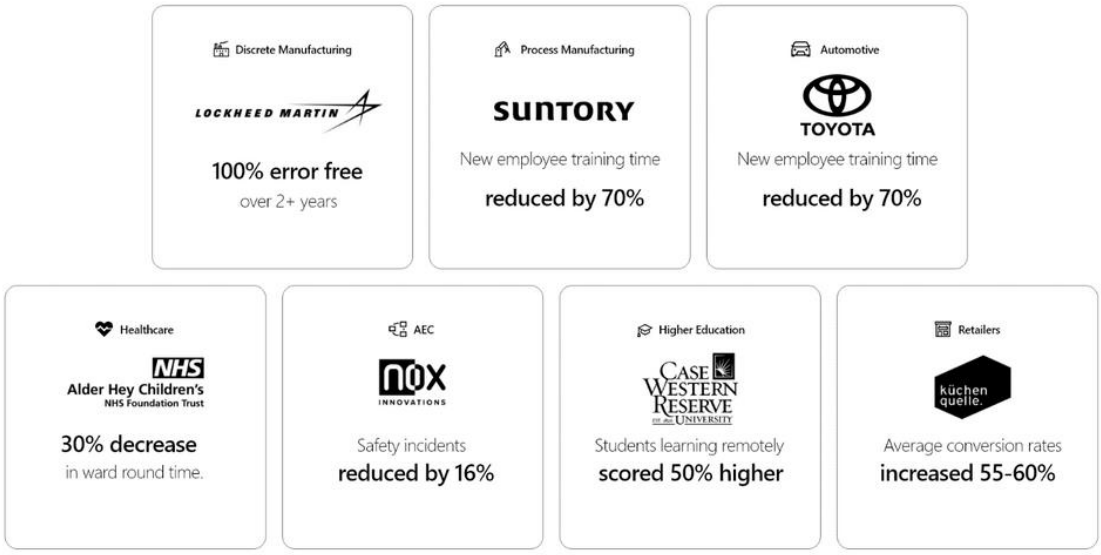
- „Select“ – Uključuje kursor pogleda putem kojega je moguće pogledati na drugi hologram te ponovo reći „select“ kako bi se selektirao.
- „Go to Start“ – Otvara startni izbornik.
- „Restart device“, „Shutdown device“ – Ponovo pokreće ili gasi uređaj
- „Volume up/down“ – Povećavanje odnosno smanjivanje glasnoće zvuka. Na isti način je moguće i mijenjati svjetlinu ekrana ako se kaže „Brightness up/down“.
- „Take a picture“ – Sprema sliku onoga što korisnik trenutno vidi u mješovitoj stvarnosti.
- „Take a video“ – Pokreće snimanje videa onoga što korisnik trenutno vidi u mješovitoj stvarnosti. Video je moguće zaustaviti i spremiti ako se kaže „Stop recording“.
- „Smaller“, „Bigger“ – Kontrola veličine holograma koje korisnik trenutno gleda.
- „Follow me“ – Hologram kojega korisnik trenutno gleda počinje pratiti korisnika. Ovo je moguće zaustaviti ako se kaže „Stop following“.
- „What can I say?“ – Komanda koja prikazuje što je sve moguće izreći i kontrolirati sa glasovnim komandama.

Neke od ovih komandi mogu se također koristiti i kako bi se kontrolirali ne samo hologrami nego i prozori. Primjerice, korisnik može narediti prozoru kojega gleda da ga prati ili da se smanji odnosno poveća. Također, glasovno je moguće i pisati tekst na sličan način kako je to moguće i sa mobilnim uređajima. Ukoliko se prilikom pisanja glasovne poruke želi dodati točka, zarez i slični znakovi, korisnik mora eksplicitno reći „comma“, „period“, itd. Za prirodniju interakciju sa sustavom se može koristiti asistent Cortana. Primjerice, za provjeru trenutne prognoze može se reći „*Hey, Cortana, what's the weather like today"* ili ako želimo provjeriti

razinu baterije „*Hey, Cortana, how much battery do I have left?*“. Cortanu je moguće isključiti kroz postavke uređaja ako je potrebno. [31]

4.3. Mogućnosti

Kao što se moglo vidjeti u prošlim poglavljima, moć Hololens uređaja je povećanje produktivnosti zaposlenika pomoću aplikacija mješovite stvarnosti. Hololens ima brojne solucije za različite probleme u sektorima prodaje, marketing, u planiranju, sastancima i slično. Time se zaposlenike uvodi u novi ekosustav koji nudi povrat ulaganja u kratkotrajno vrijeme. Microsoft navodi kako nezavisni dobavljači softvera koji grade svoje aplikacije mješovite stvarnosti na njihovoj Azure platformi postaju veći profitabilniji te navode brojne kompanije kao primjere i njihove vezane uspjehe korištenjem Hololens 2 ekosustava [32].



Slika 13. Primjeri postignuća poduzeća koja koriste Hololens [32]

Skup jednostavnih ali i kompleksnih mogućnosti ovog uređaja čine jedan takav ekosustav u kojemu su ovakva postignuća moguća. Jedna od tih mogućnosti koja je opisana u prijašnjem poglavlju je praćenje ruku i očiju prema čemu korisnici mogu bez ikakvih dodatnih kontrolera ili alata manipulirati digitalne objekte u prostoru te u isto vrijeme rukovati sa fizičkim prostorom. Takva mogućnost čini mješovitu stvarnost posebno pogodnu za rad u primjerice skladištima prilikom razvrstavanja dijelova ili u medicini prilikom operacije. No naravno, kako bi digitalni objekti ostali u prostoru točno na mjestu na kojemu su i postavljeni od strane korisnika, koristi se mogućnost „usidravanja“ objekata. Naime, objekti ostaju postavljeni na

točnoj lokaciji na koju ih korisnik postavi. Kako bi to Hololens omogućio, on koristi razvijene algoritme praćenja prostora u kojemu se korisnik nalazi bez korištenja markera. Takvo praćenje naziva se *inside-out* praćenje gdje kamere koje djeluju kao senzori, analiziraju prostor i stvaraju informacije o objektima u sobi ili prostoru koji mogu služiti kao markeri za usidravanje objekata. Također, spomenuto je u ranijem poglavlju kako je Hololens uređaj koji podržava glasovne komande što omogućava *hands-free* iskustvo korištenja što može biti vrlo korisno ako korisnik uređaja trenutno nema slobodne ruke. No ovdje se ne radi samo o softverskim solucijama. Važno je napomenuti da je Hololens dizajniran na način da bude što lakši i udobniji za nošenje kako bi zaposlenicima bilo lagodnije koristiti ovaj uređaj bez da se osjećaju da imaju težak i neudoban objekt na glavi. Prilikom ranih godina nastanka sličnih uređaja, većina ih je bila vezana za eksterna računala sa kablovima što nije omogućavalo veliku slobodu kretanja. Takvo nešto u radnim prostorima, a pogotovo na velik broj uređaja, bi samo stvaralo problem. Hololens u tom smislu koristi internu bateriju koja traje nekoliko sati, što ga čini kompletno mobilnim. Mogu se navesti još neke bitnije mogućnosti uređaja [32]:

- Pogonjen Azure platformom – Hololens ima mogućnost streamanja 3D objekata sa Azure platforme koji su vidljivi svim korisnicima. Takva mogućnost je ono što Hololens čini veoma konkurentnim naspram drugih sličnih uređaja.
- Snimanje mješovite stvarnosti u obliku videozapisa ili slike, te spremanje iste ili dijeljenje na društvenim mrežama, ili između kolega.
- Windows Hello autentifikacija – Biometrijska autentifikacija bazirana na skeniranju oka.
- Windows Autopilot – Još jedna korisna opcija kojom je moguće prekonfigurirati uređaj sa servisima kako bi se mogao koristiti na raznim lokacijama bez ikakvog dodatnog podešavanja.

Osim navedenoga, Microsoft izbacuje redovita ažuriranja za sustav jednako kao i za Windows platforme. Moguće je i upravljati više Hololens 2 uređaja odjednom korištenjem servisa kao što su Microsoft Intune, VMware i drugi.

4.3.1. Orijentacija i mapiranje prostora

Glavna funkcionalnost koja čini Hololens uređajem mješovite stvarnosti je mogućnost mapiranja prostora. Hololens koristi kartezijev koordinatni sustav gdje se u prostoru objekti orijentiraju putem X, Y i Z osi. Generalno kartezijev koordinatni sustav može biti desnog ili lijevog tipa. U slučaju mješovite stvarnosti, Windows koordinatni sustav naziva prostorni koordinatni sustav desnog tipa gdje je X os uvijek uperena u desno, Y os gore i Z os prema korisniku. Uz pomoć ovakvog koordinatnog sustava te ugrađenih kamera koje služe kao

senzori, Hololens smješta i orijentira digitalne predmete u stvarnom prostoru korisnika. No, nije sve tako jednostavno. Jedan takav globalni koordinatni sustav je savršen za korištenje u primarno virtualnim aplikacijama gdje se već poznaje geometrija i svaki objekt te njegova lokacija, no ako se radi o aplikacijama mješovite stvarnosti, takav slučaj je malo kompliciraniji. Naime, samostalan uređaj kao što je Hololens mora moći koristiti senzore kako bi dinamično mapirao prostor korisnika, jer kako je poznato, ni jedan prostor u kojemu će se ovakav uređaj koristiti nije isti te se ne može unaprijed znati njegova geometrija. Dakle, Hololens dinamično mapira prostor korisnika dok se on kreće i stvara geometrijsku sliku pomoću koje smješta objekte u prostor. Međutim, ovdje dolazimo do drugog problema gdje se primjerice inicijalno smještaju digitalni predmeti na udaljenosti od 2 metra dok u međuvremenu Hololens uređaj nauči kako se u stvarnosti zapravo nalaze 1.8 metara jedno od drugoga pa će prema tome ta dva digitalna predmeta uvijek biti za 0.2 metra krivo reprezentirana. Microsoft je za Hololens ovakav problem riješio, i poboljšao praćenje, dodavanjem prostornih sidra. [33]

Prostorna sidra (eng. *Spatial anchors*) služe kao prostorni markeri koji se postavljaju u prostor i označavaju bitne točke koje Hololens treba konstantno pratiti. Ti markeri se zatim koriste kako bi se pravilno pozicionirali objekti u prostoru na način da se uz dodatne informacije prostora, pozicije markera pomiču kako bi se korigirale. Tako postavljeni markeri mogu se koristiti za pozicioniranje objekata u velikim prostorima te dijeliti između uređaja putem Azure platforme kako bi bili dostupni i drugim osobama. Za razliku od ovakvog koordinatnog sustava postoje i stacionarni referentni okviri koju koriste Hololens aplikacije kako bi pozicionirale objekte u fizički manjem prostoru ovisno o kretanju korisnika. No kako se ovdje drže veze između postavljenih objekata, može doći do prevelikom pomicanja objekata kako bi oni na izgled izgledali stabilni. Također, još imamo i referentni okvir pozornice koji isto drži vezu između postavljenih objekata međutim, suprotno stacionarnom okviru, ovdje se objekti drže stabilnima samo ako se korisnik nalazi unutar prostora kojega je označio prilikom postavljanja uređaja. Tako možemo reći da za razliku od stacionarnog pozicioniranja, prostorno i putem sidra optimizira poziciju objekata na bazi njihove originalne pozicije. No ni ovaj sustav nije savršen i ukoliko se objekti renderiraju pre daleko od korisnika, svaki mali pomak pozicije objekata ovisi o udaljenosti, što će značiti da objekti dalje od korisnika dobivaju veći pomak. Prema tome Microsoft preporučuje da se većina objekata drži na udaljenosti do 3 metra od korisnika kako bi njihova pozicija bila što točnija. Ono što također valja uzeti u obzir kod korištenja uređaja je da:

- Prostor u kojemu se koristi uređaj ima dovoljno svjetlosti.
- Su senzori čisti i bez prljavštine.

- Prostor oko korisnika ima dovoljno teksture kako bi se kreiralo dovoljno prostornih referenci.

Ukoliko nešto od navedenoga nije osigurano, Hololens će, kao i svaki sličan uređaj, imati problema sa praćenjem prostora korisnika. [33]

Prilikom dizajniranja aplikacija za Hololens, treba uzeti u obzir uobičajenu oporabu prostornog mapiranja, pa je tako prva i najjednostavnija postavljanje objekata u prostor. Objekti koji se postavljaju u prostor moraju biti postavljeni uz skeniran prostor odnosno granice jer ako se takve granice ne poštuju, onda će postavljeni objekti prolaziti kroz fizičke elemente prostora korisnika te će iluzija mješovite stvarnosti biti narušena. Tako kada korisnik prstom pokaže u nekom smjeru, aplikacije bi trebale koristiti podatke o skeniranom prostoru kako bi pravilno postavile objekt i po mogućnosti, smanjili brzinu približavanja odnosno udaljavanja objekta ukoliko korisnik pokazuje na daleko. Na taj način korisnik može postavljati primjerice virtualne objekte kao što je mobitel na fizičke, kao što je to radni stol. Valja uzeti u obzir i korištenje automatskog postavljanja objekata u prostor no uz manualnu korekciju korisnika ukoliko se primjerice sklopka za svjetla koja se postavi na zid automatski postavi pre daleko od korisnika da ju aktivira. Vizualne povratne informacije prilikom postavljanja objekata su također bitne kako bi se korisniku signaliziralo gdje i kako se određeni objekt postavlja u prostor što Hololens čini vrlo dobro. [33]

Druga uporaba je okluzija. Ovdje se radi o mogućnosti sakrivanja renderiranih objekata ako se oni nalaze iza fizičkih elemenata u prostoru korisnika ili iza drugih renderiranih objekata. Na taj način se realnost podiže na višu razinu jer je logično da korisnik očekuje da se digitalni objekt postavljen na stol ne vidi ukoliko su i stol i objekt iza zida u kojega gleda korisnik. Međutim, ponekad je potrebno vidjeti elemente čak i ako se oni nalaze iza drugih elemenata. Takav slučaj se obično pronalazi u kontroliranju izbornika, jer ako se izbornici ne vide, onda se ne mogu ni kontrolirati. Tu Hololens nudi mogućnost renderiranja takvih elemenata sa primjerice određenim postotkom transparentnosti kako bi i dalje bio vidljiv korisniku. [33]

Još jedna uporaba je fizika. Naravno da se očekuje da digitalno postavljena loptica koja se kotura po stolu padne sa stola i možda odskoči par puta ili da se čaša puna tekućine prolije ako se prevrne. Uporaba fizike kod objekata stvara dodatnu percepciju realnosti spoja digitalnih i fizičkih objekata. Slično tome, Hololens nudi mogućnost da postavljeni objekti navigiraju prostorom na isti način kako bi to činio i korisnik. Takva mogućnost naravno ovisi o implementaciji od strane aplikacije no moguće je korisniku prikazati navigaciju direktno na površini asfalta ili navigirati virtualno postavljenu digitalnu životinju u prostoru korisnika. Postavljeni objekti u prostoru također uključuju sjene kako bi izgledali što stvarnije. [33]

5. Implementacija mješovite stvarnosti

Ovo poglavlje obuhvaća praktični dio rada i prikaz razvoja aplikacije mješovite stvarnosti za Hololens 2 uređaj. Prilikom postavljanja okoline i implementacije aplikacije, korištena je MRTK dokumentacija dostupna na Microsoft stranicama, implementirani primjeri dostupni u MRTK Examples dodatku kao i upute te dokumentacija za korištenje Unity softvera.

5.1. Opis aplikacije

Cilj ovog rada je prikazati mogućnosti Microsoft Hololens uređaja i načine implementacije istih u mješovitoj stvarnosti. Za potrebe toga, napraviti će se tri aplikacije koje će biti dio jedne sveukupne aplikacije, HoloPlayground. Aplikacija će koristiti MRTK višescenski sustav koji je ranije spomenut kako bi korisnik mogao navigirati između glavnog izbornika i preostalih aplikacija. Osim vlastito izrađenih modela i izbornika, aplikacija će koristiti UI elemente i skripte iz MR toolkita. Na glavnom izborniku aplikacije nalaziti će se tabela sa natpisom te tri gumba koja će voditi do ostalih aplikacija. Kako bi prebacivanje između aplikacija i prikazivanje menija bilo što intuitivnije, biti će ih moguće micati rukom ili iz daljine te će se startna pozicija za svaku od aplikacija prilagođavati poziciji korisnika u prostoru. S obzirom da MR toolkit ne podržava više od jednog izvora usmjerene svjetlosti, u svakoj sceni nalaziti će se jedan izvor svjetlosti za sve objekte te scene. Svi elementi u aplikaciji koristiti će MRTK standardni shader kako bi se optimizirale performanse aplikacije za Hololens uređaj.



Slika 14: Logo aplikacije [Izvor: vlastita izrada]

Prva aplikacija nazvana je „Pocket Basketball“, klasična igra zabijanja koša i dobivanja poena u mješovitoj stvarnosti. Prilikom pokretanja igre, korisnik će postaviti koš na željenu

lokaciju putem Air Tap gesture Hololens uređaja nakon čega će mu biti dostupna igra. Sama igra će sadržavati plutajući meni koji će slijediti korisnika ili će stajati statično na mjestu gdje ga je korisnik postavio. Cilj igre je zabiti što više koševa u određeno vrijeme. Za svaki zabijeni koš gleda se fizička udaljenost igrača od koša kako bi se dodijelili adekvatni poeni, odnosno jedan poen za blizinski zabijene koševe te dva ili tri poena za udaljeno zabijene koševe. Vrijeme igre biti će moguće podesiti kroz izbornik aplikacije kao i pregledavati trenutačno najveći highscore te tabelu sa rezultatima koja će sadržavati neke od detalja kao što su broj poena, vrijeme igre, broj blizinskih ili udaljenih šuteva i slično. Na meniju će se također nalaziti dodatne kontrole za pomicanje koša, pronalaženje lopte u slučaju da ju korisnik izgubi, resetiranje lopte ispred korisnika te pokretanje i zaustavljanje igre. Svaki od elemenata na izborniku će biti moguće aktivirati na daljinu ili dodirom prsta. Kako je bitno da se ovakva igra ukomponira u mješovitu stvarnost, lopta će imati fiziku koja reagira na skenirani prostor oko korisnika dok će se postavljanje koša „lijepiti“ na površine zidova. Također, ukomponirati će se vizualni i zvukovni elementi za zabijanje koša, odbrojavanje vremena te zvuk pri odbijanju lopte od površine. Ukoliko korisniku dosadi aplikacija, na izborniku će se nalaziti gumb za povrat na glavni izbornik.

