



УНИВЕРЗИТЕТ У НОВОМ САДУ ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА



УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА
НОВИ САД
Департман за рачунарство и аутоматику
Одсек за рачунарску технику и рачунарске комуникације

ЗАВРШНИ (BACHELOR) РАД

Кандидат: Урош Вишекруна
Број индекса: РА84-2013

Тема рада: Реализација апликације виртуелне стварности за приказ географских мапа

Ментор рада: доц. др Милан Ђелица

Нови Сад, август 2017



КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, РБР:			
Идентификациони број, ИБР:			
Тип документације, ТД:	Монографска документација		
Тип записа, ТЗ:	Текстуални штампани материјал		
Врста рада, ВР:	Завршни (Bachelor) рад		
Аутор, АУ:	Урош Вишекруна		
Ментор, МН:	Милан Ђелица		
Наслов рада, НР:	Реализација апликације виртуелне стварности за приказ географских мапа		
Језик публикације, ЈП:	Српски / латиница		
Језик извода, ЈИ:	Српски		
Земља публиковања, ЗП:	Република Србија		
Уже географско подручје, УГП:	Војводина		
Година, ГО:	2017		
Издавач, ИЗ:	Ауторски репринт		
Место и адреса, МА:	Нови Сад; трг Доситеја Обрадовића 6		
Физички опис рада, ФО: (поглавља/страница/цитата/табела/слика/графика/прилога)	7/33/11/10/16/0/0		
Научна област, НО:	Електротехника и рачунарство		
Научна дисциплина, НД:	Рачунарска техника		
Предметна одредница/Кључне речи, ПО:	Виртуелна стварност, Андроид		
УДК			
Чува се, ЧУ:	У библиотеци Факултета техничких наука, Нови Сад		
Важна напомена, ВН:			
Извод, ИЗ:	У овом раду је реализовано решење апликације виртуелне стварности у оперативном систему Андроид. Апликација омогућава преглед тродимензионалног модела планете Земље који је аугментиран корисним информацијама, попут временске прогнозе и насељености градова. Ове информације апликација добавља коришћењем Веб сервиса.		
Датум прихватања теме, ДП:			
Датум одбране, ДО:			
Чланови комисије, КО:	Председник:	Доц. др Иван Каштелан	
	Члан:	Доц. др Миодраг Ђукић	Потпис ментора
	Члан, ментор:	Доц. др Милан Ђелица	



KEY WORDS DOCUMENTATION

Accession number, ANO:		
Identification number, INO:		
Document type, DT:	Monographic publication	
Type of record, TR:	Textual printed material	
Contents code, CC:	Bachelor Thesis	
Author, AU:	Uroš Višekruna	
Mentor, MN:	Milan Bjelica	
Title, TI:	Realisation of VR application for displaying geographic maps	
Language of text, LT:	Serbian	
Language of abstract, LA:	Serbian	
Country of publication, CP:	Republic of Serbia	
Locality of publication, LP:	Vojvodina	
Publication year, PY:	2017	
Publisher, PB:	Author's reprint	
Publication place, PP:	Novi Sad, Dositeja Obradovica sq. 6	
Physical description, PD: (chapters/pages/ref./tables/pictures/graphs/appendices)	7/33/11/10/16/0/0	
Scientific field, SF:	Electrical Engineering	
Scientific discipline, SD:	Computer Engineering, Engineering of Computer Based Systems	
Subject/Key words, S/KW:	Virtual Reality, Android	
UC		
Holding data, HD:	The Library of Faculty of Technical Sciences, Novi Sad, Serbia	
Note, N:		
Abstract, AB:	This paper presents realization of VR application in Android operating system. Application provides an overview of the three dimensional planet Earth model that is upgraded with useful information, such as weather forecast and population density. This information is supplied by using Web services.	
Accepted by the Scientific Board on, ASB:		
Defended on, DE:		
Defended Board, DB:	President:	Ivan Kaštelan, PhD
	Member:	Miodrag Đukić, PhD
	Member, Mentor:	Milan Bjelica, PhD
		Menthor's sign

Zahvalnost

Zahvaljujem se doc. dr Milanu Bjelici na pomoći, savetima i podršci tokom izrade projekta.

Posebno se zahvaljujem Goranu Stuparu, Dejanu Nađu i Veljku Ilkiću na idejama i stručnim savetima tokom analiziranja problema na koje sam nailazio tokom izrade projekta.

Takođe se zahvaljujem porodici, devojci i svim kolegama na ulepšanim studijama.

SADRŽAJ

1. Uvod.....	1
2. Teorijske osnove.....	3
2.1 Virtuelna stvarnost	3
2.1.1 Istorija	3
2.1.2 Primena	4
2.2 Računarska grafika.....	6
2.2.1 Grafičko korisničko okruženje	6
2.2.2 3D računarska grafika	6
2.2.3 OpenGL ES	7
2.2.4 LibGDX	7
2.3 Android platforma.....	8
2.3.1 Istorija	8
2.3.2 Korisnički interfejs	8
2.3.3 Operativni sistem.....	8
2.3.4 Softverski stek	9
2.4 Radno okruženje	10
2.4.1 Android studio	10
2.4.2 C-More framework	11
2.4.2.1 Pozicija C-More razvojnog okruženja u Android aplikacijama	11
2.4.2.2 Arhitektura softvera	12
3. Koncept rešenja	14
3.1 Analiza zadatka.....	14
3.2 Koncept aplikacije i dizajn	14
3.3 Modul za vremensku prognozu	17
3.4 Modul za informacije o populaciji.....	18
3.5 Keširanje podataka.....	19
4. Programsко rešenje.....	20

4.1	Opis programa	20
4.2	Spisak modula i fajlova koji ih opisuju.....	20
4.3	Modul za inicijalizaciju sveta.....	20
4.4	Modul za vremensku prognozu	21
4.5	Modul za populaciju.....	23
4.6	Crtanje i mapiranje gradova	24
4.7	Provider	25
5.	Rezultati	28
5.1	Testiranje	30
6.	Zaključak.....	32
7.	Literatura	33

SPISAK SLIKA

Slika 1. Prikaz VR seta za igranje igara	5
Slika 2. Prikaz operativnog sistema Android.....	10
Slika 3. C-More pozicija u Android softverskom steku	12
Slika 4. Glavni moduli okruženja C-More	13
Slika 5. – Predstava modela kada je kamera udaljena od modela.....	15
Slika 6. Prikaz modela kada se kamera nalazi na poziciji (0.0.0)	15
Slika 7. Slika modela kada se kamera nalazi u tački (0,0,0) ali je fokus pomeren ka dole čime se vidi da je korisnik zapravo unutar modela	16
Slika 8. Na slici su crvenim tačkama prikazani neki od gradova iz baze podataka	16
Slika 9 Blok dijagram sa prikazom modula i njihovog funkcionisanja	17
Slika 10. Prikaz informacija o vremenskoj prognozi	18
Slika 11. Prikaz izgleda baze podataka	18
Slika 12. Slika sa informacijom o populaciji	19
Slika 13. Prikaz formula za dobijanje koordinata	24
Slika 14. Formula za računanje fokusa kamere	24
Slika 15. Prikaz okruženja za dobijanje informacija o potrošnji memorije.....	29
Slika 16. Prikaz opreme za testiranje aplikacije	30

SPISAK TABELA

Tabela 1 Prikaz koji fajlovi programskog koda implementiraju pojedinačne module....	20
Tabela 2 API funkcije <i>start()</i>	21
Tabela 3 API funkcije <i>readUrl()</i>	22
Tabela 4 API funkcije <i>read()</i>	22
Tabela 5 API funkcije <i>readCities()</i>	25
Tabela 6 API funkcije <i>getPopulation()</i>	26
Tabela 7 API funkcije <i>getWeatherInfo</i>	26
Tabela 8. Prikaz potrošnje memorije.....	29
Tabela 9. Prikaz smanjenja broja frejmova u sekundi sa povećanjem učitanih gradova	29
Tabela 10. Testni slučajevi	31

SKRAĆENICE

API – *Application Programming Interface* - Programska sprega

CLI – *Command Line Interface* – Okruženje komandne linije

FPS – *Frames Per Second* - Broj slika u sekundi

GUI – *Graphical User Interface* - Grafički korisnički interfejs

IDE - *Integrated Development Environment* - Integrisano razvojno okruženje

LCD - *Liquid-crystal display* - Ekran zasnovan na tehnologiji tečnih kristala

MIT - *Massachusetts Institute of Technology*

OpenGL ES – *Open Graphics Library for Embedded Systems* - Otvorena grafička biblioteka za razvoj ugrađenih uređaja

PC – *Personal computers* - Lični računar

UI – *User Interface* – Korisnički interfejs

URL - *Uniform Resource Locator* – Jedinsveni lokator resursa

VR – *Virtual reality* - Virtuelna stvarnost

3D - *Three-dimensional space* – Trodimenzionalni prostor

1. Uvod

Upotreba tehnologije virtuelne stvarnosti (*Virtual Reality - VR*) je danas u intenzivnom porastu. Proizvođači uređaja potrošačke elektronike omogućavaju nove VR funkcionalnosti, te se povećava i potreba za razvojem novih rešenja od strane programera. Zadatak rada je realizacija jedne aplikacije virtuelne stvarnosti, koja prikazuje informacije o vremenskoj prognozi u odabranim gradovima, kao i informacije o trenutnoj populaciji u gradovima u realnom vremenu, primenom VR tehnologije. Potrebno je omogućiti lako proširenje aplikacije dodatnim podacima o gradovima.

