Proces izrade interaktivne 3D vizualizacije prostora

Bošnjak, Mladen

SVEUČILIŠTE U ZAGREBU

GRAFIČKI FAKULTET

ZAVRŠNI RAD

Mladen Bošnjak

Sveučilište u Zagrebu
Grafički fakultet

Smjer: Dizajn grafičkih proizvoda

ZAVRŠNI RAD

Proces izrade interaktivne 3d vizualizacije prostora

Mentor:

doc. dr. sc. Tajana Koren Ivančević

Student:

Mladen Bošnjak

Zagreb, 2015.

Rješenje o odobrenju teme završnog rada

SAŽETAK

Kao demonstracija interaktivne 3d vizualizacije prostora izrađuje se dio unutrašnjosti Grafičkog fakulteta i dokumentira se cijeli proces. Uspoređuju se metode izrade 3d modela i bira se optimalna. Objašnjava se izrada 3d modela poligonalnim modeliranjem. Izrađeni model se teksturira i prenosi u 3d pokretač (*engine*). Upotrebom raznih tehnika postiže se što realniji prikaz prostora i optimalno korištenje resursa hardvera na kojem se aplikacija pokreće. Nastala scena renderira se u realnom vremenu i omogućava se interakcija korisniku. Navode se i objašnjavaju moguće primjene stvorene aplikacije na suvremenim operativnim sustavima i uređajima – PC, Mac, iOS, Android, Web.

ključne riječi:

3d model

pokretač

renderiranje

interakcija

ABSTRACT

As a demonstration of interactive 3d visualization of space a part of the interior of the Graphics faculty is made and the whole process is documented. Methods of making 3d models are compared and the optimal one is chosen. Polygonal 3d modeling is explained. The finished model is textured and brought into the 3d engine. A realistic representation of space and optimal resource usage of the hardware on which the application is run is achieved using various techniques. The resulting scene is rendered in real time and interaction is granted to the user. Possible uses for the application on modern operating systems and devices – PC, Mac, iOS, Android and Web are listed and explained.

Keywords:

3d model

engine

rendering

interaction

SADRŽAJ

1. UVOD

Vizualizacija je tehnika komunikacije kroz slike. Interaktivne 3d vizualizacije često se koriste u arhitekturi kako bi projektanti klijentima iz prve ruke prikazali kako će neko rješenje izgledati u stvarnosti. U industrijama poput kemijske i medicinske služe za obuku radnika, simulaciju mogućih nezgoda na radu i kvarova strojeva. [1] Koriste se u promotivne svrhe kako bi se prikazao izgled objekta koji je tek u projektnoj fazi. [2] Mogu se izrađivati za sve uređaje i operativne sustave koji imaju dovoljno snažan hardver za renderiranje 3d sadržaja u realnom vremenu. Za izradu interaktivnih 3d vizualizacija potrebno je znanje stvaranja 3d aplikacija. Pod to spadaju izrada 3d modela, 2d tekstura i poznavanje rada u programima za stvaranje interaktivnih 3d aplikacija (*3d graphics engine*). Ako projekt uključuje pretvaranje već postojećeg, stvarnog prostora u virtualni prostor, potrebno je detaljno fotografiranje i mjerenje postojećeg prostora kako bi bio što točnije prikazan u konačnoj aplikaciji.
U teorijskom dijelu ovog rada objašnjavaju se metode izrade svih dijelova interaktivnih vizualizacija i tehnologija koja ih pokreće.
Eksperimentalni dio rada demonstrira izradu interaktivne vizualizacije tako što se u 3d-u izrađuje unutrašnjost zgrade Grafičkog fakulteta i omogućava se interakcija korisniku.

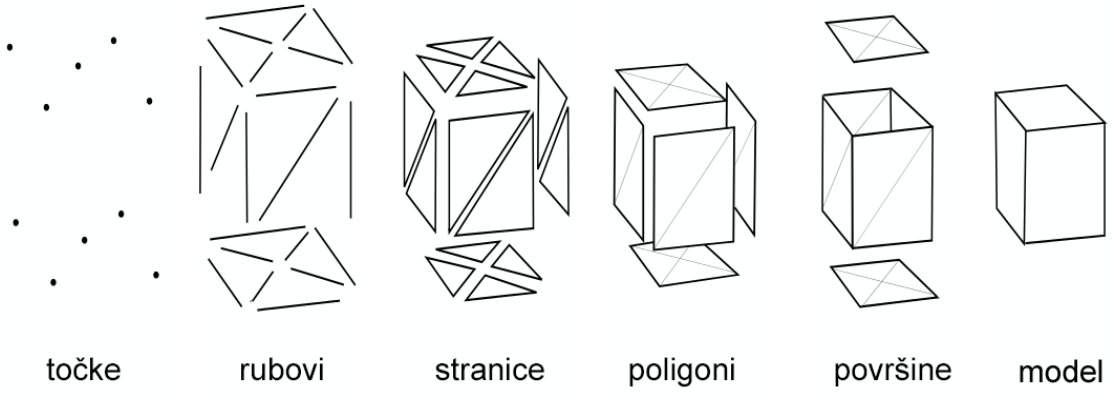
2. TEORIJSKI DIO

2.1. METODE IZRADE 3D MODELA

U računalnoj grafici 3d modeliranje je proces stvaranja matematičke prezentacije bilo koje trodimenzionalne površine objekta pomoću specijaliziranog softvera. Nastali proizvod naziva se 3d modelom. On se može prikazati kao dvodimenzionalna slika kroz proces nazvan 3d renderiranje. Fizički ga možemo izraditi koristeći 3d printer.

Postoje četiri glavna načina izrade 3d modela:

Poligonalno modeliranje – najčešći način modeliranja. Kod ovog načina površina objekta prikazuje se poligonima. Poligonalni model je skupina točaka, rubova i stranica koji definiraju oblik, ilustrirano na slici 1.



Slika 1. dijelovi poligonalnog modela

Modeliranje krivuljama – površine su definirane krivuljama na koje se utječe kontrolnim točkama.

Digitalno skulpturiranje –najnoviji način izrade 3d modela koji postaje sve popularniji zbog umjetničke slobode koju nudi. Većina programa za digitalno skulpturiranje poput ZBrusha ili Mudboxa koriste *voxele* za prikaz modela. *Voxel* je ekvivalent pikselu u 3d prostoru, što znači da osim x i y koordinata ima z koordinatu koja označava dubinu.

Skeniranje stvarnih objekata – metoda koja se sve više koristi pri izradi računalnih igara i interaktivnih vizualizacija jest skeniranje stvarnih objekata pomoću 3d skenera ili posebnih programa koji interpretiraju fotografije objekta kao 3d model. Kako bi se dobila vjerna interpretacija odbijanja svjetlosti s materijala na objektu, kod 3d skeniranja uvjeti osvjetljenja moraju biti idealni.

