UNIVERZITET U NIŠU
ELEKTRONSKI FAKULTET

Aleksandar M. Dimitrijević

RINGO - ARHITEKTURA ZA TRODIMENZIONALNU VIZUELIZACIJU TERENA VELIKIH RAZMERA

doktorska disertacija

Niš, 2015.

UNIVERZITET U NIŠU
ELEKTRONSKI FAKULTET

Aleksandar M. Dimitrijević

RINGO – ARHITEKTURA ZA TRODIMENZIONALNU VIZUELIZACIJU TERENA VELIKIH RAZMERA

doktorska disertacija

Niš, 2015.

UNIVERSITY OF NIŠ
FACULTY OF ELECTRONIC ENGINEERING

Aleksandar M. Dimitrijević

RINGO – ARCHITECTURE FOR THREE-DIMENSIONAL LARGE SCALE TERRAIN VISUALIZATION

doctoral dissertation

Niš, 2015.

Podaci o mentoru i članovima komisije

Mentor: prof. dr Dejan D. Rančić, Univerzitet u Nišu, Elektronski fakultet

Član komisije:

Član komisije:

Član komisije:

Član komisije:

Datum odbrane:

Podaci o doktorskoj disertaciji

Naslov doktorske disertacije

RINGO – arhitektura za trodimenzionalnu vizuelizaciju terena velikih razmera

Rezime

Ova doktorska disertacija predstavlja originalni pristup u vizuelizaciji terena planetarnih razmera. Razvijen je algoritam i prateća softverska arhitektura koja omogućuje geodetski ispravan i precizan prikaz čitave planete u realnom vremenu, bez obzira na rastojanje posmatrača od njene površine. Verifikacija i validacija rezultata ostvarena je korišćenjem formalnih i preciznih metoda, kroz praktičnu implementaciju sa realnim podacima za planetu Zemlju, definisanje metrike, pribavljanje numeričkih vrednosti i konačno publikovanje rezultata u specijalizovanom naučnom časopisu.

Vizuelizacija je bazirana na modifikovanom algoritmu geometrijskih klipmapa. Predložena modifikacija omogućuje generisanje subpikselski preciznog WGS84 referentnog elipsoida na grafičkom procesoru, primenom aritmetike jednostruke tačnosti. Bez obzira na ograničenu preciznost ovakve aritmetike, predloženi metod omogućuje veoma brzu, preciznu i konzistentnu vizuelizaciju WGS84 referentnog elipsoida od kosmičkih do ekstremno malih rastojanja u odnosu na njegovu površinu. Podela elipsoida na tri particije, uz primenu cilindričnih ekvidistantnih projekcija za svaku od particija, minimizuje distorziju osnovne rešetke nad kojom je uzdignut teren, a time i distorziju primenjenih tekstura. Opisani algoritam omogućuje visoku i stabilnu frekvenciju osvežavanja prikaza, čak i pri korišćenju skromnijeg hardvera, uz vrlo nisko zauzeće centralnog i efikasnu primenu grafičkog procesora. Predložen je i efikasan metod i prateći softverski protočni sistem za pribavljanje i ažuriranje tekstura, čime je vizuelizacija učinjena nezavisnom od rezolucije primenjenog teksturnog-pokrivača.

Višegodišnji proces istraživanja metoda vizuelizacije terena pratio je konstantni razvoj i

unapređenje odgovarajućeg softvera, koji je omogućavao validaciju svih ideja nastalih tokom istraživanja. Softver je razvijan u programskom jeziku C++, u MS Visual Studio okruženju. Kao interfejs prema grafičkom podsistemu korišćen je OpenGL API. Zbog specifičnosti implementacije, neophodan je OpenGL u verziji 3.3 ili noviji, kao i grafički hardver koji to podržava. U procesu vizuelizacije korišćeni su realni podaci dobijeni satelitskim i avionskim snimanjem površine planete Zemlje, kao i trenutno najprecizniji modeli i podaci vezani za procenu njene veličine i oblika.

Verifikacija preciznosti implemetacije izvršena je na osnovu uvedene metrike i vrednosti preuzetih nakon transformacija na grafičkom procesoru. Efikasnost implementacije procenjena je korišćenjem specijalne, u tu svrhu razvijene biblioteke, koja je u stanju da precizno izmeri vreme proteklo između dva funkcijska poziva, kako na centralnom, tako i na grafičkom procesoru. Obzirom da ova dva procesora rade potpuno asinhrono, neophodno je uporedno praćenje njihovih vremena izvršenja. Osim toga, performanse značajno zavise od stanja u kome se sistem nalazi. Zbog toga su u analizu uključeni i drugi parametri i stanja grafičkog podsistema, pribavljeni korišćenjem specijalizovanih biblioteka proizvođača hardvera. Dodatno je izvršeno i poređenje efikasnosti predloženog algoritma i njegove implementacije sa trenutno vrhunskim algoritmima u ovoj oblasti.

Konačnu verifikaciju predloženih ideja i implementacije predstavlja prihvatanje i publikovanje rezultata istraživanja u eminentnom časopisu koji se bavi računarskom grafikom i vizuelizacijom - Computers & Graphics Journal (UK).

Ključne reči

Teren, vizuelizacija, arhitektura, ažuriranje tekstura, nivo detalja, klipmape

Naučna oblast

Tehnološke nauke

Uža naučna oblast

Sistemski inženjering i računarska tehnologija

UDK broj i klasifikaciona oznaka za datu naučnu oblast

T 120

Hronologija odobravanja teme doktorske disertacije

Imenovanje Komisije za ocenu naučne zasnovanosti teme doktorske disertacije

Na I sednici Nastavno-naučnog i Izbornog veća Elektronskog fakulteta u Nišu (zapisnik br. 07/03-003/11-001), održanoj 05.11.2010. godine, imenovana je Komisija za ocenu naučne zasnovanosti teme doktorske disertacije kandidata mr Aleksandra Dimitrijevića pod naslovom „3D vizuelizacija terena velikih razmera u specijalizovanim informacionim sistemima".

Usvajanje Izveštaja Komisije za ocenu naučne zasnovanosti teme doktorske disertacije

Na II sednici Nastavno-naučnog i Izbornog veća Elektronskog fakulteta u Nišu (zapisnik br. 07/01-003/11), održanoj 23.12.2010. godine, razmotren je i usvojen Izveštaj Komisije za ocenu naučne zasnovanosti teme doktorske disertacije kandidata mr Aleksandra Dimitrijevića pod naslovom „RINGO – arhitektura za trodimenzionalnu vizuelizaciju terena velikih razmera".

Saglasnost Naučno-stručnog veća za tehničko-tehnološke nauke Univerziteta u Nišu na usvajanje teme doktorske disertacije

Naučno-stručno veće za tehničko-tehnološke nauke Univerziteta u Nišu, na sednici održanoj 31.01.2011. godine, dalo je saglasnost (NSV 8/20-01-001/11-009) na Odluku Fakulteta o usvajanju teme doktorske disertacije pod naslovom „RINGO – arhitektura za trodimenzonalnu vizuelizaciju terena velikh razmera" kandidata mr Aleksandra Dimitrijevića.

Objavljivanje saglasnosti Naučno-stručnog veća za tehničko-tehnološke nauke Univerziteta u Nišu na usvajanje teme doktorske disertacije

Na III sednici Nastavno-naučnog i Izbornog veća Elektronskog fakulteta u Nišu (zapisnik br. 07/01-004/11-006), održanoj 24.02.2011. godine, objavljena je saglasnost Naučno-stručnog veća za tehničko-tehnološke nauke Univerziteta u Nišu na Odluku o usvajanju teme doktorske disertacije kandidata mr Aleksandra Dimitrijevića pod naslovom „RINGO – arhitektura za trodimenzionalnu vizuelizaciju terena velikih razmera".

Doctoral Dissertation Data

Title

RINGO – architecture for the three-dimensional large scale terrain visualization

Abstract

This doctoral dissertation presents an original approach to the planet-sized terrain visualization. The developed algorithm and associated software architecture allow geodetically correct and precise representation of the entire planet in real time, regardless of the viewer's distance from its surface. Verification and validation of the results are achieved by the use of formal and precise method, through the practical implementation based on the real data for the planet Earth, defining metrics, obtaining numerical values and finally publishing the results in a specialized scientific journal.

The visualization is based on the modified geometry clipmap algorithm. The proposed modification enables the generation of sub-pixel precise WGS84 reference ellipsoid on the graphics processor, using single precision arithmetic. Regardless of the limited accuracy of this arithmetic, the proposed method allows very fast, accurate and consistent visualization of the WGS84 reference ellipsoid, from cosmic to extremely small distances to its surface. Dividing the ellipsoid into three partitions, with the use of equidistant cylindrical projection for each of the partitions, minimizes the distortion of the basic grid upon which the terrain is elevated, and thus the distortion of the applied textures. The described algorithm allows high and stable frame-rate, even when modest hardware is used, with a very low utilization of the CPU and an effective usage of the GPU. An efficient method for texture streaming and accompanying software pipeline are proposed, making the visualization independent of the applied texture-coverage resolution.

A long-term research of terrain visualization methods has been followed by the constant development and improvement of appropriate software, which allowed the validation of all the

ideas generated during the research. The software is developed in C++ programming language, using MS Visual Studio environment. As an interface to the graphics subsystem, the OpenGL API is used. Due to the specific implementation, OpenGL version 3.3 or later is required and graphics hardware that supports it. The visualization uses real data gathered by satellite and aerial remote sensing, as well as currently the most accurate models and data related to the evaluation of the size and shape of the planet Earth.

Verification of the implementation accuracy was based on established metrics and values taken after the transformation on the graphics processor. The efficiency of implementation is estimated using a special library, developed for this purpose, that is able to accurately measure the time elapsed between two function calls, both at the central and the graphics processor. Since these two processors are completely asynchronous, it is necessary to monitor their execution time in parallel. The performance significantly depends on the current state of the system, hence, the analysis includes other parameters and graphics subsystem states, obtained by specialized library provided by hardware manufacturers. In addition, the efficiency of the proposed algorithm and its implementation are compared with the current state-of-the-art algorithms in this field.

The final verification of the proposed ideas and implementation represents the acceptance and publication of research results in the eminent journal for computer graphics and visualization - Computers & Graphics Journal (UK).

Keywords

terrain, visualization, architecture, texture streaming, level-of-detail, out-of-core, clipmap

Scientific Area

Technological sciences

Scientific Sub-Area

Systems engineering and computer technology

UDK and Classification Code for the Scientific Sub-Area

T 120

Chronology of the doctoral dissertation approval

Appointment of the Commission for the evaluation of the scientific basis of the doctoral dissertation

In the first sessions of the Academic and Election Council of the Faculty of Electronic Engineering in Niš (record no. 07/03-003/11-001), held on November 5th 2010, the Committee was appointed for the evaluation of the scientific basis of the doctoral dissertation proposed by MSc Aleksandar Dimitrijević and entitled „3D large-scale terrain visualization in specialized information systems".

Approval of the Report of the Commission for the evaluation of the scientific basis of the doctoral dissertation

In the second sessions of the Academic and Election Council of the Faculty of Electronic Engineering in Niš (record no. 07/01-003/11), held on December 23rd 2010, the Report of the Commission was approved on the evaluation of the scientific basis of the doctoral dissertation proposed by MSc Aleksandar Dimitrijević and entitled „RINGO – architecture for the three-dimensional large scale terrain visualization".

The consent of the Scientific Expert Council for Technical and Technological Sciences of the University of Niš on the adoption of the doctoral dissertation

The Scientific Expert Council for Technical and Technological Sciences of the University of Niš, at the meeting held on January 31st 2011, gave its consent (NSV 8/20-01-001/11-009) on the Faculty's decision on the adoption of the doctoral dissertation entitled „RINGO – architecture for the three-dimensional large scale terrain visualization" for the candidate MSc Aleksandar Dimitrijević.

Disclosure of the consent of the Scientific Expert Council for Technical and Technological Sciences of the University of Niš on the adoption of the doctoral dissertation

In the third session of the Academic and Election Council of the Faculty of Electronic Engineering in Niš (record no. 07/01-004/11-006), held on February 24th 2011, it was announced the approval of the Scientific Expert Council for Technical and Technological Sciences of the University of Niš to the Decision on approval of the doctoral dissertation for the candidates MSc

Aleksandar Dimitrijević entitled „RINGO – architecture for the three-dimensional large scale terrain visualization”.

Sadržaj

Slike

Tabele

Uvod

Vizuelizacija terena predstavlja važan aspekt mnogih aplikacija, počevši od 3D geografskih informacionih sistema, preko simulatora i igara, do 3D virtuelnog turizma. Neke od njih koriste ravnu osnovu nad kojom je superponiran reljef, obezbeđujući jednostavnu i brzu implementaciju. Međutim, zanemarivanje zakrivljenosti površine planete ima za posledicu netačan prikaz, koji postaje vrlo uočljiv kada se posmatrač nalazi visoko iznad terena ili gleda u daljinu. Druge aplikacije, pak, koriste sfernu aproksimaciju. Ona predstavlja prilično konzistentnu vizuelnu reprezentaciju, ali dovodi do nepreciznog geodetskog pozicioniranja. Usaglašavanje načina predstavljanja podataka i potrebe za efikasnim protokom, uz primenu izabranog matematičkog modela na čitavu planetu i očuvanje zahtevane preciznosti, još uvek je izazov za sve algoritme za prikaz terena velikih razmera.

Ova doktorska disertacija predstavlja originalni pristup u vizuelizaciji terena planetarnih razmera. Osnovni zadatak je prikaz planete Zemlje u realnom vremenu, baziran na WGS84 (eng. *World Geodetic System 1984*) referentnom elipsoidu [76]. Elipsoid je podeljen u tri particije, bešavno spojene u jedinstvenu celinu, pri čemu svaka od particija koristi novi pristup u primeni geometrijskih klipmapa [64, 6] za superponiranje visina terena na elipsoidnu rešetku. Rešetka se generiše u letu na grafičkom procesoru, tačnije, u okviru verteks šejdera, korišćenjem aritmetike realnih brojeva jednostruke preciznosti. Predloženi metod garantuje subpikselsku preciznost izračunavanja i prikaza referentnog elipsoida, bez obzira koliko je posmatrač udaljen od njegove površine. Implementacija zasnovana na klipmapama omogućuje konzistentno keširanje, brz protok i predstavljanje podataka na način pogodan za korišćenje na grafičkom procesoru, dok primena ekvidistantne cilindrične projekcije (takođe poznate kao *geografska* ili *lat/lon*) za izvorne podatke minimizuje pretprocesiranje i obezbeđuje nizak-do-srednji nivo distorzije tekstura.

Autor ove disertacije se bavi vizuelizacijom terena velikih razmera već više od deset godina. Prva verzija algoritma objavljena je u radovima [90, 91] 2006. godine pod zvučnim nazivom – RINGO. Naziv je potekao od prstenaste organizacije blokova na odgovarajućem nivou

detalja. Iako je ova verzija algoritma bila prilično efikasna [30] i implementirana u više razvojnih i komercijalnih projekata, autor potpuno odustaje od nje početkom 2011. godine. U spomen na ove početke, kao i zbog činjenice da je teza prijavljena u vreme dok je prvobitna verzija algoritma bila aktuelna, naziv RINGO se javlja i u naslovu ove disertacije. Od 2011. godine, autor korenito menja pristup u vizuelizaciji, omogućavajući mnogo dinamičnije kreiranje terena i utirući put prikazu geodetski preciznog terena planetarnih razmera. Da je promena bila opravdana potvrđuje prihvatanje i publikovanje rada [34], koji opisuje osnovne postavke novog algoritma i dokazuje njegovu preciznost i efikasnost, u eminentnom časopisu za računarsku grafiku. Iako nivoi detalja i dalje imaju prstenastu strukturu i podeljeni su u blokove, zbog podrške za vizuelizaciju elipsoidnih planeta i preciznije odrednice da se radi o algoritmu zasnovanom na geometrijskim klipmapama, nova verzija algoritma nosi naziv – *Elipsoidne KlipMape* (EKM).

Vizuelizacija terena implementirana je korišćenjem OpenGL [96] programskog interfejsa aplikacije (eng. *Application Programming Interface* – API) prema grafičkom hardveru. OpenGL je nastao 1992. godine i vrlo brzo postao široko prihvaćeni industrijski standard za računarsku grafiku visokih performansi. Do verzije 2.0, OpenGL je podržavao samo tzv. „fiksnu funkcionalnost", tj. predefinisani skup algoritama za vizuelizaciju. Od verzije 2.0, uvedena je podrška za programsko upravljanje izvršenjem pojedinih operacija na grafičkom procesoru. Jezik koji omogućuje pisanje takvog programskog koda, a koji je kompatibilan sa OpenGL programskim interfejsom, naziva se *OpenGL Shading Language* (GLSL) [56]. Nezavisne kompilacione jedinice napisane u ovom jeziku nazivaju se šejderi (eng. *shader*).

Šejderi nisu vezani samo za OpenGL, već su karakteristični za sve savremene grafičke sisteme, a vode poreklo od Piksarovog (eng. *Pixar*) softvera za izradu vizuelnih specijalnih efekata za filmsku industriju – *RenderMan* [5]. *RenderMan* je još krajem 80-tih godina prošlog veka koristio poseban jezik za generisanje fotorealističnih slika kroz specifikaciju osobina materijala i svetlosti. Sa razvojem grafičkog hardvera i otvaranjem njegovog protočnog sistema za programiranje, početkom ovog veka, stvoreni su uslovi za pojavu novih proceduralnih jezika za pisanje šejdera, koji se izvršavaju u realnom vremenu [88]. Savremeni GLSL podržava više tipova šejdera, i to: verteks šejdere (za obradu pojedinačnih temena), teselacione šejdere (za dodavanje nivoa detalja), geometrijske šejdere (za rukovanje čitavim primitivama), fragment šejdere (za obradu pojedinačnih fragmenata/piksela) i šejdere za izračunavanje. Osnovna varijanta algoritma za vizuelizaciju terena, predstavljena u ovoj disertaciji, koristi samo verteks i fragmet šejdere. Verteks šejder služi za kreiranje geometrije na grafičkom procesoru, na osnovu teksture koja

u sebi sadrži samo visine terena, a fragment šejder omogućuje primenu teksturnog-pokrivača sastavljenog od satelitskih i avionskih ortografskih snimaka.

Disertacija je organizovana u devet poglavlja. Prva dva poglavlja daju premise za razvoj algoritma kroz pregled postojećih rešenja i načina predstavljanja planete Zemlje. Naredna četiri poglavlja opisuju sam algoritam vizuelizacije. Evaluacija predloženog rešenja data je u poglavljima 7 i 8, dok poslednje poglavlje donosi konačne zaključke. U nastavku sledi detaljniji pregled sadržine kroz kratak opis svakog od poglavlja.

Poglavlje 1 predstavlja istoriju razvoja algoritama za vizuelizaciju terena velikih razmera. Prikazana su rešenja koja su bila okosnice razvoja, ali i algoritmi koji su i danas aktuelni. Najveći deo poglavlja posvećen je klipmapama, njihovim glavnim karakteristikama u svakoj novoj inkarnaciji algoritma, kao i nedostacima koji su prouzrokovali dalja unapređenja.

Poglavlje 2 opisuje proces predstavljanja planete Zemlje. Definisan je referentni elipsoid, kao osnovni matematički model, korekcija tog modela kroz odstupanje geomagnetne ekvipotencijalne površine (model geoida) i konačno definisanje površine superponiranjem reljefa. Drugi deo poglavlja posvećen je projekcijama koje se koriste za organizaciju i pristup izvornim podacima.