Veliki izazov pri izradi rada jeste realizacija aplikacije sa korisnički prihvatljivim vremenom odziva, i da aplikacija ne zasiće korisnika prevelikim brojem informacija u jednom trenutku. Samim tim potrebno je pronaći pravu meru informacija koje će biti prikazane da bi rad aplikacije bio optimalan u pogledu količine prikazanih podataka i da bi odziv bio zadovoljavajući. Takođe potrebno je omogućiti što realniji prikaz zbog poboljšanja VR iskustva, što se posebno odnosi na sloj augmentacije koji treba da bude usklađen sa pomeranjem pogleda kamere u VR okruženju a da ne dolazi do izobličenja modela ili slika.

Rad je podeljen u 7 poglavlja:

- ❖ Uvod daje kratak opis i motivaciju rada, i predstavlja kratak pregled rada;
- ❖ Teorijske osnove – daju uvid u korišćeno okruženje za razvoj (Android), kao i u koncepte virtualne stvarnosti i okruženja;
- ❖ Koncept rešenja –predlog rešenja;
- ❖ Programsко rešenje – detaljan opis programskog rešenja i funkcionalnih modula;
- ❖ Rezultati – opis i rezultati ispitnih slučajeva;
- ❖ Zaključak – kratak pregled onoga što je urađeno u radu, predlog mogućih unapređenja;
- ❖ Literatura – pregled korišćene literature tokom izrade rada.

2. Teorijske osnove

U ovom poglavlju dat je uvod u virtuelnu stvarnost, objašnjene su osnove kompjuterske grafike i dat je opis okruženja za razvoj grafičke korisničke sprege na kome je zasnovana ova aplikacija, operativni sistem Android, radno okruženje za razvoj aplikacije kao i grafičko okruženje C-More.

2.1 Virtuelna stvarnost

Kada se kaže virtuelna stvarnost misli se na svet generisan od strane računara sa kojim korisnik može da vrši interakciju. Svet virtuelne stvarnosti je trodimenzionalno virtuelno okruženje u kojem korisnik može da manipuliše delovima tog sveta i da ima osećaj da se stvarno u njemu nalazi. Pored vizuelnog iskustva u nekim slučajevima virtuelne stvarnosti korisnik može da ima dodatna zvučna ili dodirna čulna iskustva.

2.1.1 Istorija

Tačno poreklo virtualne stvarnosti je sporno, delimično zbog toga što je teško tačno formulisati šta je to virtualna stvarnost. Elementi virtuelne stvarnosti se pojavljuju 1860-ih, kada je francuski pisac Antonen Arto zauzeo stav da se iluzija ne razlikuje od realnosti, zalažući se da gledaoci predstava treba da posmatraju predstavu kao stvarnost a ne nešto što je izmišljeno. Moderni koncept virtuelne stvarnosti prvi put se pojavljuje u pri povetci naučne fantastike. Stenli G. Weinbaum je 1935. napisao kratku priču u kojoj opisuje virtualnu stvarnost baziranu na naočarima sa holografskim snimanjem, uključujući miris i dodir.

Prvi mehanički uređaj koji je imao svojstva slična uređajima virtuelne stvarnosti je Senzorama koju je napravio Morton Hajlig 1950. godine. Senzorama je predstavljala mini bioskop koji pored puštanja filma angažuje više čula (vid, zvuk, miris i dodir). Nakon Senzorame pojavljuje se takozvani Damaklov mač (*Sword of Damocles*) tj. prvi set za

virtuelnu stvarnost koji podseća na današnje setove. Njegova glavna karakteristika je to što se postavlja na glavu korisnika, ali najveća mana je njegova težina i konstrukcija koja je morala da bude zakačena za plafon, što umanjuje VR iskustvo. Nakon toga *SEGA* i *Nintendo* prave svoje VR setove uglavnom za potrebe igranja video igara i zabave korisnika. 2010. godine Palmer Laki konstruiše prvi Okulus Rift i tako nastaje moderni VR set.

Do 2016. godine postojalo je 230 kompanija koje su se bavile razvojem VR tehnologije. Sam Fejsbuk ima preko 400 zaposlenih koji se bave ovim razvojem.

Sony je ove godine podneo zahtev za patentom gde predstavljaju prvi bežični VR set.

2.1.2 Primena

VR tehnologija je doživela pravu eksploziju u poslednjih par godina, potrebe za VR-om su sve veće a primena VR tehnologije varira od primene za samo zadovoljstvo korisnika do situacija u kojima VR spašava ljudske živote, neke od primene VR tehnologije su:

Edukacija i trening:

VR se koristi za učenje u virtuelnom okruženju gde učenici mogu da razvijaju svoje veštine bez posledica realnog sveta.

Kontrolisanje bola:

VR može da ometa centar bola kod ljudi sa akutnim bolom i time smanjuje bol.

Zdravstvo i klinički tretmani:

Zdravstvo bi mogao biti jedan od narednih tržišta VR tehnologija. Neki VR uređaji se već koriste u terapijama i rezultati su značajni. [1]

Tretiranje anksioznog poremećaja:

VR se koristi za lečenje poremećaja anksioznosti kao što je post-traumatski stresni poremećaj i fobije. Studije su pokazale da pacijenti koji kombinuju VR sa drugim metodama lečenja iskuse smanjenje simptoma. U nekim slučajevima dolazi do potpunog oporavka pacijenta.

Pripreme za put u svemir:

NASA koristi VR već 20 godina. Najznačajnija upotreba VR je za vreme treniranja astronauta dok su još uvek na Zemlji. Neki tipovi treninga su simulacija izloženosti beztežinskom stanju, simulacija rada sa nekim alatima u svemiru.

Simulacije leta:

Upotrebom VR uređaja obuka pilota je mnogo jednostavnija i jeftinija. Postoji cela simulacija leta svih kritičnih situacija u kojima pilot može da se nađe bez ikakvog rizika po njegov i život drugih ljudi. Pored simulacije leta postoje i simulacije vožnje raznih vojnih vozila koja nisu sigurna i jednostavna za upotrebu. Realnost situacije u kojoj piloti i vojnici mogu da se nađu za vreme simulacije je omogućena VR tehnologijom.

Video igre:

Nakon rasprostranjenja komercijalnih VR setova u svetu video igara su videli da to mogu da okrenu u svoju korist i da maksimalno iskoriste ovu tehnologiju tako što će svoje video igre prilagoditi za VR. Korišćenjem VR seta za vreme igranja neke igre korisnik postaje deo igre i aktivno učestvuje pomerajući delove svog tela i time kontroliše pomeranje svog karaktera u samoj video igri. Izgled jednog VR seta za igranje igara može da se vidi na slici 1.



Slika 1. Prikaz VR seta za igranje igara

Kinematografija i zabava:

Korišćenje VR seta za vreme gledanja filma koji je napravljen za to dozvoljava posmatraču pogled od 360 stepeni za vreme svake scene, što stvara osećaj kao da je korisnik deo filma koji gleda i time se povećava zadovoljstvo korisnika.

Neki zabavni parkovi rade sa VR tehnologijom, simulirajući razne akrobacije i zanimljivosti za njihove posetioce.

Ostale primene VR tehnologije: umetnost, inženjerstvo, arheologija, arhitektura, muzika, marketing i tako dalje. [2]

2.2 Računarska grafika

Računarska grafika je polje vizuelnog računarstva gde se pomoću računara stvara slika. Ona predstavlja stvaranje, obradu, doradu, modelaciju i vizuelizaciju sintetičkih slika, kao i animaciju u dve ili tri dimenzije uz pomoć računara i računarskih programa koji su namenjeni za tu svrhu. Računarska grafika danas ima široku primenu u nauci, industriji, arhitekturi i dizajnu. [3]

U ovom radu pojam *model* predstavlja virtualni objekat definisan skupom tačaka koji se može posmatrati iz različitih pravaca, presecati, uobičavati i slično. On sadrži određeni skup podataka. Ti podaci mogu sadržati podatke o geometriji, tački gledišta, teksturi i podatke o osvetljenju. Engleska reč “*Render*” je postupak stvaranja slike od nekog modela uz pomoć posebnog programa. Takođe se koristi kod opisivanja procesa izračunavanja specijalnih efekata u obradi videa. Ovaj proces se koristi kod računara i računarskih igara, filmova, simulatora kao i u vizualizaciji dizajna.

2.2.1 Grafičko korisničko okruženje

Grafičko korisničko okruženje (*Graphical user interface – GUI*) je softversko okruženje koje omogućava korisniku komunikaciju sa kompjuterom. Najzaslužniji za njegov razvoj su razvoj *Apple Machintosh* i *Microsoft Windows* operativnih sistema. Da bi korišćenje računara bilo jednostavno za sve korisnike grafičko korisničko okruženje je zamenilo okruženje komandne linije (*Command line interface - CLI*) koje koristi samo tekst i rukovanje isključivo preko tastature. Kod grafičkog korisničkog okruženja se pored tastature za rukovanje računarom koriste miš, kugle za upravljanje, pokazivačke ploče. Ikonica je mala slika ili simbol koja u grafičko korisničkom okruženju predstavlja program, priključeni uređaj, direktorijum ili fajl. Prednost grafičkog korisničkog okruženja je što olakšava korišćenje računara (npr. Prebacivanje fajla iz jednog direktorijuma u drugi je moguće izvršiti klikom na fajl I njegovim prevlačenjem u drugi direktorijum, dok bi preko komandne linije bilo potrebno poznavanje dugačke komande da bi se ovaj postupak obavio). [4]

2.2.2 3D računarska grafika

3D računarska grafika je predstavljanje trodimenzionalnog prostora i trodimenzionalnih oblika u dvodimenzionalnom prostoru ekrana. To se postiže tako što se dvodimenzionalnom

pikselu ekrana koji je određen pozicijom (x, y), bojom, i osvetljenošću dodeli nova osobina, a to je pozicija na imaginarnoj z osi. Time se dobija da dvodimenzionalni piksel postaje trodimenzionalni, pošto njegovu poziciju sada određuju tri koordinate (x, y, z). Kada se formira više trodimenzionalnih piksela, rezultat je trodimenzionalna površina, koja se drugačije naziva tekstura (*texture*). Neke trodimenzionalne računarske grafike u zavisnosti od snage računara dozvoljavaju senčenje objekata što stvara realističniji izgled kompjuterski napravljenih oblika i modela.