Druga aplikacija nazvana je „AirPaint“. Radi se o aplikaciji crtanja uz nekoliko zanimljivih opcija prilagođenih za mješovitu stvarnost. Prilikom ulaska u aplikaciju, pred korisnikom će se nalaziti platno za crtanje uz statični vertikalni izbornik. Platno će biti moguće premještati ručno ili putem Air Tap gesture, lijepiti na prostor korisnika. Platno će se također moći rotirati u prostoru kao i skalirati po želji. Na izborniku će se prikazivati dodatni izbornici za promjenu debljine kista i promjenu željene boje uz slidere koji će predstavljati miks crvene, zelene i plave boje. Kako bi stvar bila još zanimljivija, dodati će se i opcija crtanja sa prijelaznim bojama koje će se nasumično generirati. Korisnik će moći također promijeniti mod crtanja između 2D na platnu i 3D u prostoru putem izbornika. Svo crtanje moći će se resetirati. Kao i prijašnja aplikacija, svi gumbi i elementi izbornika biti će osjetljivi na dodir te će biti moguće vratiti se na glavni izbornik.

Finalna aplikacija nazvana je “Weather 3D”. Radi se o implementaciji izbornika i pregleda prognoze putem OpenWeatherMap API-ja u mješovitoj stvarnosti. Izbornik će sadržavati nekoliko elemenata koji će se istovremeno prikazivati korisniku. Glavni element koji će biti ispred korisnika je pregled trenutne prognoze za odabrani grad zajedno sa odgovarajućom ikonom vremena. Iznad toga će se prikazivati dodatni izbornik za prebacivanje između trenutnog vremena i prognoze za pet dana, svakih 3 sata. Takva prognoza biti će prikazana u skrolabilnom pogledu gdje je korisnik prstom ili Air Tap gesturom moći skrolati prikaz prognoze. Lijevo od glavnog elementa nalaziti će se izbornik za pretraživanje gdje će korisnik moći, klikom na polje, pozvati sistemsku tipkovnicu kako bi upisao naziv grada kojega

želi pretražiti. Pretraživanje će se obavljati na JSON skupu podataka koji će sadržavati nazive i identifikatore gradova preuzetih sa OpenWeatherMap API-ja. Kako sistemska tipkovnica nije dostupna u Unity editoru, neposredno ispod polja za pretraživanje će se nalaziti nekoliko sugestija gradova koje će se moći izabrati kako bi se promijenio prikaz prognoze. Desno od glavnog izbornika će se nalaziti još dva elementa, od kojih je jedan vertikalni izbornik sa opcijama osvježavanja prikaza vremena, pomicanja izbornika, uključivanje Air Tap gesture kako bi korisnik mogao zalijepiti izbornik na zidove skenirane u prostoru te povratak na glavni izbornik aplikacije. Drugi element će biti prikaz vremena na manjem 3D modelu kojega će korisnik moći micati i rotirati. Vrijeme će se na modelu mijenjati ovisno o trenutnoj prognozi za izabrani grad te će uz animacije imati i odgovarajuće zvukove.

5.2. Korišteni alati

U ovom poglavlju opisati će se alati koji su korišteni kod izrade eksperimentalne aplikacije za proširenu, odnosno mješovitu stvarnost, te specifični alati koji se koriste za izradu aplikacija za Hololens uređaj. Kako gotovo svi spomenuti alati zahtijevaju velike sistemske resurse, izrada aplikacije sa istima odrađena je na adekvatnom stolnom računalu.

5.2.1. Unity

Unity je game engine kreiran od strane Unity Technologies koji omogućava izradu 2D i 3D igara ili aplikacija za brojna okruženja. Prvi puta je predstavljen 2005. godine na Apple-ovoj Worldwide Developers konferenciji za Mac OS x. Od tada se proširio na druge platforme kao što su iOS, Android, desktop, konzole, VR i AR uređaji i sl. Unity je prije koristio sistem verzioniranja od 1.0 pa do 5.0 dok se nisu odlučili za moderniji format gdje se verzije označavaju po godini i iteraciji. Aplikacije se pišu u programskom jeziku C#, dok Unity sam po sebi koristi C++. Kroz godine, Unity je u svoj game engine doveo brojne značajke koje su posebno korisne za nove developere sa šablonama za kompliciranije tehnike kao što su multiplayer biblioteke, volumetrični oblaci, vrhunsku grafiku sa High Definition Render Pipeline-om i slično [34]. Ono što ovaj game engine čini vrlo pogodnim za izradu aplikacije za Hololens je što je izuzetno popularan izbor za izradu AR i VR aplikacija te je dostupno mnoštvo dokumentacije na istu temu. Također, prema Microsoftovoj stranici, Unity je jedini trenutni game engine koji podržava sve značajke, primjerice Azure Remote Rendering ili World locking tools, koje drugi, kao što su Unreal ne podržavaju [35]. Također razlog izbora ovog game engine-a je što je intuitivan i jednostavan za početnike. Moguće je napraviti aplikaciju u nekoliko koraka uz vrlo malo pisanja skripti. Prema tome, Unity kao takav je moguće koristiti i od strane dizajnera ili 3D animatora za izradu filmova, animacija, 3D renderiranih slika. Valja

napomenuti još jednu korisnu značajku, a to je trgovina assetima koja omogućava pretraživanje i kupovanje skripti, modela i dodataka za Unity koji su kreirani od strane drugih ljudi. Naravno, postoje i besplatni dodaci od kojih su neki korišteni i kod izrade aplikacije u ovom radu kako bi se ubrzao razvoj ali i fokus prebacio na bitniji dio aplikacije. Unity je moguće koristiti besplatno no isto tako se nudi i više planova od kojih su neki za timove, firme ili pak studente [36]. U izradi aplikacije ovog rada korištena je besplatna osobna licenca te verzija Unity programa 2021.3.6f1.

5.2.2. Visual Studio

Visual Studio je Microsoftov IDE za programiranje primarno u .NET i C++ jezicima no može se koristiti za brojne druge jezike. Izabran je za korištenje pri izradi aplikacije za ovaj rad iz razloga što se koristi uz Unity prilikom izrade skripti ali i kasnije prilikom puštanja i pokretanja aplikacija na uređaju ili emulatoru. Visual Studio ima različite licence no također ima i Community free verziju koja je ovdje korištena. Prilikom izrade aplikacije, korištena je verzija Visual Studio 2022 Community.

5.2.3. Mixed Reality Feature Tool

Mixed reality feature tool je alat od Microsofta koji olakšava dodavanje MRTK i ostalih dodataka vezanih za izradu aplikacija za proširenu stvarnost u Unity. Prilikom pokretanja moguće je izabrati specifičan Unity projekt te dodatke koji se žele dodati u isti kao i njihove verzije i dependencije koje zahtijevaju. Na zadnjem koraku korištenja, alat automatski dodaje definirane dodatke u manifest datoteku koja sadržava sve dodatke koji se koriste u projektu, te kopira odabrane datoteke dodataka u projekt [37]. Prilikom postavljanja okoline za izradu aplikacije ovog rada korištena je verzija alata 1.0.2206.1-Preview.

5.2.4. MRTK (Mixed Reality Toolkit)

MRTK odnosno Mixed Reality Toolkit je proširenje za Unity odnosno projekt od Microsofta koji sadrži set komponenti i dodataka kojima se ubrzava razvoj aplikacija proširene stvarnosti. MRTK u Unity dodaje nekoliko korisnih stvari. Prva stvar koju dodaje je simulacija okruženja mješovite stvarnosti prilikom testiranja aplikacija čime se omogućava brže prototipiranje ali i testiranje aplikacija u samom editoru bez posjedovanja uređaja mješovite stvarnosti kao što je to Hololens. Međutim glavna značajka MR toolkita je što sadrži brojne skripte, gotove modele i primjere koji se mogu koristiti u izradi vlastitih projekata za mješovitu ili proširenu stvarnost. Dodatke koji se žele dodati te sam MRTK se dodaje kroz prijašnje opisani Microsoftov alat Mixed Reality Feature Tool. Kako bi cijeli proces bio još jednostavniji, tijekom prvog paljenja Unity projekta u koji je dodan MR toolkit, otvara se prozor koji vodi

korisnika kroz kratko postavljanje projekta za izradu aplikacija mješovite stvarnosti [38]. Toolkit nudi i druge brojne mogućnosti od kojih je velika većina korištena u aplikaciji ovog rada, kao što su [38]:

- Simulacija input sustava putem kontrolera, pogleda i ruku.
- Gotovi profili za Hololens 1 i 2 te OpenXR ili druge platforme mješovite stvarnosti.
- Gotovi UI elementi kao što su responzivni gumbi, meniji, skrolabilni meniji, slideri, tooltipovi i sl.
- Skripte za razne akcije koje se mogu obavljati u okruženju mješovite stvarnosti kao što su uzimanje, rotacija i skaliranje objekata, dodirivanje objekata, praćenje korisnika s menijima, pozivanje sistemske tastature i sl.
- Više scenski sustav za izradu aplikacija s više scena.
- Mogućnost za uključivanje i korištenje glasovnih komandi i gestura.
- Fizika responzivna na ruke ili objekte u sceni
- Sustav skeniranja prostora te korištenje istog u razne svrhe kao primjerice lijepljenje menija za zidove ili koliziju sa hologramima.

Kako bi se podalje olakšalo korištenje dodataka i značajka MRTK, uključene su i brojne scene koje služe kao primjeri implementacije navedenih značajki. Svaki od tih dodataka moguće je odvojeno uključiti ili isključiti iz projekta te svaka skripta ili dodatak sadrži detaljnu dokumentaciju o funkcionalnosti i mogućnostima skripte odnosno dodatka. Valja naglasiti da korištenje MR toolkita nije ekskluzivno za Hololens i WMR (eng. *Windows Mixed Reality*) uređaje već se može koristiti i za ostale platforme kao što su Oculus, OpenXR, OpenVR, Ultraleap te mobilne uređaje bazirane na iOS ili Android operacijskim sustavima [38].

Prilikom izrade aplikacija za mobilne uređaje kao što je to Hololens, koristi se MRTK2 standardni shader koji zamjenjuje Unity-jev shader koji je općenito pre težak za korištenje na takvim uređajima. MRTK2 shader je poseban po tome što koristi samo uključene značajke i time se ubrzava njegov rad. Kao i Unity-jev shader, MRTK2 shader podržava vrste prikaza objekata kao što su neproziran, proziran, prijelazni (eng. *Fade*), aditivni i sl. Kako shader radi jednostavne aproksimacije za svjetla, on ne računa njihovu fizičku točnost te je brži za mobilne uređaje. To dolazi s ograničenjima korištenja dinamičkih svjetala gdje je moguće koristiti samo jedno usmjereno svjetlo te nije moguće koristiti druga dinamična svjetla kao što su reflektori. No kako bi bilo moguće koristiti više statičnih svjetala, MRTK je uključio unaprijed definirane skripte za lebdeća i blizinska svjetla. Svaki od mogućnosti shadera se može podešavati na

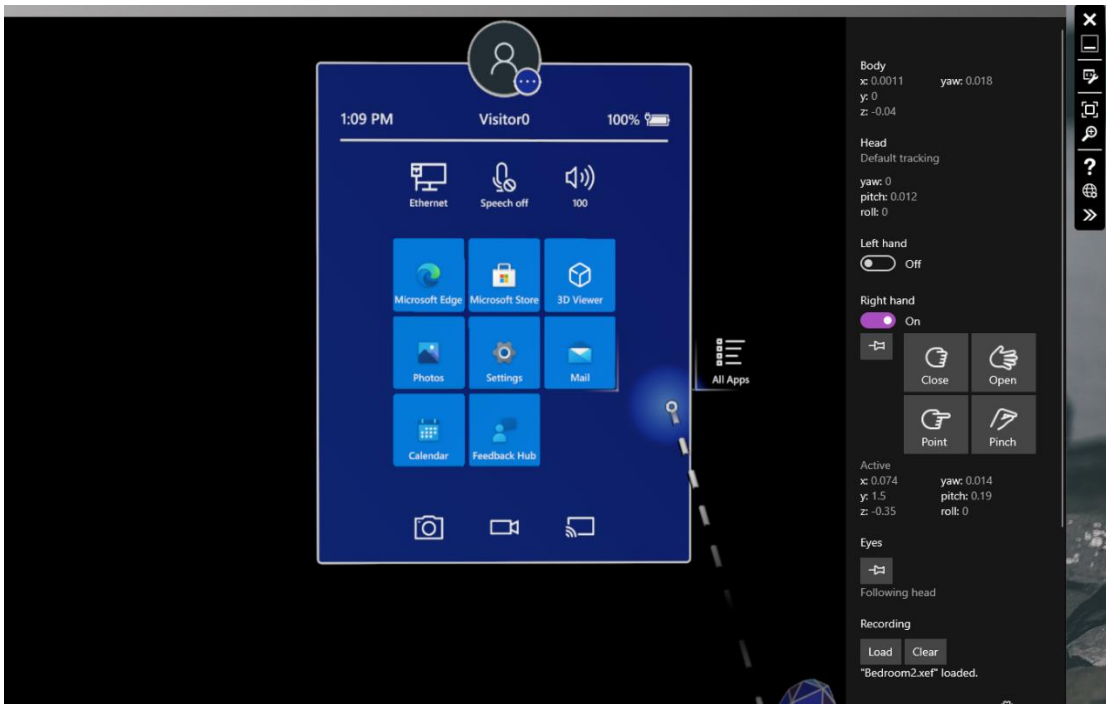
materijalu na koji je shader dodan u Unity properties prozoru jednako kao i za ostale shadere. Usprkos ograničenjima, shader nudi i dodatne mogućnosti od kojih su neke [38]:

- Primitivni izrez (eng. *Primitive clipping*) – omogućava izrezivanje objekata uz pomoć drugih objekata na vanjskoj ili unutrašnjoj strani.
- Mrežasti obrisi (eng. Mesh outlines) – dodaje obrise oko objekata uz pomoć skripti koje ne koriste nikakve naknadne obrade kako bi performanse bile adekvatne za mobilne uređaje.
- Instancirana podrška za boje (eng. *Instanced color support*) – Dodaje unikatne značajke materijalima za svaki od tisuće instanciranih GPU objekata čime se stvaraju varijacije objekata.
- Drugi efekti kao što su jednostrani portali i triplanarno mapiranje tekstura

U vrijeme izrade aplikacije mješovite stvarnosti za ovaj rad, korištena je verzija MRTK 2.8.2.0. Verzija 3.0 je trenutno u izradi i testiranju koja je dostupna javnosti no kako je još pre rano za njezino ekstenzivno korištenje, nije korištena u izradi aplikacije ovog rada. Sva dokumentacija vezana za MRTK te razni primjeri korištenja dostupni su na Microsoftovoj dokumentaciji, kao i javni forumi gdje korisnici mogu dijeliti znanja i postavljati pitanja.

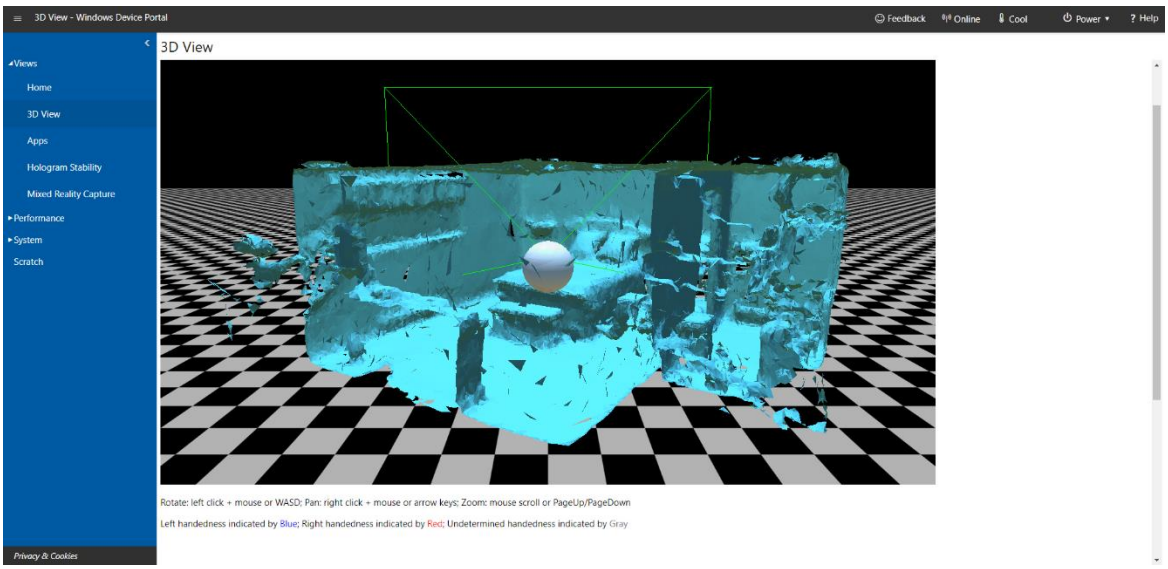
5.2.5. Hololens 2 Emulator

Hololens emulator napravljen je u svrhu testiranja aplikacija za Hololens bez posjedovanja uređaja. Emulator zahtjeva da CPU računala na kojemu se pokreće podržava Hyper-V virtualizaciju. Također, emulator nije lagan program i zahtjeva puno hardverskih resursa za pokretanje. Kontrole su simulirane putem tipkovnice, miša ili kontrolera. Moguće je kretati se u prostoru, povlačenjem miša gledati u nekom smjeru, klikom miša izvesti Air Tap gesturu i selektirati sadržaj, kontrolirati obje ruke odjednom, skrolati sadržajem putem miša, kontrolirati pogledom ali i koristiti drugi Windows Mixed Reality uređaj kako bi se simulirale kontrole pravog Hololens uređaja. Kako aplikacija za sve te kontrole koristi Hyper-V virtualizaciju, aplikacija učitana na emulator neće raspoznati da se ne radi o pravom Hololens uređaju. Glavni prozor emulatora uz kontrole zumiranja i uputa, sadrži dvije vrste panela. Prvi panel je panel simulacije koji sadrži informacije o trenutnom trackingu, ruci koja se koristi, kontrolama ruku i pogleda. Emulator, kao i pravi uređaj, simulira skeniranje prostora na način da učitava predefinirane podatke o skeniranim prostorima sa pravog uređaja kojih ima nekoliko na izbor. Sa glavnom menija moguće je doći i do dodatnih postavki koje omogućuju uključivanje akcelerirane grafike, NAT konfiguracija te informacije o mrežnim postavkama uređaja [39].