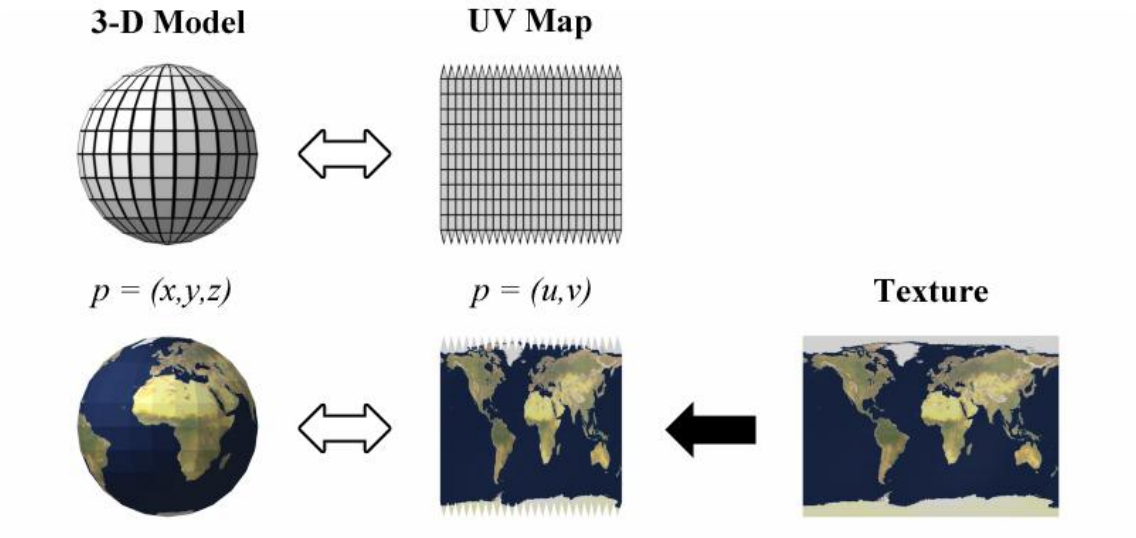
U izradi ovog završnog rada koristi se metoda poligonalnog modeliranja. Ona je optimalna za izradu unutrašnjosti Grafičkog fakulteta zbog geometrijskog oblika objekata koje je potrebno modelirati. Budući da većina 3d pokretača prikazuje objekte kao poligonalne modele neće biti potrebno konvertirati model u drugu tehniku.

2.2. TEKSTURIRANJE MODELA

Teksturiranje (eng. *Texture mapping*) je tehnika dodavanja detalja, površinske teksture ili boje 3d modelu. Pikseli teksture primjenjuju se na stranice objekta, slično omotavanju kutije u papir s uzorkom.

2.2.1. UV mapiranje

UV mapiranje je proces primjene 2d slike kao površine 3d modela. Demonstrirano na slici 2 primjenom na kuglu. [3]

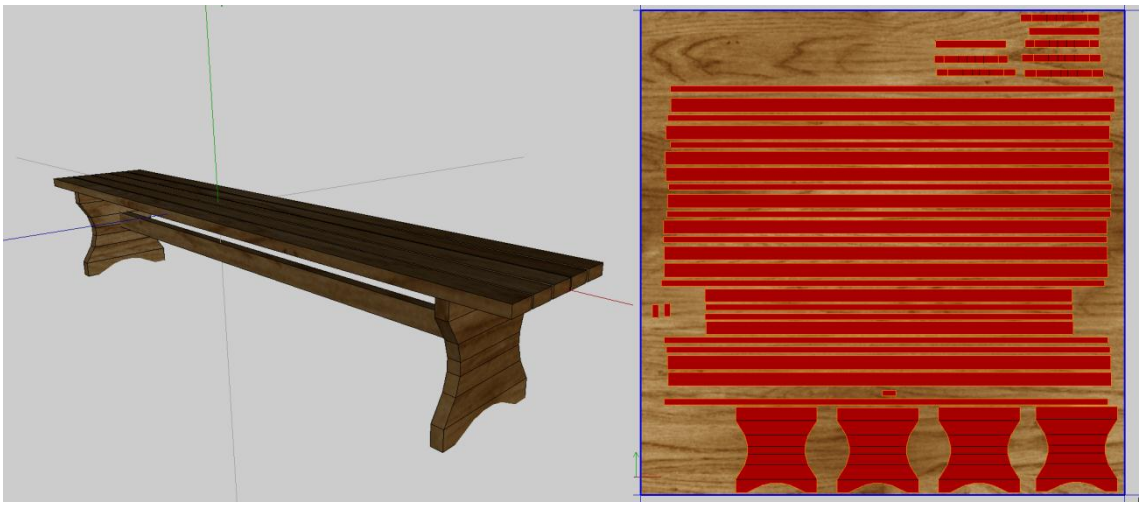


Slika 2. prikaz primjene teksture na kuglu

UV mapiranje nazivamo tako zato što se x, y i z slova koriste za označavanje koordinata 3d modela, pa se slova u i v koriste za koordinate teksture.

Većina programa za 3d modeliranje ima vlastiti sustav za UV mapiranje, a postoje i programi izrađeni posebno za tu namjenu.

Za potrebe ovog rada koristi se sustav UV mapiranja integriran u program Wings3d prikazan na slici 3.



Slika 3. lijevo - 3d model s primijenjenom teksturom, desno - UV mapa svih stranica 3d modela translatiranih u 2d prostor na teksturu

2.2.2. Izrada teksture

Teksture je moguće stvoriti na više načina; digitalnom manipulacijom fotografija, ručnom izradom u programima za digitalno crtanje te automatskim generiranjem u računalnim programima.

Digitalna manipulacija fotografija

Kod digitalne manipulacije fotografija prvo se fotografira površina koju želimo prenijeti na 3d model. Budući da nastala fotografija gotovo uvijek na sebi sadrži neželjene sjene, objekte i drugo, potrebno je digitalnom manipulacijom u programu poput Adobe Photshopa stvoriti finalnu sliku koja se može koristiti kao tekstura na modelu. Razlika između originalne fotografije i finalne slike prikazana je na slici 4.

Slika 4. lijevo - originalna fotografija, desno - tekstura dobivena digitalnom manipulacijom originalne fotografije u programu Adobe Photoshop