Poglavlje 3 je prvo od četiri poglavlja koja opisuju originalni pristup u vizuelizaciji terena planetarnih razmera, zasnovanoj na elipsoidnim klipmapama. U ovom poglavlju prikazana je struktura elipsoidnih klipmapa, podela planete na particije, blokovska organizacija particija, kao i koordinatni sistemi koji se koriste prilikom pristupa podacima i u procesu vizuelizacije. Data je i globalna struktura softverskog protočnog sistema koji definiše sam proces vizuelizacije terena korišćenjem elipsoidnih klipmapa. Tri strukturne celine ovog protočnog sistema detaljno su obrađene u zasebnim poglavljima.

Poglavlje 4 prikazuje deo softverskog protočnog sistema za vizuelizaciju terena koji se izvršava na grafičkom procesoru. Opisan je proces generisanja blokova u verteks šejderu i primena teksturnog-pokrivača u fragment šejderu.

Poglavlje 5 prikazuje deo softverskog protočnog sistema za vizuelizaciju terena koji se izvršava na centralnom procesoru. Opisan je algoritam koji izvršava nit za vizuelizaciju, kroz osnovne zadatke koje obavlja pri svakom osvežavanju prikaza, kao i postupak iscrtavanja particija i matrice-blokova.

Poglavlje 6 objašnjava proces ažuriranja klipmapa, na način realizovan u okviru predloženog algoritma za vizuelizaciju terena. Detaljno je prikazan softverski protočni sistem za pribavljanje tekstura, kao i način predstavljanja podataka. Predloženo rešenje sprečava da veli-

čina primenjenih tekstura utiče na vreme iscrtavanja, čime omogućuje vizuelizaciju korišćenjem teksturnih-pokrivača visoke rezolucije.

Poglavlje 7 potvrđuje preciznost generisanja rešetke zasnovane na WGS84 referentnom elipsoidu. Rešetka je osnova blokova u odnosu na koju se superponira reljef. Obzirom da se ona formira u okviru verteks šejdera, uz korišćenje aritmetike jednostruke tačnosti, važno je pokazati da je moguće ostvariti vizuelizaciju referentnog elipsoida, na bilo kojoj udaljenosti od njegove površine, sa subpikselskom preciznošću.

Poglavlje 8 prikazuje detalje vezane za efikasnost implementacije. Dati su rezultati ispitivanja na konkretnoj računarskoj opremi, kao i poređenje sa drugim, trenutno aktuelnim, algoritmima i aplikacijama za vizuelizaciju terena velikih razmera.

Poslednje poglavlje predstavlja zaključak čitavog rada, pregled doprinosa i prednosti predloženog algoritma, kao i sugestija za dalja istraživanja i unapređenja. Iza zaključka slede: bibliografija, prilog koji sadrži kompletan programski kod verteks i fragment šejdera geometrijskih klipmapa i kratka biografija autora.

Poglavlje 1

Algoritmi za vizuelizaciju terena

Čini se da je vizuelizacija terena mnogo jednostavnija od prikaza proizvoljnog 3D objekta, pre svega zbog njegove strukture i veoma ograničene geometrije. Međutim, kontinualna priroda terena i osobina da se vidi u gotovo svakom delu scene, uz prostiranje u daljinu dokle god pogled seže, nameće visoke zahteve specijalizovanim algoritmima za vizuelizaciju. Oni moraju, zavisno od usmerenja pogleda i daljine pojedinih delova terena, optimizovati prikaz korišćenjem različitih nivoa detalja (eng. *level-of-detail* (LOD)) i eliminisati delove koji nisu u vidnom polju, kako bi ostvarili potreban kvalitet prikaza i nivo performansi.

1.1 Kratka istorija razvoja

Rani metodi vizuelizacije vršili su dekompoziciju terena u međusobno susedne blokove, od kojih je svaki predstavljen u nekoliko različitih nivoa detalja. Blokovi su smeštani na disku i učitavani u matricu blokova u operativnoj memoriji, u skladu sa kretanjem posmatrača [37]. Iscrtavanje blokova se vrši samo ukoliko su oni u vidnom polju, a rezolucija je birana u zavisnosti od rastojanja u odnosu na posmatrača. Algoritmi koji su usledili, unapredili su iscrtavanje korišćenjem kvad-stabala (eng. *quadtree*) za predstavljanje terena i mnogo kompleksnije metrike za izbor nivoa detalja [62].

U narednim godinama pojavilo se nekoliko novih tehnika, baziranih na restriktivnim triangulacijama kvad-stabala [85] ili binarnih stabala [35]. Predloženi su vrlo napredni koncepti spajanja i podela trouglova, kao i metrika grešaka zasnovana na pogledu, čime je obezbeđena reprezentacija terena sa kontinualnom promenom nivoa detalja. Uprkos svojim naprednim konceptima, algoritmi sa kontinualnom promenom nivoa detalja nameću veliko opterećenje

centralnom procesoru i zahtevaju kontinualni transfer podataka ka grafičkim karticama. Zbog toga, nakon napretka grafičkog hardvera, oni postaju zastareli i bivaju zamenjeni algoritmima zasnovanim na grupnoj triangulaciji (eng. *clustered triangulation*), kod kojih se nivo detalja definiše na nivou blokova [102, 20, 21]. Optimalna triangulacija žrtvovana je u korist obrade grupe podataka. Korišćenje unapred pripremljenih blokova optimizovane geometrije [102], organizovanih u kvad-stabla, još uvek je vrlo popularna tehnika u vizuelizaciji masivnih terena [23, 55], pogotovu kada se podaci pribavljaju sa udaljenih servera. Pukotine između blokova različitih nivoa detalja često se popunjavaju spuštanjem ivica, bez primoravanja susednih blokova da se uzajamno podudaraju na svojim granicama.

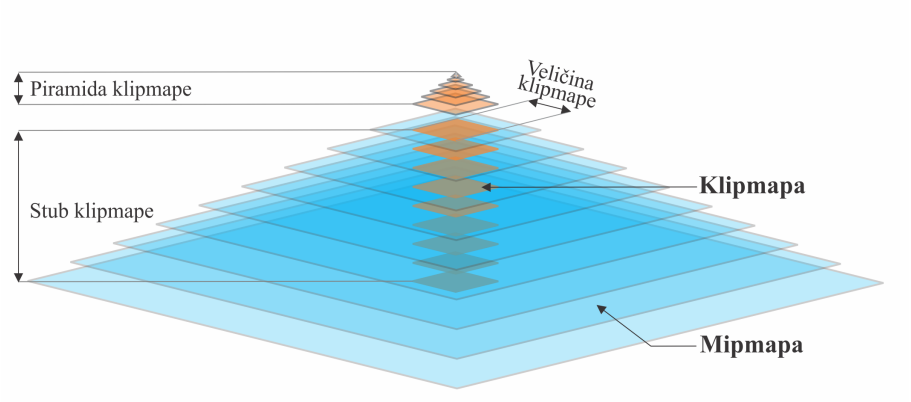
Dalji napredak grafičkog hardvera doneo je nove algoritme sa kontinualnom promenom nivoa detalja, zasnovane na hardverskoj teselaciji [16]. Izračunavanje nivoa detalja potpuno je preneto na stranu grafičkog procesora, čime je omogućeno kreiranje površina bez pukotina na bazi kvadratnih blokova poravnatih sa koordinatnim osama, modifikovanih uzorcima iz mape visina. Faktori teselacije se usklađuju na granici kvadratnih blokova, mogu se lako menjati u skladu sa metrikom greške i omogućuju formiranje trouglova uniformnih veličina u prostoru ekrana, saglasno sa izabranim odnosom performansi i detaljnosti scene.

Ekrani vrlo visoke rezolucije, u kombinaciji sa vrlo preciznom metrikom greške u prostoru ekrana, mogu zahtevati prikaz ogromnog broja trouglova, koji značajno nadmašuje mogućnosti rasterizacije trenutnog hardvera. U tom slučaju, metode zasnovane na emitovanju zraka (eng. *ray-casting*) mogu prevazići ovo usko grlo i omogućiti višu frekvenciju iscrtavanja [25]. Obzirom da emitovanje zraka, u većini slučajeva, ne nadmašuje po performansama standardnu rasterizaciju, predložen je i hibridni metod [26]. Hibridni metod je zasnovan na blokovski organizovanom hijerarhijskom terenu, gde se svaki blok prikazuje zasebno, u zahtevanoj rezoluciji, metodom koja je odabrana na bazi procene vremena iscrtavanja.

1.2 Klipmape

Drugi tok veoma popularnih tehnika za vizuelizaciju terena baziran je na teksturnim *klipmapama* [100]. Termin klipmapa odnosi se na dinamičku reprezentaciju teksture, koja se koristi da efikasno kešira proizvoljno veliku količinu podataka u konačnom prostoru fizičke memorije. Implementira se kao parcijalna reprezentacija mipmape [107], koja se može kontinualno ažurirati i čiji je svaki nivo odsečen na specificiranu veličinu, pretvarajući piramidu mipmape u

obelisk (Sl. 1.1). Bez gubitka kvaliteta kompletne mipmape, klipmapa raste linearno sa svakim novim nivoom, umesto eksponencijalnog rasta kompletne mipmape. Nivoi klipmape grupisani su u dva skupa: *piramidu klipmape* – skup statičkih nivoa niske rezolucije smeštenih u formi potpune mipmape, i *stub klipmape* – skup dinamičkih nivoa više rezolucije, centriranih u odnosu na fokalnu tačku, koji se toroidno ažuriraju sa pomeranjem date tačke.

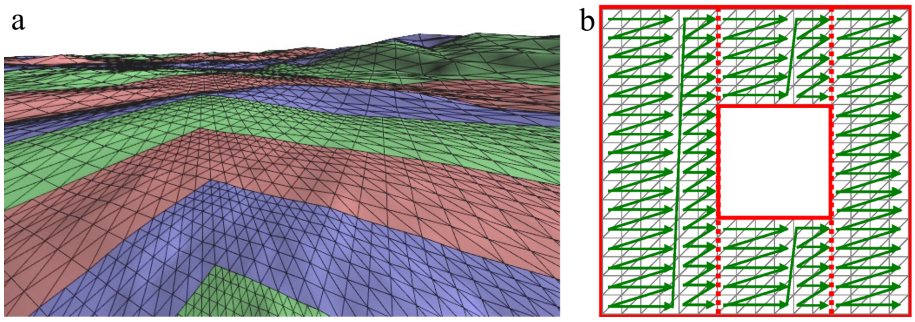


Slika 1.1: Klipmapa – dinamička reprezentacija parcijalne mipmape.

Klipmape su, kao koncept, po prvi put uvedene 1998. godine, u radu Tanera (eng. *Tanner*) i njegovih saradnika [100]. U tom radu, autori opisuju anatomiju klipmapa i predstavljaju implementaciju u kontekstu njihovog grafičkog softverskog alata visokog nivoa – *IRIS Performer*-a, kasnije poznatog pod nazivom *SGI OpenGL Performer* [36]. To je bio zaista revolucionaran koncept, ali je implementacija zahtevala specijalizovani hardver, ograničavajući primenu na *InfiniteReality* sisteme. Ovi sistemi rukovali su klipmapama na isti način kao da se radi o regularnoj mipmapi, što je omogućilo visoke performanse i kvalitet vizuelizacije. Numerički opseg i preciznost hardvera za rukovanje teksturama nametnuli su uvođenje virtuelnih klipmapa. Virtuelizacija je podrazumevala čuvanje samo podskupa nivoa klipmapa u operativnoj memoriji (odgovarajućih 16 nivoa kompletnog skupa) i dodavanje skaliranja i translacije u procesu obrade teksturnih koordinata.

Iako je opis rukovanja teksturama bio vrlo kompletan, u radu Tanera i njegovih saradnika nije razmatrana mogućnost korišćenja klipmapa za rukovanje geometrijom terena. To je po prvi put predloženo šest godina kasnije, u radu [64] dvojice Majkrosoftovih (eng. *Microsoft*) istraživača: Frenka Losasoa (eng. *Frank Losasso*) i Hjuz Hopija (eng. *Hugues Hoppe*). Za razliku od originalnih klipmapa, *geometrijske klipmape* nisu zahtevale poseban hardver i fokusirale su se na generisanje geometrije korišćenjem šejder-model 2 (SM2) grafičkih kartica opšte namene. Osim proširenja koncepta klipmapa na rukovanje geometrijom (Sl. 1.2), geometrijske

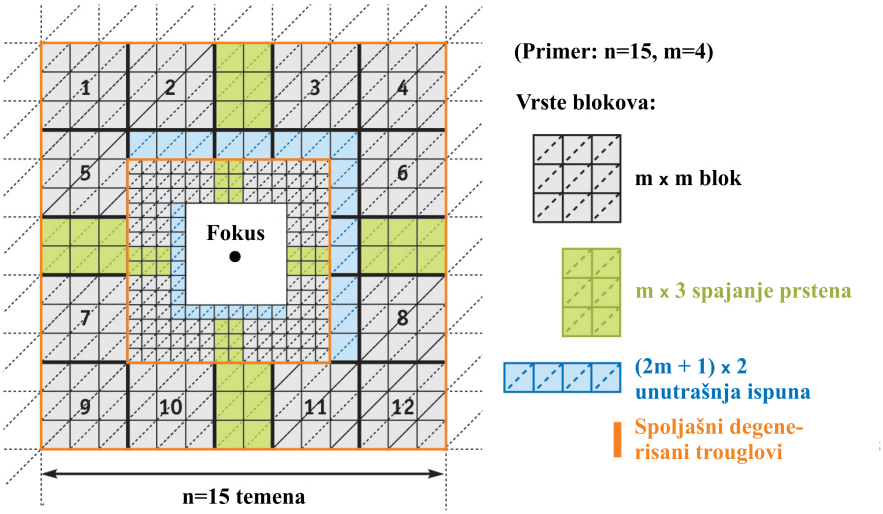
klipmape demonstrirale su i vrlo efikasne metode dekompresije i sinteze, omogućujući direktno smeštanje velikih količina podataka u operativnu memoriju, izbegavajući time učitavanje sa sporih medijuma. Korišćenje SM2 hardvera onemogućilo je očitavanje tekstura u verteks šejderu, nameću̇ći toroidno ažuriranje bafera temena umesto tekstura, a takođe i ažuriranje kompletnog indeksnog bafera pri svakom pokretu posmatrača.



Slika 1.2: Izgled terena kreiranog geometrijskim klipmapama sa nivoima detalja označenim različitim bojama (a) i postupak formiranja trouglova u okviru jednog nivoa detalja. Delovi slike preuzeti su iz [64].

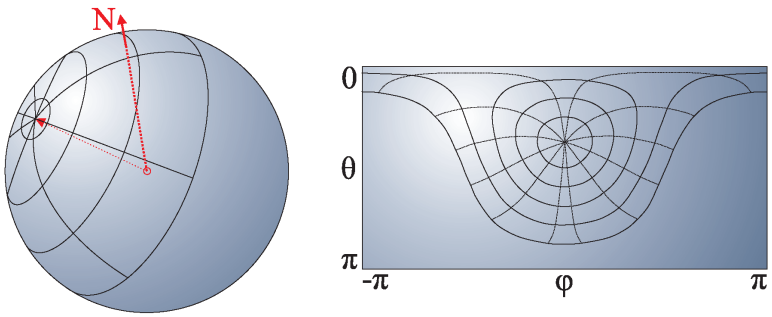
Očitavanje tekstura u verteks šejderu i neka druga unapređenja SM3 hardvera omogućila su gotovo potpunu implementaciju geometrijskih klipmapa na grafičkom procesoru, kao što je opisano u radu Asirvatama (eng. *Asirvatham*) i Hopija [6] iz 2005. godine. Jedina operacija koja je ostala na strani centralnog procesora bila je dekompresija podataka. Baferi temena za smeštanje podataka o visinama zamenjeni su dvodimenzionalnim (2D) teksturama, a mali skup statičkih bafera temena i indeksa korišćeni su za definisanje rešetke u odnosu na koju je izdignut teren (Sl. 1.3). Implementacija je bila vrlo efikasna i postala je inspiracija mnogim autorima u narednim godinama. Jedina ozbiljna ograničavajuća karakteristika algoritma bilo je korišćenje ravne 2D rešetke. Naime, originalni algoritam nije mogao da se koristi za vizuelizaciju terena planetarnih razmera. Iako je interpretacija 2D rešetke u verteks šejderu kao lat/lon rešetke mogla biti korišćena za prikaz delova sfernog terena, dva značajna problema su onemogućila širu primenu, i to: nedovoljna preciznost za rukovanje gustim rešetkama, gušćim od 3 lučne sekunde po koraku rešetke, i skupljanje u pravcu polova [23, pog.13].

Tokom 2006. godine, samo godinu dana nakon publikacije geometrijskih klipmapa implementiranih na grafičkom procesoru, dvojica istraživača sa berlinskog Zuse instituta: Malte Klasen (nem. *Malte Clasen*) i Hans-Kristijan Hege (nem. *Hans-Christian Hege*) predložili su novu reviziju algoritma, pogodnu za terene planetarnih razmera [22]. Algoritam je zasnovan na činjenici da bez obzira koliko je posmatrač udaljen od planete, on može videti samo jednu hemisferu. Zato je rešetka formirana u obliku kružnih segmenata jedinične hemisfere, sa severnim



Slika 1.3: Struktura jednog nivoa detalja terena, pri implementaciji geometrijskih klipmapa na grafičkom procesoru. Slika je preuzeta iz [6] uz prevod originalnih termina sa engleskog jezika.

polom na poziciji posmatrača (Sl. 1.4). Definisane su i odgovarajuće transformacije koordinata za preslikavanje globalnog u lokalni koordinatni sistem. Glavni problem ovog algoritma bio je nedostatak korespondencije između temena u rešetki i uzoraka izvornih podataka (u geografskoj projekciji), što je sprečilo njegovo šire prihvatanje. Dodatni problem predstavlja anizotropija uzorkovanja, a veličina nivoa klipmapa zavisi od geografske širine posmatrača.



Slika 1.4: Sferne klipmape. Osnovna rešetka je u obliku hemisfere, centrirane na poziciji posmatrača, dok su izvorni podaci u geografskoj (lat/lon) projekciji. Delovi slike preuzeti su iz [22].

Sa pojavom SM4 hardvera, krajem 2006. godine, grafičke kartice opšte namene konačno postaju sposobne da potpuno implementiraju klipmape korišćenjem *polja tekstura* [12]. Polje tekstura je kolekcija 1D ili 2D tekstura identičnih dimenzija i formata, organizovanih u slojeve, kojima se pristupa kao jedinstvenoj celini korišćenjem šejdera. Minimalni broj slojeva propisanih specifikacijom bio je 64, međutim, čak i niskobudžetne kartice podržavale su do 512 slojeva. Virtuelizacija predložena u [100] postaje nepotrebna. Demonstracija novih sposob-

nosti i popularnost klipmapa uticali su na uključivanje odgovarajućeg demo-programa [65] sa potpunim izvornim kodom u okviru NVIDIA D3D10 SDK početkom 2008. godine. U okviru demo-programa, piramida teksturne klipmape je implementirana kao zasebna 2D tekstura sa mipmapama, dok je stub klipmape smešten u polje 2D tekstura. Odgovarajući nivo je izabran na osnovu izvoda promena teksturnih koordinata u prostoru ekrana. Demo-program je prikazao samo rukovanje teksturama, sa četiri različite varijante filtriranja, bez namere da se demonstrira primena na geometriju terena. Primenjene na sferu, klipmape su zahtevale ažuriranje izuzetno velike količine podataka sa svakim malim pokretom oko polova.

U kasnijim godinama, klipmape su našle svoju primenu u: obradi u realnom vremenu i organizaciji velike količine satelitskih snimaka [11] u 2008. godini, upravljanju teksturama koje se povećavaju [39] u 2011. godini i algoritmu za vizuelizaciju terena zasnovanom na jednoprolaznom emitovanju zraka [38] u 2012. godini. Na žalost, nije bilo pokušaja da se unapredi primena klipmapa u prikazu terena planetarnih razmera.