Slika 15: Hololens 2 emulator sa otvorenim panelom simulacije [Izvor: vlastita izrada]

Drugi panel koji se može otvoriti je portal uređaja (eng. Windows Device Portal). Taj panel sadrži detaljne informacije o trenutno pokrenutoj sesiji uređaja odnosno emulatora. Moguće je pregledati instalirane aplikacije, performanse uređaja, trenutno pokrenute procese, IP adresu uređaja, logove, podatke na uređaju, snimati ekran uređaja i slično [39]. Također je moguće u stvarnom vremenu gledati 3D prikaz prostora u kojemu se nalazi uređaj zajedno sa podacima o skeniranom prostoru kao što je to prikazano na slici niže:



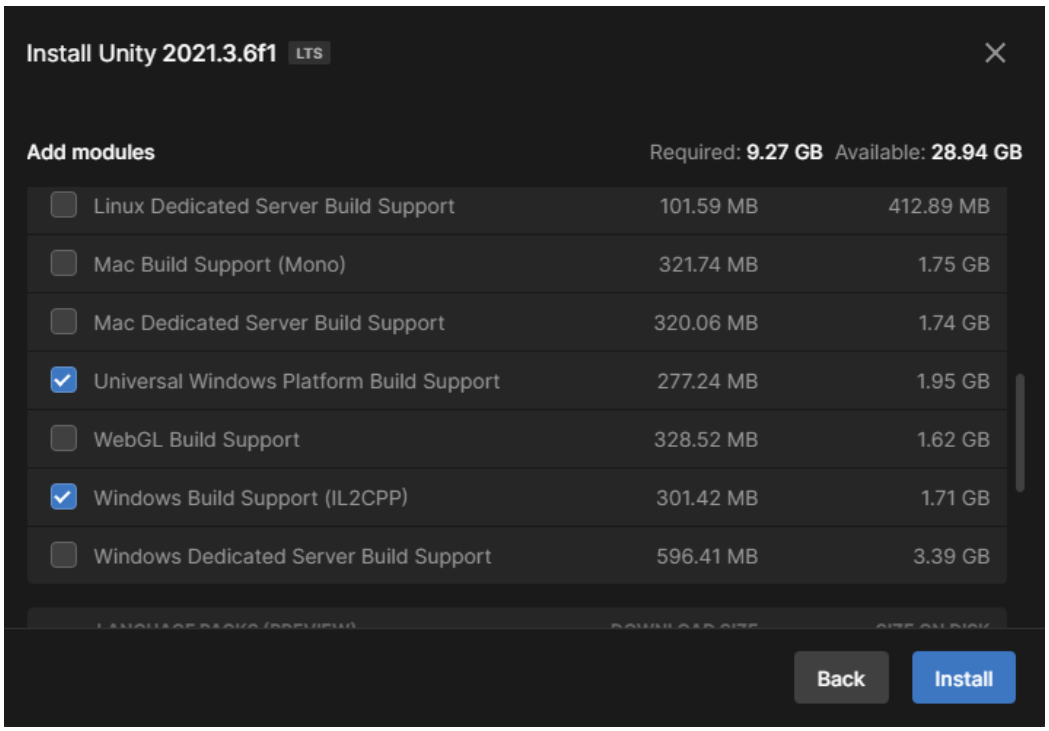
Slika 16: Panel sa 3D prikazom skenirane sobe u stvarnom vremenu [Izvor: vlastita izrada]

Prilikom testiranja aplikacija, iste je moguće pokrenuti na emulatoru kroz Visual Studio ili direktno kroz kontrolni panel uređaja učitavanjem zapakirane aplikacije. Kako je korištenje emulatora vrlo neintuitivno, primarno će se aplikacija ovog rada testirati u Unity okruženju uz simulaciju skeniranja prostora uz povremeno testiranje i na emulatoru. Verzija emulatora koja je korištena za rad je 10.0.20348.1513 (Windows Holographic 22H1).

5.3. Postavljanje razvojne okoline

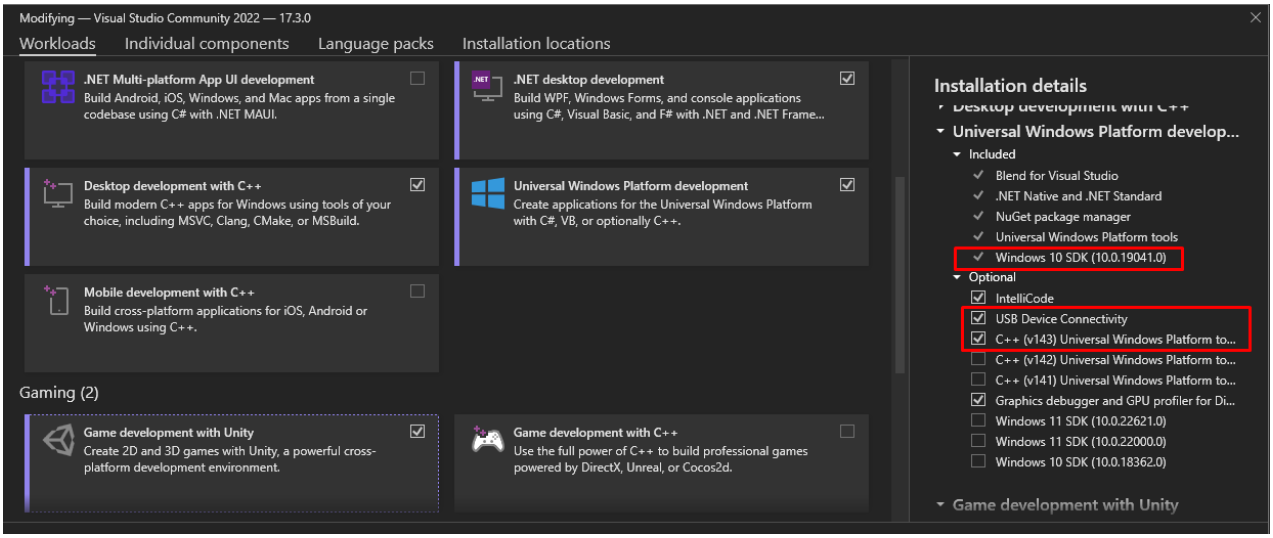
Prije početka razvoja aplikacije, potrebno je prijašnje navedene alate instalirati i podesiti kako bi se mogli koristiti za razvoj aplikacija mješovite stvarnosti. Prije svega, na računalu na kojemu se obavlja razvoj potrebno je imati operacijski sustav Windows 10 ili 11, a u ovom slučaju koristiti će se zadnja verzija Windows 11.

Zatim je potrebno preuzeti Unity Hub aplikaciju sa službene stranice i instalirati na računalo. Nakon toga otvaramo Unity Hub, odabiremo tab „Installs“ i opciju „Add“. Preporučena verzija za izradu Hololens aplikacija je 2020.3.x LTS, no ovdje će se koristiti novija verzija 2021.3.6f1. Nakon odabiranja opcije „next“, pokazuje se prozor sa modulima gdje je potrebno pod „platforms“ odabrati opcije „Universal Windows Platform Build Support“ i „Windows Build Support (IL2CPP)“ te nakon toga instalirati.



Slika 17: Instaliranje alata Unity [Izvor: vlastita izrada]

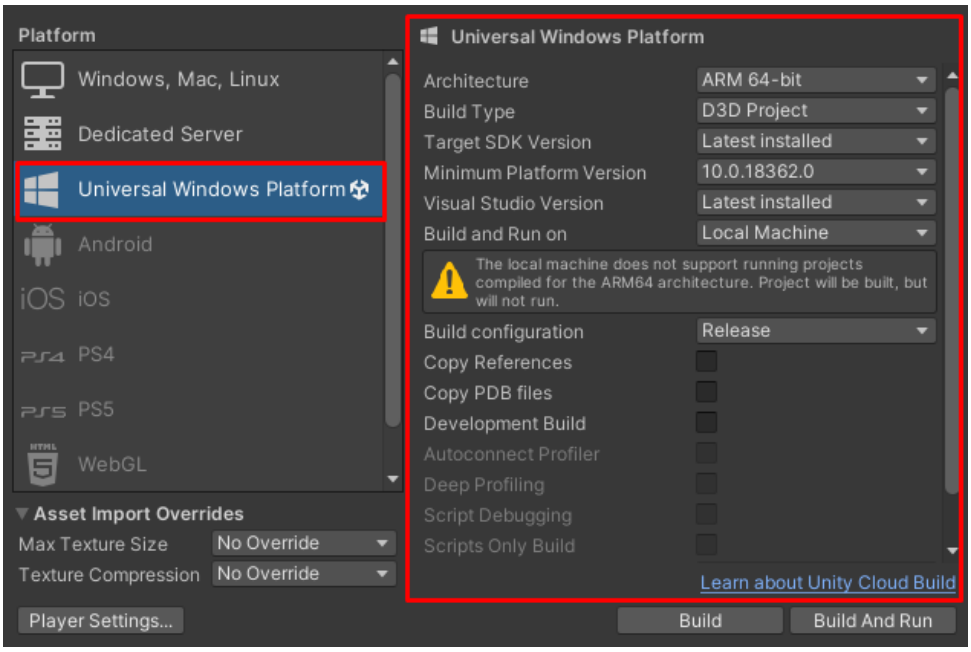
S instaliranjem Unity aplikacije može se odabrati Visual Studio 2019, no ovdje je zasebno instaliran Visual Studio Installer preko kojega je instaliran Visual Studio Community 2022. Nakon instalacije, preko installera je potrebno odabrati „modify“ opciju i otvara se prozor sa dodacima gdje je potrebno odabrati „.NET desktop development“, „Desktop development with C++“, „Universal Windows Platform development“ i „Game development with Unity“. U desnom prozoru za dodatak Universal Windows Platform development je potrebno dodatno odabrati opcije „USB Device Connectivity“, što se koristi kod testiranja aplikacije na Hololens uređaju, i „C++ (v143) UWP tools“. Nakon toga kliknuti „Modify“ i pričekati da se svi dodaci instaliraju.



Slika 18: Modificiranje dodataka za Visual Studio [Izvor: vlastita izrada]

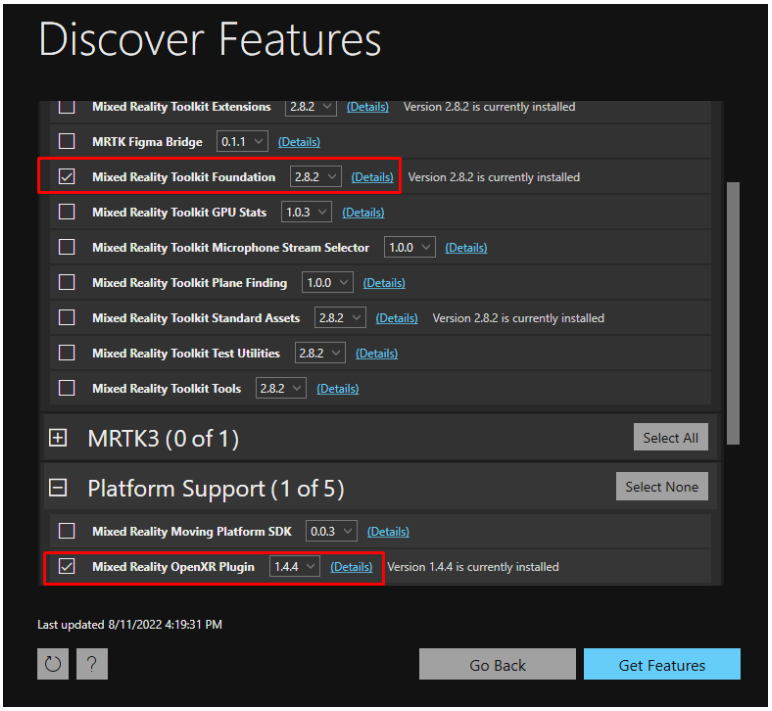
Ukoliko se planira testirati aplikaciju bez Hololens uređaja, potrebno je instalirati i Hololens Emulator čija se posljednja verzija može preuzeti iz dokumentacija od Microsoft-a. Posljednji alat koji je potrebno instalirati, a koji će se koristiti kako bi se Unity projekt pripremio za MRTK razvoj, je Mixed Reality Feature Tool koji je također dostupan u dokumentaciji Microsoft-a. Za sada nije potrebno pokretati MR Feature Tool, već se prelazi na postavljanje Unity projekta.

Pokretanjem Unity Hub aplikacije, odabire se opcija “New project”. Pod template je potrebno izabrati “3D Core”, upisati ime projekta i lokaciju spremanja projekta te kreirati projekt. Nakon što Unity stvori novi projekt, otvara se prozor editora. Sada je potrebno podesiti nekoliko opcija u editoru. Potrebno je otvoriti postavke buildanja projekta pod File -> Build settings, odabrati „Universal Windows Platform“ pod platforme i podesiti postavke kako je prikazano na slici ispod. Nakon promjena postavki, na dnu će se prikazati gumb „Switch platform“ kojega je potrebno kliknuti i pričekati da Unity promjeni platformu.



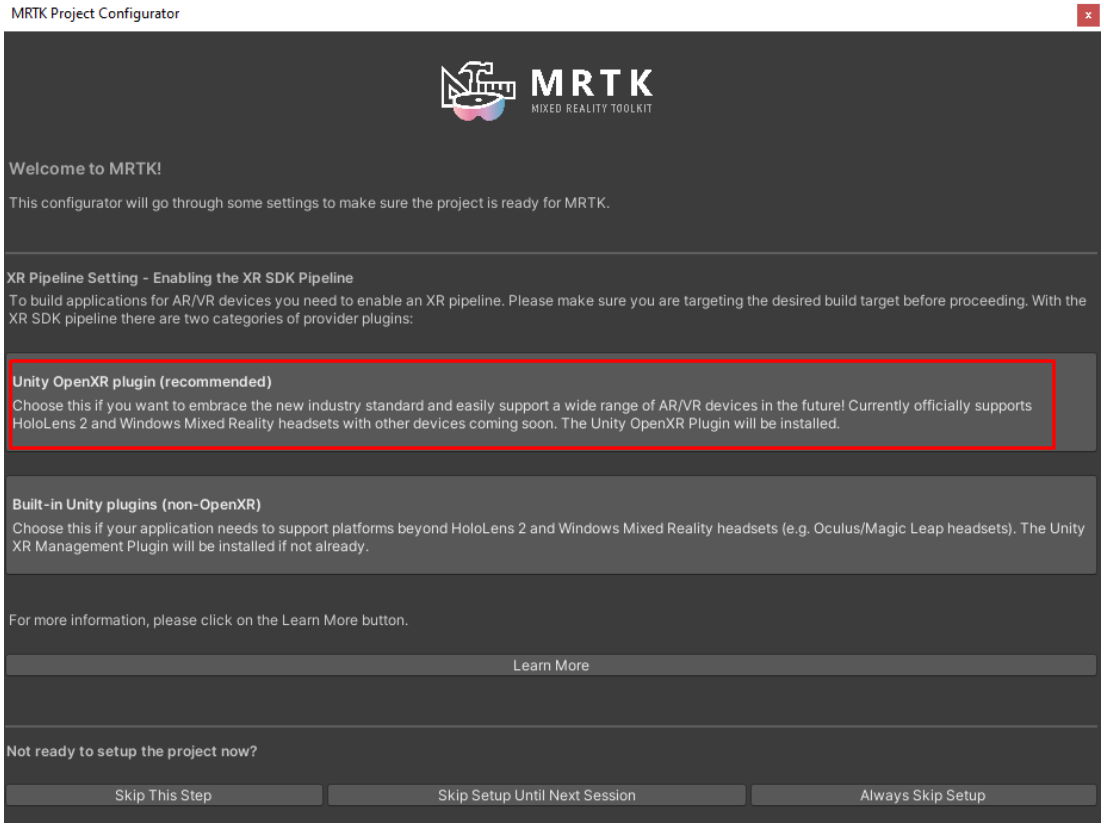
Slika 19: Build postavke Unity projekta [Izvor: vlastita izrada]

Unity se sada ostavlja otvorenim i prelazi se na Mixed Reality Feature Tool kojega je potrebno pokrenuti. Pokretanjem će alat pokrenuti osvježavanje kataloga dodataka i nakon što završi je potrebno odabrati „Start“. Odabire se Unity projekt koji se u prijašnjem koraku kreirao i opcija „Discover features“. U ovom prozoru je kao minimalni zahtjev rada sa MRTK potrebno pod Mixed Reality Toolkit sekcijom odabrati „Mixed Reality Toolkit Foundation“ i pod sekcijom Platform support se odabire „Mixed Reality OpenXR plugin“.



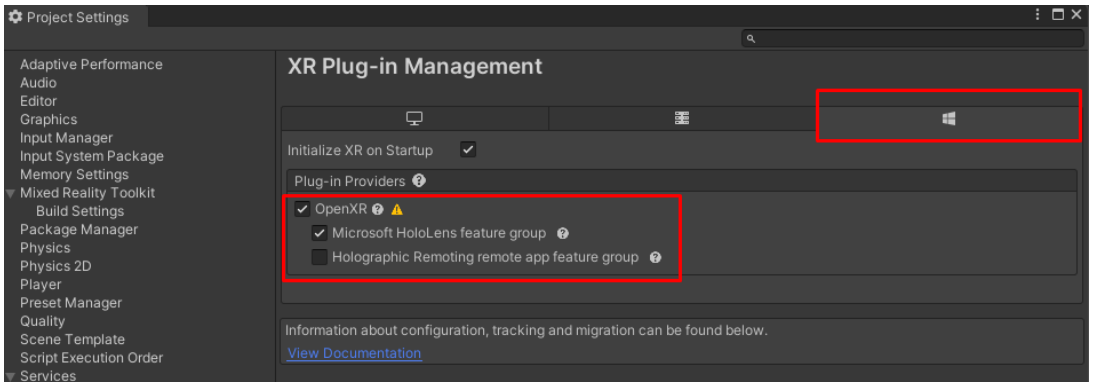
Slika 20: Mixed Reality Feature Tool obavezni dodaci [Izvor: vlastita izrada]

Na idućem koraku će alat prikazati dependencije koji će se dodati uz odabrane dodatke. Nakon provjere, može se odabrati opcija „Import“, i nakon toga „Approve“. Nakon što alat završi ubacivanje dodataka, potrebno je kliknuti na prozor Unity editora kako bi Unity obavio osvježavanje projekta i dodavanje dodataka. Nakon što završi, pojaviti će se prozor na kojemu je potrebno odabrati opciju „Yes“ da se uključe backendi, čime će se restartati Unity. Kada se Unity ponovo pokrene, automatski će se pokrenuti prozor za uvod i postavljanje MRTK u projekt. Na prozoru MRTK konfiguratora je potrebno odabrati opciju „Unity OpenXR plugin“ što je preporučeni način izrade aplikacija za Hololens i WMR uređaje.