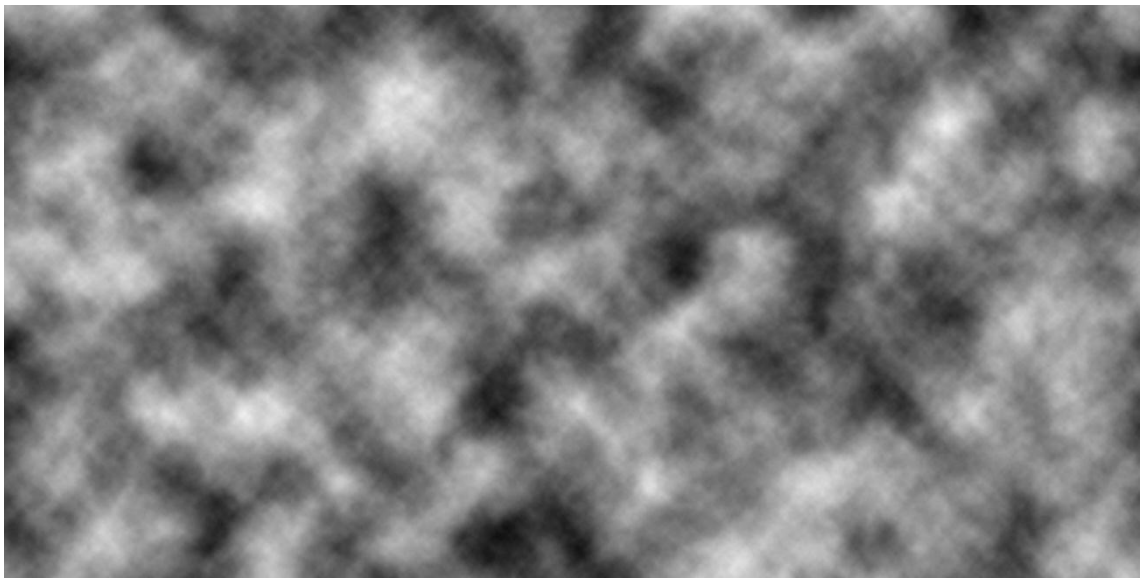
Ručna izrada u programima za digitalno crtanje

Ukoliko cilj projekta nije imitiranje stvarnosti tekstura se može stvoriti ručno u nekom od programa za digitalno crtanje ili tradicionalnim tehnikama slikarstva. To je moguće vidjeti na slici 5, gdje je autor cijelu teksturu stvorio ručno. [4]

Slika 5. tekstura kamena nacrtana ručno u programu za digitalno crtanje. Autor: Jairo Sanchez

Automatsko generiranje u računalnim programima

Mnogi programi za digitalnu manipulaciju slika imaju mogućnost renderiranja tekstura primjenom matematičkih jednadžbi i *filtera*. Na slici 6 prikazana je tekstura generirana putem *filtera Render Clouds* u programu Adobe Photoshop.



Slika 6. tekstura „oblaka“ generirana u programu Photoshop

Postoje mnoge Web stranice s besplatnim teksturama za korištenje u raznim projektima.

2.3. USPOREDBA I ODABIR POKRETAČA

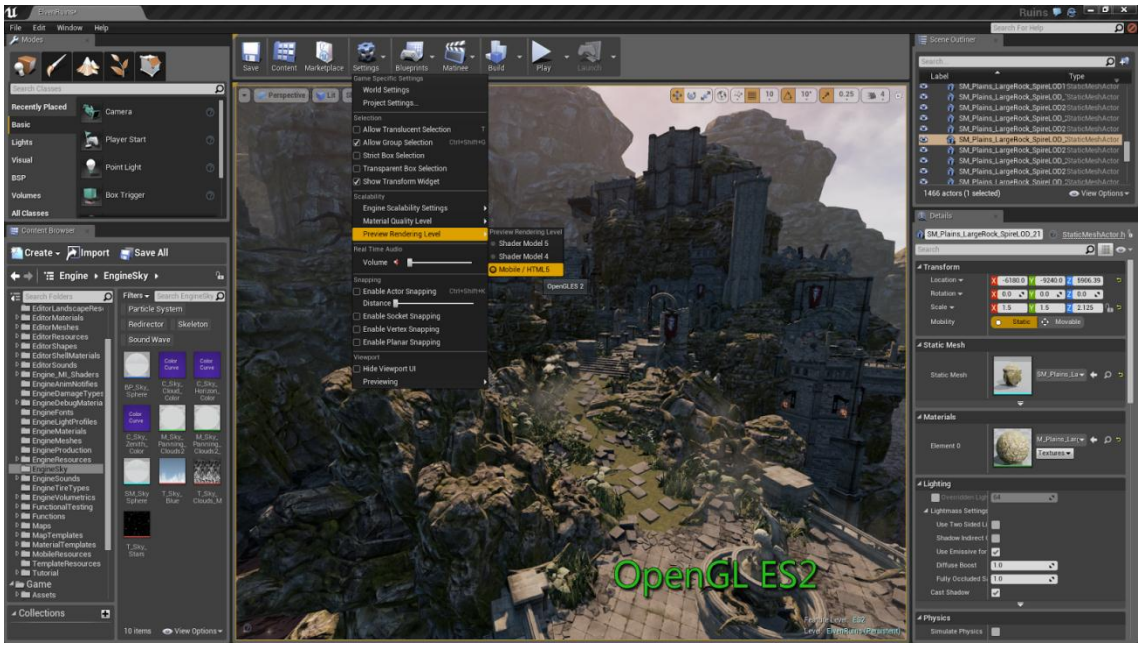
2.3.1. 3d pokretač (eng. *3d graphics engine*)

3d pokretač je softver dizajniran primarno za izradu računalnih igara, no često se koristi u vizualizaciji, arhitekturi te ostalim industrijama gdje su potrebne interaktivne simulacije. Osnovna tehnologija koju pruža pokretač jest renderiranje za 2d ili 3d grafiku, simulator fizike, zvuka, animacije, umjetne inteligencije i skriptiranje.
U većini slučajeva pokretač dolazi sa setom vizualnih razvojnih alata za bržu izradu željene aplikacije.

2.3.2. Najpopularniji 3d pokretači

Unreal Engine

Unreal Engine najpopularniji je 3d pokretač. Koristi se u širokom spektru aplikacija poput računalnih igara, virtualnih simulacija treninga u raznim industrijama, vizualizacija u arhitekturi i drugo. Dostupan je putem mjesečne pretplate od 19 američkih dolara i 5% zarade od komercijalnih proizvoda razvijenih u ovom pokretaču. Besplatan je za škole i sveučilišta, uključujući osobne primjerke za studente razvoja računalnih igara, arhitekture, programiranja, vizualizacije i sličnih. Na slici 7 prikazano je grafičko sučelje pokretača. [5]



Slika 7. grafičko sučelje Unreal Engine pokretača

Unity

Unity 3d pokretač najviše koriste nezavisni studiji i manje tvrtke kod izrade 3d aplikacija. Podržava mnoge operativne sustave kao što su Windows, Mac OS X, Linux, iOS, Android i Web. Postoje dvije verzije programa.