U ovoj disertaciji predložena je nova primena klipmapa za visoko-preciznu vizuelizaciju elipsoidnih planeta. Koncept je dokazan kroz vizuelizaciju planete Zemlje, zasnovanu na WGS84 referentnom elipsoidu, korišćenjem OpenGL-a [96].

Poglavlje 2

Predstavljanje planete Zemlje

Svaka vizuelizacija, pa i vizuelizacija terena, zahteva postojanje algoritma iscrtavanja i modela koji se predstavlja, definisanog matematičkim formulama, skupovima podataka ili kombinovano. Pre detaljnog obrazlaganja algoritma vizuelizacije, koji je fokus ove disertacije, potrebno je upoznati se sa modelom planete Zemlje. U ovom poglavlju biće opisano kako se geodetski precizno modelira planeta Zemlja, koji su osnovni skupovi podataka neophodni za takvo modeliranje i kako oni treba da budu organizovani, da bi mogli efikasno da se koriste.

2.1 Model planete Zemlje

Predstava o obliku Zemlje menjala se kroz vekove. U ranoj istoriji ljudske civilizacije, Zemlja je uglavnom smatrana ravnom površinom. Rana egipatska i mesopotamijska kultura predstavljale su Zemlju u vidu diska okruženog vodom i pokrivenog nebeskim „krovom". U staroj Grčkoj se koncept „ravne Zemlje" zadržao do tzv. klasičnog perioda, a u drugim krajevima sveta i znatno duže. U Kini, recimo, čak do 17. veka.

2.1.1 Zemlja kao sfera

Koncept sfernog oblika Zemlje najverovatnije potiče iz pitagorejske škole, a po prvi put se u pisanom obliku javlja u Platonovom (gr. $\Pi\lambda\alpha\tau\omega\nu$) dijalogu „Fedon" (gr. $\Phi\alpha\iota\delta\omega\nu$) kroz Sokratovu aluziju na oblik i veličinu Zemlje [27]. U ovom dijalogu, Sokrat se suprotstavlja tada široko prihvaćenom stavu da je Zemlja ravna ploča, rečima da ga je „neko ubedio" da je Zemlja sfera koja stoji u centru vrtloga. Prva numerička procena veličine, na oko 400 000 stadiona (gr. $\sigma\tau\alpha\delta\iota\sigma\nu$) u obimu, javlja se u Aristotelovom (gr. $A\rho\iota\sigma\tau\sigma\tau\varepsilon\lambda\eta\varsigma$) delu „Na nebu" (gr.

$\Pi\varepsilon\rho\iota\ o\upsilon\rho\alpha\nu o\upsilon$), dok su se kasnije procene kretale od 300 000 stadiona kod Kleomeda (gr. $K\lambda\varepsilon o\mu\eta\delta\eta\varsigma$), pa do preko 3 000 000 stadiona kod Arhimeda (gr. $A\rho\chi\iota\mu\eta\delta\eta\varsigma$).

Prvi precizan proračun dimenzija Zemlje izvršio je grčki matematičar Eratosten (gr. $E\rho\alpha\tau o\theta\varepsilon\nu\eta\varsigma$), koji se često smatra i osnivačem geografije, u obliku kakvom je danas poznajemo. Oko 240. godine pre nove ere, uz pretpostavku da je savršena lopta i uz sve nepreciznosti merenja rastojanja u tadašnje vreme, izračunao je obim Zemlje na 252 000 stadiona. Još uvek postoje polemike oko toga koliko je precizno Eratostenovo izračunavanje. Prema Herodotovim (gr. $H\rho o\delta o\tau o\varsigma$) spisima, jedan stadion sadrži 600 stopa. Međutim, dužina stope u antičkom svetu nije bila jedinstvena i menjala se zavisno od lokacije i epohe, pa se procena greške, u odnosu na WGS84 elipsoid, čiji obim na ekvatoru iznosi 40 075.0167 km, kreće od 11% do 31.7%. Tabela 2.1 daje prikaz greške Eratostenovog proračuna zavisno od procenjene dužine stope.

Tabela 2.1: Pregled relativnih grešaka Eratostenovog proračuna obima Zemlje u odnosu na WGS84 elipsoid, zavisno od procene dužine stope.

Tip stope	Dužina [mm]	Stadion [m]	Obim Zemlje [km]	Greška [%]
Olimpijska	294	176.4	44 452.8	10.92
Atinsko-rimska	308	184.8	46 569.6	16.21
Vavilonsko-persijska	327	196.2	49 442.4	23.37
Feničansko-egipatska	349	209.4	52 768.8	31.68

Sa prosečnom greškom od 16%, Erotostenova procena oblika i veličine Zemlje predstavlja neverovatno dostignuće antičkog sveta.

2.1.2 Zemlja kao elipsoid

Do značajnog napretka u definisanju oblika Zemlje došlo je zahvaljujući francuskom matematičaru i filozofu Pjer-Luj Moro de Mopertiju (fr. *Pierre-Louis Moreau de Maupertuis*, 1698-1759), koji je 1738. godine, na osnovu Njutnove mehanike i svojih ličnih merenja, prvi predložio elipsoidni oblik Zemlje, spljoštene na polovima. Proračun dužine velike poluose razlikovao se za samo 0.3% u odnosu na WGS84, ali je greška u proceni spljoštenosti bila veća (vidi Tab. 2.2). Naime, Moperti je verovao da je Zemlja spljoštenija nego što ona zaista jeste.

Sa razvojem geodezije, tokom 18-tog i 19-tog veka, dolazi i do unapređenja procena

veličine i oblika Zemlje, dovoljno preciznih da se još uvek koriste u geodeziji. Engleski matematičar i astronom Džordž Bidel Ejri (eng. *George Biddell Airy*, 1801-1892), na osnovu sopstvenih merenja u Engleskoj, definisao je 1830. godine referentni elipsoid – *Airy 1830*, koji se i danas koristi za mapiranje u Engleskoj, Škotskoj i Velsu, zbog svog dobrog poklapanja sa lokalnim nivoom mora u tim oblastima. Iste godine Džordž Everest (eng. *George Everest*, 1790-1866) predlaže elipsoid koji je korišćen u velikom trigonometrijskom premeru Indije – *Everest 1830*. Everestov elipsoid doživeo je veliki broj modifikacija kroz vrlo fina podešavanja dužine velike poluose u narednih 139 godina, kako bi se dobilo bolje poklapanje sa nivoom mora u konkretnim zemljama (Indija i Nepal, Pakistan, Malezija i Singapur). Godine 1841. nemački matematičar i astronom Fridrih Vilhem Besel (nem. *Friedrich Wilhelm Bessel*, 1784-1846) izvodi sopstveni proračun elipsoida koji je imao dobru preciznost za široko područje Evrope i Azije – *Bessel 1841*. Nakon toga usledilo je mnoštvo predloga elipsoida, svaki sa ciljem da minimizuje odstupanje od srednjeg nivoa mora na području određene države ili dela sveta. Tab. 2.2 daje pregled najvažnijih referentnih elipsoida.

Osim dimenzija elipsoida, za precizno definisanje koordinatnog sistema bilo je potrebno odrediti referentnu tačku na Zemlji u kojoj će se poklopiti površina elipsoida sa nivoom mora, kao i orijentacija elipsoida kako bi odstupanje bilo minimalno. U geodeziji se ovako definisani referentni koordinatni sistem naziva *datum*. *Horizontalni datum* određuje položaj date tačke na površini Zemlje izražen najčešće u longitudinalnom i latitudinalnom (lat/lon) pomeraju u odnosu na zadati koordinatni početak, dok *vertikalni datum* određuje visinu, ili dubinu, u odnosu na srednji nivo mora. Prema nekim procenama [7] trenutno se u svetu koristi preko 450 dobro definisanih horizontalnih datuma. OGP EPSG geodetski registar beleži 388 uvedenih datuma u periodu od 1859. do 2013. godine, od kojih je 50% formirano u poslednjih pola veka, a čak 91 nakon 1984. godine.

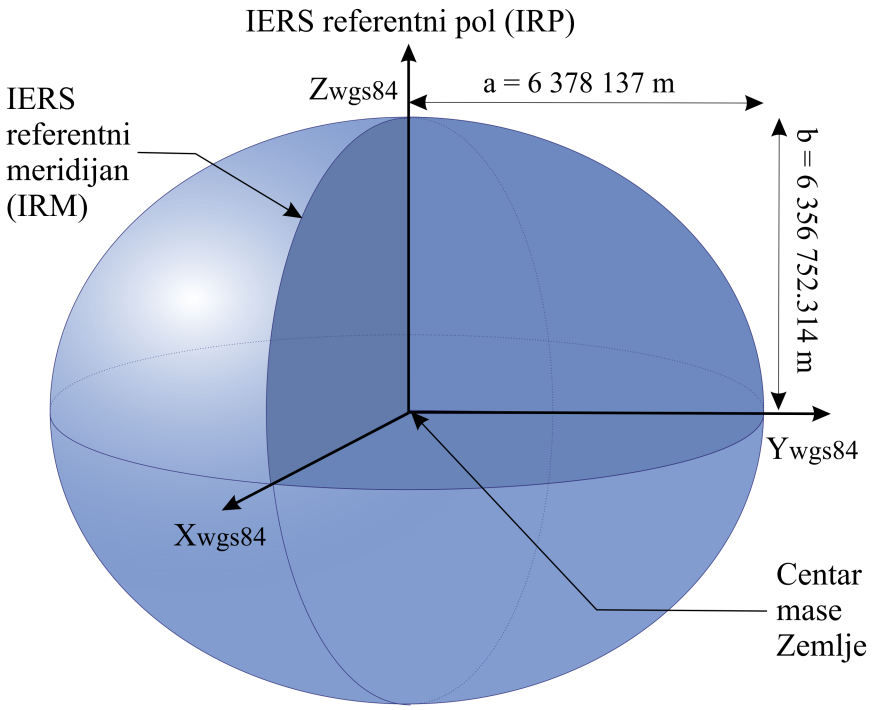
Pedesetih godina prošlog veka javila se potreba za formiranjem jedinstvenog koordinatnog sistema za čitavu planetu. Razloga je bilo mnogo, a najznačajniji su svakako internacionalizacija istraživanja svemira i razvoj astronautike, zatim, potreba za globalnim navigacionim mapama u avijaciji i geografiji, kao i pripreme za globalno ratovanje u vremenu zategnutih odnosa supersila. To je dovelo do usvajanja svetskog geodetskog sistema od strane Ministarstva odbrane SAD – *World Geodetic System 1960* (WGS60). WGS60 je definisan na osnovu raspoloživih gravitacionih i astro-geodetskih merenja, uz težnju da se ostvari što bolje poklapanje sa skupom unapred izabranih datuma. Po prvi put su korišćeni podaci o putanji satelita u proceni spljoštenosti

Tabela 2.2: Pregled referentnih elipsoida.

Naziv elipsoida	Godina	Velika poluosa (m)	Spljoštenost
Maupertuis	1738	6 397 300	1 / 191
Plessis	1817	6 376 523	1 / 308.64
Airy	1830	6 377 563	1 / 299.32
Everest	1830	6 377 276	1 / 300.801 7
Bessel 1841	1841	6 377 397.155	1 / 299.152 815
Bessel 1841 (Namibia)	1841	6 377 483.865	1 / 299.152 815
Clarke 1866	1866	6 378 206	1 / 294.98
Clarke 1880	1880	6 378 249	1 / 293.46
Helmert 1906	1906	6 378 200	1 / 298.3
International	1924	6 378 388	1 / 297
Krassowsky	1940	6 378 245	1 / 298.3
Everest 1948 (Malezija i Singapur)	1948	6 377 304.063	1 / 300.801 7
Everest 1956 (Indija i Nepal)	1956	6 377 301.243	1 / 300.801 7
Everest 1969 (Malezija)	1969	6 377 295.664	1 / 300.801 7
Hough 1960	1960	6 378 270	1 / 297
WGS60	1960	6 378 165	1 / 298.3
Fischer 1960	1960	6 378 155	1 / 298.3
WGS66	1966	6 378 145	1 / 298.25
GRS67	1967	6 378 160	1 / 298.25
Fischer 1968	1968	6 378 150	1 / 298.3
WGS72	1972	6 378 135	1 / 298.26
GRS80	1979	6 378 137	1 / 298.257 222 101
WGS84	1987	6 378 137	1 / 298.257 223 563

elipsoida. Napredovanje tehnologije, povećanje količine podataka dobijenih merenjem, kao i povećanje preciznosti merenja uticalo je na usavršavanje referentnog elipsoida kroz nekoliko revizija (WGS66 i WGS72), sve do trenutno aktuelne verzije WGS84 (Sl. 2.1). WGS84 je usvojen 80-tih godina prošlog veka kao osnova za globalni sistem za pozicioniranje (GPS).

WGS84 [76] je konvencionalni zemaljski referentni sistem (eng. *Conventional Terrestrial Reference System* – CTRS) originalno ustanovljen 1987. godine. Do današnjih dana NGA (eng.



Slika 2.1: WGS84 referentni elipsoid.

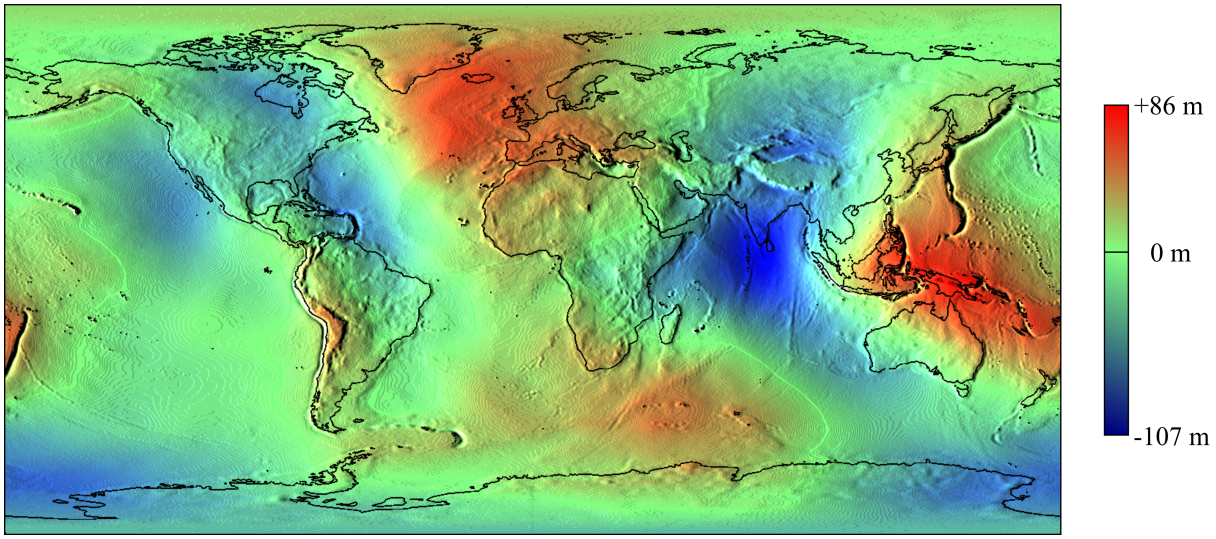
National Geospatial-Intaligence Agency) neprekidno radi na održavanju maksimalne moguće tačnosti, uzimajući u obzir nova preciznija merenja i napredak u geofizičkom modeliranju, kako bi se adekvatno predstavilo kretanje površine Zemlje. Trenutno aktuelna revizija – WGS84 (G1762), definisana je 16. oktobra 2013. godine i predstavlja šesto ažuriranje referentnog okvira od njegovog inicijalnog uspostavljanja (WGS84 → WGS84 (G730) → WGS84 (G873) → WGS84 (G1150) → WGS84 (G1674) → WGS84 (G1762)). Slovo G u oznaci ukazuje da su GPS merenja korišćena za dobijanje koordinata, dok broj u nastavku predstavlja GPS nedelju kada su merenja potvrđena od strane NGA i uključena u implementaciju WGS84. Bez obzira na povećanje preciznosti, osnovni parametri elipsoida WGS84 (velika poluosa i spljoštenost) nisu menjani od njegovog ustanovljavanja.

2.1.3 Zemlja kao geoid

Iako referentni elipsoid vrlo dobro aproksimira površinu Zemlje, ipak postoji izvesno odstupanje. Oblik koji bolje predstavlja „pravu" površinu Zemlje označava se terminom *geoid*. Površina geoida se definiše kao geomagnetna ekvipotencijalna površina koja bi se poklopila sa srednjim nivoom okeana, da su oni konstantne gustine, bez struja, međusobno povezani kanalima prosečenim kroz kontinente. Iako nema fizičkog smisla za područja kontinenata, geodeti prilično

precizno, korišćenjem atmosferskog pritiska, mogu meriti visinu bilo koje tačke na Zemlji u odnosu na tu zamišljenu površinu. Ona nije direktno vezana za reljef, već je definisana anomalijama u gravitacionim polju.

Za aproksimaciju oblika geoida najčešće se koriste sferni harmonici. Skup koeficijenata potrebnih za formiranje sfernih harmonika elipsoida poznat je pod nazivom *gravitacioni model Zemlje* (eng. *Earth Gravity Model* – EGM). I ovaj model je evoluirao u protekle tri decenije, od EGM84, preko EGM96 do EGM2008. Tekuća revizija, EGM2008 [86], takođe objavljena od strane NGA, predstavlja gravitacioni model kompletan do sfernog harmonika 2159-tog reda, a sadrži i dodatne koeficijente koji ga proširuju do 2190-tog stepena. Zbog lakše primene, odstupanje geoida od WGS84 elipsoida moguće je preuzeti sa zvaničnih lokacija [75] i u obliku rasterske datoteke, sa trenutno maksimalnom rezolucijom od 1 lučne minute. Sl. 2.2 predstavlja senčeni prikaz odstupanja geoida od WGS84 elipsoida. Zbog lakšeg prostornog orijentisanja, na slici su prikazane i konture kontinenata.



Slika 2.2: Odstupanja EGM96 geoida od referentnog WGS84 elipsoida.

Kao što se sa slike može videti, odstupanja su vrlo mala, ali se ipak moraju uzeti u obzir. I globalni pozicioni sistem (GPS) koristi korekciju geoida pri određivanju nadmorske visine GPS prijemnika. Obzirom da GPS sateliti orbitiraju oko centra gravitacije, visina koja se može meriti pomoću njih je isključivo relativna u odnosu na geocentrični referentni elipsoid. Bez korekcije, GPS prijemnik na brodu, koji se nalazi na površini mora ili okeana, ne bi prikazivao nultu visinu, već visinu koja može biti i veća od 100 metara.

2.1.4 Reljef

Konačni oblik Zemljine površine dobija se dodavanjem reljefa, koji je obično definisan skupom visina terena. Obzirom da su te visine merene u odnosu na srednji nivo mora, njihove vrednosti se moraju sabrati sa odstupanjem geoida pre primene na WGS84 elipsoid. Visine terena, a pogotovu odstupanje geoida, su vrlo male vrednosti u odnosu na poluprečnik Zemlje, tj. u odnosu na veličinu elipsoida, te se zato obično smatraju „pokrivačem" i skladište u formatima pogodnim za dvodimenzionalne (2D) podatke.