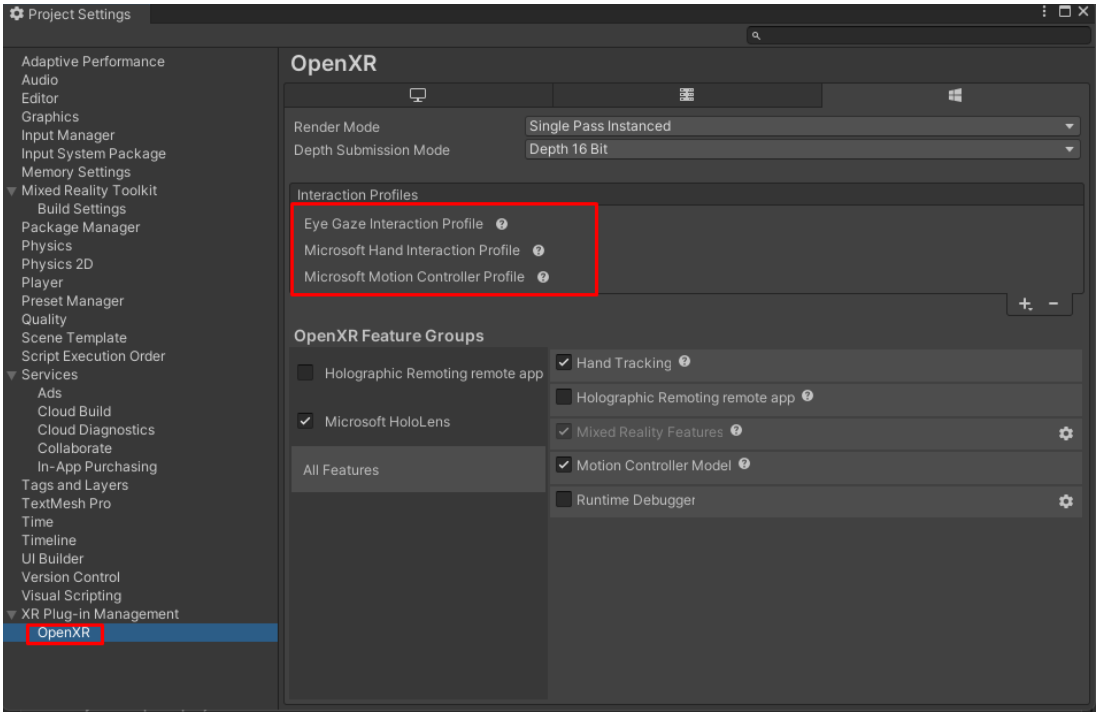
Slika 21: Unity MRTK konfigurator [Izvor: vlastita izrada]

Unity će sada instalirati OpenXR dodatak te otvoriti novi prozor gdje je potrebno kliknuti na opciju „Show XR Plug-in Management Settings“ što će otvoriti prozor sa postavkama OpenXR dodatka. Ovdje se odabire tab sa UWP postavkama te stavlja kvačica kod opcija „OpenXR“ te ispod toga „Microsoft Hololens feature group“ što će pripremiti OpenXR za Hololens.



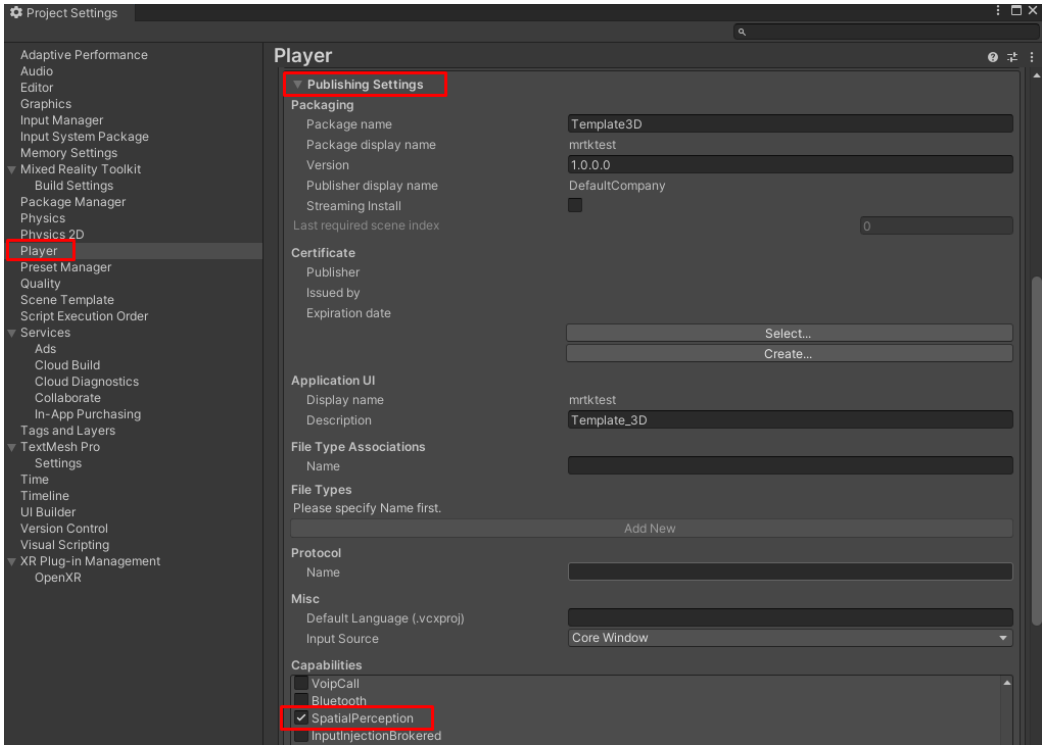
Slika 22: Glavne postavke XR dodatka [Izvor: vlastita izrada]

Ukoliko pored OpenXR opcije bude žuti trokut upozorenja, potrebno je kliknuti na njega te na prozoru koji se prikaže, opciju „Fix all“ što će popraviti sve osim jednog upozorenja, a to je da fale interakcijski profili. Za rješavanje tog problema potrebno je u postavkama projekta otići pod sekciju OpenXR te pod „Interaction profiles“ dodati tri profila kao na slici ispod. Također, pod OpenXR feature groups treba provjeriti da je odabran Microsoft Hololens te opcije s desna „Hand Tracking“ i „Motion Controller Model“. Poželjno je i opciju „Depth Submission Mode“ postaviti na „Depth 16-bit“ kako bi se povećale performanse Unity editora pri testiranju aplikacije.



Slika 23: Dodatne postavke XR dodatka [Izvor: vlastita izrada]

Nakon zatvaranja prozora postavki, MRTK konfigurator će detektirati promjenu postavki i prikazati prozor gdje je potrebno odabrati opciju „Apply settings“ kako bi se postavile optimalne postavke za razvoj aplikacija za Hololens 2 uređaj. Idući korak MRTK konfiguratora obavijestiti će korisnika kako MRTK koristi dodatak TextMeshPro za dodavanje teksta te je tu potrebno odabrati opciju „Import TMP Essentials“. Time je završena priprema projekta za MRTK razvoj i otvara se zadnji korak konfiguratora koji sadrži poveznice na dokumentaciju i API te opciju „Done“ koja zatvara isti. S obzirom da će se u ovom projektu koristiti mapiranje prostora, potrebno je još dodatno u postavkama projekta otići pod Player -> Publishing Settings -> Capabilities i označiti opciju „SpatialPerception“ kako bi se prilikom korištenja aplikacije na emulatoru ili uređaju ispravno aktivirao sustav skeniranja prostora. Na istom ekranu moguće je mijenjati i ostale postavke kao što je ime i opis aplikacije, ime paketa, ikone, verzija i slično.



Slika 24: Postavke aplikacije [Izvor: vlastita izrada]

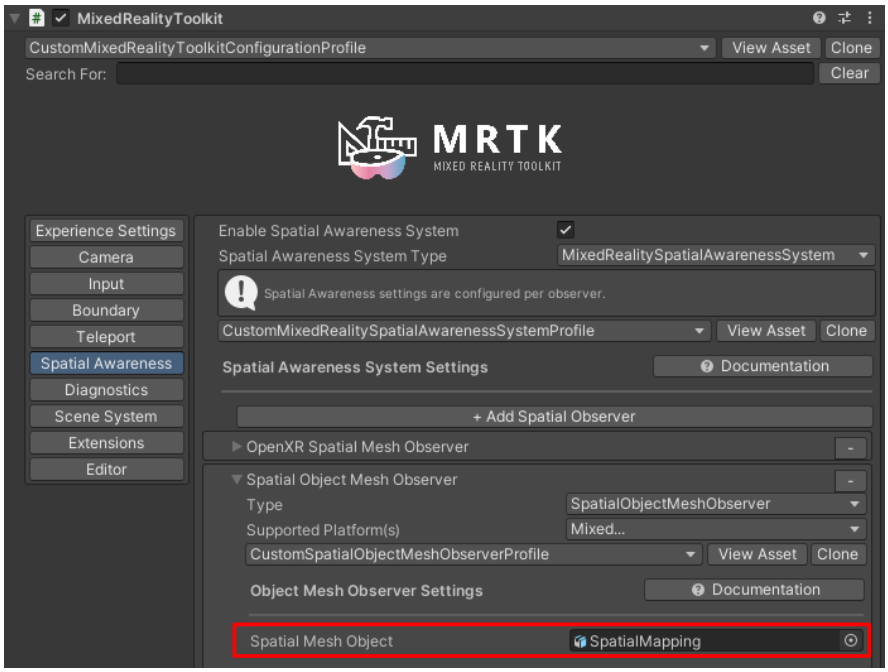
Time je u potpunosti završeno postavljanje projekta te je sada moguće krenuti na razvoj aplikacije.

5.4. Razvoj aplikacije

Razvoj aplikacije započinje dodavanjem MRTK u trenutnu scenu što se obavlja putem izbornika na vrhu Mixed Reality -> Toolkit -> Add to Scene And Configure. Time MRTK dodaje obavezne objekte u scenu:

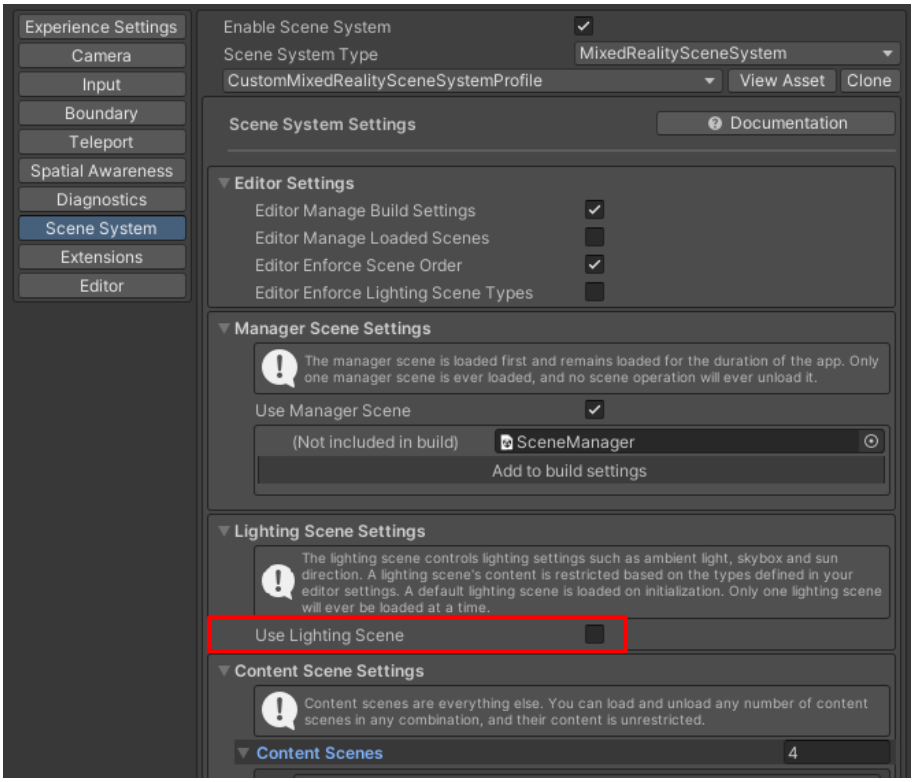
- MixedReality Toolkit – Glavni MRTK objekt koji sadrži Toolkit skriptu sa svim opcijama vezanim za toolkit.
- MixedRealityPlayspace – Objekt koji sadrži kameru korisnika
- MixedRealitySceneContent – Sadrži skriptu koja osigurava da sav sadržaj postavljen unutar ovog objekta bude poravnan u prostoru korisnika i pravilno prikazan. Ovdje se stavljaju svi objekti scene.

Sljedeće što je napravljeno je podešavanje postavki MRTK na Toolkit objektu. MRTK nudi nekoliko predefiniranih preseta postavki koje je moguće izabrati. Ovdje će se koristiti generički profil za većinu uređaja s nekoliko modifikacija. Prvotno, kako će aplikacija koristiti sustav skeniranja prostora (eng. Spatial awareness), isti se uključuje te se dodaju potrebni observeri. Uz OpenXR observer, dodan je i „Spatial Object Mesh Observer“ kako bi isti sustav radio i u Unity editoru, te je tom observeru dodijeljen objekt koji će pri pokretanju predstavljati skenirani prostor.



Slika 25: Postavke sustava skeniranja prostora [Izvor: vlastita izrada]

Sljedeća bitna stvar koja se ovdje podešava je multiscenski sustav. Pošto će aplikacija sadržavati više scena koje će korisnik mijenjati u jednoj aplikaciji, isti je potrebno uključiti i podesiti putem „Scene System“ postavki na istom prozoru. Kada se prvi puta uključi, MRTK će automatski dodati dvije scene u projekt, jednu koja sadrži globalni objekt koji će osvjetljavati sve scene i jednu koja će sadržavati sve glavne MRTK objekte opisane ranije. Jedini objekt koji se može rekreirati i koristiti više puta u drugim scenama je „MixedRealitySceneContent“ za poravnavanje sadržaja. Unutar postavki za multiscenski sustav moguće je odabrati koje scene reprezentiraju koji dio sustava te se niže u postavkama moraju dodati sve scene koje se budu izmjenjivale. Ovdje valja napomenuti da je korišten vlastiti izvor svjetla za svaki dio aplikacije pa je prema tome isključeno korištenje zadane scene u postavkama višescenskog sustava.



Slika 26: Postavke multiscenskog sustava [Izvor: vlastita izrada]

Sljedeće je napravljena nova scena koja će predstavljati početni meni koji će se otvoriti kada korisnik upali aplikaciju te će unutar te scene moći upaliti jednu od tri scene koje želi pokrenuti. Za potrebe ove aplikacije, napravljen je jednostavni početni zaslon sa porukom dobrodošlice te tri gumba od kojih svaki vodi do druge scene. Kako MRTK dolazi sa mnoštvo predefiniranih modela za gumbe, slidere, menije i slično, iskorišten je model gumba iz toolkita sa dodatno dodanom 3D ikonom aplikacije. Slično modelima, MRTK nudi gotove skripte za

interakcije sa modelima koje se mogu dodatno podešavati za razne akcije. Prema tome, gumbi koriste skriptu „NearInteractionTouchable“ kako bi bili osjetljivi na dodir u prostoru, te „LoadContentScene“ koja se okida na dodir i putem koje je moguće postaviti koja scena će se pokrenuti dodirom tog gumba. Scene je moguće otvarati aditivno, no ovdje je postavljeno da se izmjenjuju. Također, kako ista skripta ima opciju pokretanja direktno kada korisnik upali aplikaciju, napravljen je prazan objekt „SwitchToMainMenu“ koji je postavljen u „SceneManager“ scenu te mu je dodana ista skripta sa referencom na scenu početnog zaslona i opcijom „Load On Startup“. Time će se automatski prvo pokrenuti scena sa glavnim izbornikom. Na roditeljski objekt glavnog izbornika dodana je vlastito kreirana skripta koja ima samo jednu ulogu, da prilikom paljenja scene reorijentira objekte scene prema lokaciji korisnika u prostoru. Kod skripte prikazan je ispod.

```
using Microsoft.MixedReality.Toolkit;
using UnityEngine;

/**
 * Kontroler za main menu aplikacije
 */
public class MainMenuController : MonoBehaviour
{
    void Start()
    {
        // reorijentiraj scenu prema kameri
         this.transform.root.gameObject.GetComponent<MixedRealitySceneCo
         ntent>().ReorientContent();
    }
}
```

Unity djeluje na način da se objektima dodaju skripte koje će upravljati njihovim ponašanjem, dok su sami objekti svrstani u stablastu strukturu što znači da svaki objekt može imati djecu i roditelja. Svaka skripta mora implementirati „MonoBehaviour“ što označava startnu klasu za sve Unity skripte. Unutar tih skripti moguće je definirati neke od gotovih metoda koje se pokreću u određenim slučajevima. U ovom primjeru se nalazi funkcija „Start()“ koja se poziva samo jednom i to prilikom instanciranja objekta. Svaki objekt u aplikaciji je vrste GameObject te je putem istoga, funkcijom „GetComponent<>()“ moguće dohvatiti bilo koju komponentu postavljenu na isti objekt. U ovom slučaju se dohvaća korijen odnosno roditelj objekta preko kojega se dolazi do „MixedRealitySceneContent“ komponente koja nudi funkciju „ReorientContent()" kojom se poravnava sav sadržaj scene prema korisniku.