Prva je Unity Personal, besplatna inačica za osobe ili tvrtke koje zarađuju manje od 100,000 američkih dolara godišnje od svojih projekata. Druga je Unity Pro, profesionalna inačica pokretača.
Najnovija inačica pokretača, Unity 5.0 podržava mnoge moderne značajke poput globalne iluminacije, naprednog animiranja 3d modela, dinamičnih zvučnih efekata i drugih. Slika 8 prikazuje grafičko sučelje pokretača. [6]

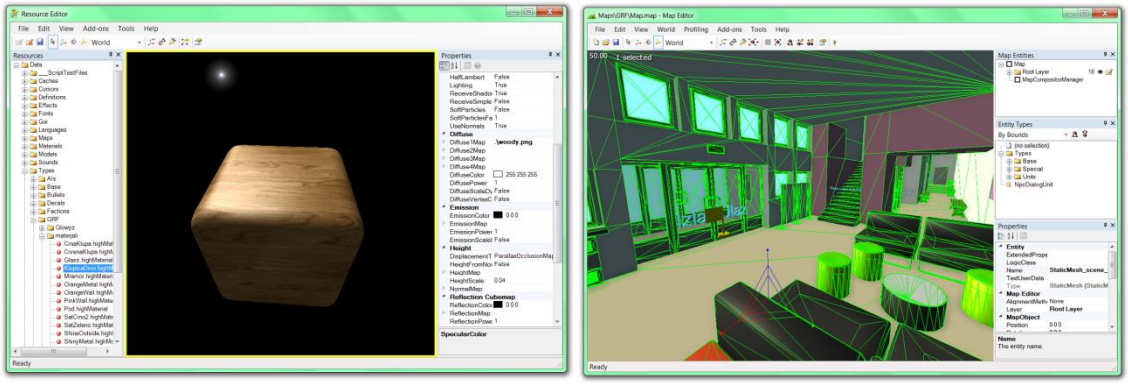
Slika 8. grafičko sučelje Unity pokretača

NeoAxis Engine

NeoAxis Engine nije popularan kao gore navedeni pokretači, no zbog svoje jednostavnosti korištenja i mogućnosti brzog stvaranja prototipa odabran je za demonstraciju procesa izrade 3d vizualizacije prostora u ovom radu. Podržava Windows i Mac OSX operativne sustave.

NeoAxis 3d Engine odvojen je u dva primarna dijela. Prvi je *Resource Editor*, program koji služi za stvaranje entiteta koji se kasnije koriste u simulaciji. Pod entitete spadaju modeli, teksture, materijali, fizika objekata i grafička sučelja.

Drugi je *Map Editor*, program u kojem se izrađuju mape za simulaciju. U ovom programu na mapu se postavljaju modeli, stvara se teren (u ovom radu ne izrađuje se vanjski prostor, zbog čega nije potrebno stvarati teren) i skriptira se logika simulacije. [7] Oba *editora* prikazani su na slici 9.



Slika 9. lijevo – *Resource Editor*, desno – *Map Editor* NeoAxis Enginea

2.4. POSTAVKE U POKRETAČU

2.4.1. Materijali

Ovisno o pokretaču kojeg koristimo moguće je osim pridodavanja teksture modelu kontrolirati i materijal površine. Na ovaj način manipulira se boja, prozirnost, refleksije, emitiranje svjetlosti i druge značajke površine. Na slici 10 vidljive su različite postavke grubosti površine i načina difuzije svjetlosti.

Slika 10. ista tekstura s tri različite postavke materijala

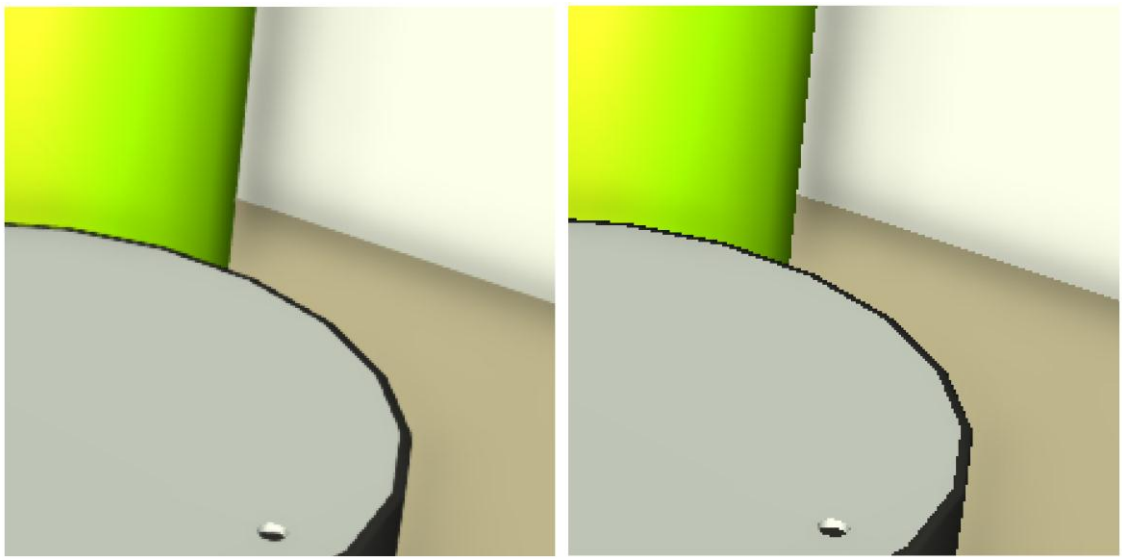
2.4.2. Post-efekti *(*eng. *posteffects)*

Post-efekti su efekti koje grafička kartica primjenjuje na scenu nakon što ju renderira. Postoje mnogi post-efekti koji približavaju vizualizaciju prostora stvarnosti. Nabrajaju se i objašnjavaju efekti koji se koriste u demonstraciji vizualizacije u ovom radu.

Anti-aliasing

Ako se povuče dijagonalna linija kvadratnim pikselima njihovi oštri rubovi stvaraju nazubljeni „stepenasti“ efekt što se vidi na desnoj strani slike 11. Ova pojava naziva se *aliasing*. Kada bi rezolucije monitora bile mnogo više taj efekt ne bi bio primjetan kao što je vidljivo na lijevoj strani slike 11, no s današnjim monitorima potrebno je pronaći rješenje za sakrivanje ovog negativnog efekta.

Anti-aliasing pronalazi rubove renderiranih objekata u sceni i zamućuje rubne piksele kako bi smanjio efekt *aliasinga*.[8]



Slika 11. lijevo – uključen *anti-aliasing* efekt, desno – isključen *anti-aliasing* efekt

SSAO *(Screen Space Ambient Occlusion)*

Ambijentalna okluzija tehnika je renderiranja kojom se izračunava koliko je svaka točka u sceni izložena ambijentalnoj svjetlosti. Najbolji primjer za objašnjavanje ambijentalne okluzije je cijev. Unutrašnjost cijevi manje je izložena svjetlosti i što dublje ulazimo u cijev količina difuzne svjetlosti sve je manja što taj dio cijevi čini mračnijim.

Screen space ambient occlusion pokušava, izračunavajući dubinu renderiranih piksela, izračunati do kojih dijelova scene difuzna svjetlost teže dopire, te na tim dijelovima scene dodaje meke sjene. [8] Te sjene vidljive su na kutovima objekata na slici 12.

Slika 12. lijevo – uključen SSAO efekt, desno – isključen SSAO efekt

Bloom

Bloom efekt simulira način na koji vrlo intenzivno svjetlo promatraču izgleda kao da izlazi iz okvira objekta koji emitira svjetlost. Ova je pojava vidljiva na materijalu stakla na slici 13. [8]

Slika 13. lijevo – uključen *bloom* efekt, desno – isključen *bloom* efekt

3. EKSPERIMENTALNI DIO

3.1. FOTOGRAFIRANJE REFERENCI

Prvi korak u procesu izrade vizualizacije je fotografiranje referenci za buduće modele. Svaki objekt koji će biti modeliran fotografira se zasebno. Uz to se objekti fotografiraju i zajedno kako bi se lakše odredio odnos veličina, što je vidljivo na slici 14.