2.2.3 OpenGL ES

OpenGL je programska sprega (*Application Programming Interface - API*) koja služi za renderovanje dvodimenzionalne i trodimenzionalne vektorske grafike. *API* se obično koristi za interakciju sa *GPU* da bi se postiglo hardverski ubrzano renderovanje.

OpenGL Embedded System (*OpenGL ES*) je verzija *OpenGL-a* koja omogućava potpuno funkcionalnu dvodimenzionalnu i trodimenzionalnu grafiku, a dizajnirana je za ugrađene (*embedded*) sisteme kao što su pametni telefoni, tablet uređaji, igračke konzole i lični digitalni asistenti. *API* je raspoloživ na različitim platformama i podržan od strane različitih jezika. Ukoliko je potrebno više kontrole prilikom iscrtavanja na ekran ili prilikom postavljanja objekata u 3D grafici, standardni alati nisu dovoljni. *OpenGL ES API* obezbeđen od strane Androida framework-a nudi skup alata za prikaz grafike. [5][6]

2.2.4 LibGDX

LibGDX je razvojno okruženje (*Framework*), u vidu biblioteke, za razvijanje igara napisan u programskom jeziku Java, sa primesama C i C++ komponenti za optimizacije određenih komponenti. Omogućava implementacije na više platformi (*cross-platform*) tako da je moguće razvijati igre za desktop računare (*Windows, Linux, Mac OS X*) kao i za mobilne platforme (*Android, iOS*) a i za web bazirane platforme oslanjajući se na *WebGL*. *LibGDX* je okruženje otvorenog koda pod licencem *Apache 2.0*, što u suštini znači da onaj ko želi da kopira, modifikuje, ili koristi izvorni kod za neki drugi projekat, ima to pravo, pod uslovom da njegov projekat mora da bude licenciran pod istom tom licencem. Ovo se odnosi na izvorni kod, a ne i za korišćenje same biblioteke.

Pored aplikativnog interfejsa koji biblioteka nudi, *LibGDX* pruža i direktni pristup fajl sistemu, ulazno izlaznim uređajima kao i *OpenGL* interfejsu, i time omogućava korisniku biblioteke mogućnost optimizacije na niskom nivou. [7]

2.3 Android platforma

Android je trenutno najrasprostranjeniji operativni sistem za pametne telefone. Popularan je i među kompanijama koje zahtevaju gotove, jeftine, prilagodive i lage operativne sisteme za svoje visokotehnološke uređaje. Pored toga što je prvenstveno namenjen za pametne telefone i tablet uređaje, Android je našao primenu i kod televizora, igračkih konzoli, digitalnih kamera i druge elektronike. Takođe je našao i primenu i u računarima u automobilu.

Zbog sve veće potrebe za trodimenzionalnim aplikacijama, Android se proširuje sa *OpenGL ES*, *libGDX* kao i sa *C-More* tehnologijama koje baš to omogućavaju. VR tehnologija je za sada bazirana na Android operativnom sistemu, upravo zbog toga što su uređaji koji rade na ovom operativnom sistemu mali i lako prenosivi i samim tim pogodni da se koriste u VR setovima.

2.3.1 Istorija

Android je mobilni operativni sistem kompanije Gugl zasnovan na Linuks jezgru, prvenstveno dizajniran za mobilne uređaje sa ekranom osetljivim na dodir, kao što su pametni telefoni i tablet uređaji. Android je razvila istoimena kompanija (*Android Inc.*) koju je kompanija Gugl finansijski podržavala, a kasnije i kupila 2005. godine. Android je predstavljen 2007. godine zajedno sa osnivanjem udruženja Open Handset Alians (*Open Handset Allianse - OHA*), konzorcijuma hardverskih, softverskih i telekomunikacionih kompanija posvećenih razvoju otvorenih standarda za mobilne uređaje. Prvi Android telefon je prodat u oktobru 2008. godine i to je bio *T-Mobile G1* kompanije *HTC*.

2.3.2 Korisnički interfejs

Korisnički interfejs Android uređaja je zasnovan na direktnoj manipulaciji objektima na ekrantu, korišćenjem ulaza u vidu dodira koji odgovaraju pokretima u realnom svetu. Dodatni hardver, kao što su akcelerometar, žiroskop i senzor blizine, se koristi za dodatne zahteve korisnika, kao na primer podešavanje orijentacije ekrana u zavisnosti od položaja uređaja. Android dozvoljava korisnicima slobodno uređivanje početnih ekrana prečicama do aplikacija, što omogućava prikazivanje sadržaja uživo, kao što su informacije vezane za elektronsku poštu ili vremensku prognozu. Aplikacije mogu dalje slati obaveštenja korisniku, sa relevantnim informacijama o , na primer, pristigloj SMS poruci ili elektronskoj pošti.

2.3.3 Operativni sistem

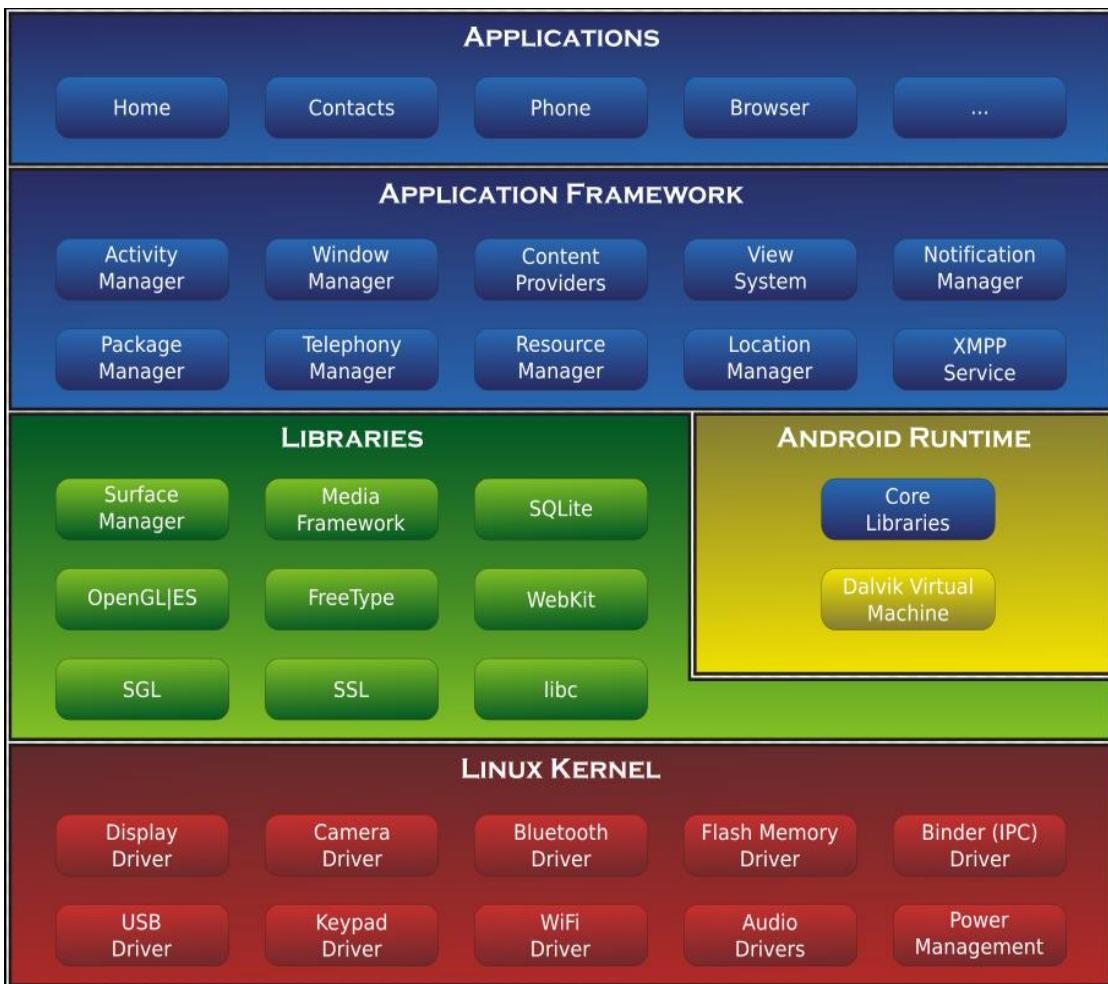
Android je operativni sistem otvorenog koda dostupan pod Apači licencom(*Apache License*). Ona dopušta slobodnu izmenu i distribuciju softvera od strane proizvođača uređaja,

telekomunikacionih operatera i programera entuzijasta. I pored toga većina Android uređaja dolazi sa dodatnim komercijalnim softverom. Android takođe, ima veliku zajednicu programera aplikacija, koje proširuju funkcionalnost uređaja i najčešće se razvijaju u programskom jeziku Java. U oktobru 2012. godine, bilo je dostupno približno 700 000 aplikacija za android platformu, dok je broj preuzimanja aplikacija sa Androidove prodavnice aplikacija Gugl plej (*Google play*), oko 25 milijardi. Istraživanje koje je sprovedeno među programerima u periodu april – maj 2013. godine, pokazalo je da je Android najpopularnija platforma kod 71% programera. Danas na Gugl plej prodavnici može da se preuzme 3 044 893 različitih aplikacija. [8]

2.3.4 Softverski stek

Na vrhu linuks jezgra nalazi se posredni softver (*Middleware*), biblioteke i aplikativni programski interfejs (*API*) napisani u programskoj jeziku C, kao i aplikativni softver pokrenut na aplikacionom okviru(engl. Application framework) koji sadrži Java-kompatibilne biblioteke bazirane na Apači Harmoniji (*Apache Harmony*).Android koristi Dalvik virtualnu mašinu na kojoj izvršava tzv. deks-kod(*Deh, Dalvik Executable*), koji se najčešće prevodi iz Java bajtkoda. Android 5.1 ne radi na Dalviku, već na ART virtuelnoj mašini.