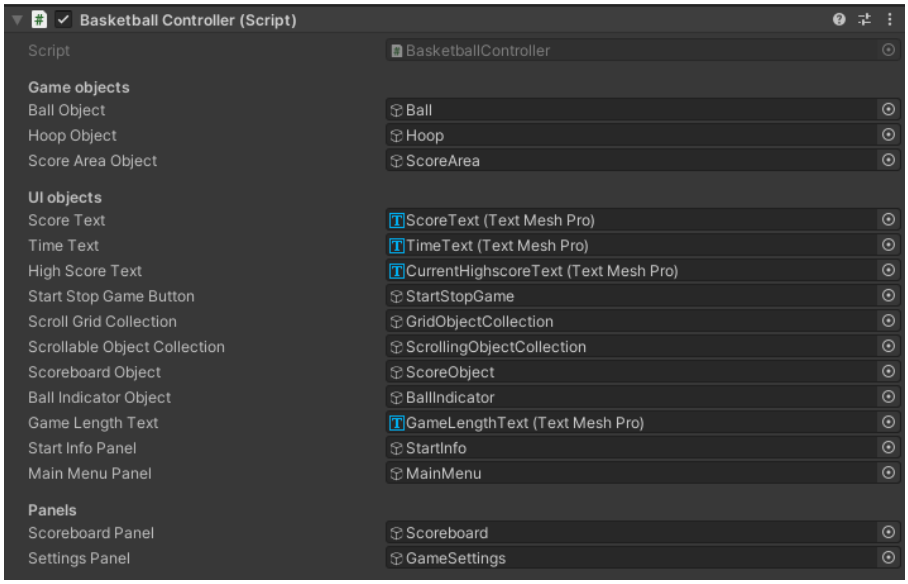
Slika 27: Prikaz strukture objekata i glavnog izbornika aplikacije [Izvor: vlastita izrada]

5.4.1. Pocket Basketball

Razvoj Pocket Basketball aplikacije započet je kloniranjem objekta MixedRealitySceneContent i dodavanjem u novu scenu. Dizajn izbornika napravljen je tako da bude sličan gotovim UI elementima iz MRTK dodatka.

Glavni dio izbornika sastoji se od gumba preuzetih iz MRTK dodatka te modificiranih sa promijenjenim ikonama, nazivima i akcijama koje obavljaju koje su opisane ranije u ovom radu. S desne strane od glavnog izbornika nalazi se ploča sa dvije dodatne akcije, povratak na glavni izbornik i zaustavljanje praćenja korisnika, što će fiksirati izbornik u prostoru. Iznad su smješteni dodatni izbornici za prikazivanje ploče rezultata i postavki igre koji će se prikazivati ovisno o tome da li je korisnik pritisnuo gumb. Objekti gumba složeni su na način da su sadržani u jednom objektu nazvanom „ButtonCollection“ koji sadrži skriptu „GridObjectCollection“. Ta skripta omogućava svrstavanje gumbi u grid ovisno o postavkama, što može biti korisno kod dodavanja dodatnih gumbi te održavanja jednake udaljenosti od ostalih. Elementi gumba iz MRTK dodatka došli su s nekoliko skripti koje ih čine funkcionalnim, no od bitnijih su to „NearInteractionTouchable“, kako bi gumb reagirao na dodir korisnika, „Audio Source“, kako bi se pustio zvuk prilikom pritiska gumba te „Button Config Helper“ koja pojednostavljuje postavljanje ikona, naziva i akcija na jednom mjestu. Da bi se kontroliralo objektima u ovoj sceni, stvoren je novi prazan objekt na vrhu hijerarhije te mu je pridodana nova skripta koja će biti glavni kontroler ove scene. Definiranjem objekata kao javnih unutar skripte govori Unity-ju da te objekte prikaže u postavkama skripte preko čega je moguće postaviti reference unutar editora. Također, korištenjem oznaka [Header(“naziv”)] govori Unity-

ju da postavi naslov iznad skupa objekata kako bi se bolje organizirale postavke skripte. Primjer izgleda postavki referenca objekata za kontroler prikazan je niže.



Slika 28: Postavke kontrolera za Pocket Basketball [Izvor: vlastita izrada]

Ono što je prvo uzeto u obzir kod izrade ove aplikacije jest da korisnik mora prije svega postaviti obruč na pravo mjesto kako bi igra krenula. Na objekt obruča postavljene su skripte „Tap To Place“ i „SolverHandler“. One omogućavaju postavljanje objekta na površinu skeniranu od strane Hololensa, što uključuje i zidove što bi ovdje bio cilj. Prvi dio skripte glavnog kontrolera igre sadržava metode „Start()“ i „Update()“. Za metodu „Start()“ je već spomenuto ranije u radu. Metoda „Update()“ se izvršava svaku sličicu izvršavanja aplikacije, odnosno ako se radi o 60 sličica po sekundi, toliko puta će se izvršiti metoda u sekundi.

```
private void Start()
    {
        scoreEffectObject =
scoreAreaObject.transform.Find("ScoreEffect").gameObject;
        scoreEffectObject.SetActive(false);
        scoreboardPanel.SetActive(false);
        scoreboardObject.SetActive(false);
        settingsPanel.SetActive(false);
        ballObject.SetActive(false);
        mainMenuPanel.SetActive(false);
        ballObject.SetActive(false);
        startInfoPanel.SetActive(true);

        timer = gameLength;

        // reorijentiraj scenu prema kameri

this.transform.root.gameObject.GetComponent<MixedRealitySceneContent>().
ReorientContent();
```

```
        // odmah pokreni Tap to place kako bi se postavio obruč
        StartHoopPlacement();
    }

    private void Update()
    {
        UpdateGameStats();
    }
```

Unutar „Start()“ metode se najprije korištenjem funkcije „SetActive()“ isključuju objekti koji ne trebaju biti aktivni. Postoje dvije pomoćne varijable, gameLength i timer, koji drže referencu postavljene dužine igre i trenutne vrijednosti brojila vremena. Dalje se scena kao i kod glavnog izbornika reorijentira putem metode „ReorientContent()“ pozvane na roditeljskom objektu, te se poziva metoda pod imenom „StartHoopPlacement()“.

```
public void StartHoopPlacement()
    {
        hoopObject.GetComponent<TapToPlace>().enabled = true;
        hoopObject.GetComponent<TapToPlace>().StartPlacement();
    }
```

Metoda pali „Tap To Place“ funkcionalnost na objektu obruča i poziva metodu „StartPlacement“ istoimene gotove skripte. Kako je ovu metodu potrebno pozvati i putem gumba na glavnom izborniku, postavljena je na public te da se poziva kada se okine slušač da je korisnik pritisnuo gumb. Funkcija koja se poziva svaku sličicu u metodi „Update()“ je „UpdateGameStats()“:

```
 private void UpdateGameStats()
    {
        if (currentGame != null) // ako je igra aktivna, ažuriraj timer
        {
            if (timer > 0)
            {
                timer -= Time.deltaTime;
                timeText.text = "Time: " + Math.Round(timer,
2).ToString();
                if (timer < 10f && !isNearEnd)
                {
                    isNearEnd = true; // pomoćna varijabla
                    Debug.Log("playing audio");
                    hoopObject.GetComponent<AudioSource>().Play();
                }
            }
            else
            {
                hoopObject.GetComponent<AudioSource>().Stop();
                // ažuriraj tablicu rezultata
                currentGame.number = ++gameCount;
                currentGame.gameTime = gameLength;
                highscore = highscore < currentGame.score ?
currentGame.score : highscore;
                highScoreText.text = "Current highscore: " +
highscore.ToString();

                AddToScoreboard(currentGame);
```

```
                ResetValues();
            }
        }
    }
```

Unutar skripte postoji objekt pod nazivom currentGame tipa “Game” koji sadržava informacije o igri koja se trenutno igra. Struktura objekta je sljedeća:

```
private class Game
    {
        public int number; // redni broj
        public int score; // trenutni broj koševa
        public int fieldGoals; // 3 poena
        public int freeThrows; // 1 poen
        public float gameTime; // vrijeme igre
    }
```

U funkciji „UpdateGameStats()“ najprije se gleda da li je taj objekt prazan jer ako nije, igra je u tijeku. Zatim se gleda da li je timer veći od nule ili nije. Ako je timer veći od nule, igra je i dalje u tijeku te je potrebno smanjiti brojač, postaviti tekst kako bi se vizualno ažurirao brojač korisniku te pokrenuti zvuk odbrojavanja putem komponente „AudioSource“ ako je brojač ispod deset. Ako je brojač nula ili manji, znači da je igra završila te je potrebno postaviti redni broj igre, duljinu igre, trenutni najveći broj koševa prikazan na ekranu rezultata dodati objekt igre u panel rezultata putem metode „AddToScoreboard()“:

```
private void AddToScoreboard(Game game)
    {
        GameObject scoreboardItem = Instantiate(scoreboardObject,
scrollGridCollection.transform);

scoreboardItem.transform.Find("Number").GetComponent<TextMeshPro>().text
= game.number.ToString() + ".";

scoreboardItem.transform.Find("Score").GetComponent<TextMeshPro>().text
= game.score.ToString();

scoreboardItem.transform.Find("Time").GetComponent<TextMeshPro>().text =
game.gameTime.ToString() + "s";

scoreboardItem.transform.Find("FreeThrows").GetComponent<TextMeshPro>().
text = game.freeThrows.ToString();

scoreboardItem.transform.Find("FieldGoals").GetComponent<TextMeshPro>().
text = game.fieldGoals.ToString();
        scoreboardItem.SetActive(true);


scrollGridCollection.GetComponent<GridObjectCollection>().UpdateCollecti
on();

scrollableObjectCollection.GetComponent<ScrollingObjectCollection>().Upd
ateContent();
    }
```

Prije svega, ovdje se instancira novi objekt rezultata koji se nalazi u „ScrollingObjectCollection“ objektu u panelu rezultata. Skripta koja omogućava skrolanje po tom prozoru je istoimena objektu i dolazi uz MRTK dodatak. Dalje se za svaki tekstualni element objekta rezultata postavljaju vrijednosti igre te se pozivaju metode „UpdateCollection()“ i „UpdateContent()“ kako bi se osvježio skrolabilni prikaz. Na posljetku se u metodi „UpdateGameStats()“ poziva metoda „ResetValues()“:

```
private void ResetValues()
    {
        timeText.text = "Time: -";
        scoreText.text = "Score: 0";
        timer = gameLength;
        currentGame = null;
        isNearEnd = false;

startStopGameButton.GetComponent<ButtonConfigHelper>().MainLabelText =
"Start game";

startStopGameButton.GetComponent<ButtonConfigHelper>().SetQuadIconByName
("IconDone");
    }
```

Metoda resetira tekstualne elemente tabele s vremenom, te preostale pomoćne varijable. Kako se ova ista metoda poziva i kroz metodu pokretanja igre, resetiraju se ikona i naziv gumba.

```
public void StartStopGame()
    {
        if (currentGame != null) // igra je u tijeku, zaustavi ju
        {
            ResetValues();
        } else
        {
            currentGame = new Game();

startStopGameButton.GetComponent<ButtonConfigHelper>().MainLabelText =
"Stop game";

startStopGameButton.GetComponent<ButtonConfigHelper>().SetQuadIconByName
("IconClose");
        }
    }
```

Metoda za pokretanje odnosno zaustavljanje igre jednostavno pregledava da li je trenutno igra aktivna te ili kreira novu igru ili poziva prijašnje opisanu „ResetValues()“ funkciju.

Kako je jedan od zahtjeva aplikacije bio napraviti obruč koji će dodavati bodove korisnika samo ako se zabije koš kroz gornji dio obruča, koriste se dva collider objekta sa oznakom „isTrigger“ kako bi se smatrala samo okidačima. Na oba collidera je postavljena nova skripta „DelegateTriggerCall“:

```
public class DelegateTriggerCall : MonoBehaviour
{
    private Collider caller;
    private void Awake()
    {
        caller = GetComponent<Collider>();
    }

    public UnityEvent<OnTriggerDelegation> Enter;

    void OnTriggerEnter(Collider other) => Enter.Invoke(new
OnTriggerDelegation(caller, other));
}

public struct OnTriggerDelegation
{
    public OnTriggerDelegation(Collider caller, Collider other)
    {
        Caller = caller;
        Other = other;
    }

    public Collider Caller { get; private set; }
    public Collider Other { get; private set; }
}
```

Jednostavno rečeno, skripta služi samo da bi se delegirao poziv „OnTriggerEnter()“ metode na zadanu funkciju u postavkama skripte. Toj funkciji se onda prosljeđuje „OnTriggerDelegation“ struktura koja sadrži pozivatelja funkcije, ovdje su to jedan od dva collidera, i objekt koji je ušao u prostor collidera, u ovom slučaju je to lopta. Funkcija koja se poziva je upravo „OnScoreAreaTriggerEnter(OnTriggerDelegation delegation)“ unutar glavnog kontrolera:

```
public void OnScoreAreaTriggerEnter(OnTriggerDelegation delegation)
    {
        if (delegation.Other.name.Equals("Ball"))
        {
            if (delegation.Caller.name.Equals("ScoreCollider"))
            {
                if (hoopDown)
                {
                    hoopDown = false;
                }
                else
                {
                    hoopUp = true;
                    if (scoreEffectObject.activeSelf)
                    {
                        scoreEffectObject.SetActive(false);
                    }
                    scoreEffectObject.SetActive(true);
                    scoreAreaObject.GetComponent<AudioSource>().Play();

                    if (currentGame != null) // samo kada je igra
aktivna
                    {
                        AddScore();
                    }
```

```
                }
            }
            else if (delegation.Caller.name.Equals("UnderHoopCollider"))
            {
                if (hoopUp)
                {
                    hoopDown = false;
                    hoopUp = false;
                }
                else
                {
                    hoopDown = true;
                }
            }
        }
    }
```

Dakle, ako se radi o lopti, te „ScoreCollider“ objektu, potrebno je pokrenuti vizualni efekt i zvuk zabijanja koša te dodati poene putem funkcije „AddScore()“. Ako se pak radi o „UnderHoopCollider“ objektu, onda se ne dodaju poeni već se postavlja stanje pomoćnih varijabli koje služe samo da se može provjeriti da li je prilikom dodirivanja lopte i „ScoreCollider“ objekta, lopta prije toga dotaknula „UnderHoopCollider“. Time se vodi računa da igrač ne može zabijati koševe ispod samog obruča. Funkcija za dodavanje poena je sljedeća:

```
private void AddScore()
    {
        // odredi broj poena
        float distance =
Vector3.Distance(scoreAreaObject.transform.position, lastShootPosition);
        int score = 0;
        if (distance < 3)
        {
            score = 1;
        } else if (distance < 4)
        {
            score = 2;
        } else if (distance > 4)
        {
            score = 3;
        }

        currentGame.score += score;
        scoreText.text = "Score: " + currentGame.score;
        if (score == 1) currentGame.freeThrows++;
        if (score == 2 || score == 3) currentGame.fieldGoals++;
    }
```

Prvotno se gleda udaljenost između zadnje zabilježene pozicije kamere igrača kada je lopta puštena iz ruke te pozicije objekta obruča. Temeljeno na dobivenoj udaljenosti se dodaju poeni te ažuriraju informacije trenutne igre i vizualni prikaz poena igraču. Kamera igrača bilježi se u funkciji „OnBallHoldEnded()“:

```
public void OnBallHoldEnded()
    {
        // ovdje može ići ili pozicija kamere ili pozicija lopte
        lastShootPosition = Camera.main.transform.position;
    }
```

Funkcija se poziva svaki put kada je okinut događaj puštanja lopte iz ruke te se iz globalno dostupnog objekta kamere uzima pozicija u prostoru. Postoji još nekoliko javnih funkcija koje se pozivaju pritiskom na gumbe. Primjerice, funkcija „BringToPlayer()“ se okida kada igrač pritisne gumb za resetiranje lopte i postavlja poziciju lopte ispred igrača na udaljenosti od jedan metar ili funkcija „ToggleBallIndicator()“ koja se okida kada igrač pritisne gumb za pronalazak lopte i aktivira objekt strelice koji prati objekt lopte putem „SolverHandler“ skripte iz MRTK dodatka te funkcije „ToggleScoreboard()“ i „ToggleSettings()“ koje jednostavno aktiviraju i deaktiviraju panele sa rezultatima i postavkama. Posljednja metoda koja se okida kada završi postavljanje obruča je „OnTapToPlaceEnded()“:

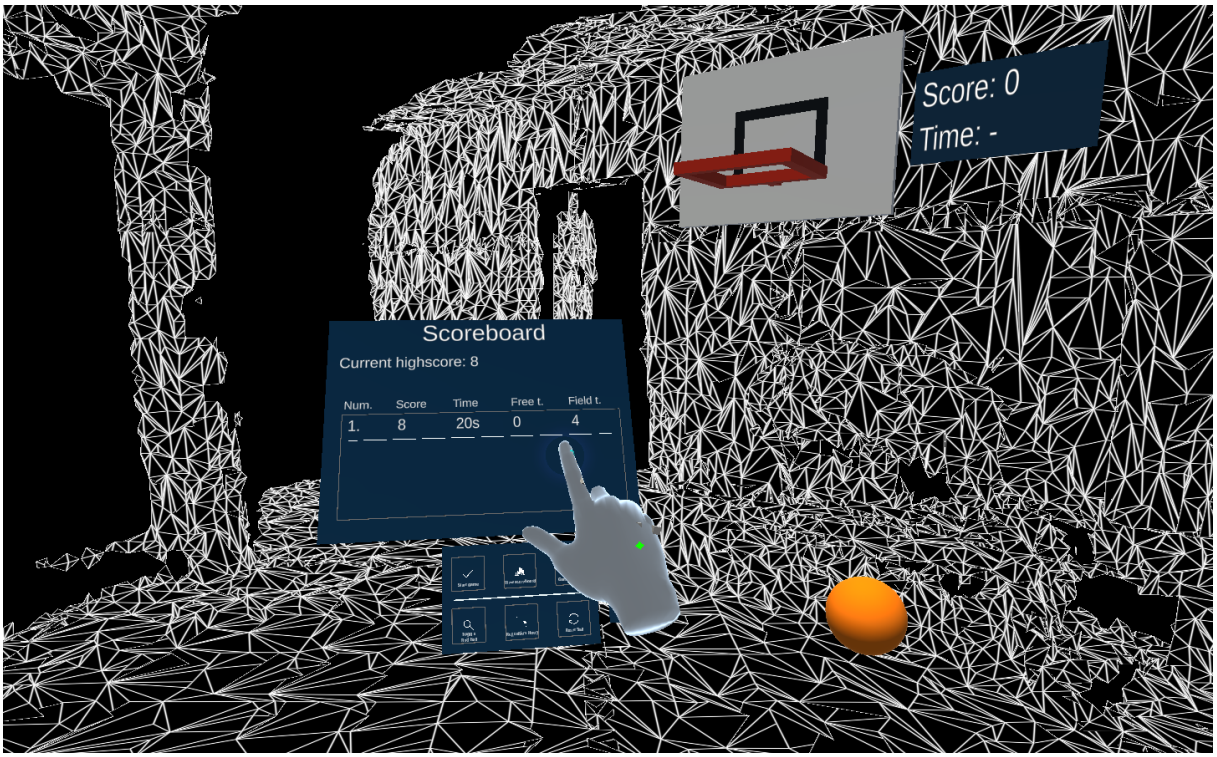
```
public void OnTapToPlaceEnded()
    {
        hoopObject.GetComponent<TapToPlace>().enabled = false;
        if (startInfoPanel.activeSelf)
        {
            startInfoPanel.SetActive(false);
            mainMenuPanel.SetActive(true);
            BringToPlayer();
            ballObject.SetActive(true);
        }
    }
```

Ovdje se jednostavno onemogućava „Tap To Place“ funkcionalnost i, ukoliko je bio aktivan informativni panel na početku igre, se aktivira glavni izbornik. Uključuje se lopta i dovodi igraču pomoću sljedeće funkcije:

```
public void BringToPlayer()
    {
        ballObject.GetComponent<Rigidbody>().velocity = Vector3.zero;
        ballObject.GetComponent<Rigidbody>().angularVelocity =
Vector3.zero;

        Camera camera = Camera.main;
        ballObject.transform.position = camera.transform.position +
camera.transform.forward * ballDistance;
    }
```

Prvi dio funkcije služi da bi se lopti resetiralo kretanje kako ne bi odletjela prilikom resetiranja pozicije, a drugi dio funkcije postavlja poziciju lopte ispred kamere igrača postavljene na udaljenosti zadanoj u varijabli „ballDistance“ što je ovdje jedan metar.