Standardni formati digitalnih podataka o visinama terena (eng. *Digital Terrain Elevation Data* – DTED), u kojima se danas uglavnom javljaju izvorni podaci, potekli su od američke vojske i potrebe usklađivanja podrške za različite sisteme naoružanja i trenažere. Trenutno aktuelna verzija standarda MIL-PRF-89020B [77] definiše zahteve za tri nivoa DTED podataka. Prema standardu, horizontalni datum podataka mora biti WGS84, a vertikalni srednji nivo mora (eng. *Mean Sea Level* – MSL) u skladu sa definicijom EGM96 geoida. Podaci moraju biti podeljeni u datoteke veličine $1^\circ \times 1^\circ$, koje ne smeju presecati celobrojne vrednosti geografskih širina ili dužina, ne smeju imati pukotina i mogu se preklapati na granicama segmenata, pri čemu podaci na granicama segmenata moraju biti identični. Gustina uzorkovanja podataka zavisi od definisanog nivoa. DTED nivo 0 podrazumeva uzorkovanje na 30 lučnih sekundi ("), DTED nivo 1 na 3", a DTED nivo 2 na 1". Uzorkovanje po geografskoj širini je uvek konstantno, dok po geografskoj dužini postaje sve ređe sa približavanjem polovima. Definisano je ukupno 5 zona, prikazanih u Tab. 2.3.

Tabela 2.3: Pregled rezolucija uzorkovanja po zonama za DTED nivo 2.

Zona	Latituda (°)	Korak po latitudi (")	Korak po longitudi (")
I	0 – 50	1	1
II	50 – 70	1	2
III	70 – 75	1	3
IV	75 – 80	1	4
V	80 – 90	1	6

DTED nivo 0 ima nominalnu rezoluciju od oko 1 km i dobija se usrednjavanjem vrednosti preciznijeg modela (nivo 1). Izvorno je bio namenjen federalnim agencijama, ali se vrlo brzo došlo do zaključka da se ovi podaci mogu učiniti i javno dostupnim bez ograničenja. Zbog svoje

niske rezolucije, ne smeju se koristiti u automatskom navođenju letelica ili bilo čemu što može ugroziti javnu bezbednost.

DTED nivo 1 ima nominalnu rezoluciju od oko 90 m i približno odgovara podacima dobijenim digitalizacijom izohipsi na kartama razmere 1:250 000. Kao model terena srednje rezolucije, namenjen je planiranju različitih vojnih aktivnosti.

DTED nivo 2 je digitalni model terena više rezolucije (nominalno oko 30 m) i odgovara podacima dobijenim digitalizacijom izohipsi na kartama razmere 1:50 000. Za razliku od prethodna dva nivoa, obično pokriva samo odabrane delove terena.

Tabela 2.4: Pregled besplatnih globalnih digitalnih modela terena.

Oznaka	Godina	Rezolucija (")	Izvor
ETOPO5	1988	300	NGDC/NOAA
TerraBase	1994	300	NGDC/NOAA
GTOPO30	1996	30	EROS/USGS
GLOBE	1999	30	NOAA
ACE GDEM	2000	30	EAPRS
SRTM30	2003	30	NASA/USGS
SRTM3 v1	2003	3	NASA/USGS
SRTM3 v2	2005	3	NASA/USGS
SRTM3 v2.1	2009	3	NASA/USGS
SRTM3 v3	2013	3	NASA/USGS
SRTM3 v4	2008	3	CIAT/USGS
SRTM3 v4.1	2008	3	CIAT/USGS
ACE2 GDEM	2008	3	EAPRS
ETOPO1	2009	60	NOAA
ASTER GDEM1	2009	3	METI/ERSDAC
ASTER GDEM2	2011	1	METI/ERSDAC
SRTM1	2014	1	NASA/USGS

Počev od 1988. godine do danas kreirano je više javno dostupnih globalnih digitalnih modela terena (eng. *Global Digital Elevation Model* – GDEM). Njihova rezolucija i kvalitet predstavljenih podataka je vremenom rastao, tako da su već sada na raspolaganju DTED modeli nivoa 2 za čitav svet.

Prvi globalni digitalni modeli terena nastali su krajem 80-tih i početkom 90-tih godina prošlog veka. ETOPO5 [74] i TerrainBase [95] imali su rezoluciju od svega 5 lučnih minuta (približno 10 km), ali su zbog odsustva preciznijih podataka u svoje vreme predstavljali standarde za GDEM. Oba modela formirala je NGDC (eng. *National Geophysical Data Center*) u okviru NOAA (eng. *National Oceanic and Atmospheric Administration*).

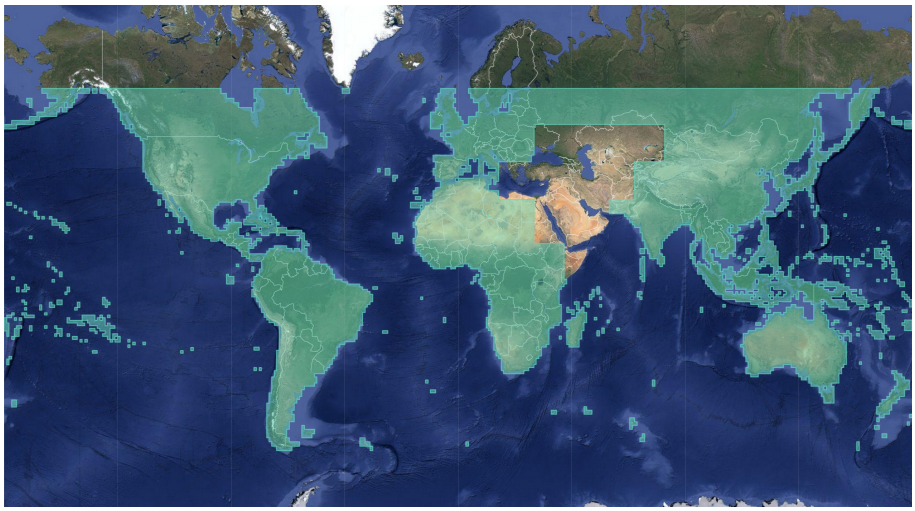
GTOPO30 je najstariji javno dostupni globalni digitalni model terena sa rezolucijom 30". Rezultat je trogodišnjeg rada osoblja Centra za resurse planete Zemlje, posmatranje i nauku (eng. *Center for Earth Resources Observation and Science* – EROS), u okviru USGS (eng. *U.S. Geological Survey*). Izveden je na osnovu različitih izvora, pre svega rasterskih i vektorskih topografskih karata, kompletiran i učinjen javno dostupnim 1996. godine. Radi lakše distribucije, podaci su podeljeni u blokove $40^\circ \times 50^\circ$ (osim za južni pol, gde su blokovi veličine $60^\circ \times 30^\circ$).

GLOBE (eng. *Global Land One-km Base Elevation*) [50] je objavljen 3 godine kasnije (1999), kao rezultat internacionalnog projekta vođenog od strane NOAA. GLOBE kombinuje različite, do tada raspoložive, izvore podataka. Osamnaest izvora podataka je sklopljeno u mozaik i detaljno dokumentovano.

ACE (eng. *Altimeter Corrected Elevations*) je unapređeni GDEM, nastao u EAPRS laboratoriji De Monfort univerziteta u Lečesteru (UK). Radi povećanja horizontalne i vertikalne preciznosti, korigovani su podaci prethodna dva modela (GTOPO30 i GLOBE), korišćenjem nezavisne altimetarske baze podataka sa preko 100 miliona pojedinačnih merenja i odgovarajućeg ekspertskog sistema. Osam godina kasnije, objavljena je i druga verzija modela (*ACE2*), bazirana na SRTM podacima.

SRTM (eng. *Shuttle Radar Topography Mission*) je internacionalni projekat u kome su učestvovali američka nacionalna aeronautička i svemirska agencija (NASA), americka nacionalna agencija za slikanje i mapiranje (NIMA), nemačka (DLR) i italijanska (ASI) svemirska agenicija. Cilj projekta bio je kreiranje digitalnog modela terena od 56° južne geografske širine do 60° severne geografske širine. Tokom 11 dana, februara 2000. godine, Spejs Šatl Endevor (eng. *Space Shuttle Endeavour*) prikupio je potrebne podatke sa rezolucijom 1 lučne sekunde. Prva verzija SRTM digitalnog modela, podeljena u sekcije po formatima DTED nivoa 1 i 2, sadržala je neobrađene izvorne podatke, sa mnoštvom loših vrednosti usled refleksije, kao i površine bez podataka (tzv. *praznine*). Verzija 2 je nastala nakon obrade podataka od strane NGA, uz bolje definisanje vodenih površina i obala, i eliminisanje igličastih grešaka (greške koje zahvataju samo jedan uzorak). I u ovom modelu bile su prisutne praznine. U verziji 2.1 izvršeno je

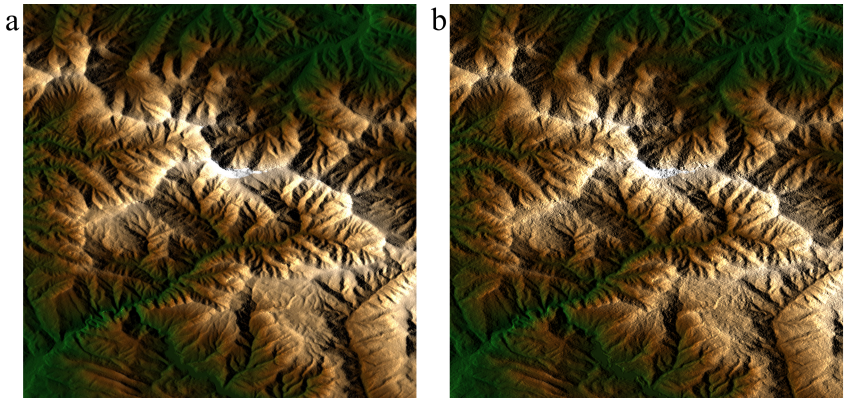
preračunavanje, usrednjavanje vrednosti i eliminisanje još nekih od grešaka. Popunjavanje praznina ostvareno je tek u verziji 3, korišćenjem podataka iz drugih izvora, a pre svega ASTER GDEM2 i USGS GMTED2010 modela. Ovaj model je poznat i kao *SRTM Plus*. U verziji 3 pomerena je i rešetka uzorkovanja za polovinu ćelije, da bi ponovo bila vraćena na staro u verziji 4. Verzija 4 dodatno unapređuje popunjavanje praznina i uvodi nove interpolacione tehnike. Na klimatskom samitu Ujedinjenih nacija, 23. septembra 2014. godine, Vlada Sjedinjenih Američkih Država objavila je da će učiniti javno dostupnim i SRTM podatke sa rezolucijom jedne lučne sekunde za čitav svet. Objavljivanje podataka ide postepeno i do septembra 2015. godine trebalo bi da svi podaci budu dostupni. Na Sl. 2.3 prikazano je tekuće stanje pokrivenosti sveta SRTM1 podacima, preuzeto sa portala EarthExplorer [103] maja meseca 2015. godine. Zelenom bojom označene su postojeće sekcije.



Slika 2.3: Pokrivenost sveta SRTM1 podacima. Prikazano je tekuće stanje maja meseca 2015. godine, preuzeto sa portala EarthExplorer.

ASTER (eng. *Advanced Spaceborne Thermal Emission and Reflection Radiometer*) u verziji 1 publikovan je juna 2009. godine i predstavlja najkompletnije mapiranje planete Zemlje, obzirom da pokriva 99% njene površine (od 83° južne geografske širine do 83° severne geografske širine). Iako veće nominalne rezolucije od SRTM, nekoliko izvora je potvrdilo da je prava rezolucija zapravo mnogo manja. Verzija 2 (ASTER GDEM2), objavljena oktobra 2011. godine, donosi unapređenje horizontalne i vertikalne preciznosti i redukovanje grešaka. Obzirom da su SRTM1 podaci novi (i još ne u celosti publikovani), poređenja u naučnim radovima su ograničena na SRTM3 i ASTER GDEM2 podatke. Kako ASTER ima veću prostornu rezoluciju, uočena je i njegova veća preciznost na planinskim padinama i u području neravnog terena [93].

Međutim, u vizuelnom poređenju sa SRTM1 podacima, korišćenjem senčenog reljefa (Sl. 2.4), može se jasno videti daleko veće prisustvo šuma u ASTER podacima.



Slika 2.4: Uporedni prikaz dela reljefa senčenog korišćenjem SRTM1 (a) i ASTER GDEM2 (b) podataka.

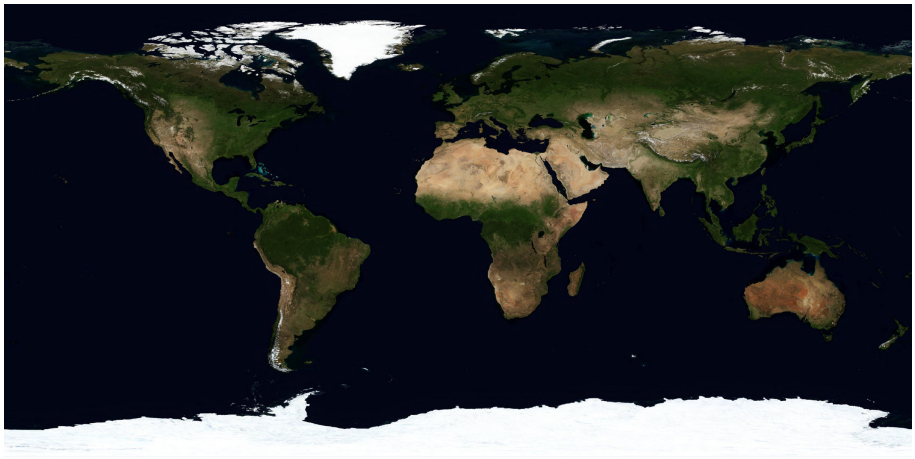
Detaljniji digitalni modeli terena, dobijeni korišćenjem LIDAR [17] (eng. *LIght Detection And Ranging*) i INSAR [94] (eng. *INterferometric Synthetic Aperture Radar*) tehnologijama, nisu dostupni na nivou čitavog sveta, već prvenstveno za područje severno-američkog kontinenta. Nacionalni skup podataka o visinama terena za teritoriju Sjedinjenih Američkih Država (eng. *National Elevation Dataset* – NED) objedinjuje podatke sa rezolucijom od 1/3 lučne sekunde (DTED nivo 3) za teritoriju SAD, Havaja, delova Aljaske i teritorijalnih ostrva SAD, i rezolucijom 1/9 lučne sekunde (DTED nivo 4) za teritoriju SAD i vrlo malog dela Aljaske. U toku je proces prikupljanja još detaljnijih podataka, sa rezolucijom od 1 m za područje SAD. DTED nivoi 3, 4 i 5 predstavljaju samo predlog standarda i nisu ušli u MIL-PRF-89020B [77].

2.1.5 Teksturni pokrivač

Koliko god bio detaljan digitalni model terena, on ne može verno predstaviti površinu Zemlje, uz istovremeno zadržavanje dobrih performansi vizuelizacije, ukoliko se ne koristi teksturni-pokrivač u vidu satelitskih ili aero-foto snimaka.

Najpoznatiji izvor podataka koji sadrži slike čitave planete Zemlje u realnim bojama je *Blue Marble Next Generation* [98] serija slika rezolucije 86400×43200 piksela, odnosno 15 lučnih sekundi po pikselu (reda 500 m u ekvatorijalnom području). Napravljen je na osnovu NASA Terra MODIS (eng. *MODerate resolution Imaging Spectroradiometer*) kolekcije iz 2004. godine, sa potpuno uklonjenim oblacima i organizovane po mesecima u 12 zasebnih slika. Ovaj izvor podataka se vrlo često koristi u procesu vizuelizacije čitave planete Zemlje i to posebno slike iz jula ili avgusta meseca. One imaju najmanje snežnog pokrivača, a čitava površina planete

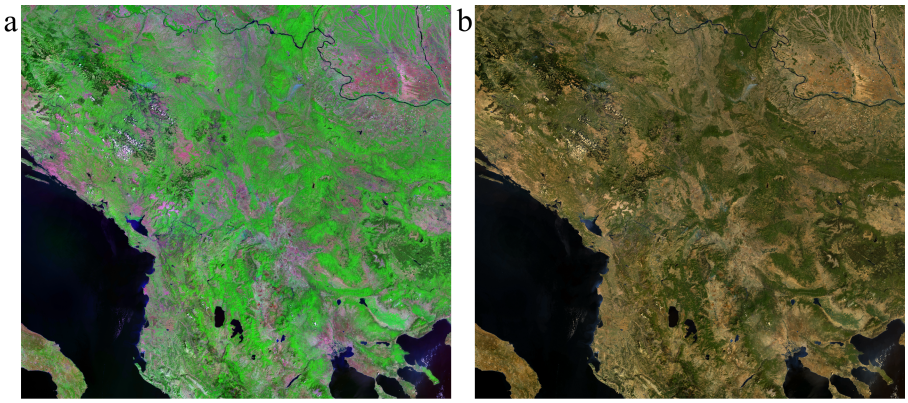
je mnogo zelenija, pogotovu područje jugoistočne Azije, gde zbog perioda monsuna dolazi do značajnog povećanja količine vegetacije. Na Sl. 2.5 prikazana je planeta Zemlju u julu mesecu 2004. godine. Osim osnovnih slika, postoje i varijante sa senčenjem reljefa ili morskog dna.



Slika 2.5: Blue Marble Next Generation – slika planete Zemlje u realnim bojama, mesec jul 2004. godine.

NASA je od 70-tih godina pa do kraja prošlog veka radila na formiranju ortorektifikovanog i geodetski preciznog globalnog skupa satelitskih snimaka površine Zemlje. Rezultati treće faze mapiranja, iz 2000. godine (± 3 godine), organizovani su u skup podataka poznat kao *Orthorectified Landsat Enhanced Thematic Mapper (ETM+) Compressed Mosaics* [60]. Ovi podaci imaju rezoluciju od 14.25 m po pikselu i nastali su na bazi tri kanala Landsat ETM+ senzora: kanala 7 (infracrvena svetlost srednje talasne dužine) prikazan crvenom bojom, kanal 4 (infracrvena svetlost manje talasne dužine) prikazan zelenom bojom i kanal 2 (vidljiva zelena svetlost) prikazan plavom bojom. Podaci iz sva tri kanala su dodatno izoštreni korišćenjem panhromatskog kanala, a primenjen je i poseban algoritam, kompanije EarthSat, za povećanje kontrasta. Mozaik se sastoji od blokova u UTM/WGS84 projekciji veličine $6^\circ \times 5^\circ$ (južno od 60° severne geografske širine) i $12^\circ \times 5^\circ$ (severno od 60° severne geografske širine). Obzirom da podaci ne predstavljaju vidljivi deo spektra, potrebna je dodatna obrada zbog boljeg uklapanja sa ostalim izvorima podataka koji se koriste u vizuelizaciji. Na Sl. 2.6 prikazan je primer modifikacije boja, koja je korišćena u vizuelizaciji opisanoj u ovoj disertaciji.

Savremeni satelitski senzori visoke rezolucije omogućuju vrlo precizno snimanje površine Zemlje u vidljivom spektru. Trenutno najpreciznije, komercijalno dostupne, snimke omogućuje WorldView-3 satelit, sa rezolucijom 0.31 m po pikselu. Još detaljniji snimci zahtevaju snimanje sa manjih visina, tačnije iz aviona. Za gradska područja se obično izrađuju ortografski snimci sa rezolucijom 10 cm po pikselu, dok su vangradska područja pokrivena snimcima rezolucije 40 cm



Slika 2.6: Jedan blok iz mozaika Landsat satelitskih snimaka u izvornom obliku (a) i nakon modifikacije boja (b) za potrebe primene u vizuelizaciji opisanoj u ovoj disertaciji.

po pikselu. Na Sl. 2.7 dat je uporedni prikaz ortografskih snimaka različitih rezolucija, preuzetih sa portala geoSerbia [92]. Snimci Sl. 2.7 (a) i Sl. 2.7 (b) su georeferencirani ortografski snimci načinjeni senzorima postavljenim na avion, dok je snimak Sl. 2.7 (c) načinio satelit SPOT-5.