Android koristi Bionic biblioteku umesto standardne C biblioteke, koju je svojevremeno Gugl razvio za Android, kao derivaciju BSD standardne C biblioteke koda. Bionic poseduje nekoliko osnovnih funkcionalnosti specifičnih za Linuks jezgro i njegov razvoj se nastavlja nezavisno od izvornog koda Androida. Glavne prednosti korišćenja Bionic biblioteke umesto *GNU C* biblioteke je drugačije licenciranje i optimizacija za procesore sa manjom frekvencijom. Slojevi Androida mogu da se vide na slici 2.



Slika 2. Prikaz operativnog sistema Android

2.4 Radno okruženje

U ovom delu rada biće opisani alati i radni okviri koji su korišćeni za izradu aplikacije.

2.4.1 Android studio

Android studio je zvanično integrисано razvojно okružење (*IDE, Integrated Development Environment*) за razvoј Android aplikacija. Pored snažног kod editora i raznih razvojnih alata, Android studio nudi još raznih dodataka koji poboljшавају produktivност програмера u izradi Android aplikacija.

Neki od tih dodataka su:

- Brz emulator sa mnogo dodataka;
- Ujedinjeno okružење u kom se razvijaju program za sve Android uređaje;
- Proširiv framework i alati za testiranje.

2.4.2 C-More framework

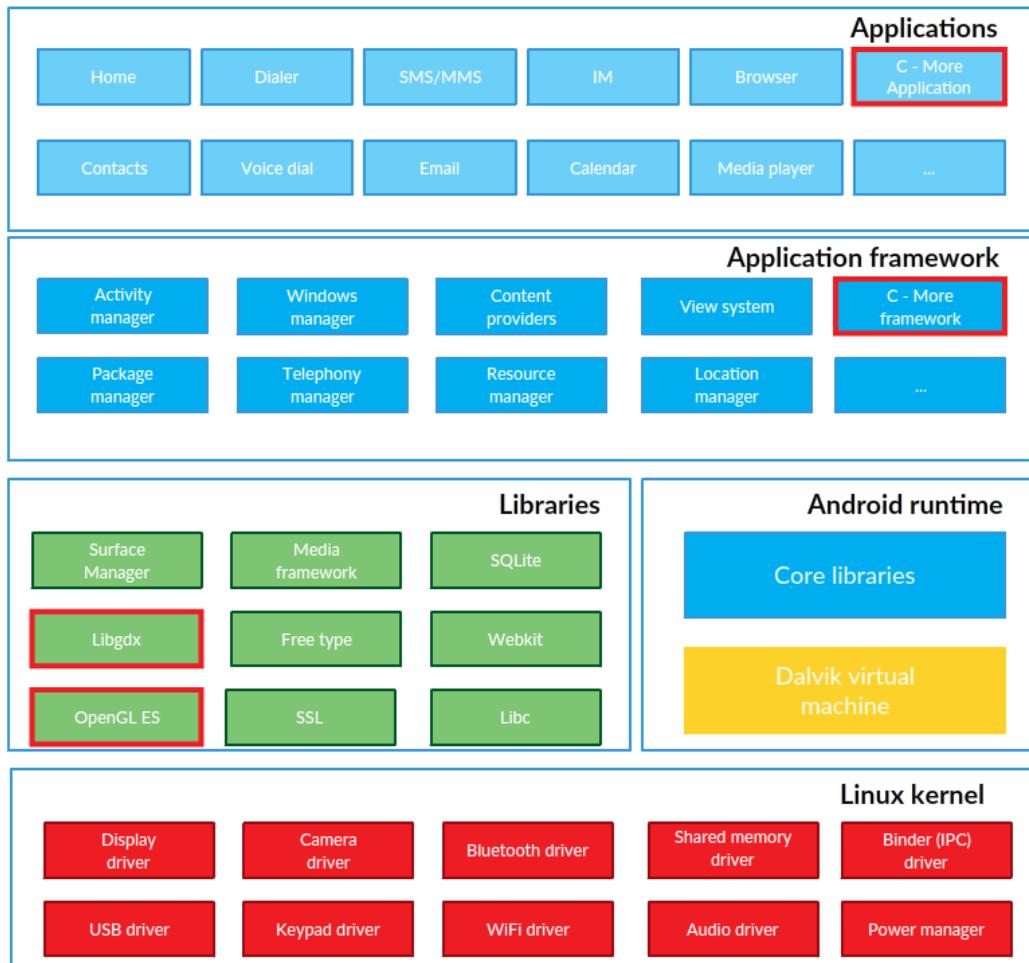
C-more razvojno korisničko okruženje za razvoj grafičke korisničke sprege je 3D Java razvojno okruženje za razvoj grafičke korisničke sprege. Okruženje je raspoloživo na različitim platformama bazirano na *OpenGL* standardu, koja radi na Windows, Linux, Mac, OS X, Android, iOS platformama kao i na pregledačima interneta koji podržavaju *WebGL*. *C-more* razvojno okruženje je razvijano u okviru Istraživačko-razvojnog instituta RT-RK.

2.4.2.1 Pozicija C-More razvojnog okruženja u Android aplikacijama

C-More softver je integriran u Android-u na različitim nivoima:

- Aplikacioni nivo: sadrži aplikacije napisane koristeći Android *API*
- Aplikaciono okruženje: sadrži Java biblioteke koje podržavaju Android aplikacije. Ovaj nivo sadrži C-More razvojno okruženje.
- Nativne biblioteke: sadrže nativne biblioteke koje podržavaju Android aplikaciono okruženje. Ovaj nivo sadrži *libGDX* i *OpenGL ES* (standardna Android biblioteka) koje koristi C-More razvojno okruženje.

Grafički prikaz pozicije *C-More* razvojnog okruženja u Android softveru se nalazi na Slika 3.



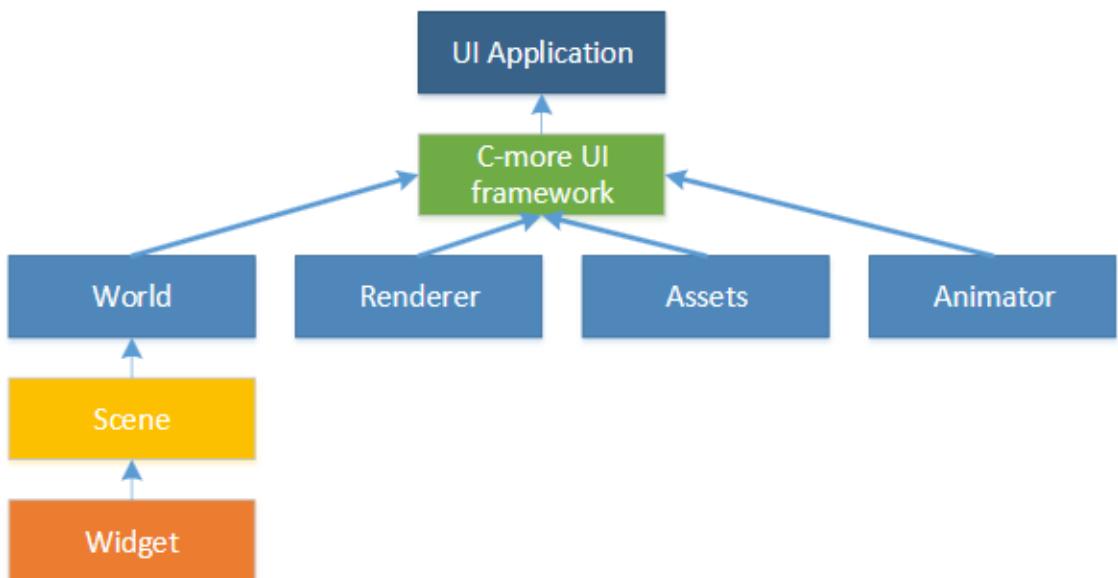
Slika 3. C-More pozicija u Android softverskom steku

2.4.2.2 Arhitektura softvera

C-More razvojno okruženje se koristi za razvijanje 3D *UI* aplikacija na raznim platformama kao što su PC i Android. Prednosti *C-More* razvojnog okruženja su da programeri mogu da naprave veoma napredne 3D *UI* aplikacije pisane u Javi, koristeći sve prednosti OpenGL-a koje omogućavaju sjajne preformanse.

C-More UI razvojno okruženje za razvoj grafičke strukture je usko povezan sa *libGDX* razvojnim okruženjem za razvoj grafičke strukture koji je podržan od strane različitih platformi i približava *API* za Android platformu.

Na Slika 4 su prikazani glavni moduli *C-More* razvojnog okruženja za razvoj grafičke korisničke sprege.



Slika 4. Glavni moduli okruženja C-More

UI aplikacije se razvijaju u Javi za ciljanu platform. Aplikacije za grafičko renderovanje koriste API koji je izведен iz modula *C-More* grafičkog okruženja.