Slika 29: Pokretanje aplikacije Pocket Basketball [Izvor: vlastita izrada]

5.4.2. AirPaint

Kao i sa prijašnjom aplikacijom, ovdje je prvo stvoren MixedRealitySceneContent objekt koji će držati sve ostale objekte u sceni. Za lokalno crtanje, napravljen je jednostavan model platna koje će plutati u prostoru te će se moći lijepiti na prostor korisnika. Implementacija crtanja napravljena je uz pomoć Unity Line Renderer komponente putem koje je moguće crtati linije u prostoru dodavanjem točki s koordinatama. Komponenti je moguće mijenjati boje, debljinu linije, lokaciju linija, zaobljenost kutova i slično. Podaci o crtanju će se spremati kao klonirani objekti u jedan od dva prazna objekta koji će biti zaduženi za spremanje 3D ili 2D crtanja. Desno od platna napravljen je vertikalni izbornik sa gumbima za otvaranje izbora boja, debljine kista, resetiranja crteža i ostalih akcija. Izbornik je stavljen kao dijete objekta koji sadrži platno kako bi se kretao zajedno sa platnom ako ga korisnik pomiče. Također je napravljen objekt koji sadrži glavnu skriptu cijele aplikacije. Kao i kod prijašnje aplikacije, skripta sadrži reference na sve bitne objekte unutar aplikacije kroz Unity.

```
private void Start()
    {
        timer = lineDelay; // inicijaliziraj timer

        // pobrini se da objekti koji trebaju biti isklju?eni u startu
stvarno jesu
        colorMenu.SetActive(false);
        brushMenu.SetActive(false);
```

```
        targetRenderer = colorPreviewObject.GetComponent<Renderer>();

        // reorijentiraj scenu prema kameri

this.transform.root.gameObject.GetComponent<MixedRealitySceneContent>().
ReorientContent();
    }
```

U startu se inicijalizira brojač u varijabli timer koji će voditi računa o broju točaka koje se postavljaju kada korisnik vuče liniju. To rješava problem postavljanja previše točaka u sekundi te su time performanse poboljšane. Deaktiviraju se dodatni izbornici te se reorijentiraju objekti prema kameri korisnika. Također se uzima komponenta Renderer iz objekta koji nudi prikaz odabrane boje kako bi korisnik vizualno vidio koju je boju izabrao. Objekt koji će predstavljati liniju je sljedeće strukture:

```
private class Line // trenutna linija sa listom vektora
    {
        public GameObject currentLine;
        public List<Vector3> pointList = new List<Vector3>();
    }
```

Kako je u planu bilo izvesti mogućnost crtanja putem Air Tap funkcije koju nudi Hololens, bilo je potrebno iskoristiti nekoliko Handlera koje nudi MRTK. Prvi handler koji je zadužen za dohvaćanje koordinata i crtanje je „Pointer Handler“. To je gotova MRTK skripta koja izlaže „Pointer Down“, „Pointer Up“ i „Pointer Dragged“ događaje sa odgovarajućim objektom koji nudi informacije o lokaciji i vrsti Pointera. Handler je postavljen na objekt platna u Unity te su stvorene metode u glavnoj skripti.

```
public void PointerDown(MixedRealityPointerEventData eventData)
    {
        if (!resizeToggled && !tapToPlaceToggled)
        {
            var result = eventData.Pointer.Result;
            eventData.Pointer.IsTargetPositionLockedOnFocusLock = false;
            eventData.Pointer.IsFocusLocked = false;

            lineDict.Remove(eventData.Pointer.PointerId); // makni ako
je ostala linija od prije sa istim identifikatorom
            drawing = true;
            Line lineClone = CreateLine();
            lineDict.Add(eventData.Pointer.PointerId, lineClone);
            if (rainbowMode)
            {

lineClone.currentLine.GetComponent<LineRenderer>().endColor =
GetRandomColor();

lineClone.currentLine.GetComponent<LineRenderer>().startColor =
GetRandomColor();
            }
        }
    }
```

Putem pomoćnih varijabla, prvo se gleda da li je trenutno upaljeno skaliranje platna ili funkcija Tap To Place jer ne želimo da korisnik crta ako se platno pozicionira. To se radi u svakoj od Pointer funkcija. Objekt „MixedRealityPointerEventData“ sadrži sve podatke koji su ovdje potrebni uključujući i pointer. Postavljaju se vrijednosti varijabli „IsTargetPositionLockedOnFocusLock“ i „IsFocusLocked“ na false kako bi se spriječilo lijepljenje pointera za platno kao i crtanje izvan platna. Također se miče linija iz liste ako je preostala, kreira se nova linija koja se dodaje u listu i, ako se radi o rainbow crtanju, generiraju se nove boje svakim stvaranjem linije odnosno povlačenjem kista. Nova linija kreira se funkcijom koja izgleda ovako:

```
private Line CreateLine()
    {
        Line lineClone = new Line();
        lineClone.currentLine = Instantiate(lineObject, drawingMode3D ?
drawing3dData.transform : drawingData.transform);
        lineClone.currentLine.name = "Data (" + lineCount + ")";
        lineCount++;
        return lineClone;
    }
```

Dakle, instancira se novi objekt vrste Line u odgovarajući objekt roditelja ovisno o vrsti crtanja. Postavlja se ime i vraća istoimena linija. Sljedeća je funkcija povlačenja pointera:

```
public void PointerDragged(MixedRealityPointerEventData eventData)
    {
        if (!resizeToggled && !tapToPlaceToggled)
        {
            if (!drawing && !drawingMode3D)
            {
                lineDict.Remove(eventData.Pointer.PointerId);
                drawing = true;
                PointerDown(eventData);
            }
            timer -= Time.deltaTime;
            if (timer <= 0)
            {
                // 2D crtanje treba koristiti local space pointere
                var position = drawingMode3D ?
eventData.Pointer.Position :
eventData.Pointer.Result.Details.PointLocalSpace;

                Line lineClone;
                if (lineDict.TryGetValue(eventData.Pointer.PointerId,
out lineClone))
                {
                    lineClone.pointList.Add(position);

lineClone.currentLine.GetComponent<LineRenderer>().positionCount =
lineClone.pointList.Count;

lineClone.currentLine.GetComponent<LineRenderer>().SetPositions(lineClon
e.pointList.ToArray());

                    timer = lineDelay;
                }
```

```
            }
        }
    }
```

Ovdje valja napomenuti da ako korisnik dok crta napusti platno, i dalje se stvaraju linije jer tako funkcionira fokusiranje pointera kroz MRTK, pa je bilo potrebno iskoristiti „Focus Handler“ na objekt platna. Na „FocusExit“ događaj je postavljeno samo da se varijabla „drawing“ postavlja na false kako bi se znalo da je korisnik napustio platno. Dalje se u funkciji „PointerDragged“ gleda da li se ne radi o 3D slikanju i da li je korisnik napustio platno te se kreira novi objekt s linijom. Time se dobiva realnija linija bez spajanja sa prijašnjom kada se korisnik vrati na crtanje platna. Dalje se brojač oduzima te ako je manji ili jednak nuli se ulazi u dio koji zapravo obavlja crtanje linije dohvaćanjem ili lokalne ili world pozicije, ovisno o tome radi li se o 2D ili 3D slikanju, te se dodaje nova pozicija liniji u listi i resetira brojač.

```
public void PointerUp(MixedRealityPointerEventData eventData)
    {
        if (!resizeToggled)
        {
            lineDict.Remove(eventData.Pointer.PointerId);
            if (lineDict.Count == 0)
            {
                drawing = false;
            }
        }
    }
```

Zadnja Pointer metoda se okida kada korisnik podigne prst i time se trenutna linija miče iz liste. Za postavljanje platna, dodane su „Tap To Place“, „NearInteractionGrabbable“ i „ObjectManipulator“ gotove MRTK skripte na objekt platna. Također, kako MRTK nudi i mogućnost postavljanja ograničenja putem skripte „Constraint Manager“, postavljena su ograničenja minimalne i maksimalne veličine putem skripte „MinMaxScaleConstraint“ te rotacija putem skripte „Rotation Axis Constraint“. Ta dva ograničenja postavljena su direktno na objekt platna kako se ne bi desilo da izbornik završi u krivoj poziciji naspram platna. Funkcije koje su zadužene za funkciju „Tap To Place“ u glavnom kontroleru su sljedeće:

```
public void EnableTapToPlace()
    {
        tapToPlaceToggled = true;
        boardObject.GetComponent<BoxCollider>().enabled = false;
        drawingBoardObject.GetComponent<TapToPlace>().enabled = true;
        drawingBoardObject.GetComponent<TapToPlace>().StartPlacement();
    }

public void OnTapToPlaceEnded()
    {
        tapToPlaceToggled = false;
        boardObject.GetComponent<BoxCollider>().enabled = true;
        drawingBoardObject.GetComponent<TapToPlace>().enabled = false;
    }
```

Stanje aplikacije se prvotno postavlja na „tapToPlaceToggled“ da se onemogući crtanje prilikom pomicanja. Gasi se Collider samog objekta ploče jer on blokira micanje cijelog objekta ploče te se pali funkcija „Tap To Place“. Nakon što korisnik završi sa postavljanjem, poziva se funkcija „OnTapToPlaceEnded()“ kako bi se ponovo postavilo stanje aplikacije, uključio Collider i onemogućilo postavljanje. Svi gumbi unutar ove aplikacije ponašaju se jednako kao i u prijašnjoj jer su preuzeti iz MRTK dodatka, odnosno osjetljivi su na dodir i okidaju funkcije.

Pritiskom na gumb resetiranja crteža poziva se sljedeća funkcija:

```
public void ResetDrawing()
    {
        foreach (Transform child in drawing3dData.transform)
        {
            Destroy(child.gameObject);
        }
        foreach (Transform child in drawingData.transform)
        {
            Destroy(child.gameObject);
        }
    }
```

U funkciji se jednostavno prolazi kroz svu djecu objekata „drawingData“ i „drawing3dData“ koji sadrže sve podatke o crtanju odnosno sve linije i iste se brišu pozivanjem Unity „Destroy()“ funkcije koja briše objekte iz trenutne sesije.

```
public void SwitchDrawingMode()
    {
        drawingMode3D = !drawingMode3D;

boardObject.GetComponent<CustomPointerHandler>().SetFocusRequired(!drawi
ngMode3D);
        lineObject.GetComponent<LineRenderer>().useWorldSpace =
drawingMode3D;
    }
```

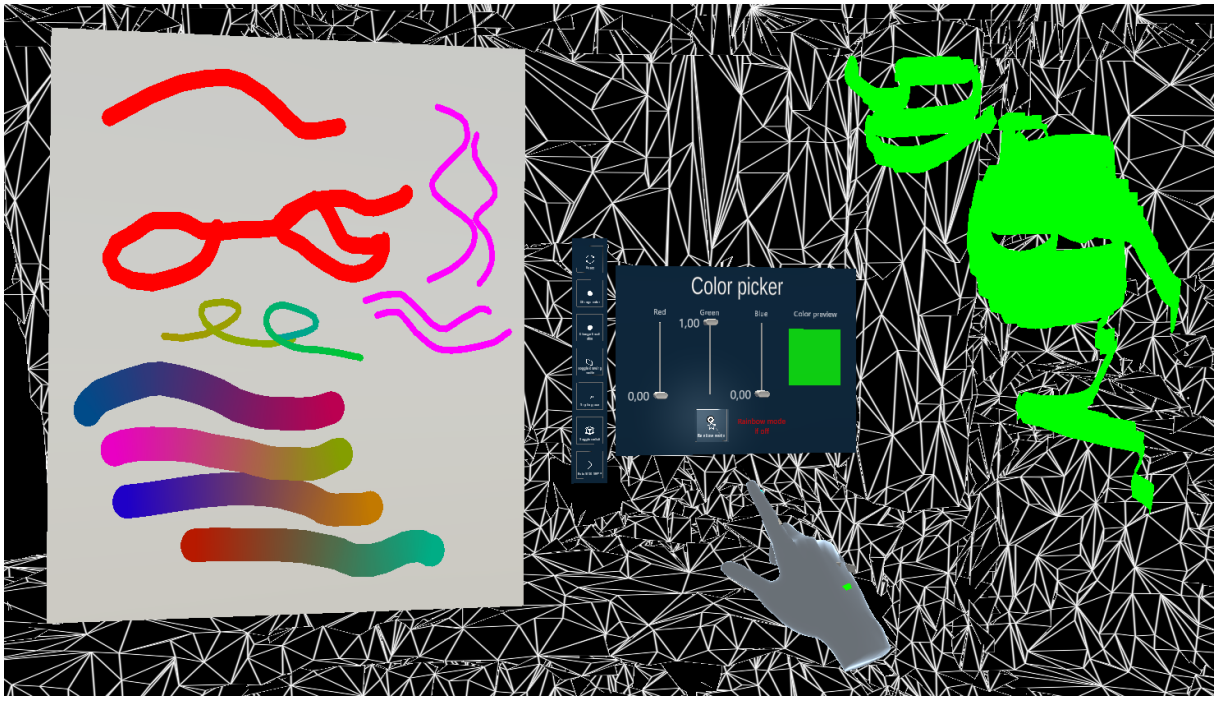
Pritiskom na gumb promjene vrste crtanja poziva se gore prikazana funkcija koja mijenja vrstu crtanja iz 2D u 3D i obrnuto. U funkciji se postavlja stanje aplikacije kroz varijablu „drawingMode3D“ te se mora pozvati „SetFocusRequired()“ na Pointer Handler komponenti objekta ploče i postaviti da fokus na ploču nije nužan ako se radi o 3D crtanju. Zadani MRTK Pointer Handler je imao vidljivost iste funkcije postavljene na privatno, pa je napravljena nova klasa sa kopiranim kodom iz originalne klase uz modifikaciju da postoji javno dostupan setter. Također uz fokus je potrebno postaviti da Line Renderer komponenta koristi world koordinate ako se radi o 3D crtanju.

Za izbornik biranja boje, postavljeni su slideri za crvenu, zelenu i plavu boju čijom kombinacijom korisnik bira koju će boju koristiti. Slider objekti su preuzeti iz MRTK dodatka i sadrže skriptu „PinchSlider“ koja omogućava postavljanje opcija ali i funkcija koje će se pozivati

kada se pomiču slideri. Prema tome su napravljene tri funkcije, svaka za jednu od boja, te će se ovdje prikazati samo jedna s obzirom da su gotovo jednake:

```
public void OnSliderUpdateBlue(SliderEventData eventData)
    {
        if ((targetRenderer != null) && (targetRenderer.material != 
null))
        {
            targetRenderer.material.color = new 
Color(targetRenderer.sharedMaterial.color.r, 
targetRenderer.sharedMaterial.color.g, eventData.NewValue);
            if (!rainbowMode)
            {
                lineObject.GetComponent<LineRenderer>().startColor = 
targetRenderer.material.color;
                lineObject.GetComponent<LineRenderer>().endColor = 
targetRenderer.material.color;
            }
        }
        blueText.text = $"{eventData.NewValue:F2}";
    }
```

Jednostavno rečeno, na Renderer koji je postavljen u startu se postavlja isti materijal samo sa promijenjenom bojom te ako se ne radi o rainbow modu rada se ista boja odmah postavlja i na Line Renderer. Odabrana boja se vizualno također vidi na istom prozoru. Funkcija za promjenu debljine kista radi na sličnom principu te neće ovdje biti prikazana, kao i funkcije za prikazivanje odnosno gašenje prozora čija se funkcionalnost ne razlikuje od one prikazane u kodu prijašnje aplikacije.



Slika 30: Pokretanje aplikacije AirPaint [Izvor: vlastita izrada]

5.4.3. Weather 3D

Ovdje je u planu bilo napraviti detaljniji pregled vremena i prognoze sa kontrolama na nekoliko ekrana ispred korisnika, pa su tako prijašnje opisane funkcionalnosti postavljene na nekoliko panela. Neki od elemenata na panelu, kao i za prijašnje aplikacije, su preuzeti iz MRTK dodatka i modificirani, primjerice reaktivni gumbi i polje za pretraživanje. Za 3D model prikaza trenutnog stanja vremena iskorišten je besplatan paket asseta iz Unity trgovine koji sadrži modele drveća, kuća, cesta i slično. Vremenski efekti su samostalno kreirani osim efekta munje čija je besplatna biblioteka također preuzeta iz Unity trgovine. Slično prijašnjim aplikacijama, na objekte prozora i 3D modela vremena, postavljene su skripte „Object Manipulator“, „SolverHandler“, „Tap To Place“ i „NearInteractionGrabbable“ iz MRTK dodatka što će omogućiti interakciju i postavljanje prozora i modela od strane korisnika. Dodatno, na 3D model vremena postavljene su skripte „Maintain Apparent Size Constraint“ i „MinMaxScaleConstraint“, također iz MRTK dodatka, što će onemogućiti korisniku da mijenja veličinu modela kako se ne bi poremetili efekti. Naravno, uz glavne objekte, stvoren je i prazan objekt koji će sadržavati glavnu skriptu za cijelu aplikaciju.