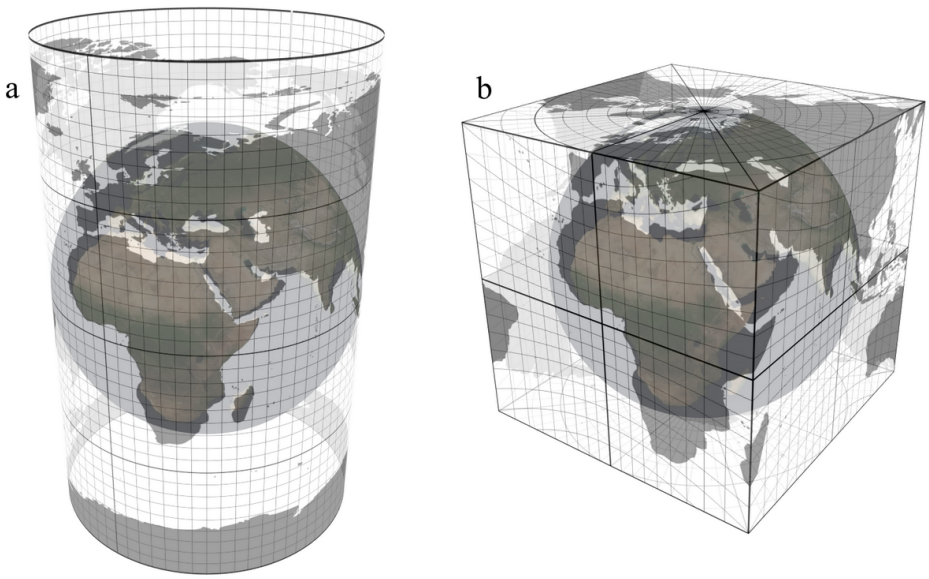
Slika 2.7: Uporedni prikaz ortografskih snimaka različitih rezolucija, i to: 10 cm (a), 40 cm (b) i 2.5 m (c), preuzetih sa portala geoSerbia.

Količina podataka potrebna za pokrivanje planete Zemlje teksturama visoke rezolucije je neverovatno velika. Snimci samo kopnene površine sa rezolucijom 10 cm imali bi ukupno oko 1.5E16 piksela i u nekompresovanom obliku zauzimali oko 40 pentabajta (1 PB = 1E15 B). Zbog toga se često koriste vrlo visoki stepeni kompresije za skladištenje snimaka na diskove, kao i ograničavanje visoke rezolucije samo na odabrane oblasti.

Svi pokrivni podaci (odstupanje geoida, digitalni model terena i ortografski snimci) moraju biti projektovani na dvodimenzionalnu površinu, poželjno pravougaonog oblika, kako bi bili pogodni za smeštanje i rukovanje. U nastavku ovog poglavlja biće prikazane neke standardne projekcije koje se koriste u vizuelizaciji planete Zemlje.

2.2 Projekcija površine Zemlje

Vrlo važan problem u vizuelizaciji planete, ali takođe i u raznim drugim disciplinama koje se bave prikupljanjem i prikazom planetarnih podataka, jeste kako organizovati podatke koji predstavljaju njenu površinu. Uzorci sferoidne površine moraju biti projektovani na ravan, kako bi bili efikasno smešteni u memoriji računara. Ovo je problem koji su kartografi imali još od antičkih vremena, smišljajući brojne kartografske projekcije ne bi li što bolje površinu Zemlje preneli na kartu, uz očuvanje nekih njenih bitnih karakteristika. Projekcije koje očuvavaju površinu (eng. *equal-area*) projektuju oblasti na površini Zemlje na proporcionalno jednake površine na karti. Konformne (eng. *conformal*) projekcije, na drugoj strani, očuvavaju oblike. Projekcija ne može istovremeno biti i konformna i da očuvava površinu. Upravo zbog toga, mnoge projekcije su nešto između i pokušavaju da smanje distorziju ili očuvaju neku drugu karakteristiku (ne površinu ili oblik). Najpopularnije projekcije koje se koriste u vizuelizaciji terena planetarnih razmera su: ekvidistantna cilindrična (Sl. 2.8 (a)) i sferna kocka (Sl. 2.8 (b)).



Slika 2.8: Najčešće projekcije korišćene za vizuelizaciju terena planetarnih razmera su: ekvidistantna cilindrična projekcija (a) i sferna kocka (b).

2.2.1 Ekvidistantna cilindrična projekcija

Ekvidistantna cilindrična projekcija koristi geografsku rešetku sa konstantno razmaknutim podeocima, poravnatu sa paralelama i meridijanima, za predstavljanje podataka. To je verovatno najintuitivniji način poimanja Zemljine površine za ljude. Štaviše, reprezentacija je koherentna

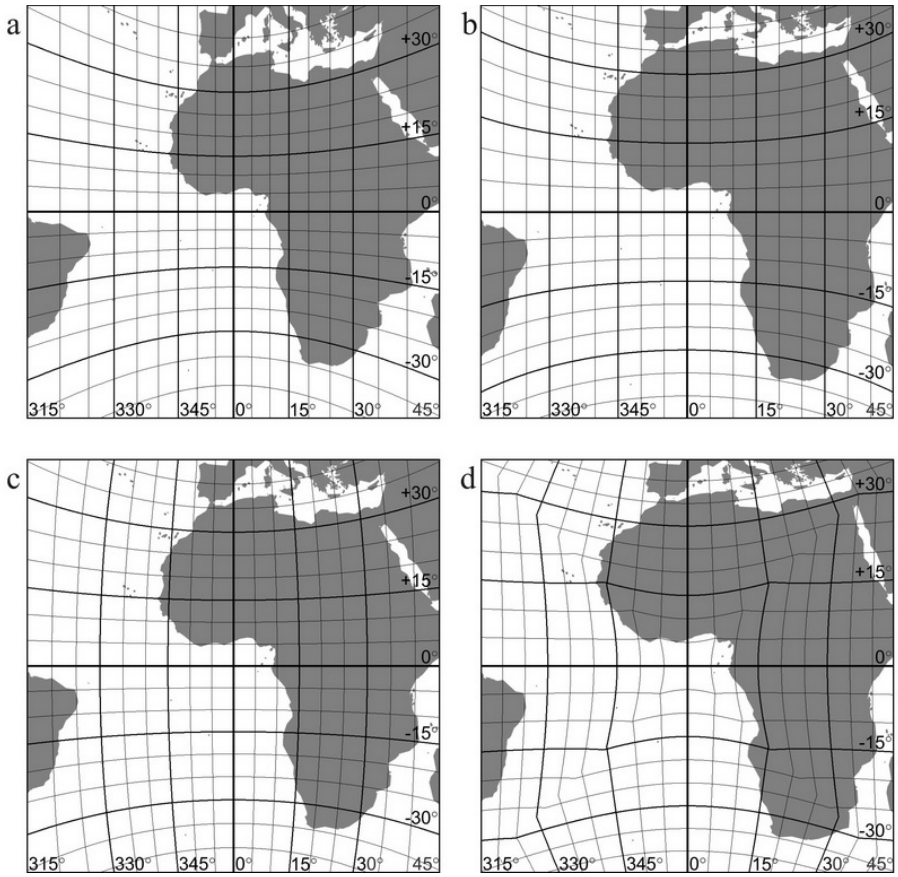
i jedinstvena na nivou čitave planete. Ova projekcija nije ni konformna niti očuvava površinu. Distorzija raste sa rastojanjem od ekvatora i postoje singulariteti u polovima. Tokom vizuelizacije, kako se posmatrač približava polovima raste količina nepotrebnih podataka, a performanse opadaju u skladu sa ukupnim porastom količine podataka. Takođe, vrlo tanki trouglovi stvaraju probleme u senčenju i teksturisanju na polovima. Da bi se prevazišao ovaj problem, Gerstner [43] je predložio kompenzaciju, množenjem vrednosti greške na osnovu koje se definiše veličina trougla sa kosinusom latitude, čime se ostvaruje efekat povećanja trouglova sa približavanjem polovima. Iako je gustina rešetke korigovana, Gerstnerov pristup ne omogućuje 1:1 mapiranje na izvorne podatke, jer zadržava geografsku rešetku, samo sa razređenim uzorkovanjem.

2.2.2 Sferne kocke

Jedini način da se smanji distorzija jeste da se poveća broj ravni na koje se projektuje sferoidna površina. Poliedarske projekcije koriste stranice opisanih (ili upisanih) poliedara kao projekcione ravni. Među njima, svakako najpopularnije su projekcije zasnovane na kocki (heksahedralne projekcije), poznatije kao *Sferne kocke* (SK). Postoji mnoštvo SK projekcija. Najnaivniji pristup, baziran na šest gnomoničkih projekcija na stranice opisane kocke, datira od pre nekoliko vekova. Ova projekcija nije ni konformna niti očuvava površinu, a distorzija značajno raste prema ivicama kocke (Sl. 2.9 (a)). Efekti distorzije mogu se smanjiti (Sl. 2.9 (b)) ako se uzorkovanje vrši u sfernim koordinatama, a ne u ravni projekcije [61].

Obzirom da je očuvanje površine jedna od bitnih karakteristika projekcije, bilo je pokušaja razvoja projekcije zasnovane na sfernoj kocki sa ovom osobinom. Čen (eng. *Chan*) i O'Nil (eng. *O'Neill*) [19] su 1975. godine razvili projekciju za potrebe američke mornarice, koja je približno očuvavala površinu. Uz izvesne modifikacije, ova projekcija je kasnije korišćena i u okviru NASA projekta KOBE (eng. *Cosmic Background Explorer* - COBE). Na žalost, KOBE SK projekcija je vrlo neprecizna, ili bar u slučaju korišćenja jednačina i koeficijenata objavljenih u [15]. Maksimalna greška u pozicioniranju od oko 1.4 km diskvalifikuje ovu projekciju za primenu u geografskim informacionim sistemima. O'Nil i Loubšer (eng. *Laubscher*) [84] su razvili SK projekciju u zatvorenom obliku, koja očuvava površinu i ima vrlo malu ugaonu distorziju. Iako je korišćena u skorašnjim naučnim istraživanjima i vizuelizacijama planete Zemlje [59], ova projekcija ima izuzetnu distorziju aspekta, a postoje i diskontinuiteti na dijagonalama stranice kocke (Sl. 2.9(d)).

Za potrebe Outerra projekta [55], razvijena je konformna SK projekcija. Oblici konti-



Slika 2.9: Prednja stranica sferne kocke korišćenjem: Tangencijalne sferne kocke – TSC (a), Prilagođene sferne kocke – PSK (b), Outerra sferne kocke – OSK (c) i Kvadrilateralizovane sferne kocke – KSK (d).

nenata su potpuno očuvani, kao što se može videti na Sl. 2.9 (c). Međutim, kako projekcija ne može istovremeno očuvavati površinu i biti konformna, Outerra SK ispoljava distorziju površine koja se radijalno povećava sa rastojanjem od centra stranice kocke.

Kao što se iz svega prethodnog može videti, ne postoji idealna projekcija. Što je neka karakteristika doslednije ispoljena, druge karakteristike izložene su većim izobličenjima. U nastavku će biti prikazane još neke projekcije koje su prezentovane u naučnim radovima i našle svoju primenu u vizuelizaciji terena planetarnih razmera.

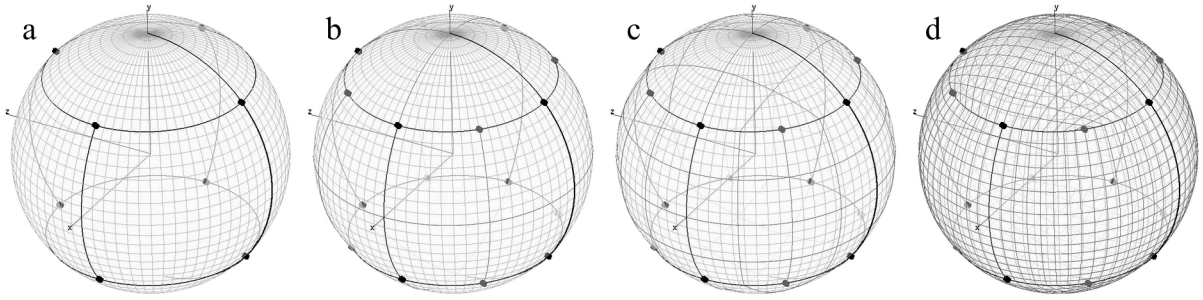
2.2.3 Primeri drugih projekcija

Cilindrične projekcije i sferne kocke sigurno predstavljaju najzastupljenije projekcije u trodimenzionalnoj vizuelizaciji planete Zemlje. Cilindričnu projekciju koriste trenutno verovatno najjači predstavnici na tržištu: *Google Earth* [46] i *Microsoft Virtual Earth 3D* [66]. *Google Earth* koristi standardnu ekvidistantnu cilindričnu projekciju, dok *Microsoft Virtual Earth 3D* koristi Merkatorovu projekciju (obzirom da je zasnovan na *Bing Maps* servisu, koji mape predstavlja u

obliku sekcija definisanih u Merkatorovoj projekciji [71]). Razlog za izbor cilindričnih projekcija je očigledan – jednostavno uklapanje u postojeće izvore podataka.

Sferne kocke nalaze veću primenu kod okruženja sa ograničenim skupovima podataka, naučnim projektima i eksperimentima. Za vizuelizaciju planete Zemlje, sferne kocke koriste: *Outerra* [59], *ECM* [59] i *Eve Flight Simulator* [49]. Međutim, postoji i mnoštvo drugih projekcija i metoda vizuelizacije koji zavređuju pažnju. U nastavku će biti prikazani pristupi koji nalikuju projekciji predstavljenoj u sledećem poglavlju, mada se od nje suštinski razlikuju.

Džejms Miler (eng. *James Miller*) i Tom Gaskins (eng. *Tom Gaskins*), u svom radu [67], predlažu izbegavanje singulariteta u polovima uvođenjem polarnih regiona, sa drugačijim postupkom kreiranja rešetke od ostatka planete. Proces formiranja osnovne rešetke i podele površine na sitnije celine (u daljem tekstu – *teselacija*) počinje, kako tvrde autori, od kocke (Sl 2.10 (a)). Međutim, ovde se ne radi o sfernoj kocki, što se jasno može videti sa slike, jer ivice prate uporednike, što nije slučaj ni sa jednom SK projekcijom. Zato je tačnije reći da četiri bočne stranice koriste ekvidistantnu cilindričnu projekciju, jer, i po tvrdnjama autora, rešetka prati mrežu meridijana i uporednika. Rešetka polarnih regiona kreirana je tako da ne preseca meridijan na 180° i da se inicijalno čitav region deli na četiri sektora (trougla). Sektori se pretvaraju u četvorouglove dodavanjem temena na sredini ivica okrenutih ka bočnim stranama „kocke" (Sl 2.10 (b)). Svaki od novonastalih četvorouglova se dalje deli dodavanjem temena na sredini svake stranice i njihovim spajanjem sa sredinom četvorougla (Sl 2.10 (c)). Proces se nastavlja do postizanja odgovarajuće gustine rešetke (Sl 2.10 (d)).

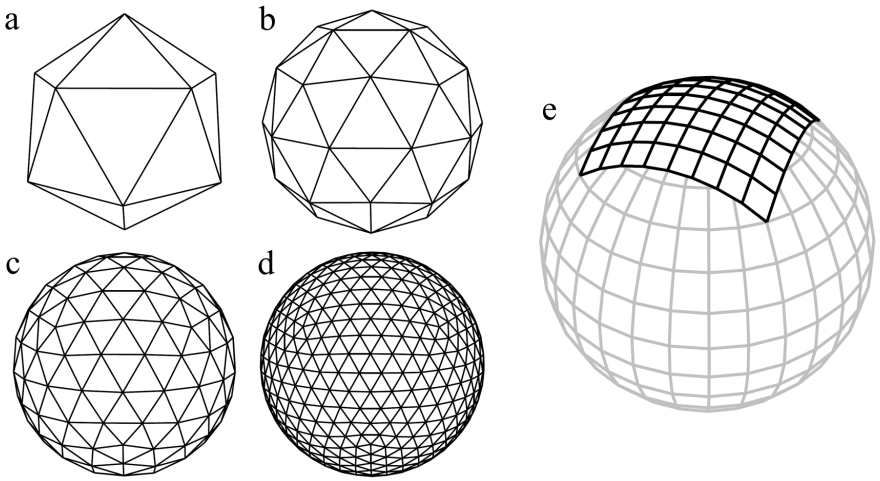


Slika 2.10: Koraci u teselaciji elipsoida: mapiranje „kocke" na elipsoid (a), prvi nivo podele (b), drugi nivo podele (c) i četvrti nivo podele (d). Prikazana je kompozicija pojedinačnih slika preuzetih iz [67], uz neznatnu modifikaciju kako bi se poboljšala uočljivost detalja.

Da ne bi došlo do pojave pukotina, dužina ivica trouglova koji se generišu na osnovu polarne rešetke mora da odgovara dužini ivica susednih trouglova iz četiri bočna regiona. To se može postići ako su granične linije postavljene na približno $\pm 40°$ geografske širine. Osnovna

ideja o potrebi za uvođenjem polarnih regiona, sa drugačujom teselacijom u odnosu na ostatak planete, korišćena je i u projekciji predloženoj u ovoj disertaciji. Međutim, predloženo rešenje, za razliku od [67], ne zahteva rekurzivnu podelu površine i koristi isti metod za generisanje rešetki svih regiona.

Robert Kojma (eng. *Robert Kooima*) je u svojoj doktorskoj disertaciji [58] ukazao na postojanje problema u akviziciji podataka u okolini polova pri korišćenju cilindričnih projekcija (ili kako ih on neformalno naziva – *sfernih projekcija*) i predložio korišćenje polarne stereografske projekcije za date oblasti (Sl 2.11 (e)). Razlog više za takav predlog predstavljali su, verovatno, i skupovi podataka sa kojima je raspolagao. Međutim, umesto da izvori podataka budu osnov za podelu planete, predlaže se uniformna teselacija, koja ne zavisi od podataka. Za osnovnu aproksimaciju planete koristi se ikosaedar (Sl 2.11 (a)). Svaka od trougaonih stranica se zatim rekurzivno deli, dodavanjem temena na sredini ivica i njegovim udaljavanjem od prvobitnog položaja, do poklapanja sa površinom planete (Sl 2.11 (b), (c) i (d)).

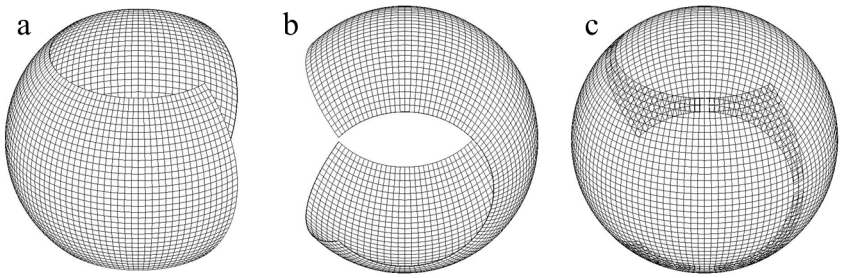


Slika 2.11: Formiranje geometrije planete na osnovu ikosaedra: početni ikosaedar (a), prvi nivo podele (b), drugi nivo podele (c) i treći nivo podele (d). Nezavisno od formirane geometrije, podaci o visinama terena i boje teksturnog-pokrivača uzorkuju se iz skupova podataka (e) u cilindričnoj projekciji (svetlija rešetka) ili polarnoj stereografskoj projekciji (tamnija rešetka). Elementi slike preuzeti su iz [58].