C-More framework sadrži četiri glavna modula:

- Modul *Svet* (*World module*): ovaj modul je mesto gde su postavljene sve *UI* aplikacione komponente. Svet je napravljen od scena u 3D svetu od različitih grafičkih elemenata. Scene mogu biti predstavljene kao nezavisne grafičke celine unutar aplikacije. Ovaj modul sadrži kameru koja omogućava korisniku da se kreće kroz 3D svet;
 - Scene mogu da sadrže različite grafičke elemente (slika, tekst, lista, dugme) i oni mogu biti postavljeni relativno svuda unutar 3D sveta. Ovo omogućava programerima da kreiraju napredne i interesantne efekte.
- Modul za iscrtavanje (*Renderer module*): Ovaj modul je odgovoran za sve što se pojavljuje na ekranima uređaja;
- Modul za resurse (*Assets module*) je odgovoran za rukovođenje svim neophodnim resursima aplikacije (na primer tekstura, zvuk, 3D modeli). Modul za resurse efikasno koristi memoriju što je veoma bitno kada se radi o uređajima sa nedostatkom memorije i procesne snage;
- Animator je odgovoran za animacije objekata. Animator može da animira poziciju, rotaciju, faktor skaliranja, transparentnost kao i ručno napravljene animacije.[9][10]

3. Koncept rešenja

U ovom poglavlju predstavljena je analiza problema i uslovi koje aplikacija treba da zadovolji. Izloženi su glavni zahtevi koje aplikacija treba da zadovolji i predstavljene su slike rešenja.

3.1 Analiza zadatka

Ova aplikacija je zamišljena kao VR aplikacija, sa osnovnom ulogom da prikazuje augmentaciju geografske mape u realnom vremenu. Ova augmentacija se sastoji od informacija vezanih za vremensku prognozu i populaciju stanovnika po gradovima. Pored ovih informacija aplikacija može da prikazuje različite druge geografske informacije. Sve informacije je potrebno prikazati na modelu planete Zemlje. Planeta je prikazana kao sferni 3D model koji je napravio tim dizajnera u okviru Instituta RT-RK. Potrebno je obezbediti da korisnik ima utisak da se nalazi u središtu modela i pogledom tj. pomeranjem glave pomera kameru i time određuje na koji deo planete želi da gleda. Kako korisnik pomera kameru tako se iscrtavaju gradovi za taj deo planete na koji je fokus kamere postavljen. Postavljanjem fokusa kamere tačno iznad nekog od gradova pojavljuju se informacije o tom gradu, vezane za vremensku prognozu ili za populaciju. Pored prikazivanja tih informacija napravljeno je iscrtavanje svakog grada iz baze podataka na osnovu njegove geografske širine i dužine.

3.2 Koncept aplikacije i dizajn

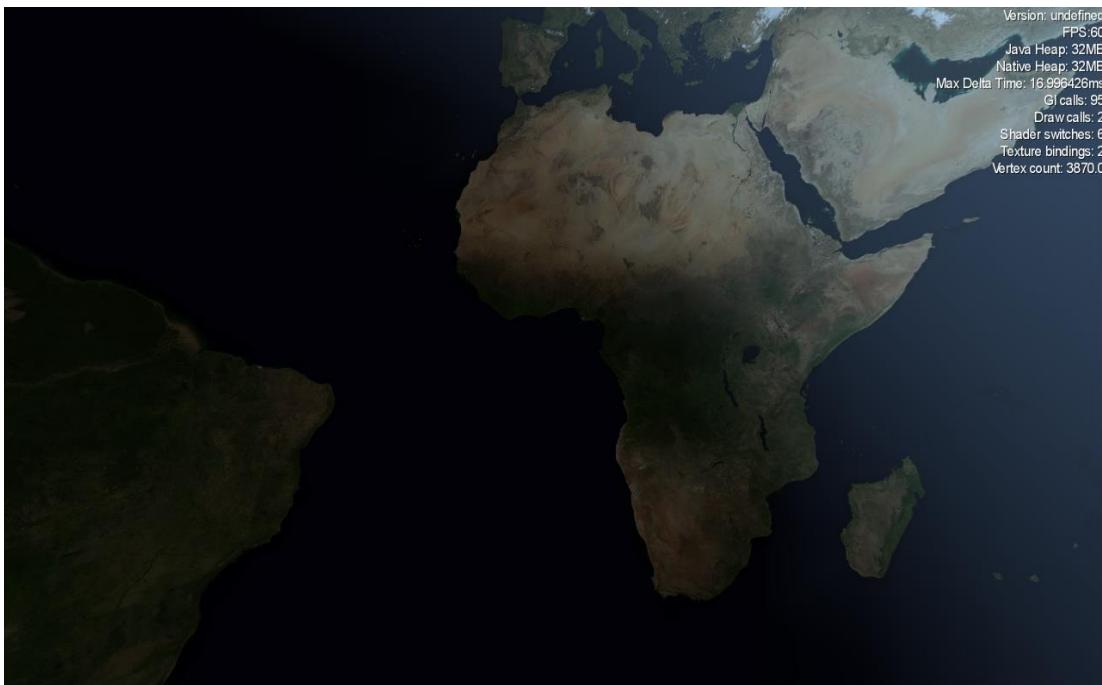
Dizajn ove aplikacije predstavlja sferni 3D model planete koji se nalazi u središtu same aplikacije i sve informacije koje zanimaju korisnika se nalaze unutar tog modela u okviru kojeg je postavljen i pogled korisnika. Potrebno je podesiti poziciju ambijentalnog osvetljenja čija uloga je da imitira položaj Sunca u našem solarnom sistemu. Glavni zahtev je da gradovi budu precizno mapirani unutar planete. Zahvaljujući pogledu koji je postavljen unutar

modela, slika kontinenata koja je zapepljena na 3D sferni model mora da bude invertovana i prilagođena aplikaciji.

Izgled modela, i aplikacije bez prikaza gradova prikazani su na slikama 6, 7 i 8. .



Slika 5. – Predstava modela kada je kamera udaljena od modela



Slika 6. Prikaz modela kada se kamera nalazi na poziciji (0.0.0)



Slika 7. Slika modela kada se kamera nalazi u tački (0,0,0) ali je fokus pomeren ka dole čime se vidi da je korisnik zapravo unutar modela

Nakon implementacije modela i njegovog pozicioniranja u 3D prostoru, unutar njega prvo bitno su mapirani svi gradovi iz baze podataka. To može da se vidi na Slika 8.

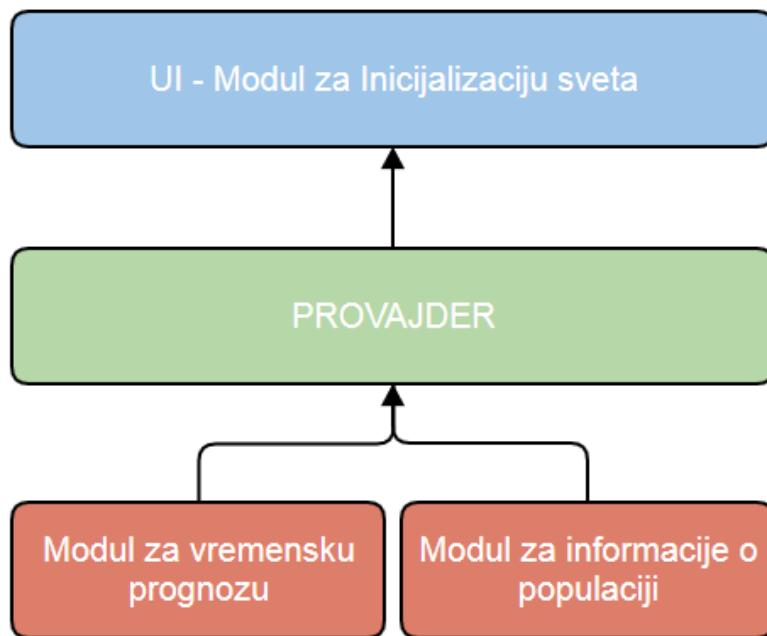


Slika 8. Na slici su crvenim tačkama prikazani neki od gradova iz baze podataka

Kao što se vidi na slici jedan od većih problema dizajna je bio da se tačno mapiraju svi gradovi unutar sfere, iz razloga što je prečnik iscrtavanja gradova morao da bude manji od prečnika sfernog 3D modela, i zbog toga je došlo da malih odstupanja.

3.3 Arhitektura rešenja

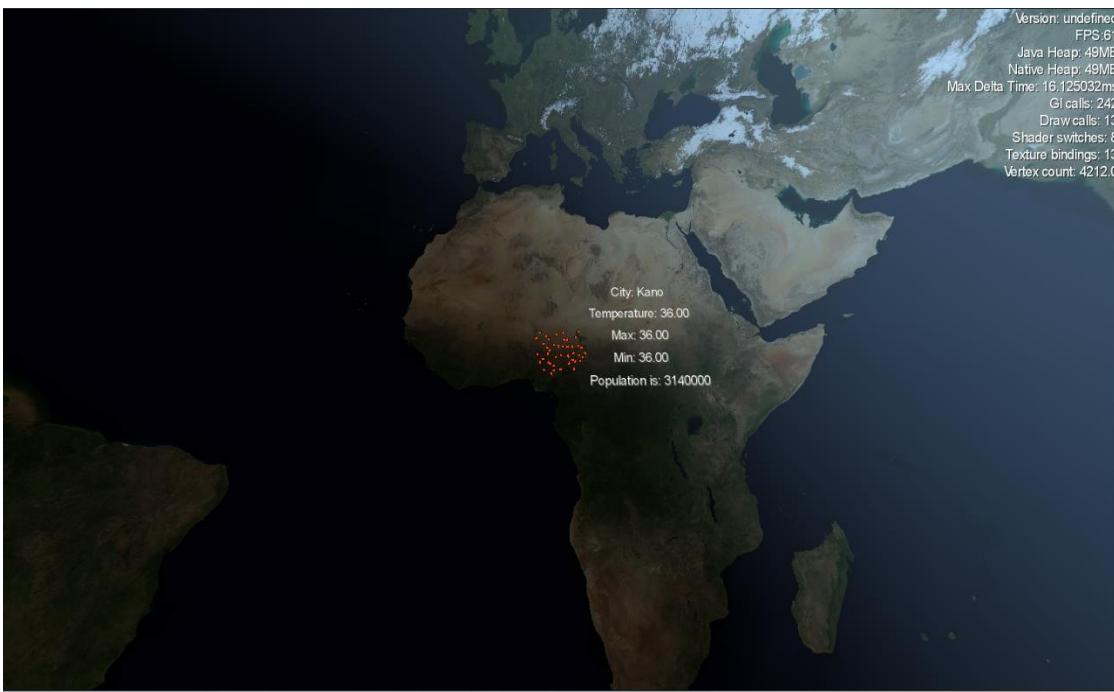
Na slici 10 prikazana je arhitektura rešenja tj. moduli koji opisuju ovo rešenje.