Kako se u ovoj aplikaciji koristi OpenWeatherMap API, iskorišten je vlastiti API ključ te su napravljeni modeli podataka koji će se dohvaćati i serijalizirati:

```
[Serializable]
public class WeatherInfo
{
    public int id;
    public string name;
    public List<Weather> weather;
    public Main main;
    public Wind wind;
    public Sys sys;
}

[Serializable]
public class Weather
{
    public int id;
    public string main;
    public string description;
    public string icon;
}

[Serializable]
public class Main
{
    public string temp;
    public string feels_like;
    public string pressure;
    public string humidity;
    public string temp_min;
    public string temp_max;
}
```

```
[Serializable]
public class Wind
{
    public string speed;
    public string deg;
}

[Serializable]
public class Sys
{
    public string country;
}

[Serializable]
public class City
{
    public string id;
    public string name;
    public string country;
}

[Serializable]
public class Cities
{
    public City[] cities;
}

[Serializable]
public class Forecast
{
    public string cod;
    public string cnt;
    public List<Flist> list;
}

[Serializable]
public class Flist
{
    public string dt;
    public string dt_txt;
    public Main main;
    public List<Weather> weather;
}
```

Ovdje će se dohvaćati samo potrebni podaci što uključuje podatke o brzini vjetra, prognozi, gradu, temperaturi, vlazi, tlaku, opisu vremena i ikoni. Prema tome, slijedi funkcija koja se pokreće pri startanju aplikacije:

```
private void Start()
    {
        weatherObjects.Add(snowObject);
        weatherObjects.Add(heavyCloudsObject);
        weatherObjects.Add(lightCloudsObject);
        weatherObjects.Add(rainObject);
        weatherObjects.Add(fogObject);
        weatherObjects.Add(clearObject);

        weather3DModel.SetActive(false);
        forecastPanel.SetActive(false);
        forecastObject.SetActive(false);
```

```
        weatherPanel.SetActive(true);

        ReadCities();
        if (weatherPanel.activeSelf)
        {
            StartCoroutine(FetchWeather(UpdateWeather));
        }
        if (forecastPanel.activeSelf)
        {
            StartCoroutine(FetchForecast(UpdateForecast));
        }

        // reorijentiraj scenu prema kameri
this.transform.root.gameObject.GetComponent<MixedRealitySceneContent>().
ReorientContent();

        // mora se dohvatiti poslije orijentacije
        modelPosition = weather3DModel.transform.localPosition;
    }
```

Prvo što se ovdje radi je dodavanje svih referenci na vremenske objekte odnosno efekte u listu što će biti potrebno kasnije. Deaktiviraju se svi elementi koji trebaju biti isključeni u startu, reorijentira se scena prema korisniku i dohvaća se izvorna pozicija 3D modela u odnosu na izbornik, kako bi se ona mogla kasnije resetirati prilikom gašenja odnosno paljenja modela. Kako je za pretraživanje gradova iskorištena lista ID-ja preuzeta u JSON formatu sa OpenWeatherMap API stranice, ona se učitava putem funkcije „ReadCities()“ koja sadrži jednu liniju poziva funkcije „JsonUtility.FromJson()“ što čita dodijeljenu JSON datoteku u Unity referencama. Nakon toga se pokreću dvije korutine unutar kojih se sinkrono dohvaćaju podaci o vremenu. Proslijeđene funkcije označavaju callback funkcije koje će se pozvati nakon što se vrijeme dohvati.

```
private IEnumerator FetchWeather(Action<WeatherInfo> onSuccess)
    {
        // dohvati vrijeme
        UnityWebRequest request =
UnityWebRequest.Get(String.Format("http://api.openweathermap.org/data/2.
5/weather?id={0}&APPID={1}&units=metric", customCityId != null ?
customCityId : CITY_ID, API_KEY));
        yield return request.SendWebRequest();
        if (request.result == UnityWebRequest.Result.ProtocolError ||
request.result == UnityWebRequest.Result.ConnectionError)
        {
            Debug.Log(request.error);
        }
        else
        {
            byte[] result = request.downloadHandler.data;
            string jsonResponse =
System.Text.Encoding.Default.GetString(result);
            WeatherInfo info =
JsonUtility.FromJson<WeatherInfo>(jsonResponse);
            request.Dispose();

            // dohvati i ikonu
```

```
            request =
UnityWebRequestTexture.GetTexture(String.Format("https://openweathermap.
org/img/wn/{0}@2x.png", info.weather[0].icon));
            yield return request.SendWebRequest();
            if (request.result == UnityWebRequest.Result.ProtocolError
|| request.result == UnityWebRequest.Result.ConnectionError)
            {
                Debug.Log(request.error);
            }
            else
            {
                Texture2D imgTexture =
((DownloadHandlerTexture)request.downloadHandler).texture;
                Sprite imgSprite = Sprite.Create(imgTexture, new Rect(0,
0, imgTexture.width, imgTexture.height), new Vector2(.5f, .5f));
                weatherIcon.sprite = imgSprite;
                request.Dispose();
                onSuccess(info);
            }
        }
    }
```

Funkcije vraćaju sučelje tipa „IEnumerator“ što je tip kojega funkcije korutina moraju vraćati. Vrijeme se dohvaća putem pozivanja „Get()“ metode kroz klasu „UnityWebRequest“. Prije pozivanja, na URL se dodaju podaci o API ključu te ID grada koji se pretražuje. Nakon uspješnog dohvaćanja vremena se vraćeni tekstualni JSON pretvara u model pozivanjem funkcije „JsonUtility.FromJson<>()“. Kako se radi o samo jednom podatku, ovdje se odmah dohvaća i ikona kroz drugi GET zahtjev te se poziva proslijeđena „onSuccess()“ funkcija. Kako je korutina za dohvaćanje prognoze gotovo ista, ona se ovdje neće prikazati. Funkcija koja se poziva nakon dohvata podataka o vremenu je sljedeća:

```
private void UpdateWeather(WeatherInfo info)
    {
        this.weatherInfo = info;

        // izaberi objekte koji se prikazuju na 3D modelu.
        switch (weatherInfo.weather[0].main)
        {
            case "Clear":
                {
                    clearObject.SetActive(true);
                    foreach (GameObject weatherObject in weatherObjects)
                    {
                        if (!weatherObject.Equals(clearObject))
                        {
                            weatherObject.SetActive(false);
                        }
                    }
                    break;
                }
            case "Clouds":

……

        string temp = weatherInfo.main.temp.Contains(".") ?
weatherInfo.main.temp.Split(".")[0] : weatherInfo.main.temp;
```

```
        string tempFeelsLike = weatherInfo.main.feels_like.Contains(".")
? weatherInfo.main.feels_like.Split(".")[0] :
weatherInfo.main.feels_like;
        string textString = "Location: " + weatherInfo.name + ", " +
weatherInfo.sys.country + "<br>";
        textString += "Weather: " + weatherInfo.weather[0].main +
"<br>";
        textString += "Description: " +
weatherInfo.weather[0].description + "<br>";
        textString += "Temperature: " + temp + "C, " + "feels like " +
tempFeelsLike + "C<br>";
        textString += "Pressure: " + weatherInfo.main.pressure + "
hPa<br>";
        textString += "Humidity: " + weatherInfo.main.humidity + "
%<br>";
        infoText.text = textString;
}
```

Prvotno se u switch funkciji obavlja izbor 3D efekata koji će se prikazati na modelu ovisno o vraćenom nazivu vremena. Lista koja je postavljena na početku a sadrži sve vremenske objekte ovdje dolazi u korist jer se svi ostali moraju isključiti. Ostatak switch funkcije nije prikazan kako kod ne bi bio pre velik, no djeluje na istom principu. Na posljetku se svi podaci postavljaju na tekstualne elemente prozora. Nakon pozivanja korutine za dohvat prognoze za drugi prozor se poziva funkcija:

```
private void UpdateForecast(Forecast forecast)
    {
        this.forecast = forecast;

        // dealociraj trenutnu listu prognoze koja se prikazuje
        foreach (GameObject forecastObj in forecastScrollableObjects)
        {
            Destroy(forecastObj);
        }
        forecastScrollableObjects.Clear();

        StartCoroutine(ContinueForecast(forecast.list));
    }
```

Ovdje se radi o sličnoj implementaciji skrolabilnog prozora kao i u tablici rezultata u Pocket Basketball aplikaciji. Svi trenutno aktivni objekti se moraju brisati, aktivira se poruka za čekanje rezultata i pokreće se nova korutina koja dohvaća sve ikone za listu dobivenih rezultata. Kako je funkcija velika, prikazati će se samo bitni dijelovi.

```
private IEnumerator ContinueForecast(List<Flist> forecastList)
    {
        // generiraj jedan scrollabilni element po jednom objektu
        foreach (Flist hourlyWeather in forecast.list)
        {
            GameObject forecastClone = Instantiate(forecastObject,
forecastGridCollection.transform);
……
```

Za svaki od objekata u listi se instancira po jedan novi objekt. Originalni objekt se ne koristi i uvijek je prisutan u listu samo je isključen da se ne bi vidio. Dalje se pozivanjem

„UnityWebRequest.GetTexture()" funkcije dobiva ikona sa URL-a uz dodanu oznaku ikone koja je dobivena u zahtjevu prije. Nakon uspješnog poziva se dohvaća tekstura iz dobivenog odgovora te se uz pomoć nje, radi novi Sprite objekt sa funkcijom „Sprite.Create()“. Sprite je 2D grafika i ona se generalno koristi u 2D aplikacijama, no ovdje je pogodna za prikazivanje ikona te ju je moguće postaviti na „SpriteRenderer“ komponentu objekta:

```
Texture2D imgTexture =
((DownloadHandlerTexture)request.downloadHandler).texture;
                Sprite imgSprite = Sprite.Create(imgTexture, new Rect(0,
0, imgTexture.width, imgTexture.height), new Vector2(.5f, .5f));
                sprite.GetComponent<SpriteRenderer>().sprite =
imgSprite;
```

Vrijeme se parsira u čitljiviji format dok se minimalna i maksimalna temperatura zaokružuju:

```
                // parsiraj vrijeme
                DateTime parsedDate;
                if (DateTime.TryParseExact(hourlyWeather.dt_txt,
TIME_FORMAT, null, System.Globalization.DateTimeStyles.None, out
parsedDate))
                {
                    timeText.GetComponent<TextMeshPro>().text =
parsedDate.ToString("g"); //g = 1.1.2022. 18:00
                }

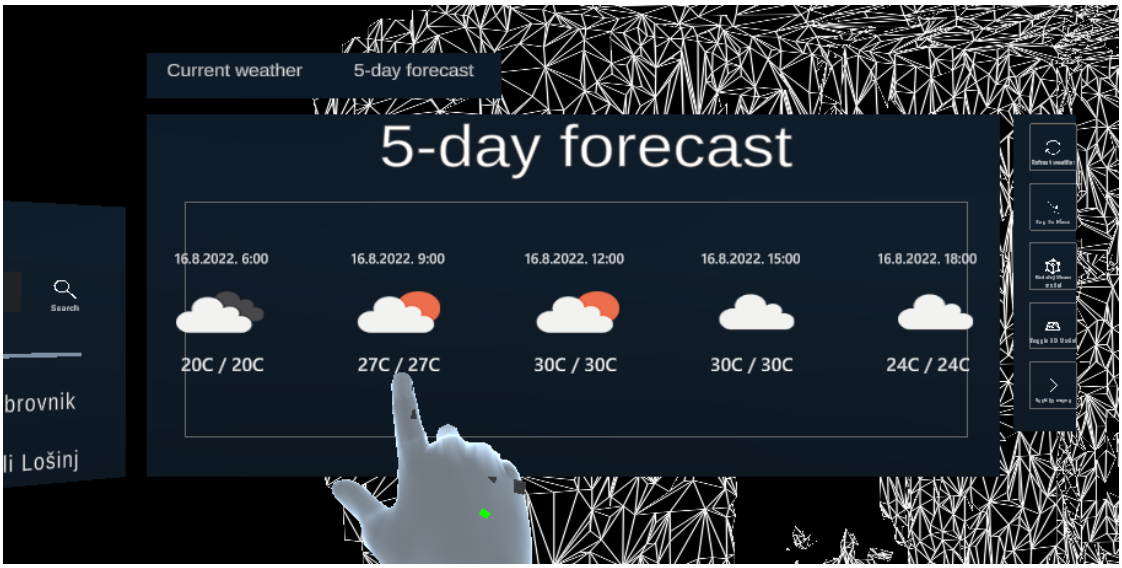
                // parsiraj temperaturu
                string temp_min =
hourlyWeather.main.temp_min.Contains(".") ?
hourlyWeather.main.temp_min.Split(".")[0] : hourlyWeather.main.temp_min;
                string temp_max =
hourlyWeather.main.temp_max.Contains(".") ?
hourlyWeather.main.temp_max.Split(".")[0] : hourlyWeather.main.temp_max;
                temperatureText.GetComponent<TextMeshPro>().text =
temp_min + "C" + " / " + temp_max + "C";
```

Za tekstualne objekte u skrolabilnim kontejnerima je potrebno uključiti cache za materijale kako se ne bi dešavale greške prilikom skrolanja:

```
timeText.GetComponent<MaterialInstance>().CacheSharedMaterialsFromRender
er = true;

temperatureText.GetComponent<MaterialInstance>().CacheSharedMaterialsFro
mRenderer = true;
```

Zatim se takav vremenski objekt stavlja u listu i proces se ponavlja za sve elemente u listi dobivenoj u prijašnjem zahtjevu. Kada se petlja završi, objekti se postavljaju kao aktivni i ažurira se skrolabilni prikaz.



Slika 31: Prikaz funkcionalnosti skrolabilnog prozora sa prognozom [Izvor: vlastita izrada]

Klikom na polje za pretraživanje se pozivom sljedeće funkcije pali sistemska tastatura:

```
TouchScreenKeyboard.Open("", TouchScreenKeyboardType.Default, false,
false, false, false);
```

Radi se o Unity funkciji koja otvara zadanu tastaturu sustava, u ovom slučaju Hololens tastaturu, kojom korisnik unosi naziv grada. Tastaturu nije moguće pozvati u Unity editoru već samo u Hololens Emulatoru pa je uz pretraživanje na istom prozoru postavljeno nekoliko gumba sa nazivima gradova koji se mogu pritisnuti umjesto utipkavanja naziva. Unutar „Update()“ funkcije se zatim svaku sličicu pregledava unos tastature ako je trenutno aktivna te se isti tekst postavlja i u polje pretraživanja. Klikom na gumb pretraživanja se poziva funkcija „SearchCityName()“ koja jednostavno prolazi kroz parsiranu listu gradova iz JSON-a te traži ID grada sa sličnim imenom. Ako je ID nađen, poziva se funkcija za osvježavanje prognoze.

Uz to se poziva i funkcija „GenerateLightning()“ koja generira munje ako je trenutno upaljen 3D model vremena i ako se radi o olujnom vremenu.

```
if (lightningTimer == lightingMaxTimer)
            {
                lightningTimer = 0;
                Random rand = new Random();
                int lightningIndex = rand.Next(1, 4);
                lightingMaxTimer = rand.Next(1000, 4000);

                switch (lightningIndex)
                {
                    case 1:

lightning1Object.GetComponent<LightningBoltScript>().Trigger();

lightning1Object.GetComponent<AudioSource>().Play();
                        break;
```

```
                    case 2:

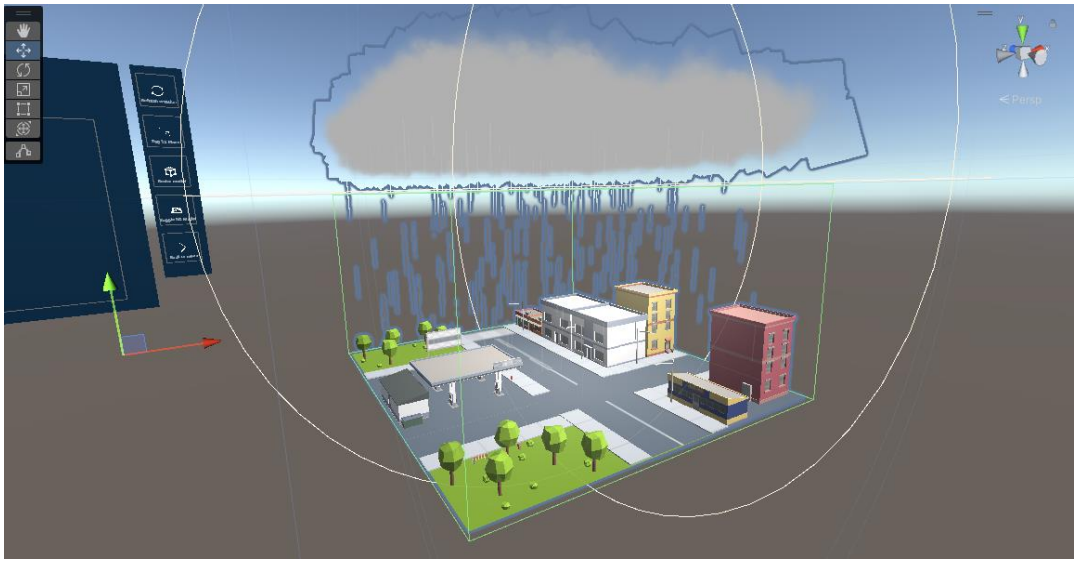
lightning2Object.GetComponent<LightningBoltScript>().Trigger();

lightning2Object.GetComponent<AudioSource>().Play();
                        break;
                    case 3:

lightning3Object.GetComponent<LightningBoltScript>().Trigger();

lightning3Object.GetComponent<AudioSource>().Play();
                        break;
                }
            }
            else
            {
                lightningTimer++;
            }
```

Gore je prikazan glavni isječak funkcije koji generira munju svakih n sličica između 1000 i 4000. Također se nasumično bira koja od tri munje na modelu će se aktivirati te se uz pozivanje „Trigger()“ metode na komponenti skripte „LightningBoltScript“ iz dodatka, pokreće i odgovarajući zvuk. Svi zvukovi su pozicionirani u prostoru zahvaljujući MRTK „Spatial Audio Spatializer“ dodatku. Prilikom aktiviranja gumba za paljenje odnosno gašenje modela, on se automatski pozicionira do prozora s vremenom postavljanjem lokalne pozicije koja je na početku aplikacije spremljena.