Slika 14. nekoliko primjera fotografija objekata

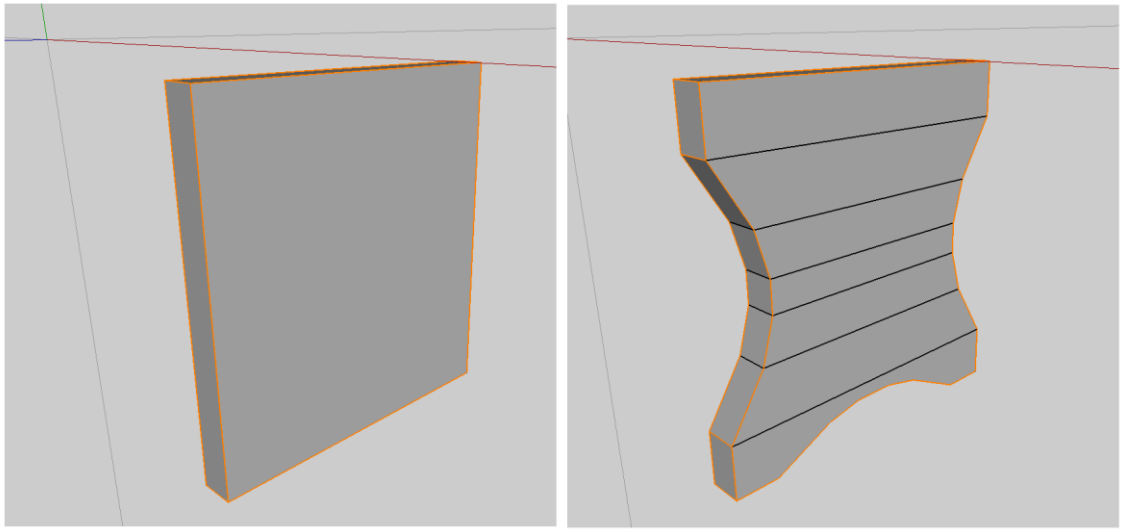
3.2. IZRADA 3D MODELA POLIGONALNIM MODELIRANJEM

Za izradu 3d modela koristi se program Wings3d zbog jednostavnosti korištenja i brzog stvaranja prototipa. Manji objekti u prostoru Grafičkog fakulteta izrađuju se zasebno od glavnog modela unutrašnjosti zgrade.

Za demonstraciju modeliranja uzima se drvena klupa prikazana na lijevoj strani slike 15.

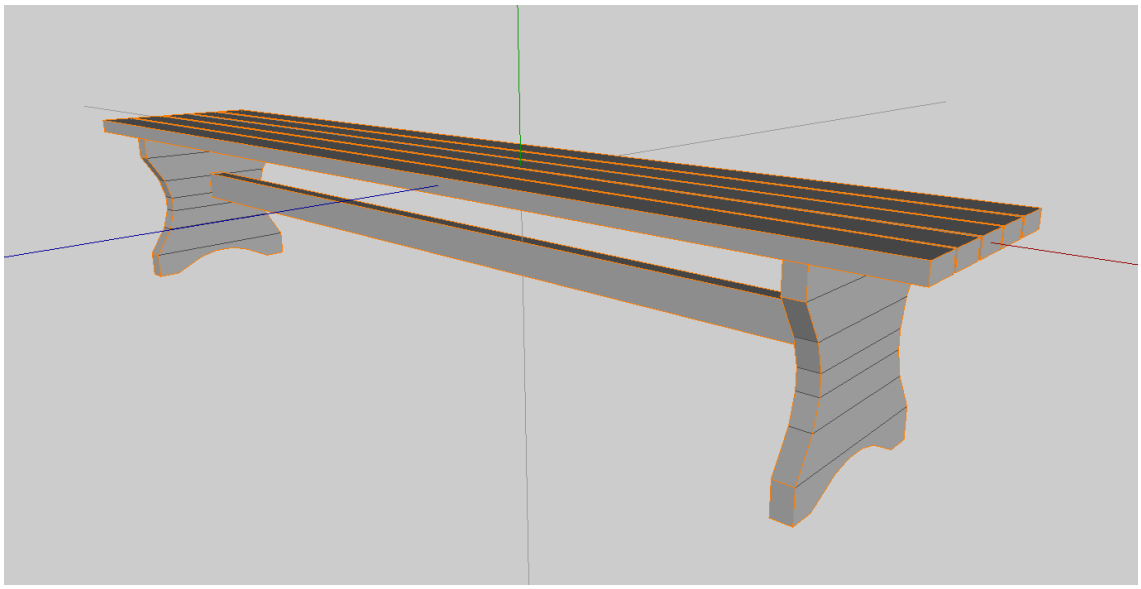
Slika 15. lijevo - fotografija objekta, desno – finalni render modela izrađenog u pokretaču

Poligonalno modeliranje uvijek počinje od jednostavnog oblika poput kocke, kugle, piramide i drugih. Na lijevoj strani slike 16 nalazi se prvotni model kvadra. Manipulacijom njegovih stranica pretvara se u oblik noge klupe, što je vidljivo na desnoj strani slike 16.



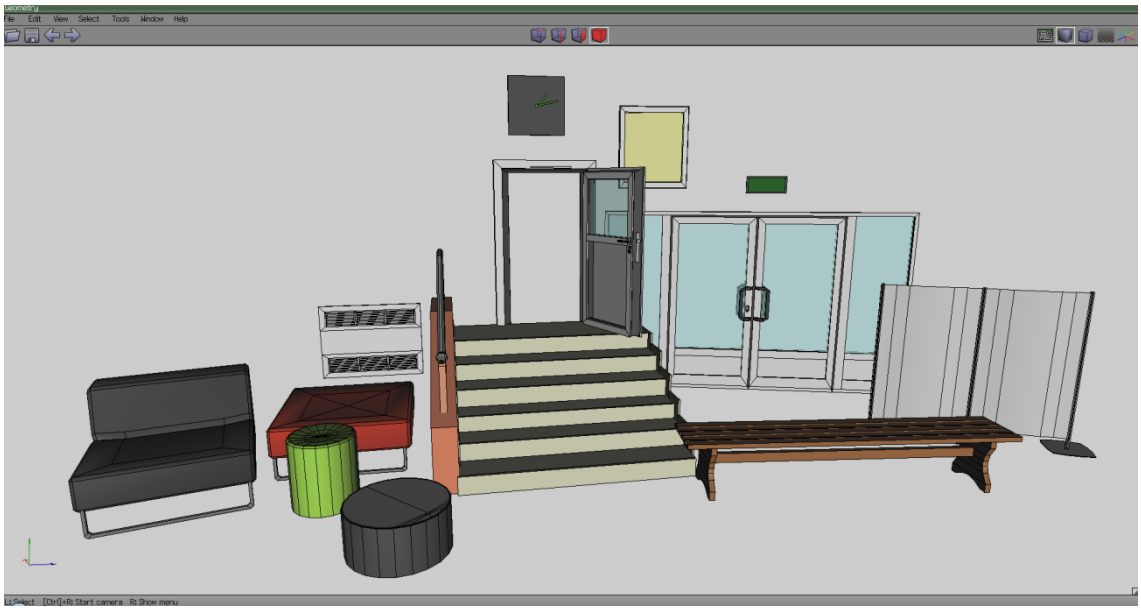
Slika 16. za osnovni model odabire se kvadar iz kojeg se dodavanjem i modificiranjem stranica dobiva oblik noge klupe

Na slici 17 prikazan je dovršen model klupe izrađen dupliciranjem noge klupe, dodavanjem dugih kvadara na gornju stranu i dodavanjem jednog poprečnog kvadra između nogu klupa.