Slika 9 Blok dijagram sa prikazom modula i njihovog funkcionisanja

3.3.1 Modul za vremensku prognozu

Ovaj modul je zadužen za prosleđivanje informacija vezanih za vremensku prognozu. U ovom modulu se informacije koje se dobavljaju preko Web servisa parsiraju i pripremaju za prikaz krajnjem korisniku. Uspostavlja se veza sa serverom sa definisanom putanjom (*Uniform Resource Locator - URL*) preko internet veze, te se parsiraju informacije vezane za minimalnu, maksimalnu i trenutnu temperaturu, vlažnost vazduha i pritisak koji se dobavljaju preko JSON poruka. Web servis koji se koristi je *Open Weather Map* koji je besplatno dostupan. Preko njega se dobavljaju navedene informacije.[11] Na slici ispod je prikazan izgled aplikacije kada se kamera pozicionira na grad Kano u Nigeriji i prikazuju informacije o tom gradu.



Slika 10. Prikaz informacija o vremenskoj prognozi

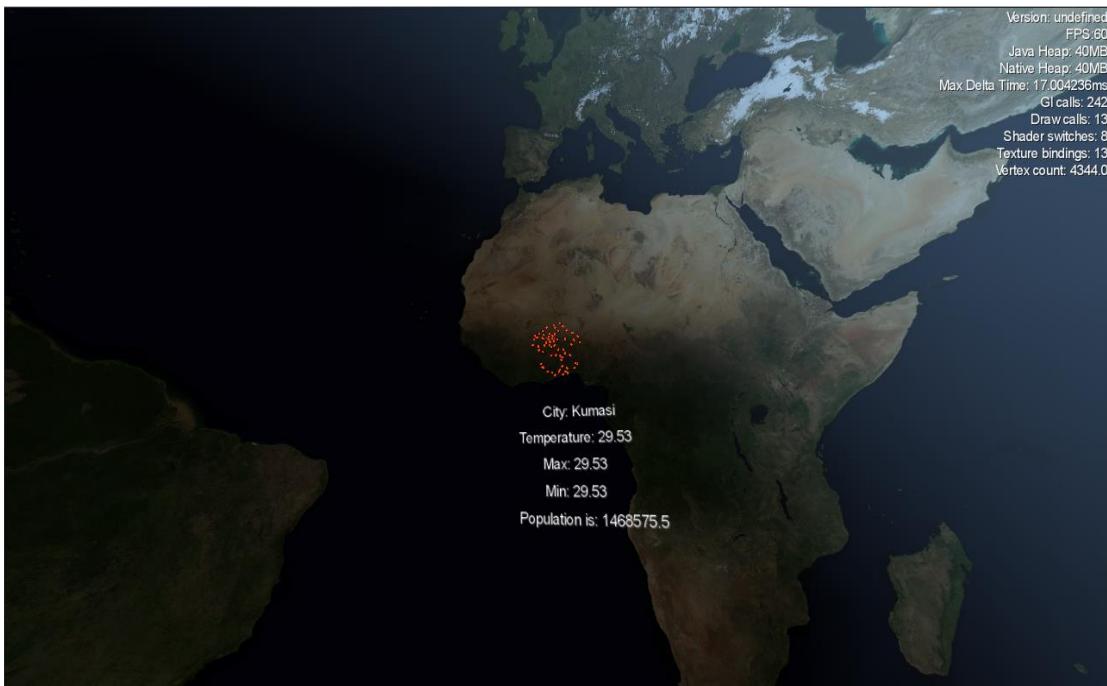
3.3.2 Modul za informacije o populaciji

Svrha ovog modula je da obezbedi informacije o populaciji svakog grada iz baze podataka. Baza podataka je *csv* baza podataka preuzeta sa interneta. Informacije o populaciji gradova se nalaze u istoj bazi podataka gde se nalaze gradovi i informacije o njima. Parsiranjem iste baze dobijaju se informacije o populaciji. Ovaj modul koristi već parsiranu bazu podataka i iz nje prosleđuje informacije o populaciji krajnjem korisniku. Na Slika 11 se vidi izgled jedne linije baze podataka.

grad	širina	dužina	populacija	država
Novi Sad,	45.2503762,	19.84994055,	220428.5,	Serbia

Slika 11. Prikaz izgleda baze podataka

Na sledećoj slici se može videti prikaz informacija za populaciju (Slika 12).



Slika 12. Slika sa informacijama o populaciji

3.3.3 Provajder

Svrha modula *Provajder* jesteda sakrije rad modula za vremensku prognozu i modula o inframacijama o populaciji, i da koristi njihova rešenja i prosleđuje ih dalje modulu za inicijalizaciju sveta, time se obezbeđuje proširivost aplikacije I dodavanje novih modula u projekat.

3.3.4 UI - Modul za inicijalizaciju sveta

Modul za inicijalizaciju sveta koristi usluge provajdera i krajnjem korisniku prikazuje krajnji izgled same aplikacije, učitavanje modela planete Zemlje, prikaz gradova, informacije o populaciji i vremenskoj prognozi.

3.4 Keširanje podataka

Podaci koji se keširaju su podaci koji se preuzimaju posredstvom Web servisa vezani za vremensku prognozu. Odlučeno je da se podaci keširaju jednom u sat vremena iz razloga što se informacije o vremenskoj prognozi ne menjaju toliko često. Ograničenja web servisa dozvoljavaju preuzimanje informacija o jednom gradu svake dve sekunde u besplatnoj verziji, što zahteva vreme da se te informacije prikupe. To je jedan od problema ove aplikacije.

4. Programsко rešenje

U ovom poglavlju opisani su detalji realizacije radnog okvira, kao i detalji realizacije pojedinačnih modula. U tabeli ispod prikazano je koji fajlovi programskog koda implementiraju pojedinačne module.

4.1 Opis programa

Pri startovanju programa ispred korisnika se pojavljuje model planete Zemlje i pomoću pomeranja kamere korisnik ima osećaj da polako ulazi u centar planete. Nakon što se korisnik tj. kamera pozicionira u nultu tačku (0,0,0) 3D prostora započinje pravo VR iskustvo.

4.2 Spisak modula i fajlova koji ih opisuju

Prikazani su svi moduli ovog rešenja kao i fajlovi koji opisuju svaki modul pojedinačno (Tabela 1).

MODUL	KLASA
Modul za inicijalizaciju sveta	WorldModelScene
Modul za vremensku prognozu	JsonWeatherParser, CityScene
Modul za informacije o populaciji	Parser, CityScene, SimpleMapDataElement

Tabela 1 Prikaz koji fajlovi programskog koda implementiraju pojedinačne module

Dalje kroz poglavlja svaki modul će biti detaljno objašnjen kao I fajlovi koji ga određuju.

4.3 Modul za inicijalizaciju sveta

Svet predstavlja modul *C-More* grafičkog okruženja za razvoj grafičke sprege. Klasa koja opisuje ovaj modul je *WorldModelScene*. U okviru sveta, pri inicijalizaciji, registruje se i

aktivira prva scena koja je zadužena za učitavanje i prikaz samog modela planete. Dolazi do pomeranja pozicije kamere na poziciju (0,0,0) i time se simulira ulazak korisnika u jezgro planete, nakon čega se koristi animacija rotacije obezbeđenja od strane *C-More* okruženja i planeta počinje da se rotira. Rotacija se vrši zbog poravnanja gradova koji treba da budu nacrtani na stvaran položaj na mapi sveta.

Pomoću metode *setOnKeyListener* koja se nalazi u ovoj sceni omogućene su animacije rotacije, na pritisak tastera određenih za rotaciju, planete u svim pravcima kao i rotacija sloja na kojem se nalaze mapirani gradovi.

4.4 Modul za vremensku prognozu

Modul će biti opisan kroz niz tabela koje opisuju svaku funkciju koja se koristi u ovom modulu:

Klasa ***JsonWeatherParser***:

BigDecimal[] start(String cityName)	
Opis	Poziva se da se dobave temperature za prosleđeni grad
Ulagni parametri	String cityName – ime grada za koji se dobavlja temperature
Izlazni parametri	/
Povratna vrednost	Povratna vrednost je niz brojeva retVal[0]=minimalna temperatrua retVal[1]=maksimalna temperatrua retVal[2]=trenutna temperatrua

Tabela 2 API funkcije *start()*

String readUrl(String urlString)	
Opis	Koristi se za dobavljanje JSON paketa preko Web servisa
Ulazni parametri	String urlString – adresa na internetu na kojoj se nalaze JSON informacije
Izlazni parametri	/
Povratna vrednost	BufferedReader reader - Povratna vrednost je string u kom se nalazi JSON sadrzaj koji se dalje parsira

Tabela 3 API funkcije *readUrl()*

Klasa **Parser**: Da bi uopšte bilo moguće pozvati prethodne dve funkcije potrebno je parsiranjem baze podataka doći do imena grada i za to je zadužena sledeća funkcija.