Slika 32: Model vremena u Unity editoru [Izvor: vlastita izrada]

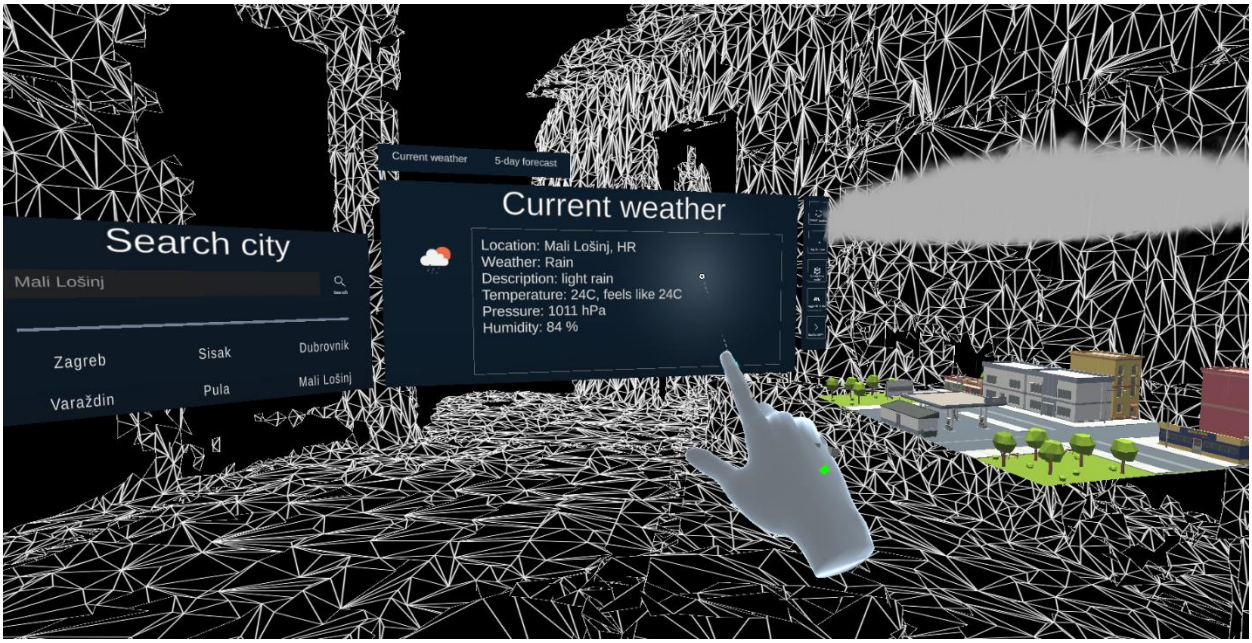
Izbornik sa gumbima ima slične akcije kao i prijašnje aplikacije, odnosno aktiviranje akcije „Tap To Place“ za premještanje prozora, opcija za rotaciju i premještanje modela što uključuje skripte „Object Manipulator“ i „BoundsControl“ te opcija povratka na glavni izbornik. Tu je i opcija osvježavanja vremena koja se može pozvati ručno klikom na gumb u izborniku

ali se poziva i tijekom otvaranja tabova te pretraživanja kako bi se osvježio prikaz prognoze. Ta je funkcija spomenuta nekoliko puta u prijašnjim isječcima koda i ona samo ponovo poziva korutine za dohvat vremena i prognoze.

```
public void RefreshWeather()
    {
        weatherInfo = null;
        forecast = null;

        if (weatherPanel.activeSelf)
        {
            infoText.text = "loading...";
            StartCoroutine(FetchWeather(UpdateWeather));
        }
        if (forecastPanel.activeSelf)
        {
            loadingForecastText.SetActive(true);
            StartCoroutine(FetchForecast(UpdateForecast));
        }
    }
```

Još postoje i funkcije prebacivanja na druge panele, te paljenja i gašenja kontrola premještanja modela koje su gotovo iste kao i kod prijašnjih dijelova aplikacije.



Slika 33: Pokretanje aplikacije Weather 3D [Izvor: vlastita izrada]

S time su prikazani i završeni svi dijelovi aplikacije HoloPlayground. Poveznice na Github repozitorij aplikacije te MRTK i Unity dokumentaciju korištenu u izradi se nalaze u prilozima. Na Github repozitoriju je također navedena lista korištenih besplatnih Unity i audio resursa.

6. Zaključak

Cilj ovog rada bio je prikazati mogućnosti Hololens uređaja i tijek razvoja jedne aplikacije za proširenu, odnosno mješovitu stvarnost. Prije samog razvoja, navedene su bitne značajke Hololens uređaja i mješovite stvarnosti općenito. Kroz razvoj HoloPlayground aplikacije je cilj rada postignut te je implementirano nekoliko mogućnosti MR toolkita od kojih je korišten i multiscenski sustav kako bi se tri aplikacije ukomponirale u jednu. Kroz sve dijelove aplikacije prikazane su mogućnosti korištenja skeniranog prostora, reaktivnosti objekata na isti, postavljanje objekata u prostor, interaktivnost korisnika sa objektima, svjetlosni i grafički elementi aplikacije te UI elementi MR toolkita. Kroz poglavlje postavljanja okoline detaljno je razrađen postupak iz čega je vidljivo da je za razvoj potrebno nekoliko alata, konfiguracija istih te puno računalnih resursa što razvoj za Hololens uređaj ne čini jednostavnim.

Usprkos tome, mogućnosti Hololens uređaja i MR toolkita su brojne, čije primjere u praksi je moguće vidjeti u poljima medicine, edukacije, proizvodnje i logistike pa i u zabavne svrhe što je opisano kroz teorijsku razradu ovog rada. Mnoštvo dostupne dokumentacije i gotovih primjera uvelike je ubrzalo razvoj aplikacije ovog rada te pružilo dobar uvid u razvoj aplikacija mješovite stvarnosti i učinilo cijelo iskustvo poučnim.

Popis literature

[1] P. Milgram, F. Kishino, „A taxonomy of mixed reality visual displays“, 1994. [Na internetu]. Dostupno: https://cs.gmu.edu/~zduric/cs499/Readings/r76JBo-Milgram_IEICE_1994.pdf [pristupano 12.06.2022.].

[2] Microsoft, „Introduction to mixed reality“, 2022. [Na internetu]. Dostupno: https://docs.microsoft.com/en-us/learn/modules/intro-to-mixed-reality/ [pristupano 12.06.2022.].

[3] J. Carmigniani, B. Furht, „Augmented reality: An Overview“, 2011. [Na internetu]. Dostupno: https://www.researchgate.net/publication/227164365_Augmented_Reality_An_Overview [pristupano 12.06.2022.].

[4] R. Carter, „Mixed Reality vs Augmented Reality – The Differences that Matter“, 2021. [Na internetu]. Dostupno: https://www.xrtoday.com/augmented-reality/mixed-reality-vs-augmented-reality/ [pristupano: 12.06.2022.].

[5] USCSchool of Cinema Arts, „Morton Heilig: Inventor VR“, (bez dat.). [Na internetu]. Dostupno: https://www.uschefnerarchive.com/morton-heilig-inventor-vr/ [pristupano 12.06.2022.].

[6] I. E. Sutherland, “A head-mounted three dimensional display”, 1968. [Na internetu]. Dostupno: https://dl.acm.org/doi/pdf/10.1145/1476589.1476686 [pristupano 12.06.2022.].

[7] L. B. Rosenberg, „Virtual Fixtures: Perceptual Tools for Telerobotic Manipulation“, 1993. [Na internetu]. Dostupno: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=380795 [pristupano: 12.06.2022.].

[8] C. Art, R. Grasset, L. Gruber, T. Langlotz, “The History of Mobile Augmented Reality”, 2015. [Na internetu]. Dostupno: https://www.researchgate.net/publication/275974448_The_History_of_Mobile_Augmented_Reality [pristupano: 12.06.2022.].

[9] NMC: The New Media Consortium, „The Horizon Report“, 2005. [Na internetu]. Dostupno: https://www.learntechlib.org/p/182024/ [pristupano 12.06.2022.].

[10] Virtual Reality Society, “What is Virtual Reality?“, (bez dat.). [Na internetu]. Dostupno: https://www.vrs.org.uk/virtual-reality/what-is-virtual-reality.html [pristupano 13.06.2022.].

[11] Virtual Reality Society, “Head-mounted Displays (HMDs)”, (bez dat.). [Na internetu]. Dostupno: https://www.vrs.org.uk/virtual-reality-gear/head-mounted-displays/ [pristupano: 13.06.2022.].

[12] Microsoft, „What is mixed reality?“, 2022. [Na internetu]. Dostupno: https://docs.microsoft.com/en-us/windows/mixed-reality/discover/mixed-reality [pristupano: 13.06.2022.].

[13] Kore, „Displays for Augmented and Virtual Reality“, 2018. [Na internetu]. Dostupno: https://hackernoon.com/displays-for-augmented-and-virtual-reality-2d77b5199a8b [pristupano: 13.06.2022.].

[14] J. Goldman, „Windows Mixed Reality headsets are here and they're affordable“, 2018. [Na internetu]. Dostupno: https://www.cnet.com/tech/computing/windows-mixed-reality-headsets-coming-this-fall-acer-lenovo-dell-hp-asus/ [pristupano: 13.06.2022.].

[15] VRCompare, „Samsung Odyssey+“, (bez dat.). [Na internetu]. Dostupno: https://vr-compare.com/headset/samsungodyssey%2B [pristupano 18.06.2022.].

[16] HP, “Reverb G2 VR Headset”, (bez dat.). [Na internetu]. Dostupno: https://www.hp.com/us-en/vr/reverb-g2-vr-headset.html [pristupano 18.06.2022.].

[17] Wikipedia, „Google Glass“, (bez dat.). [Na internetu]. Dostupno: https://en.wikipedia.org/wiki/Google_Glass [pristupano 18.06.2022.].

[18] Magic Leap, „Magic Leap 1“, (bez dat.). [Na internetu]. Dostupno: https://www.magicleap.com/magic-leap-1 [pristupano 18.06.2022.].

[19] Henry St Leger, „Hands on: Magic Leap One Review“, 2019. [Na internetu]. Dostupno: https://www.techradar.com/reviews/magic-leap-one [pristupano 18.06.2022.].

[20] Microsoft, „Hololens“, (bez dat.). [Na internetu]. Dostupno: https://www.microsoft.com/en-us/hololens/ [pristupano 18.06.2022.].

[21] S. Lang, M. Kota, D. Weigert, F. Behrendt, „Mixed reality in production and logistics: Discussing the application potentials of Microsoft HoloLens“, 2019. [Na internetu]. Dostupno: https://www.sciencedirect.com/science/article/pii/S187705091930122X [pristupano 19.06.2022.].

[22] D. Beyoglu, C. Hursen, A. Nasiboglu, „Use of mixed reality applications in teaching of science“, 2020. [Na internetu]. Dostupno: https://link.springer.com/article/10.1007/s10639-020-10166-8 [pristupano: 19.06.2022.].

[23] C.E. Hughes, C.B. Stapleton, D.E. Hughes, E.M. Smith, „Mixed reality in education, entertainment, and training“, 2005. [Na internetu]. Dostupno:

https://ieeexplore.ieee.org/abstract/document/1528429?casa_token=zAvDbhY8NBUAAAAA:_o_ukE08bhBoF0NYfKo1damzzj9s2h_XSkDcMXNQyRGBCWrxXeJN03MuvcwQZKU33aw5qzBOT9ZYwJw [pristupano: 19.06.2022.].

[24] Wikipedia, „Pokemon Go“, (bez dat.), [Na internetu]. Dostupno: https://en.wikipedia.org/wiki/Pok%C3%A9mon_Go [pristupano: 19.06.2022.].

[25] L. Greenwood, “Mixed Reality vs. Augmented Reality vs. Virtual Reality: Their Differences and Use in Healthcare”, 2021. [Na internetu]. Dostupno: https://www.brainlab.com/journal/mixed-reality-augmented-reality-virtual-reality-differences-and-use-in-healthcare/ [pristupano 19.06.2022.].

[26] L. Gallagher, J. Alford, „Mixed-reality headsets in hospitals help protect doctors and reduce need for PPE“, 2020. [Na internetu]. Dostupno: https://www.imperial.ac.uk/news/197617/mixed-reality-headsets-hospitals-help-protect-doctors/ [pristupano 19.06.2022.].

[27] J. T. Verhey, J. M. Haglin, E. M. Verhey, D. E. Hartigan, „Virtual, Augmented, and Mixed reality applications in orthopedic surgery“, 2019. [Na internetu]. Dostupno: https://onlinelibrary.wiley.com/doi/full/10.1002/rcs.2067?casa_token=HcEhiiFr_NMAAAAA%3AUOM6Q66-458tpzFMihgyZqV6AaZTDfzSoom0af-whKW2h-OQ2FITNOQqjIsLQajdk52uFk9TKEx8pUm1ug [pristupano 19.06.2022.].

[28] A. G. Campbell, K. Santiago, D. Hoo, E. Mangina, „Future mixed reality educational spaces“, 2016. [Na internetu]. Dostupno: https://ieeexplore.ieee.org/abstract/document/7821738?casa_token=3GGTprwN6agAAAAA:4DpsWJBJQfxsbTMoEGUqElv_Q9mHf9bqn53kSAwQegj4-74CTHzrsn5LAKqbwqTPEqoYlm3DnOb-aQ [pristupano 26.06.2022.].

[29] L. Chen, T. W. Day, W. Tang, N. W. John, „Recent Developments and Future Challenges in Medical Mixed Reality“, 2017. [Na internetu]. Dostupno: https://arxiv.org/pdf/1708.01225.pdf [pristupano 26.06.2022.].

[30] J. Kim, „Advertising in the Metaverse: Research Agenda“, 2021. [Na internetu]. Dostupno: https://www.tandfonline.com/doi/full/10.1080/15252019.2021.2001273 [pristupano 26.06.2022.].

[31] Microsoft, “Microsoft Hololens Documentation”, (bez dat.). [Na internetu]. Dostupno: https://docs.microsoft.com/en-us/hololens/ [pristupano 02.07.2022.].

[32] Microsoft, „Microsoft Hololens capabilities and solutions“ (bez dat.). [Na internetu]. Dostupno: https://docs.microsoft.com/en-us/hololens/hololens-commercial-features [pristupano 07.07.2022.].

[33] Microsoft, „Spatial mapping“, (bez dat.), [Na internetu]. Dostupno: https://docs.microsoft.com/en-us/windows/mixed-reality/design/spatial-mapping [pristupano 10.07.2022.].

[34] Wikipedia, “Unity (game engine)”, (bez dat.), [Na internetu]. Dostupno: https://en.wikipedia.org/wiki/Unity_(game_engine) [pristupano 10.08.2022.].

[35] Microsoft, “Choosing your engine”, 2022., [Na internetu]. Dostupno: https://docs.microsoft.com/en-us/windows/mixed-reality/develop/choosing-an-engine?tabs=unity [pristupano 10.08.2022.].

[36] Unity Technologies, “Unity”, (bez dat.), [Na internetu]. Dostupno: https://unity.com/ [pristupano 10.08.2022.].

[37] Microsoft, „Mixed Reality Feature Tool“, 2022., [Na internetu]. Dostupno: https://docs.microsoft.com/en-us/windows/mixed-reality/develop/unity/welcome-to-mr-feature-tool [pristupano 10.08.2022.].

[38] Microsoft, „What is Mixed Reality Toolkit 2?“, 2022., [Na internetu]. Dostupno: https://docs.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk2/?view=mrtkunity-2022-05 [pristupano 10.08.2022.].

[39] Microsoft, „Using the Hololens Emulator“, 2022., [Na internetu]. Dostupno: https://docs.microsoft.com/en-us/windows/mixed-reality/develop/advanced-concepts/using-the-hololens-emulator [pristupano 10.08.2022].

Popis slika

Slika 1. Spektar virtualnog kontinuuma [1] 3
Slika 2. Dijagram mješovite stvarnosti, prema: [2] 4
Slika 3. Damoklov mač [6] 5
Slika 4. Oculus Rift Development kit [8] 7
Slika 5. Uređaji na spektru mješovite stvarnosti [12] 9
Slika 6. Samsung Odyssey+ [15] 11
Slika 7. HP Reverb G2 [16] 11
Slika 8. Google Glass sa okvirima s dioptrijom [17] 13
Slika 9. Magic Leap One [18] 14
Slika 10. MR tehnologije u Orlando znanstvenom centru [23] 17
Slika 11. Kretanje MR trendova kod aplikacija u medicini [29] 20
Slika 12. Microsoft Hololens 2 [31] 22
Slika 13. Primjeri postignuća poduzeća koja koriste Hololens [32] 26
Slika 14: Logo aplikacije [Izvor: vlastita izrada] 30
Slika 15: Hololens 2 emulator sa otvorenim panelom simulacije [Izvor: vlastita izrada] 36
Slika 16: Panel sa 3D prikazom skenirane sobe u stvarnom vremenu [Izvor: vlastita izrada] 36
Slika 17: Instaliranje alata Unity [Izvor: vlastita izrada] 37
Slika 18: Modificiranje dodataka za Visual Studio [Izvor: vlastita izrada] 38
Slika 19: Build postavke Unity projekta [Izvor: vlastita izrada] 39
Slika 20: Mixed Reality Feature Tool obavezni dodaci [Izvor: vlastita izrada] 39
Slika 21: Unity MRTK konfigurator [Izvor: vlastita izrada] 40
Slika 22: Glavne postavke XR dodatka [Izvor: vlastita izrada] 41
Slika 23: Dodatne postavke XR dodatka [Izvor: vlastita izrada] 41
Slika 24: Postavke aplikacije [Izvor: vlastita izrada] 42
Slika 25: Postavke sustava skeniranja prostora [Izvor: vlastita izrada] 43
Slika 26: Postavke multiscenskog sustava [Izvor: vlastita izrada] 44
Slika 27: Prikaz strukture objekata i glavnog izbornika aplikacije [Izvor: vlastita izrada] 46
Slika 28: Postavke kontrolera za Pocket Basketball [Izvor: vlastita izrada] 47
Slika 29: Pokretanje aplikacije Pocket Basketball [Izvor: vlastita izrada] 54
Slika 30: Pokretanje aplikacije AirPaint [Izvor: vlastita izrada] 59
Slika 31: Prikaz funkcionalnosti skrolabilnog prozora sa prognozom [Izvor: vlastita izrada] 66
Slika 32: Model vremena u Unity editoru [Izvor: vlastita izrada] 67
Slika 33: Pokretanje aplikacije Weather 3D [Izvor: vlastita izrada] 68

Popis tablica

Tablica 1. Tehničke specifikacije uređaja Microsoft Hololens 2 23

Prilozi

1. **Github repozitorij aplikacije HoloPlayground:**

   https://github.com/ItsNotACoffee/HoloPlayground

2. **MRTK dokumentacija:**

   https://docs.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk2

3. **Unity dokumentacija:**

   https://docs.unity3d.com/Manual