Slika 17. dupliciranjem noge klupe i dodavanjem dugih kvadara dobivamo model sličan objektu sa referentne fotografije

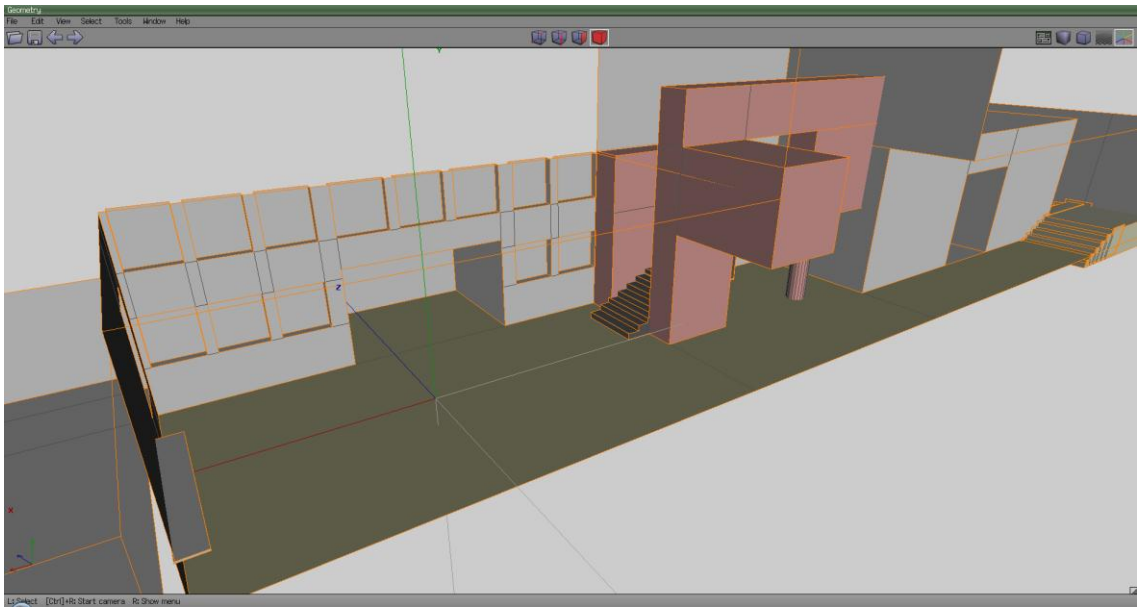
Ovim postupkom modelira se svaki pojedini objekt potreban u sceni. Završeni modeli bez tekstura prikazani su na slici 18.



Slika 18. zasebno modelirani objekti u programu Wings3d

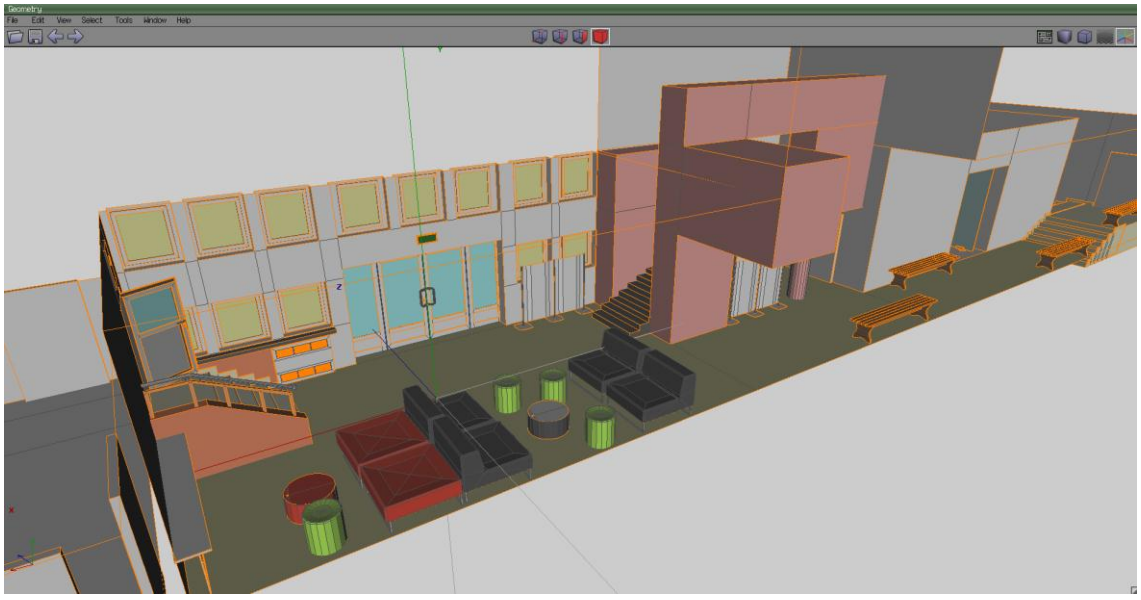
3.3. MODELIRANJE UNUTRAŠNJOSTI ZGRADE

Nakon izrade svih modela u sceni modelira se unutrašnjost zgrade. Zidovi, podovi, stropovi i stepeništa. Posebna pažnja pridaje se omjeru veličina na modelu, vidljivo na slici 19.



Slika 19. model unutrašnjosti zgrade Grafičkog fakulteta

Naposljetku se postavljaju prije modelirani objekti u model unutrašnjosti i rade se završne korekture omjera objekata, prikazano na slici 20.