List<String[]> read()	
Opis	Konstruktoru klase u kojoj se funkcija nalazi se prosleđuje ulazni stream, nad kojim se poziva ova funkcija koja služi za parsiranje fajla
Ulazni parametri	/
Izlazni parametri	/
Povratna vrednost	resultList - Povratna vrednost je lista niza stringova u kojoj se nalaze gradovi i sve informacije o njima, geografska dužina, širina, država u kojoj se grad nalazi i populacija

Tabela 4 API funkcije *read()*

Ovaj modul je zadužen za parsiranje podataka o vremenskoj prognozi sa interneta ili iz već sačuvanih *JSON* fajlova. U ovom modulu postoji klasa *JsonWeatherParser* čija osnovna svrha je da parsira podatke iz *JSON* poruka direktno sa interneta ili iz već postojećih fajlova u nekom direktorijumu. U zavisnosti da li je flag *CACHE_DATABASE* postavljen na *false* parsiranje se vrši pomoću funkcije *readUrl*. Ova funkcija preuzima podatke sa Web servisa I

smešta ih u jedan string koji se kasnije parsira. U suprotnom se čitaju informacije iz predhodno sačuvanih fajlova sa podacima o vremenskoj prognozi za svaki grad. U slučaju preuzimanja informacija sa interneta u okviru URL se prosleđuje ime grada i tako se dobijaju određene informacije.

Nakon učitavanja *JSON* sadržaja, sadržaj se parsira i pristupa se potrebnim informacijama, ovo se obavlja korišćenjem metode *fromJson*, koja prima sadržaj sa interneta u vidu stringa i popunjava klasu sa poljima koje odgovaraju informacijama u *JSON* fajlu, u ovom slučaju to su temperatura, pritisak i vidljivost. Nakon što se dobave informacije o temperaturi potrebno je izvršiti konverziju iz kelvina u stepen celzijusa, i to je urađeno tako što je od dobijene vrednosti oduzeto 273,15 i dobijena je temperatura u celzijusima.

Da bi ovo parsiranje uopšte bilo moguće potrebno je da se parsira baza podataka sa gradovima i informacijama o njima da bi njihova imena mogla da budu prosleđena u putanju za URL ili putanju do direktorijuma u kojoj se nalaze *JSON* fajlovi. To je učinjeno pomoću klase *Parser*. Osnovna uloga ove klase je parsiranje informacija o gradovima (ime države, ime grada, geografska širina, geografska dužina, populacija). Glavna metoda klase *Parser* je *read* metoda. U ovoj metodi se obavlja parsiranje CSV baze podataka.CSV (*engl. Comma-separated values*) baza podataka je baza u kojoj su informacije koje su skladištene u njoj razdvojene zarezima i svaka informacija o određenom gradu u ovom slučaju se nalazi u jednom redu i to omogućuje lako parsiranje ovakve baze. Baza je organizovana tako da se u svakom redu nalaze informacije o jednom gradu i da su informacije razdvojene zarezom, pa po toj logici *read* metoda i radi. Ova metoda vraća listu niza stringova.

SimpleMapDataElement - Ova klasa je zadužena da prilikom parsiranja *csv* baze podataka organizovano čuva informacije o gradovima. Ova klasa poseduje *get* i *set* metode za sva polja vezana za informacije o gradu (ime grada, ime države, geografska širina, geografska dužina i populacija). Nakon što se baza podataka parsira i smesti u listu niza stringova, prolaskom kroz tu listu popunjava se klasa *SimpleMapDataElement*. To omogućava čuvanje informacija vezanih za grad čime se izbegava stalno parsiranje baze podataka.

4.5 Modul za informacije o populaciji

Zbog preglednosti koda napravljena je klasa *PopulationClass* čija je osnovna metoda *getPopulation* i ona kao parameter prima element iz gore navedene klase *SimpleMapDataElement* i na osnovu njega vraća broj stanovnika za prosleđeni grad.

4.6 Crtanje i mapiranje gradova

Cilj mapiranja gradova je da se svaki grad predstavi na što stvarnijoj poziciji u odnosu na stvaran svet. i to je uspešno urađeno uz što manja odstupanja. Logika ovog dela zadatka se odvija u klasi *CityScene*, gde je osnovna metoda ove klase *addCity* koja kao parametar prima element *SimpleMapDataElement* klase, i na osnovu njegove geografske širine i dužine računa njegovu prezentaciju u 3D prostoru. Računanje koordinata se vrši pomoću formula:

$$\begin{aligned}x &= R * \cos(lat) * \cos(lon) \\y &= R * \cos(lat) * \sin(lon) \\z &= R * \sin(lat)\end{aligned}$$

Slika 13. Prikaz formula za dobijanje koordinata

gde su x , y , z koordinate koje treba da se izračunaju, R predstavlja poluprečnik tj. udaljenost koordinata od centra sfere, lat je geografska širina, lon je geografska dužina.

Na osnovu koordinata (x , y , z) iscrtavaju se svi gradovi, pa se koordinate proslede metodi *setPosition* i tako svaka tačka dobija svoje mesto u 3D prostoru *C-More* okruženja. Svaki grad je predstavljen kao crvena tačka i mapiran unutar sfernog 3D modela. Prikazuju se samo gradovi koji su u vidnom polju korisnika.

Prikazivanje gradova oko grada u fokusu je rešeno na sledeći način. Nakon što se iscrtaju svi gradovi mapira se njihova apsolutna pozicija, tj. pozicija koja određuje njihov položaj na ekranu a ne u 3D prostoru i nakon toga se sakriju svi gradovi sem gradova koji se nalaze unutar kruga tj fokusa kamere. Formula za računanje tog kruga koji predstavlja fokus kamere je:

$$(x - a)^2 + (y - b)^2 = r^2$$

Slika 14. Formula za računanje fokusa kamere

gde je x – pozicija x koordinate na ekranu, a – polovina širine ekrana, y – pozicija y koordinate, b – polovina visine ekrana, a r - predstavlja poluprečnik samog fokusa i sa njim se odrađuje koliko će gradova zapravo da bude prikazano.

Nakon što su gradovi izmapirani i nacrtani potrebno je da se prikažu informacije o gradu u koji se gleda. To je urađeno pomoću klase *AugmentedWidget*. Osnovna uloga ove

klase je da ispiše informacije o temperaturi, populaciji i ime grada korisniku. Isto može da se vidi na slikama 8 i 9.

4.7 Provider

Uloga paketa Provider je da sakrije rad modula, i da koristi njihova rešenja i prosleđuje ih dalje, čime se obezbeđuje proširivost aplikacije i dodavanje novih modula u projekat. U slučaju ove aplikacije provajder upravlja krajnjim informacijama modula za vremensku prognozu i populaciju. I omogućuje da iz svakog dela koda može da im se pristupi.

void readCities()	
Opis	Glavna metoda ove klase, služi za parsiranje baze podataka sa gradovima pomoću već ranije definisanih funkcija iz prethodnih modula koji obaljaju to parsiranje nakon što se isparsira baza, uloga ove metode je da napravi objekte klase <i>City</i> koji kao polja ima samo informacije koje opisuju jedan grad i to su ime grada, ime države u kojoj se grad nalazi, geografska širina i dužina.
Ulazni parametri	/
Izlazni parametri	/
Povratna vrednost	/

Tabela 5 API funkcije *readCities()*

String getPopulation(SimpleMapDataElement element)	
Opis	Koristi se za dobavljanje populacije za prosleđeni element, a samim tim što klasa <i>SimpleMapDataElement</i> ima u sebi geter za populaciju to je I povratna vrednost ove funkcije
Ulagni parametri	SimpleMapDataElement element - Ulagni parametar je jedan element iz klase <i>SimpleMapDataElement</i>
Izlagni parametri	/
Povratna vrednost	String populationIs - Povratna vrednost je string u kom se nalazi informacija o populaciji za prosleđeni grad

Tabela 6 API funkcije *getPopulation()*

WeatherInfo getWeatherInfo(String city)	
Opis	Obavlja parsiranje JSON paketa pomoću već ranije navedenih funkcija iz modula za vremensku prognozu i skladišti informacije o vremenskoj prognozi za prosleđeni grad
Ulagni parametri	String city- predstavlja grad za koji se dobavljaju informacije o vremenskoj prognozi
Izlagni parametri	Informacije o minimalnom maksimalno i trenutnoj temperaturi
Povratna vrednost	WeatherInfo(result[0], result[1], result[2]) - Povratna vrednost su minimalna, maksimalna i trenutna temperature redom za prosleđeni grad

Tabela 7 API funkcije *getWeatherInfo*

Glavna klasa provider paketa je *Provider* klasa, prva metoda ove klase je *readCities* koja isčitava gradove pomoću metoda za parsiranje informacija o gradovima iz ranije navedenih modula i funkcija, nakon učitavanja svakog grada i informacija o njemu u listu gradova se dodaje novi element tj. klasa *City* čija su polja ime države, ime grada, geografska širina, geografska dužina. Druga metoda ove klase je *getCities* koja vraća listu svih gradova

učitanih prethodnom metodom. Takođe tu je i metoda *getCityName* koja kao parametar prima *City*, što je jedan element *SimpleMapDatElement* i vraća ime tog elementa. Sledeća metoda je *getPopulation* koja za parametar takođe prima element iz *SimpleMapDataElement* a pošto u toj klasi postoji metoda *getPopulation* samim tim je i to povratna vrednost ove metode.