Rekurzivna podela ikosaedra se koristi samo do nivoa određenog položajem i vidnim poljem posmatrača, ali i preciznošću aritmetike jednostruke tačnosti, obzirom da se za ovaj proces koristi geocentrični koordinatni sistem. Tako dobijeni blokovi, koji su u vidnom polju, se dalje rekurzivno dele, ali se sada koristi koordinatni sistem vezan za oko posmatrača (tj. za položaj kamere). Nakon postizanja potrebnog nivoa detalja, podaci o visinama i teksturnom-pokrivaču se čitaju iz dva skupa podataka, jednog u cilindričnoj i drugog u polarnoj stereografskoj projekciji,

a zatim se vrši težinsko usrednjavanje očitanih vrednosti u skladu sa položajem posmatrača. Za razliku od ovakvog pristupa, metod predložen u ovoj disertaciji je mnogo jednostavniji, ne zahteva rekurzivnu teselaciju niti čitanje i usrednjavanje vrednosti pročitanih iz više skupova podataka sa različitim projekcijama.

Treća, prilično egzotična, ali vrlo primenljiva projekcija za vizuelizaciju planeta je *Yin-Yang* projekcija [54]. Ova projekcija deli površinu planete na dve uzajamno zarotirane particije, koje su u slučaju sfere potpuno identične (Sl 2.12). Particije nisu potpuno disjunktne, a delovi koji se preklapaju mogu se ukloniti trimovanjem.



Slika 2.12: *Yin-Yang* rešetka. Čitava planeta podeljena je u dve particije: Yang (a) i Yin (b), koje su identične i neznatno se preklapaju (c). Slika je preuzeta iz [87].

Iako je *Yin-Yang* projekcija, odnosno odgovarajuća rešetka, korišćena u više naučnih radova, nije poznato da je našla primenu u vizuelizaciji planete Zemlje. Međutim, ukoliko se za svaku particiju koristi odgovarajuća ekvidistantna cilindrična projekcija, a obzirom da ima samo dve particije, *Yin-Yang* rešetka može biti vrlo zanimljiva za primenu u EKM.

Bez obzira na mnoštvo projekcija, očigledno je da se još uvek traga za boljim načinom predstavljanja površine planete. Takva projekcija bi trebalo da balansira između što uniformnijeg uzorkovanja podataka (projekcije koje očuvavaju površinu) i što manje distorzije aspekta (konformne projekcije). Takođe, u nekim primenama, poželjno je i da broj projekcionih ravni bude što manji. U narednom poglavlju biće detaljno izložen predlog projekcije koja se efikasno može koristiti u vizuelizaciji zasnovanoj na elipsoidnim klipmapama. Ova projekcija kombinuje jednostavnost ekvidistantne cilindrične projekcije, podelu planete na tri particije i mali-do-srednji nivo distorzije tekstura.

Poglavlje 3

Elipsoidne klipmape

Elipsoidne KlipMape (EKM) predstavljaju jezgro algoritma za vizuelizaciju terena planetarnih razmera, opisanog u ovoj disertaciji. To je potpuno novi pristup u primeni klipmapa. Za razliku od prethodnih algoritama, koji su primenjivali klipmape na ravnu [64, 6] ili sfernu površinu [22], EKM po prvi put koristi elipsoidnu površinu, omogućujući geodetski preciznu vizuelizaciju planete. Elipsoidna površina podeljena je na tri particije, povezane bez šavova u monolitnu celinu.

Druga vrlo bitna razlika u odnosu na prethodne implementacije geometrijskih klipmapa, koje koriste unapred učitanu rešetku nad kojom se uzdiže teren, jeste generisanje elipsoidne rešetke u letu, u verteks šejderu. Predloženi metod garantuje subpikselsku preciznost izračunavanja referentnog elipsoida planete Zemlje, korišćenjem aritmetike jednostruke tačnosti grafičkog procesora. Primena klipmapa omogućuje konzistentno keširanje i brz protok podataka, dok korišćenje ekvidistantne cilindrične projekcije za izvorne podatke minimizuje pretprocesiranje i očuvava nizak-do-srednji nivo distorzije tekstura.

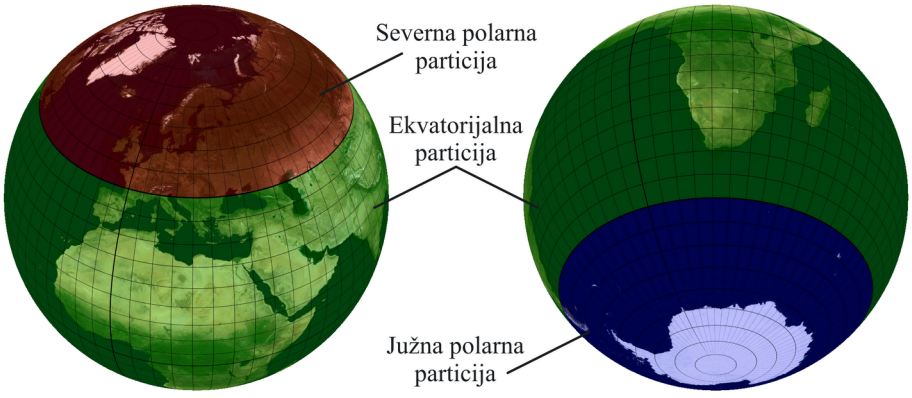
U ovom poglavlju biće detaljno predstavljena tro-particiona ekvidistantna cilindrična projekcija, korišćena u procesu vizuelizacije, i upoređena distorzija koju ona unosi u primenu tekstura na teren, u odnosu na SK projekcije. Takođe, biće prikazani koordinatni sistemi koji se koriste u pristupu podacima i vizuelizaciji, kao i konverzije iz jednog koordinatnog sistema u drugi. Razumevanje koordinatnih sistema i transformacija je vrlo važno za praćenje izlaganja u narednim poglavljima. Nakon koordinatnih sistema, biće objašnjen proces spajanja particija i mešanja nivoa detalja. Na samom kraju poglavlja, prikazana je arhitektura softverskog sistema za vizuelizaciju terena, čija je detaljna implementacija data u narednim poglavljima.

3.1 Podela planete na tri particije

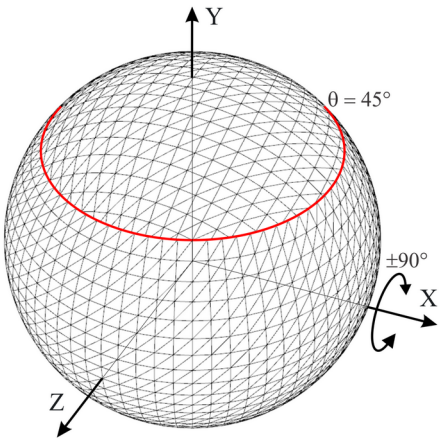
EKM pokušava da zadrži geografsku rešetku (ekvidistantna cilindrična projekcija) koliko je god to moguće uz: minimizaciju izobličenja, izbegavanje singulariteta i izazivanje male-do-srednje distorzije tekstura. Da bi se to ostvarilo, površina planete podeljena je u tri particije: jednu ekvatorijalnu i dve polarne (Sl. 3.1). Za razliku od sfernih kocki (SK), opisanih u prethodnom poglavlju, gde je broj particija šest, EKM redukuje taj broj kako bi smanjio količinu podataka u memoriji, kao i broj graničnih linija između particija. Ekvatorijalna particija se prostire od -45° do +45° geografske širine, uz korišćenje standardne geografske rešetke (ekvidistantna cilindrična). Polarne particije koriste geografsku rešetku zarotiranu za ±90° oko globalne X-ose i odsečenu na ±45° globalne geografske širine (Sl. 3.2).

Prednosti podele planete na tri particije i korišćenja geografske rešetke su brojne:

- broj skupova podataka je redukovan na tri,
- maksimalno dve particije su vidljive iz bilo koje tačke, sve do rastojanja od površine kada se cela planeta zamenjuje mnogo jednostavnijim modelom,
- oko 71.2% svih izvornih podataka je u standardnoj ekvidistantnoj cilindričnoj (lat/lon) projekciji, tako da se izvorni podaci mogu koristiti direktno, bez bilo kakve reprojekcije (slika 3.3 (d)), ukoliko su već u toj projekciji,
- reprojekcija podataka polarnih particija (slika 3.3 (a) i (b)) je jednostavna i
- distorzija aspekta i površine teksela ograničene su na male-do-srednje vrednosti.

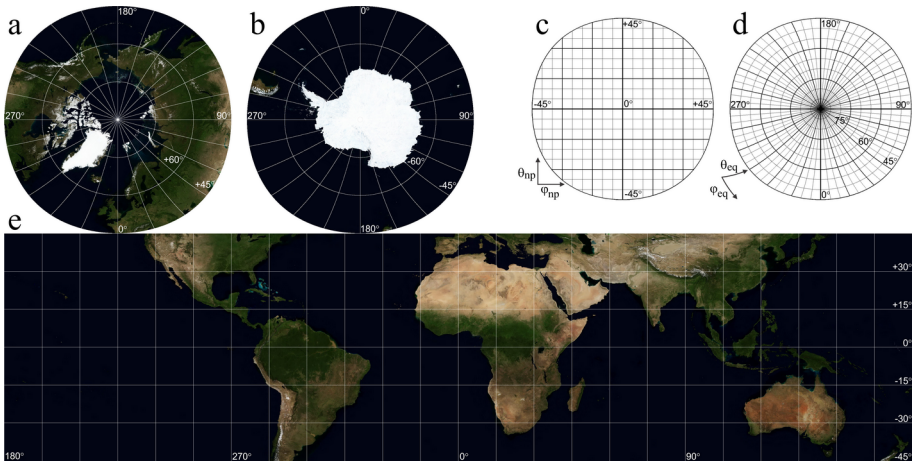


Slika 3.1: Podela na particije. Planeta se sastoji od tri particije: jedne ekvatorijalne i dve polarne (severne i južne).



Slika 3.2: Rešetka polarnih particija dobija se rotacijom ekvatorijalne rešetke za $\pm 90°$ oko globalne X-ose i odsecanjem na $\pm 45°$ geografske širine.

Broj particija je veoma bitan prilikom razmatranja zauzeća memorije. Naime, svaka particija zahteva sopstveni skup podataka, koji se sastoji od najmanje dve klipmape (za visine terena i teksturni-pokrivač) predstavljenih poljima tekstura [12]. Obzirom da se polje tekstura tretira kao jedinstvena celina od strane grafičkog drajvera, prebacivanje u ili iz grafičke memorije je veoma skupa operacija, pogotovu zbog njihove veličine. Zbog toga, trenutna implementacija EKM ograničava broj particija na tri.



Slika 3.3: Planarne projekcije particija. Oko 71.2% svih podataka je u standardnoj ekvidistantnoj cilindričnoj (lat/lon) projekciji (e), dok su polarne particije, (a) i (b), takođe u ekvidistantnoj cilindričnoj projekciji, ali rotirane za $\pm 90°$. Polarne lat/lon rešetke su ekvidistantne sa uzajamno upravnim osama φ i θ. Planarna projekcija severne polarne particije je prikazana sa polarnom (c) i globalnom/ekvatorijalnom (d) rešetkom.

Ove tri particije su razdvojene dvema odvojenim i veoma udaljenim granicama (paralelama na $+45°$ i $-45°$). Posmatrač mora biti jako visoko iznad površine, da bi mogao da vidi sve

tri particije istovremeno (iznad 3000 km za planetu Zemlju). Na takvim visinama, varijacije u visini terena su neprimetne i čitava planeta se može predstaviti jednostavnim elipsoidom.

Predložena projekcija pojednostavljuje pripremu podataka, obzirom da ne postoje direktna i inverzna transformacija za celu ekvatorijalnu particiju, ukoliko su podaci već u WGS84 (lat/lon) projekciji. Takođe, transformacije za obe polarne particije su jednostavne rotacije.

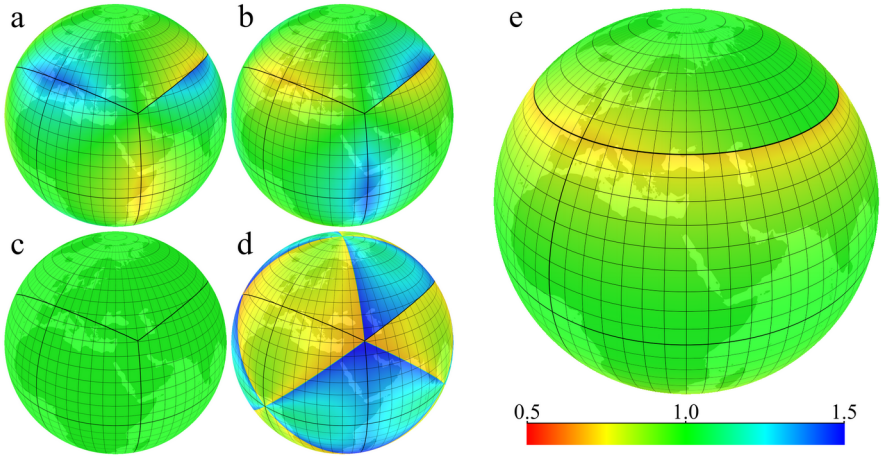
3.2 Distorzija tekstura

Distorzija je posledica projekcije sferoidne površine na ravan (direktna transformacija) i reprojekcije nazad na sferoidnu površinu (inverzna transformacija). Distorzija izazvana direktnom transformacijom je vrlo važna u kartografiji. Njene karakteristike direktno utiču na veličinu i oblik objekata na mapi. Stepen distorzije izazvane direktnom transformacijom se obično kvantitativno meri varijacijom razmere (tj. faktorom skaliranja) i ugaonom distorzijom, a vizuelno na mapi prikazuje korišćenjem Tisoovih indikatriksa [97, pog.4] (eng. *Tissot's indicatrices*).

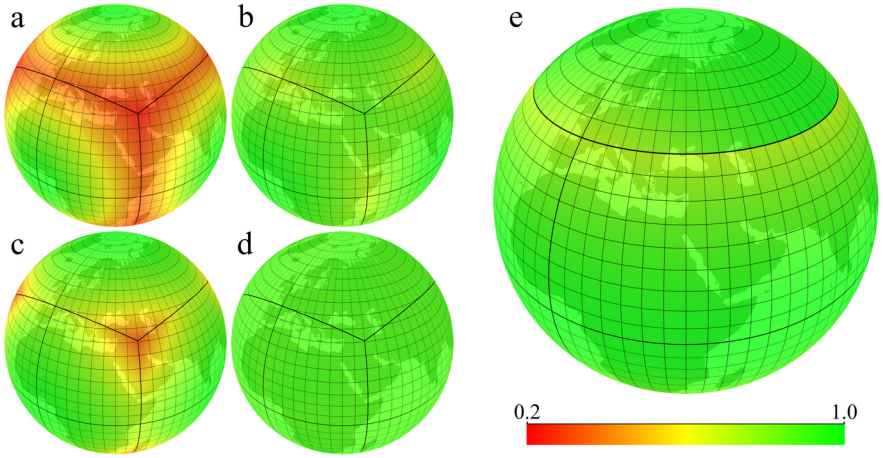
I faktor skaliranja i ugaona distorzija, u slučaju ekvidistantne cilindrične projekcije, postepeno rastu ka polovima. Počevši bez distorzije na ekvatoru, faktor skaliranja (tj. distorzija površine) raste do beskonačnosti, dok maksimalna ugaona distorzija (razlika u uglu presecanja dve linije pre i nakon projekcije) raste do 180°. Podelom površine u tri particije, kao što je prethodno predloženo, maksimalni faktor skaliranja i ugaona distorzija se ograničavaju na 1.42 i 19.76°, respektivno. Takođe, veličina prostora potrebnog za smeštanje projektovanih podataka je redukovana za aproksimativno 30%, u odnosu na standardnu jedno-particionu ekvidistantnu cilindričnu projekciju.

U kompjuterskoj grafici, sa druge strane, distorzija se manifestuje u suprotnom smeru, kroz primenu teksture na 3D model. Zbog toga ćemo u nastavku koristiti distorziju aspekta teksela (δ_{aspekt}) i distorziju površine teksela (δ_{povr}) za evaluaciju primenjene projekcije. Distorziju aspekta teksela definišemo kao odnos širine i visine teksela ($\delta_{aspekt} = \Delta_x/\Delta_y$), dok distorziju površine teksela definišemo kao odnos prostornih veličina teksela na tekućoj poziciji i teksela u centru projekcije ($\delta_{povr} = (\Delta_x\Delta_y)/(\Delta_{x0}\Delta_{y0})$). Oba distorziona faktora se računaju nakon (re)projektovanja teksela na sferoidnu površinu. Sl. 3.4 i 3.5 ilustruju distribuciju distorzionih faktora na površini Zemlje, za pomenute SK projekcije i predloženu tro-particionu ekvidistantnu cilindričnu projekciju.

Na Sl. 3.4 i 3.5 se jasno uočava da EKM unosi relativno malu distorziju u poređenju sa SK



Slika 3.4: Distribucija distorzije aspekta teksela za sledeće projekcije: Tangencijalnu sfernu kocku – $\delta^{tsk}_{aspekt} \in [0.71, 1.41]$ (a), Prilagođenu sfernu kocku – $\delta^{psk}_{aspekt} \in [0.71, 1.41]$ (b), Outerra sfernu kocku – $\delta^{osk}_{aspekt} \in [0.99, 1.00]$ (c), Kvadrilateralizovanu sfernu kocku – $\delta^{ksk}_{aspekt} \in [0.65, 1.54]$ (d) i Elipsoidne klipmape – $\delta^{ekm}_{aspekt} \in [0.71, 1.00]$ (e).



Slika 3.5: Distribucija distorzije površine teksela za sledeće projekcije: Tangencijalnu sfernu kocku – $\delta^{tsk}_{povr} \in [0.22, 1.00]$ (a), Prilagođenu sfernu kocku – $\delta^{PSK}_{povr} \in [0.71, 1.00]$ (b), Outerra sfernu kocku – $\delta^{osk}_{povr} \in [0.32, 1.00]$ (c), Kvadrilateralizovanu sfernu kocku – $\delta^{ksk}_{povr} \in [0.89, 1.00]$ (d) i Elipsoidne klipmape – $\delta^{ekm}_{povr} \in [0.71, 1.00]$ (e).

projekcijama. Distorzija aspekta teksela uzrokuje nejednako uzorkovanje teksture u različitim pravcima i zahteva veće teksture da bi se ostvarila ista prostorna pokrivenost u oba pravca. Efekat ove distorzije može biti umanjen korišćenjem anizotropnog filtriranja [79], ali zahtev za većim teksturama ne može biti eliminisan. Distorzija površine teksela sprečava konzistentno biranje nivoa detalja pri uzorkovanju tekstura, što može izazvati zamućenje prikaza. Iako EKM ima veću distorziju aspekta teksela u odnosu na Outera SK, distorzija površine teksela je znatno manja. Sa druge strane, Kvadrilateralizovana SK minimizuje distorziju površine, ali je distorzija aspekta

teksela značajna.

Iz svega prethodno navedenog može se zaključiti da EKM dobro balansira distorzije aspekta i površine teksela, uz primenu vrlo jednostavne projekcije.

3.3 Koordinatni sistemi

EKM koristi više različitih koordinatnih sistema za pristup podacima i vizuelizaciju. Obzirom da izvorni podaci predstavljaju uzorke sferoidne površine, oni su prirodno organizovani (smešteni na disku i u memoriji) i njima se pristupa korišćenjem sledeća tri lat/lon (polarna) koordinatna sistema:

- globalnog lat/lon koordinatnog sistema,
- lat/lon koordinatnog sistema tekuće particije i
- lat/lon koordinatnog sistema bloka/matrice-blokova.