Slika 20. model unutrašnjosti zgrade fakulteta sa dodanim zasebnim objektima

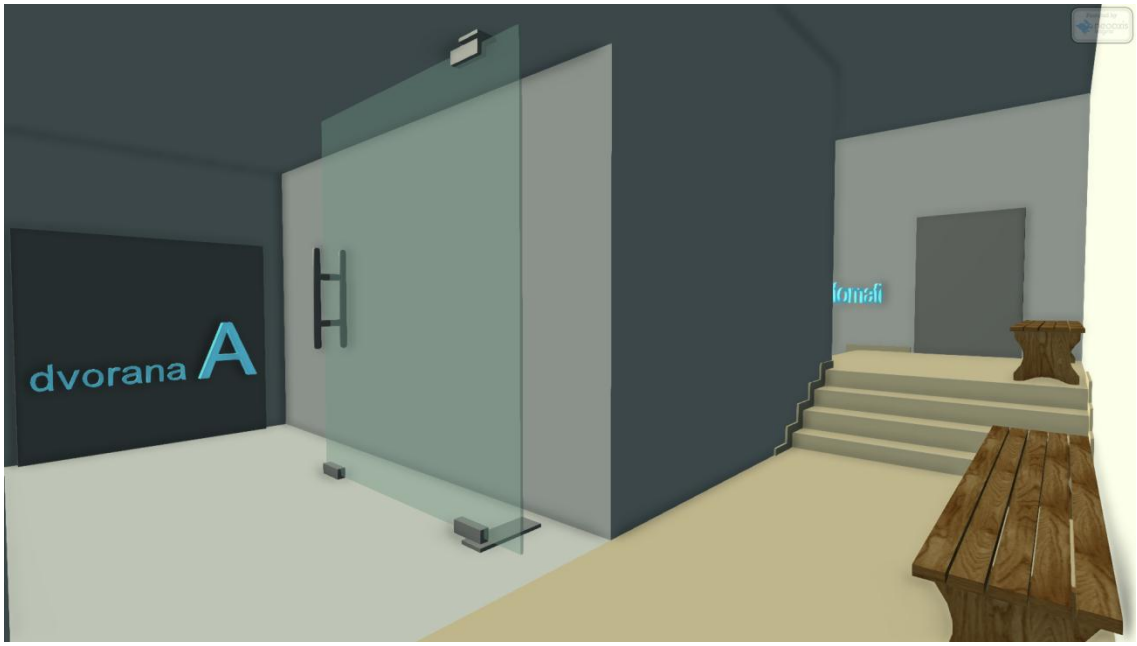
3.4. PRENOŠENJE MODELA U POKRETAČ

Stvoreni model konvertira se iz *.wings* formata koji koristi program Wings3d u format *.mesh* koji koristi pokretač NeoAxis Engine. U *Resource Editoru* pokretača stvaraju se potrebni materijali za razne površine na modelu. To podrazumijeva drvo, plastika, metal, staklo, mramor i drugi. U M*ap Editoru* pokretača postavlja se model u mapu, izvor svjetla koji simulira sunce i točka u kojoj će se stvoriti avatar korisnika.

Prilikom testiranja aplikacije zabilježene su slike 21, 22 i 23. Na njima se vidi završni izgled aplikacije kako ju vidi korisnik.

Slika 21. izgled ulaznog dijela unutrašnjosti Grafičkog fakulteta

Slika 22. pogled na ulazna vrata i stepenice koje vode na više dijelove zgrade



Slika 23. pogled na ulaz dvorane A

4. FINALIZIRANJE

4.1. OPTIMIZACIJA

Svi dijelovi hardvera uređaja koji pokreće aplikaciju koriste se kod simuliranja vizualizacije prostora. Glavni procesor zadužen je za simuliranje logike i fizike. U radnu memoriju spremljene su teksture i zvukovi koji se koriste u simulaciji. Najvažniji dio hardvera u simuliranju 3d vizualizacija je hardver zadužen za renderiranje 3d grafike. Kod računala je to grafička kartica, a kod mobilnih uređaja grafički procesor. Grafička kartica 60 puta u sekundi renderira scenu koju korisnik trenutno gleda. Što je scena kompleksnija, to je kartici teže renderirati scenu. Aplikacija koja nije optimizirana ili korištenje nedovoljno jakog hardvera može rezultirati u smanjenju broja renderiranih sličica u sekundi, što korisnik vidi kao smetnje kod kontroliranja aplikacije.

Optimizacija je pronalaženje balansa između kompleksnosti modela, veličine tekstura, rezolucije renderiranja, količine izvora svjetlosti i efekata u sceni. Osim optimizacije modela i tekstura, potrebno je optimizirati kôd aplikacije kako bi stabilno radila na svim ciljanim operativnim sustavima i uređajima.

4.2. KOMPILIRANJE

Izrađena aplikacija mora se finalizirati za ciljane operativne sustave. To se ostvaruje kompiliranjem pri čemu, umjesto da se aplikacija pokreće kroz pokretač, nastaje samostalna izvršna datoteka (*.exe file*). U procesu kompiliranja kôd aplikacije se kriptira kako ne bi moglo doći do neželjene promjene kôda. Takva aplikacija spremna je za daljnju upotrebu krajnjih korisnika. Aplikacija stvorena za ovaj rad kompilira se za Windows i Mac OSX sustave.

4.3. TESTIRANJE STVORENE APLIKACIJE

Stvorenu aplikaciju potrebno je temeljito testirati na svim operativnim sustavima za koje je predviđena. Budući da postoje nebrojene količine potencijalnih uređaja na kojima bi korisnici mogli koristiti aplikaciju (različiti pametni telefoni, prijenosna računala i hardverske konfiguracije kod osobnih računala), potrebno je ekstenzivno testiranje na što širem spektru navedenih uređaja. Često je neizvedivo testiranje na takvoj količini uređaja pa mnogi pokretači imaju ugrađene sustave za emuliranje raznih uređaja koje nazivamo *emulatorima*. *Emulator* je softver koji omogućuje jednom uređaju (domaćin) da se ponaša kao drugi uređaj (gost). Tim putem uređaj domaćin može pokretati softver dizajniran za uređaj gosta. [9]

5. PRENOŠENJE APLIKACIJE NA RAZLIČITE OPERATIVNE SUSTAVE

Microsoft Windows

Microsoft Windows najfleksibilniji je operativni sustav za izradu i pokretanje aplikacija zbog velike popularnosti i lakoće stvaranja aplikacija. Aplikacije moraju imati *.exe* datoteku (*executable file*) putem koje ih korisnik pokreće. Aplikacije se mogu preuzimati iz mnoštva izvora. Windows upozorava korisnika ako nije moguće verificirati izvor aplikacije koju korisnik pokreće. Novije verzije Windows operativnog sustava (od verzije 8 nadalje) imaju integrirani Microsoft Store preko kojeg korisnici mogu preuzimati aplikacije slično Appleovom App Storeu. 3d sadržaj na Windows operativnom sustavu prikazuje se putem 3d pokretača koji većinom koriste Microsoftov DirectX API. API (*application programming interface*) je set rutina, protokola i alata za izradu softvera. Ostali operativni sustavi većinom koriste OpenGL (*open graphics library*) API.

Apple Mac OS X

Appleov Mac OS X dolazi sa Macintosh stolnim i prijenosnim računalima. Baziran je na Unix sustavu. Kao i na Windows operativnom sustavu, aplikacije za OS X mogu se instalirati iz bilo kojeg izvora, no najpopularniji način preuzimanja aplikacija je putem Appleovog App Storea.