I poslednja metoda ove klase je *getWeatherInfo*, ova metoda je zadužena da dobavi informacije o vremenskoj prognozi i to radi tako što koristi klasu *jsonWeatherParser* za dobavljanje informacija o vremenskoj prognozi za svaki grad, nakon što dobavi informacije o trenutnoj, maksimalnoj i minimalnoj temperaturi poziva konstruktor klase *WeatherInfo* i prosleđuje mu ove tri informacije kao tri elementa niza.

WeatherInfo klasa je zadužena za prikaz informacija o vremenskoj prognozi, kao što je već rečeno njen konstruktor se pravi u metodi *getWeatherInfo* i samim tim se smeste informacije za određeni grad.

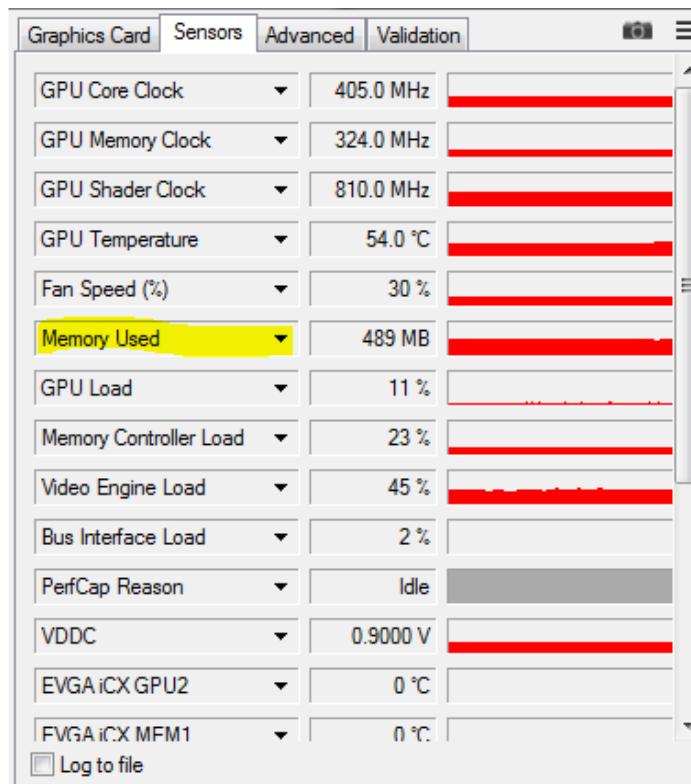
5. Rezultati

Okruženje u kom je testirana aplikacija je Samsung Galaxy S7. Mobilni telefon visokih performansi, moćnog procesora i velikog ekrana rezolucije 2560x1440. Zbog svojih karakteristika ovaj telefon je odlično okruženje za testiranje VR aplikacija.

Karakteristike mobilnog telefona Samsung Galaxy S7 su:

- Operativni sistem – Android;
- Procesor - Octa-core (4x2.3 GHz Mongoose & 4x1.6 GHz Cortex-A53);
- Sistemska memorija – 4GB RAM;
- Memorija za skladištenje – 32/64GB;
- Baterija - Non-removable Li-Ion 3000 mAh battery.

U tabelama ispod prikazani su rezultati koji su dobijeni merenjem potrošnje memorije (Tabela 8) i tabela za prikaz ponašanja FPS kada se u fokusu broj gradova povećava povećanjem fokusa (Tabela 9). Okruženje koje je korišćeno za prikaz potrošnje memorije je *TechPowerUp GPU – Z* (Slika 15). Informacije o FPS su dobijene za vreme pokretanja aplikacije na mobilnom telefonu (Samsung Galaxy S7), prikaz informacija o FPS moze da se vidi na slici 16 u gornjem desnom uglu ekrana mobilnog telefona.



Slika 15. Prikaz okruženja za dobijanje informacija o potrošnji memorije

	Bez učitanog modela	Bez učitanih gradova	Sa gradovima i modelom
Memorija(MB)	55	74	114

Tabela 8. Prikaz potrošnje memorije

Broj gradova u fokusu	FPS
100	60
200	55
300	31
400	24
1000	9

Tabela 9. Prikaz smanjenja broja frejmova u sekundi sa povećanjem učitanih gradova

Na osnovu dobijenih rezultata zaključeno je da je za rad aplikacije pogodan što manji fokus u kom će biti prikazani gradovi i informacije o njima. Samim tim što se radi o VR

aplikaciji potreban je što veći broj frejmova u sekundi za što bolji ugodaj korisnika. Po tabeli iznad, optimalan broj gradova u fokusu je između 100 i 200 i tada ne dolazi do smetnji u radu aplikacije. Kada se broj gradova poveća preko 400 aplikacija radi sa jako malim brojem frejmova u sekundi i tada ne zadovoljava preformanse za VR iskustvo.

5.1 Testiranje

Testiranje aplikacije se vršilo vizuelno i uz pomoć mobilnog telefona *Samsung Galaxy S7* i *Samsung Gear VR* seta. Izgled opreme na kojem je testirana aplikacija je prikazana na slici 16. Rezultati testiranja biće prikazani u tabeli 10.



Slika 16. Prikaz opreme za testiranje aplikacije

Testni slučaj	Opis koraka	Očekivani rezultat	Uspešnost
Učitavanje modela planete Zemlje	1-Staviti VR set na glavu 2-Pokrenuti aplikaciju	Učitan model planete zemlje i blago zarotiran zbog poravnjanja sa gradovima	Uspešno
Pomeranje fokusa kamere u negativnom smeru po X osi	Pomeriti glavu u levu stranu	Pomeranje fokusa kamere u levu stranu	Uspešno
Pomeranje fokusa kamere u pozitivnom smeru po X osi	Pomeriti glavu u desnu stranu	Pomeranje fokusa kamere u desnu stranu	Uspešno
Pomeranje fokusa kamere u negativnom smeru po Y osi	Pomeriti glavu na dole	Pomeranje fokusa kamere ka dole	Uspešno
Pomeranje fokusa kamere u pozitivnom smeru po Y osi	Pomeriti glavu na gore	Pomeranje fokusa kamere ka gore	Uspešno
Određivanje grada sa najvećom populacijom u okolini reke Nil u Egiptu, Afrika	Pomeranjem seta pozicionirati fokus kamere na reku Nil	Ispis sa najvećom populacijom i prikaz vremenske prognoze za grad Kairo	Uspešno
Provera ispravnosti podataka vezanih za temperature za predhodno učitani grad	Pomeranjem seta pozicionirati fokus kamere na reku Nil	Informacije o vremenskoj prognozi su isti kao i informacije sa web servisa sa kojeg se preuzimaju informacije	Uspešno

Tabela 10. Testni slučajevi

Pokretanjem aplikacije i upoređivanjem gradova sa Google Maps pozicijama, utvrđeno je da su gradovi tačno pozicionirani na mapi. Testiranje ispravnosti prikaza temperature upoređeno je sa temperaturom na *Open Weather Map* sajtu.

6. Zaključak

U ovom radu je predstavljeno rešenje trodimenzionalnog prikaza modela planete Zemlje na kojem su prikazane informacije o vremenskoj prognozi i populaciji u okruženju virtualne stvarnosti.

Realizacija ove aplikacije omogućava krajnjim korisnicima da u 3D okruženju mogu da vide realne i trenutne informacije o temperaturi i populaciji. Aplikacija je realizovana tako da može lako da se proširi, jer informacije o temperaturi i populaciji nisu jedine koje mogu da se prikažu na modelu planete Zemlje, što je obezbeđeno modularnošću rešenja.

Nedostatak ovog rešenja jeste krivljenje prostora pod velikim uglom vidljivosti u VR grafičkoj predstavi.

Osim toga, moguća su i brojna dodatna grafička poboljšanja, i to:

- Poboljšanje dizajna i prikaza informacija – Razvojem dizajna bi se korisniku pružila mogućnost menjanja sadržaja prikazanih informacija, i time se poboljšalo VR iskustvo korisnika.
- Optimizacija grafičkih resursa koji se koriste u aplikaciji – Došlo bi do poboljšanja preformansi i smanjenja potrošnje video memorije ukoliko bi se optimizovalo iscrtavanje raznih informacija koje ova aplikacija prikazuje.

7. Literatura

- [1]Larsen CR, Oestergaard J, Ottesen BS, Soerensen JL “*The efficacy of virtual reality simulation training in laparoscopy: a systematic review of randomized trials.*” Acta Obstet Gynecol Scand. 2012; 91:1015–1028 jul 2017.
- [2]https://en.wikipedia.org/wiki/Virtual_reality jul 2017.
- [3]Predavanje sa predmeta Digitalne multimedije 2, Visoka škola elektrotehnike i rašunarstva strukovnih studija, Beograd www.viser.edu.rs/download.php?id=18199 jul 2017.
- [4] www.linfo.org/gui.html jul 2017.
- [5]<https://www.khronos.org/opengles/> jul 2017.
- [6]<http://developer.android.com/guide/topics/graphics/opengl.html> jul 2017.
- [7]<https://github.com/libgdx/libgdx/wiki/Introduction> jul 2017.
- [8]<https://www.appbrain.com/stats/number-of-android-apps> jul 2017.
- [9]c-more_ui_framework_0.2.0(Bogeyman) documentation: C-More UI framework top level architecture jul 2017.
- [10] Đorđe Kovačević, Violeta Vukobat, Stefan Pejić and Ištvan Papp, *Building Blocks for GUI – “Novel Solution for Composite Component Mangment”*, 2nd International Conference on Electrical, Electronic and Computing Engineering IcETRAN 2015, Silver Lake, Serbia, June 8 – 11 2005 jul 2017.
- [11] <https://openweathermap.org/api> jul 2017.