Globalni lat/lon koordinati sistem služi kao jedinstveni okvir za globalno pozicioniranje. U svim formulama koje slede, njegove koordinate se predstavljaju grčkim slovima φ (longituda) i θ (latituda), bez indeksa – (φ, θ). Svaka particija ima svoj sopstveni lat/lon koordinatni sistem. Ekvatorijalni koordinatni sistem je identičan globalnom, dok su polarni zarotirani za $\pm 90^\circ$. Ukoliko se koordinate u narednim formulama odnose na bilo koju particiju, koristi se indeks *part* – $(\varphi_{part}, \theta_{part})$, a ukoliko su vezane za određenu particiju, indeks odgovara nazivu particije: $(\varphi_{eq}, \theta_{eq})$ za ekvatorijalnu, $(\varphi_{np}, \theta_{np})$ za severnu polarnu i $(\varphi_{sp}, \theta_{sp})$ za južnu polarnu particiju. Konverzija lat/lon koordinata iz severne polarne particije u koordinatni sistem ekvatorijalne particije definisana je jednačinama (3.1) i (3.2), dok jednačine (3.3) i (3.4) definišu inverznu transformaciju .

$$\theta_{eq} = \arcsin\left(\cos\left(\theta_{np}\right)\cos\left(\varphi_{np}\right)\right) \tag{3.1}$$

$$\varphi_{eq} = \arcsin\left(-\sin\left(\theta_{np}\right)/\cos\left(\theta_{np}\right)\right) \tag{3.2}$$

$$\theta_{np} = \arcsin\left(-\cos\left(\theta_{eq}\right)\cos\left(\varphi_{eq}\right)\right) \tag{3.3}$$

$$\varphi_{np} = \arcsin\left(\sin\left(\theta_{eq}\right)/\cos\left(\theta_{np}\right)\right) \tag{3.4}$$

Transformacija koordinata iz južnog polarnog u koordinatni sistem ekvatorijalne particije i inverzna transformacija su identične transformaciji iz ekvatorijalnog u severni polarni koordinatni sistem i iz severnog polarnog u ekvatorijalni, respektivno.

Zbog konačne preciznosti aritmetike centralnog procesora, količnik $sin(\theta_{eq})/cos(\theta_{np})$, u jednačini (3.4), može izaći iz opsega [-1,1]. Zbog toga, vrednost mora biti ograničena na dati opseg, kako bi izračunate vrednosti uvek bile korektne. Takođe, jednačina (3.4) nije definisana za $\theta_{np} = \pm 90^\circ$. Međutim, to nije nikakav problem, obzirom da latituda nikada ne prelazi $\pm 45^\circ$ u okvirima jedne particije ($\theta_{part} \in [-45^\circ, +45^\circ]$).

Particije se sastoje od blokova centriranih oko pozicije posmatrača. Sva rastojanja unutar bloka definisana su u lat/lon koordinatnom sistemu bloka, sa koordinatnim početkom u centru bloka. Koordinatni sistem bloka je uvek poravnat sa koordinatnim sistemom particije, tako da su blokovske koordinate $(\varphi_{blk}, \theta_{blk})$ zapravo rastojanja od centra bloka u koordinatnom sistemu date particije.

Za razliku od pristupa podacima, koji zahteva lat/lon koordinate, konačna vizuelizacija zahteva primenu pravougaonih Dekartovih koordinata. EKM koristi dva Dekartova koordinatna sistema, i to:

- geocentrični koordinatni sistem i
- topocentrični koordinatni sistem.

Globalni geocentrični koordinatni sistem se koristi u izračunavanjima koja se tiču planete kao celine i njegove koordinate su izražene bez indeksa – *(x,y,z)*, dok se topocentrični koordinatni sistem koristi za preciznu vizuelizaciju blokova i njegove koordinate su u svim narednim formulama označene indeksom t – (x_t, y_t, z_t).

Obzirom da se koristi elipsoidni model planete, transformacija iz globalnog lat/lon u geocentrični Dekartov koordinatni sistem definisana je jednačinama (3.5) do (3.8). U datim jednačinama h predstavlja visinu zadate tačke u odnosu na elipsoid, dok su a i b velika i mala poluosa elipsoida, respektivno.

$$N = \frac{a^2}{\sqrt{a^2 \cdot \cos^2\theta + b^2 \cdot \sin^2\theta}} \tag{3.5}$$

$$x = (N+h) \cdot \cos\theta \cdot \sin\varphi \tag{3.6}$$

$$y = \left(\frac{b^2}{a^2} \cdot N + h\right) \cdot \sin\theta \tag{3.7}$$

$$z = (N+h) \cdot \cos\theta \cdot \cos\varphi \tag{3.8}$$

WGS84 je na polovima spljošteni sferoid, tako da se geocentrične Dekartove koordinate (x, y, z) površine polarnih particija mogu izračunati na osnovu particionih lat/lon koordinata

$(\varphi_{np}, \theta_{np}, h)$, korišćenjem jednačina (3.9) do (3.13), uzajamnom zamenom poluprečnika u pravcu Y i Z ose i izračunavanjem koeficijenta N_{np} u odnosu na globalnu latitudu θ. Transformacija je ista za obe polarne particije.

$$\theta = \arcsin\left(\cos\theta_{np} \cdot \cos\varphi_{np}\right) \tag{3.9}$$

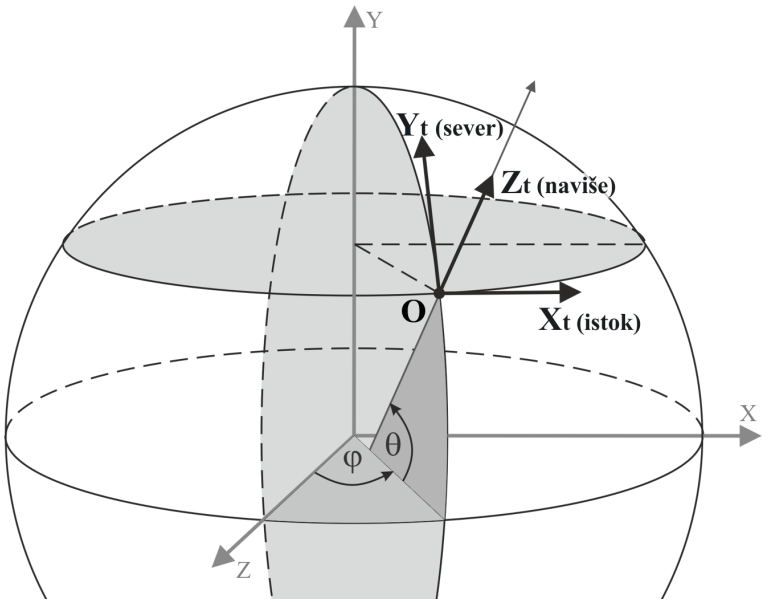
$$N_{np} = \frac{a^2}{\sqrt{a^2 \cdot \cos^2\theta + b^2 \cdot \sin^2\theta}} \tag{3.10}$$

$$x = (N_{np} + h) \cdot \cos\theta_{np} \cdot \sin\varphi_{np} \tag{3.11}$$

$$y = (N_{np} + h) \cdot \sin\theta_{np} \tag{3.12}$$

$$z = \left(\frac{b^2}{a^2} \cdot N_{np} + h\right) \cdot \cos\theta_{np} \cdot \cos\varphi_{np} \tag{3.13}$$

Da bi se sačuvala preciznost, vizuelizacija prostranih virtuelnih svetova se obično vrši korišćenjem „plutajućeg" koordinatnog početka, vezanog za poziciju kamere [101]. EKM neznatno modifikuje ovaj pristup, tako da umesto koordinatnog sistema vezanog za kameru koristi topocentrični koordinatni sistem (Sl. 3.6). Koordinatni početak topocentričnog sistema nalazi se na zadatoj tački na površini Zemlje, sa X_t-osom usmerenom ka istoku, Y_t-osom usmerenom ka severu i Z_t-osom usmerenom naviše, upravno na X_tY_t ravan.



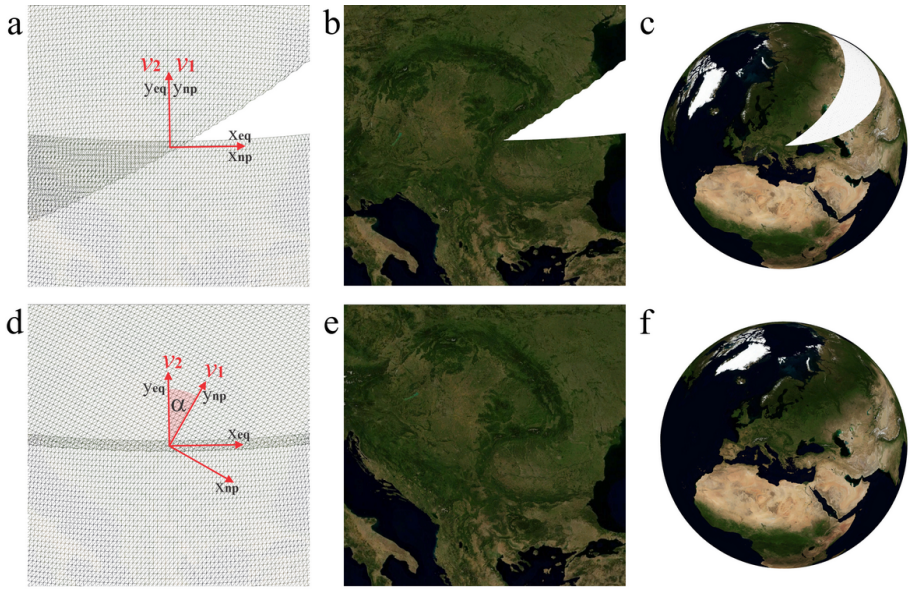
Slika 3.6: Topocentrični koordinatni sistem. Koordinatni početak postavljen je u zadatoj tački na površini Zemlje – O_t, X_t-osa je usmerena ka istoku, Y_t-osa je usmerena ka severu, a Z_t-osa naviše.

Transformacija iz globalnog geocentričnog u lokalni topocentrični koordinatni sistem može se jednostavno ostvariti korišćenjem vektorske aritmetike. Pozicija topocentričnog koordinatnog početka oduzima se od pozicije tekuće tačke, obe u geocentričnom sistemu, a zatim

rezultujući vektor treba projektovati na ose topocentričnog koordinatnog sistema (X_t, Y_t i Z_t) korišćenjem skalarnih proizvoda.

3.4 Spajanje particija

Svaka particija se vizuelizuje na osnovu najmanje dve klipmape, jedne za smeštanje visina terena, a druge za teksturni-pokrivač. Sve klipmape su nezavisne i sastoje se od nivoa (tj. slojeva) definisanih izvornim podacima, pri čemu se svaki nivo sastoji od jednog ili više blokova, koji čine matricu-blokova. Korišćenje više blokova po nivou može poboljšati performanse ukoliko se iscrtavaju samo blokovi koji su u vidnom polju (eng. *view frustum culling*). Utvrđivanje koji su blokovi u vidnom polju vrši se na centralnom procesoru.



Slika 3.7: Rotacija polarnih blokova. Svi blokovi su inicijalno centrirani u odnosu na poziciju posmatrača i poravnati sa topocentričnim osama, $X_{eq} - Y_{eq}$ za ekvatorijalnu i $X_{np} - Y_{np}$ za severnu polarnu particiju (a). To izaziva nepoklapanje na $\pm 45°$ geografske širine, obzirom da se polarna i ekvatorijalna rešetka sreću pod različitim uglovima na različitim geografskim dužinama (b), što uslovljava i rotaciju čitave polarne „kape" (c). Rotacijom polarnih blokova za ugao α, ugao pod kojim se rešetke sreću (d), površina se prikazuje kontinualno (e), a polarna kapa se vraća na svoju regularnu poziciju (f).

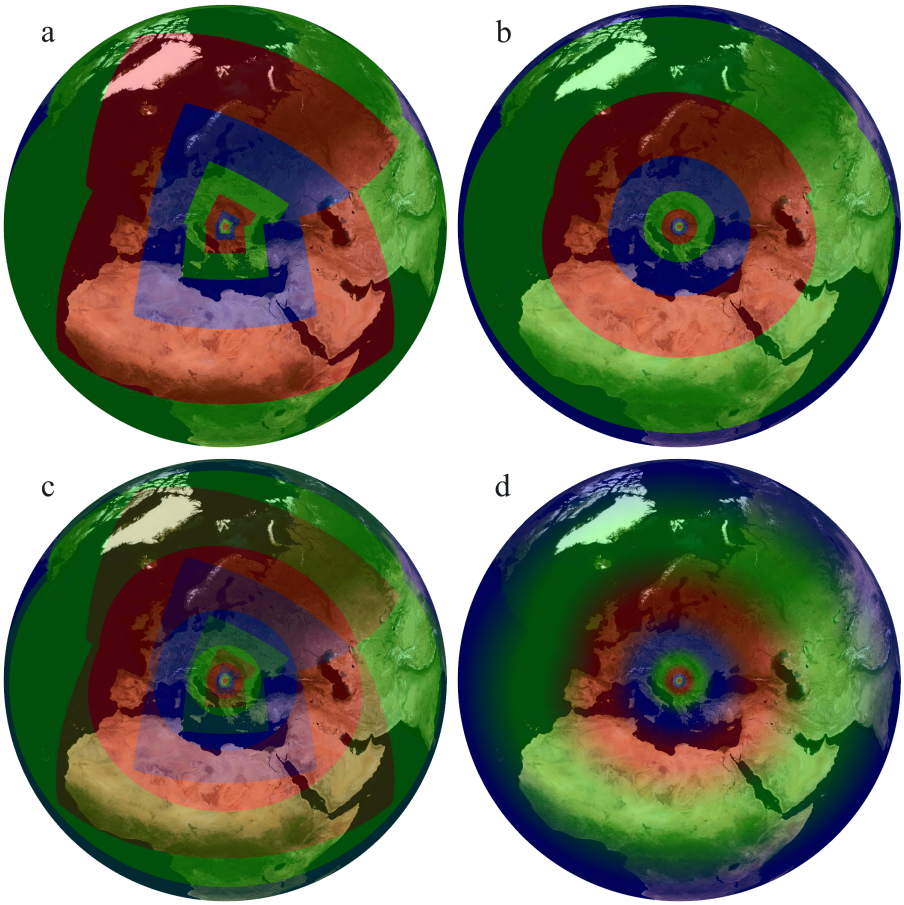
Jedna od prednosti EKM nad drugim algoritmima je mogućnost potpuno proceduralnog kreiranja gradivnih blokova površine planete, sa nivoom detalja određenim tekućim položajem posmatrača i bez rekurzivnih podela. Blokovi su centrirani u odnosu na položaj posmatrača i poravnati sa uzorcima izvornih podataka. Obzirom da se rešetke particija sreću pod različitim

uglovima na $\pm 45^\circ$ geografske širine, blokovi polarnih particija moraju biti zarotirani oko topocentrične Z_t-ose, kako bi se poklopili sa odgovarajućim blokovima iz ekvatorijalne particije (Sl. 3.7). Ugao rotacije α je ugao između X_t ili Y_t osa topocentričnih koordinatnih sistema blokova dve susedne particije. On se računa na osnovu tekuće pozicije posmatrača $(\varphi^v_{part}, \theta^v_{part})$, u lat/lon koordinatnom sistemu date particije, korišćenjem jednačina (3.14) do (3.16), u kojima $R_i(\xi)$ predstavlja rotacionu matricu u odnosu na osu i za ugao ξ.

$$v_1 = R_x(90^\circ) \cdot R_y(\varphi^v_{np}) \cdot R_x(\theta^v_{np}) \cdot [0,1,0]^{\mathrm{T}} \tag{3.14}$$

$$v_2 = R_y(\varphi^v_{eq}) \cdot R_x(-\theta^v_{eq}) \cdot [0,1,0]^{\mathrm{T}} \tag{3.15}$$

$$\alpha = \arccos\left(\frac{v_1 \cdot v_2}{|v_1| \cdot |v_2|}\right) \tag{3.16}$$



Slika 3.8: Prostiranje nivoa detalja i njihovi uzajamni odnosi: visine terena (a), teksturni-pokrivač (b), nezavisnost visina terena i teksturnog-pokrivača (c), mešanje nivoa teksturnog-pokrivača (d).

Korišćenje geografske rešetke, kao i rotacija polarnih blokova, stvaraju kompleksne i dinamički promenljive kombinovane blokove na $\pm 45^\circ$ geografske širine (Sl. 3.8 (a)). Odvajanje LOD šeme teksturnog-pokrivača od procesa kreiranja blokova (Sl. 3.8 (c)) omogućuje poravnanje

teksturnih-pokrivača na nivou fragmenata u okviru fragment šejdera, u odnosu na rastojanje od posmatrača u lat/lon prostoru particije. Aproksimacija, korišćena za kompenzaciju efekta krivljenja rešetke i rotacije, je dovoljno precizna da osigura poravnanje na nivou piksela do tekstura sa rezolucijom od 7.5 lučnih sekundi po tekselu. Za grublje teksture mogu se javiti uočljive granice nivoa detalja na prelaznoj liniji particija (Sl. 3.8 (b)), mada su one daleko od posmatrača, skrivene neravninama terena na manjim visinama ili potpuno nevidljive ako se koriste nivoi tekstura iz istog izvora podataka (tj. sa istim koloritom). Različiti izvori sa različitim koloritom mogu stvoriti diskontinuitete vidljive sa većih visina, ali čak i u takvim slučajevima mešanje susednih nivoa utiče na to da prelaz bude neprimetan (Sl. 3.8 (d)).

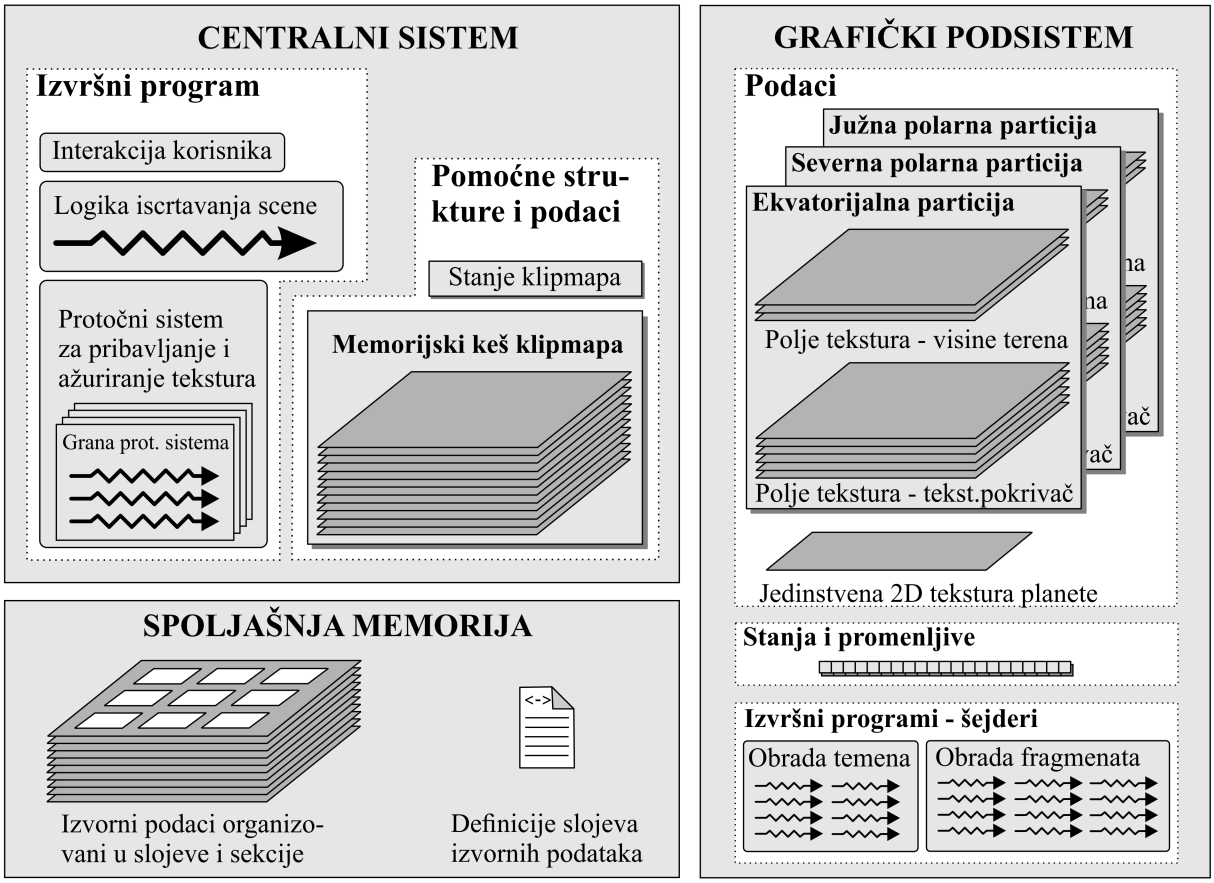
Primarni cilj mešanja nije da sakrije nesavršenosti, već da omogući primenu različitih teksturnih-pokrivača sa različitim koloritima. Predloženi algoritam vizuelizacije prevashodno je namenjen geografskim informacionim sistemima (GIS), gde izvori podataka značajno variraju, od satelitskih snimaka u vidljivom ili infra-crvenom delu spektra, preko ortografskih snimaka načinjenih iz aviona, do veštački nacrtanih slojeva. Štaviše, snimci istih oblasti koji potiču iz različitih izvora ili vremenskih perioda mogu biti prikazani u različitim bojama i sa različitim objektima (Sl. 3.9). U ovakvim primenama mešanje je neophodno kako bi se prelazi učinili glatkim.