Apple iOS

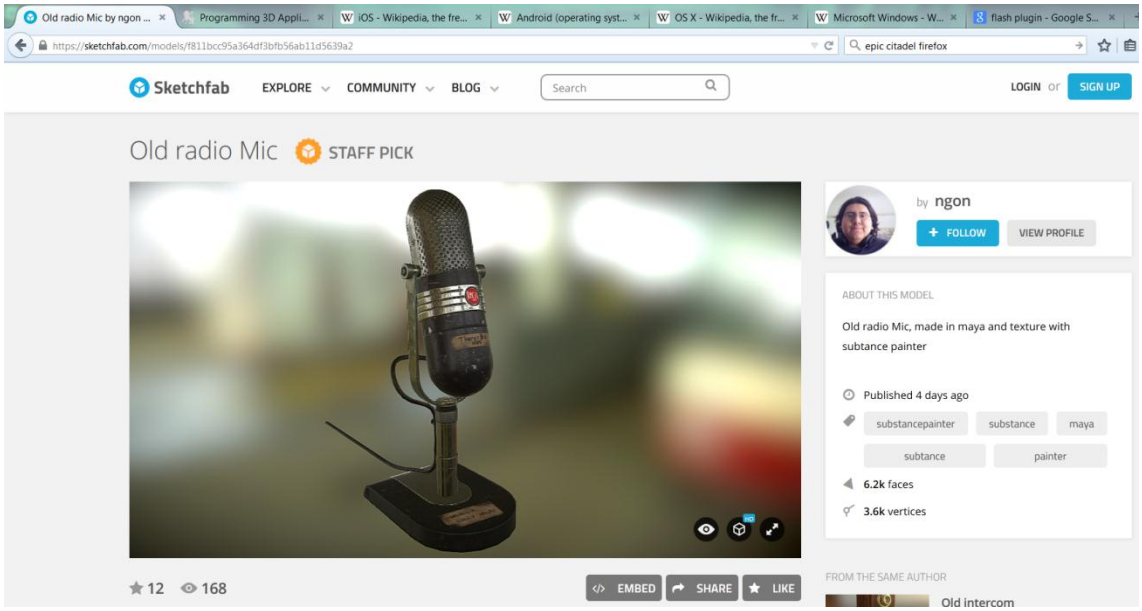
Mobilni operativni sustav iOS pokreće mnoge Appleove uređaje uključujući iPhone, iPad i iPod touch. Aplikacije za iOS korisnici preuzimaju isključivo sa Appleovog App Storea. Za izradu i objavljivanje aplikacija na App Storeu potrebno je platiti godišnju pristojbu od 99 američkih dolara. Završena aplikacija prolazi kroz Appleovu kontrolu kvalitete gdje se provjerava stabilnost i primjerenost sadržaja aplikacije.

Google Android

Android je operativni sustav trenutno u vlasništvu tvrtke Google. Baziran je na Linux *kernelu*. Nalazi se na mnogim uređajima poput pametnih telefona, tableta, televizora, automobila i satova. Aplikacije za Android preuzimaju se primarno sa Google Play Storea. Uz to većina uređaja koji koriste Android imaju opciju instaliranja aplikacija iz drugih izvora. Razvijanje i objavljivanje aplikacija na Play Storeu besplatno je i nema kontrole kvalitete. Zbog toga je lakše razvijati aplikacije za Android nego za iOS, no postoji veća mogućnost da je aplikacija koju korisnik preuzme nestabilna ili čak štetna za uređaj.

Web

Pokretanje aplikacija dizajniranih za prikazivanje i korištenje na internetu nije ovisno o uređaju na kojem se pokreću, već o internet pregledniku kojeg korisnik koristi. Postoji mnogo metoda za prikazivanje 3d sadržaja na Webu. Najčešće putem *plugina* poput Adobe Flash playera ili Unity pokretača. Sve popularnije je direktno prikazivanje 3d sadržaja na Web stranici bez preuzimanja dodatnih *plugina*. To je ostvarivo putem HTML-a 5 (WebGL ili CSS), vidljivo na slici 24. [10]



Slika 24. 3d model mikrofona renderiran u realnom vremenu na web stranici Sketchfab u pregledniku Firefox

6. ZAKLJUČAK

Osim za već postojeće primjene u arhitekturi, medicini, kemijskoj i ostalim industrijama, opisane su druge moguće primjene interaktivnih vizualizacija.
Kao što je vidljivo u teorijskom dijelu rada, za izradu kvalitetne interaktivne 3d vizualizacije osim 3d modeliranja i teksturiranja potrebno je znanje korištenja 3d pokretača i pomni odabir istog ovisno o cilju koji se želi postići aplikacijom.
Svi operativni sustavi i uređaji na koje je moguće prenijeti aplikaciju imaju svoje prednosti i nedostatke koje valja uzeti u obzir kod biranja platforme na kojoj će se aplikacija koristiti.
U praktičnom dijelu rada prikazan je razvoj i praktična primjena interaktivne vizualizacije na primjeru unutrašnjosti prostora Grafičkog fakulteta u svrhu demonstriranja kompleksnosti izrade interaktivnih 3d vizualizacija i njihovih mogućih primjena.

Sagleda li se potencijal aplikacije sa dizajnerskog stajališta, može se proširiti od običnog hodanja kroz zgradu do nečeg mnogo funkcionalnijeg i korisnijeg za korisnika. Kao primjer minimalne dodatne funkcionalnosti, u vizualizaciji su već dodani natpisi ispred vrata prostorija, što može pomoći korisniku u kretanju kroz zgradu. Integracijom mrežnog sustava kako bi se aplikacija mogla ažurirati putem interneta moguće je uvelike proširiti funkcionalnost. Plakati u simulaciji mogli bi prezentirati studentske radove, uz mogućnost mijenjanja radova kroz određeni period vremena. Prostor za novosti mogao bi prikazivati obavijesti sa web stranice fakulteta.

Budući da NeoAxis Engine podržava samo Windows i Mac OS X operativne sustave, prenošenje aplikacije u drugi pokretač (primjerice Unity) omogućilo bi kompiliranje aplikacije za iOS i Android uređaje, čime bi korisnici bili u mogućnosti koristiti aplikaciju neovisno o tome gdje se nalaze. Za uređaje sa sučeljem na dodir (mobiteli, tableti) bilo bi potrebno pojednostavniti način na koji korisnik unosi komande zbog nedostatka tipkovnice i miša. Prenošenje aplikacije na druge operativne sustave donosi niz različitih problema ovisno o operativnom sustavu na koji se aplikacija prenosi.

7. LITERATURA

1. ***http://www.neoaxis.com/showcase/products/systemotehnika_simulator - 3. 8. 2015.

2. ***http://www.hyperviz.com/showroom/ - 3. 8. 2015.

3.***http://blog.nobel-joergensen.com/2011/04/05/procedural-generated-mesh-in-unity-part-2-with-uv-mapping/ - 21. 7. 2015.

4.***http://sansajairo.blogspot.com/ - 21. 7. 2015.

5.***https://www.unrealengine.com/what-is-unreal-engine-4 - 24. 7. 2015.

6.***https://unity3d.com/unity - 24. 7. 2015.

7.***http://www.neoaxis.com/neoaxis/overview - 24. 7. 2015.

8.***http://www.pcgamer.com/pc-graphics-options-explained/ - 28. 7. 2015.

9.***http://www.emulator-zone.com/doc.php/help/commondefs.html - 3. 8. 2015.

10. ***https://sketchfab.com/models/f811bcc95a364df3bfb56ab11d5639a2 - 26. 8. 2015.