Slika 3.9: Umekšani prelaz između različitih slojeva. Aerodromska pista na levoj slici (a) je kraća (stariji satelitski snimak). Kako se posmatrač približava (b), stariji satelitski snimak se meša sa novijim avionskim ortografskim snimkom. Na desnoj slici (c) pista je produžena, a postoje i dodatni objekti sagrađeni oko nje.

3.5 Arhitektura sistema i proces vizuelizacije

Arhitektura sistema koji vrši vizuelizaciju terena je vrlo složena jer mora da obezbedi efikasno pribavljanje podataka, minimalno ažuriranje potrebnih struktura, brzo izračunavanje, paralelizaciju zadataka i, konačno, upravljanje čitavim sistemom. Sam proces vizuelizacije distri-

buiran je između grafičkog i centralnog procesora. Grafički procesor je odgovoran za kreiranje geometrije terena i teksturisanje, dok centralni procesor pribavlja i prosleđuje podatke, upravlja iscrtavanjem i vrši sva izračunavanja visoke preciznosti koje zahteva algoritam. Na Sl. 3.10 prikazana je globalna arhitektura EKM sistema za vizuelizaciju terena, podeljena u tri celine: spoljašnju memoriju, centralni sistem (koga čine centralni procesor i operativna memorija) i grafički podsistem (koga čine grafički procesor sa pridruženom memorijom). Zaobljeni parvougaonici na slici označavaju aktivne komponente, koje se izvršavaju na odgovarajućem procesoru, dok strelice u njima ukazuju na paralelizaciju zadataka i izdvajanje aktivnosti u zasebne niti.



Slika 3.10: Globalna arhitektura EKM sistema za vizuelizaciju terena.

Spoljašnja memorija služi za skladištenje podataka. Podaci su organizovani u slojeve, a svaki sloj je podeljen na sekcije. Sekcija odgovara jednoj datoteci i predstavlja minimalnu količinu podataka koja se može pročitati sa spoljašnje memorije, ali i inkrement sa kojim se vrši ažuriranje memorijskih struktura. Zbog toga je bitno da veličina datoteka ne bude suviše velika, čime bi se usporilo učitavanje i povećala veličina memorijskih struktura, ali ni suviše mala, jer bi to značajno povećalo broj datoteka i otežalo indeksiranje. Više slojeva, koji predstavljaju isti ortografski snimak, može se organizovati u monolitnu datoteku, korišćenjem ECW [52] ili

MrSID [63] formata. Ovi formati podržavaju visok stepen kompresije i interno formiraju više nivoa detalja (tj. slojeva), ali zbog narušavanja podataka prilikom kompresije nisu pogodni za skladištenje mapa visina [29].

Definicije slojeva izvornih podataka zapamćene su u posebnoj XML datoteci. Datoteka sadrži kolekciju definicija mapa (XML element *MapCollection*), a svaka definicija mape (*Map*) određena je atributima: *ime* (*name*) i *tip* (*type*). Primer definicije kolekcije mapa dat je na Sl. 3.11. Atribut *ime* referencira odgovarajuću mapu, tj. skup podataka. Minimalni broj definicija mapa za tekuću implementaciju EKM je šest, po jedna mapa visina i mapa teksturnog-pokrivača za svaku od particija. Atribut *tip* određuje format skladištenja podataka. Informacija o formatu je bitna *protočnom sistemu za pribavljanje i ažuriranje tekstura*, jer se na osnovu nje vrši konverzija, ukoliko format skladištenja nije pogodan za primenu u teksturama. Detalji vezani za način predstavljanja podataka prikazani su u poglavlju 6 – *Ažuriranje klipmapa*.

```
<MapCollection>

    <Map name="GeoEquatorTexDX1" type="DXT1">

        <Sections path="D:\WorldData\EqTexDx1\E1\" pattern="E1_%X3_%Y3.dx1"
          proj="CS_WGS84" Xll="-180.0" Yll="-50.0" secX="0" secY="0"
          secXsize="600" secYsize="600" pixXsize="0.033333333333333333"
          pixYsize="0.033333333333333333" topDown="true"/>

        <Sections path="D:\WorldData\EqTexDx1\E2\" pattern="E2_%X3_%Y3.dx1"
          proj="CS_WGS84" Xll="-180.0" Yll="-50.0" secX="0" secY="0"
          secXsize="600" secYsize="600" pixXsize="0.016666666666666667"
          pixYsize="0.016666666666666667" topDown="true"/>

        <Sections path="D:\WorldData\EqTexDx1\E3\" pattern="E3_%X3_%Y3.dx1"
          proj="CS_WGS84" Xll="-180.0" Yll="-50.0" secX="0" secY="0"
          secXsize="600" secYsize="600" pixXsize="0.0083333333333333333"
          pixYsize="0.0083333333333333333" topDown="true"/>

    </Map>

</MapCollection>
```

Slika 3.11: Primer definicije kolekcije mapa sastavljene samo od jedne mape sa tri sloja.

Svaka definicija mape u sebi sadrži definicije proizvoljnog broja slojeva. Ukoliko je sloj sastavljen od sekcija, definicija sloja je predstavljena XML elementom *Sections*. Sekcije su opisane putanjom (*path*), šablonom imena (*pattern*), kartografskom projekcijom (*proj*), prostornim koordinatama donjeg levog ugla sekcije (*Xll* i *Yll*), koordinatama u okviru matrice sekcije čiji je donji levi ugao prethodno definisan (*secX* i *secY*), veličinom sekcije u uzorcima (*secXsize* i *secYsize*), veličinom koraka diskretizacije (*pixXsize* i *pixYsize*) i redosledom zapisa vrsta u datoteci (*topDown*). EKM koristi uniformno označavanje sekcija na osnovu njihovog položaja u okviru matrice sekcija datog nivoa. Položaj koordinata u imenu definisan je ključnim poljima

%X i %Y u atributu *pattern*, a celobrojne vrednosti koje slede data polja definišu broj cifara. Na primeru prvog sloja sa Sl. 3.11, sekcija sa koordinatama (5,3) nosi naziv „E1_005_003.dx1". Prethodno prikazano definisanje slojeva rasterskih mapa predstavlja modifikovanu verziju pristupa, originalno predloženog u [28, pog.8]. Na Sl. 3.11 je prikazano definisanje samo jedne mape sa tri sloja. Broj slojeva u realnoj implementaciji je daleko veći i zavisi od maksimalne rezolucije odgovarajuće mape. U eksperimentima opisanim u poglavlju 8 – *Efikasnost implementacije*, korišćeno je po 8 slojeva za mape visina i 16 slojeva za teksturne-pokrivače.

Centralni sistem obuhvata aktivne komponente, koje se izvršavaju na centralnom procesoru, i pomoćne strukture smeštene u operativnoj memoriji. Centralnu pokretačku snagu čitavog sistema za vizuelizaciju čini *logika iscrtavanja scene*. Ona upravlja procesom iscrtavanja, implementira sve ključne elemente EKM algoritma na centralnom procesoru i direktno komunicira sa grafičkim podsistemom korišćenjem OpenGL API-ja. Proces vizuelizacije ostvaruje se samo u jednoj niti zbog nemogućnosti efikasne višenitne komunikacije sa OpenGL drajverom. Naravno, *interakcija korisnika* i *pribavljanje i ažuriranje tekstura* ostvaruje se kroz zasebne niti. Korisničke akcije asinhrono se beleže u poseban registar, koga nit za vizuelizaciju interpretira pre svakog iscrtavanja scene. Kompletan algoritam po kome se izvršava proces vizuelizacije na centralnom procesoru prikazan je u poglavlju 5 – *Iscrtavanje scene*.

Za efikasan rad čitavog sistema veoma je bitan podsistem za *pribavljanje i ažuriranje tekstura*, obzirom da teksture predstavljaju praktično jedini izvor podataka u EKM algoritmu i da vrlo dinamično menjaju svoj sadržaj. On je organizovan u obliku protočnog sistema (eng. *pipeline*) sa više odvojenih grana. Svakoj klipmapi dodeljena je po jedna grana, a svaka grana sadrži više niti, od kojih svaka obavlja određeni specijalizovani zadatak. Detaljan prikaz rada ovog protočnog sistema prikazan je u poglavlju 6 – *Ažuriranje klipmapa*.

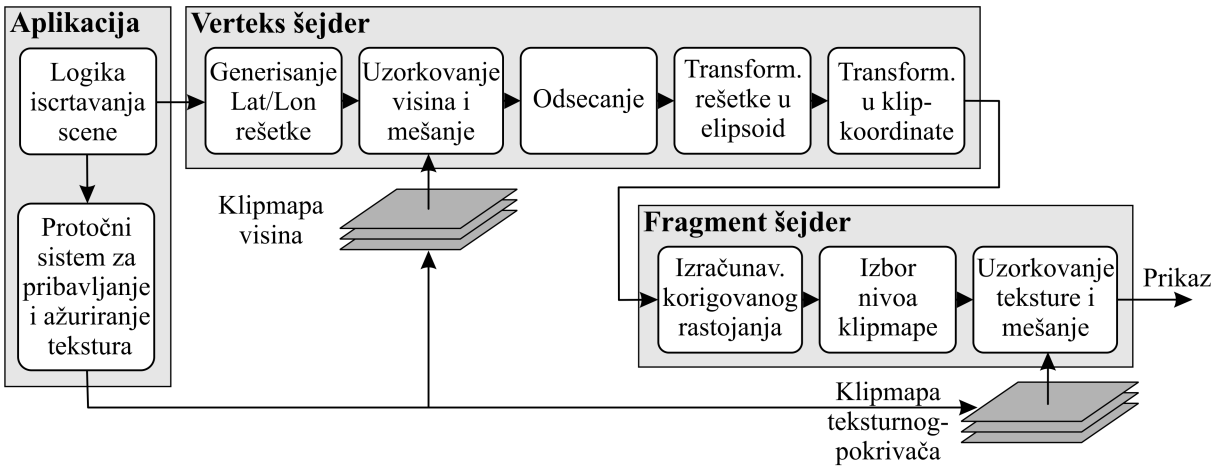
Protočni sistem za pribavljanje i ažuriranje tekstura formira *memorijski keš klipmapa* u operativnoj memoriji, popunjavajući ga podacima učitanim sa spoljašnje memorije. Na osnovu *stanja klipmapa*, uslovljenog kretanjem posmatrača i trenutnim stanjem u procesu ažuriranja, iz memorijskog keša se izdvajaju potrebni podaci i prosleđuju *grafičkom podsistemu*.

Najviše prostora u memoriji *grafičkog podsistema* zauzimaju klipmape. Svaka particija sadrži bar dve klipmape, jednu sa *visinama terena* (geometrijska klipmapa), a drugu sa satelitskim i avionskim ortografskim snimcima (*teksturni-pokrivač*). Prostor koji zauzimaju *stanja i promenljive* potrebne za iscrtavanja je zanemarljiv u poređenju sa klipmapama, koje su organizovane u *polja tekstura* [12]. Osim polja tekstura, postoji i jedna jedinstvena dvodimenzionalna (2D)

tekstura koja se koristi za vizuelizaciju planete kao celine pri velikim rastojanjima posmatrača.

Proces vizuelizacije na strani grafičkog procesora podeljen je u dve faze: kreiranje geometrije terena (kroz *obradu temena* u verteks šejderu) i primenu teksturnog-pokrivača (kroz *obradu fragmenata* u fragment šejderu). Svaka od ovih faza sastoji se od nekoliko koraka. U *verteks šejderu* se najpre formira pravougaona uniformna rešetka za svaki od blokova i za svako od temena se očitava visina iz odgovarajuće klipmape visina. Temena koja su prekrivena detaljnijim blokovima se odsecaju, kao i temena koja prelaze granice tekuće particije. Zatim se vrši deformisanje rešetke u skladu sa parametrima elipsoida i očitanim visinama. Poslednji korak obrade u verteks šejderu je izračunavanje konačnih koordinata temena množenjem matricom projekcije, modeliranja i pogleda.

Nakon verteks šejdera, vrši se rasterizacija i svaki fragment (preteča piksela, koji će biti prikazani na ekranu) se prosleđuje fragment šejderu. U fragment šejderu se najpre određuje prostorno rastojanje fragmenta od topocentričnog koordinatnog početka (projekcije stajne tačke posmatrača na površinu planete). Na osnovu prethodno određenog rastojanja, bira se nivo klipmape teksturnog-pokrivača iz koga se uzorkuju vrednosti i primenjuju na dati fragment.



Slika 3.12: EKM softverski protočni sistem za vizuelizaciju terena.

Sl. 3.12 prikazuje EKM softverski protočni sistem za vizuelizaciju terena. Detaljan prikaz svake od komponenti ovog protočnog sistema dat je u narednim poglavljima. Poglavlje 4 opisuje aktivnosti koje se izvršavaju na grafičkom procesoru, kroz faze obrade u verteks i fragment šejderu. Poglavlje 5 prikazuje algoritam logike za iscrtavanje scene, dok je poglavlje 6 posvećeno protočnom sistemu za pribavljanje i ažuriranje tekstura.

Poglavlje 4

Kreiranje i vizuelizacija blokova

EKM omogućuje potpuno generisanje terena na grafičkom procesoru, i to korišćenjem samo dva programabilna stepena 3D grafičkog protočnog sistema: verteks šejdera i fragment šejdera. Verteks šejder generiše geometriju terena superponiranjem visina smeštenih u geometrijskoj klipmapi na proceduralno formiranu elipsoidnu rešetku, dok fragment šejder primenjuje teksturni-pokrivač na površinu terena, definišući konačnu boju svakog piksela. U ovom poglavlju dat je detaljan opis formiranja i vizuelizacije blokova, osnovnih gradivnih elemenata terena, kroz procese koji se izvršavaju na grafičkom procesoru.

4.1 Verteks šejder

Blokovi su monolitski gradivni elementi geometrije terena, proceduralno generisani u verteks šejderu pri svakom iscrtavanju scene. Jedan blok ili mala grupa blokova (matrica-blokova) formira jedan nivo detalja terena. Svaki nivo centriran je u odnosu na položaj posmatrača ili neku drugu tačku fokusa, i poravnat sa rešetkom sledećeg grubljeg nivoa. Svi relevantni podaci prosleđuju se verteks šejderu korišćenjem uniformnih promenljivih, bez potrebe za atributima temena. Transformacija temena ostvaruje se kroz:

1. izračunavanje pozicije temena u okviru bloka, na osnovu *gl_VertexID*,
2. (opciono) izračunavanje vektora normale na površinu terena u datom temenu,
3. mešanje susednih nivoa na njihovom obodu,
4. izračunavanje rastojanja odsecanja:

(a) na osnovu detaljnijeg bloka/matrice-blokova,

(b) na osnovu granice particije,

5. izračunavanje pozicije temena u lokalnom topocentričnom koordinatnom sistemu i

6. izračunavanje klip-koordinata temena, kao konačnog izlaza verteks šejdera.

Prvi korak u obradi temena je njegovo pozicioniranje na rešetki i određivanje blokovskih koordinata (Sl. 4.1 (a)). Blokovske koordinate temena $(\varphi_{blk}^{vert}, \theta_{blk}^{vert})$ su zapravo lat/lon pomeraj u odnosu na centar bloka/matrice-blokova, izračunat na osnovu jednačina (4.1) and (4.2). *gl_VertexID* je ugrađena ulazna promenljiva OpenGL šejding jezika (eng. *OpenGL Shading Language* – GLSL) koja sadrži celobrojni indeks datog temena [56]. Veličina bloka (kao broj temena duž jedne stranice) definisan je promenljivom σ, dok ω određuje njegov pomeraj od centra matrice-blokova, izražen u broju blokova. Korak rešetke je postavljen promenljivom χ. Iako je moguće da koraci rešetke budu različiti duž različitih osa, tekuća implementacija podrazumeva jednak lat/lon razmak ($\chi_\varphi = \chi_\theta$). Blokovske koordinate se koriste, kao teksturne koordinate, za pristup mapi visina terena (geometrijska klipmapa) i teksturnom-pokrivaču (teksturna klipmapa), kao i za izračunavanje rotacije vektora normale.

$$\varphi_{blk}^{vert} = \Big((gl\_VertexID \% \sigma) - \frac{\sigma}{2} + \big(\omega_\varphi \cdot (\sigma - 1)\big) \Big) \cdot \chi_\varphi \tag{4.1}$$

$$\theta_{blk}^{vert} = \left(\frac{\sigma}{2} - \left\lfloor \frac{gl\_VertexID}{\sigma} \right\rfloor + (\omega_\theta \cdot (\sigma - 1)) \right) \cdot \chi_\theta \tag{4.2}$$

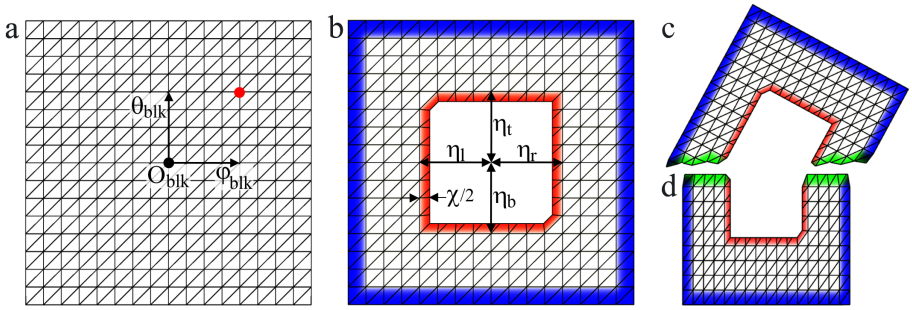
Teksturni-pokrivači terena u GIS aplikacijama su obično zasnovani na digitalnim ortografskim snimcima. Obzirom da fotografija inherentno sadrži senke, većina aplikacija ne zahteva dodatne efekte osvetljenja. Međutim, ako su ovakvi efekti ipak potrebni, neophodno je odrediti normale na datu površinu. One mogu biti unapred izračunate i smeštene u klipmapi, ili proceduralno generisane u verteks šejderu. Performanse proceduralnog pristupa, zasnovanog na centralnoj razlici [23, pog.11], predstavljene su u poglavlju 8 – *Efikasnost implementacije*. Iako se mapa visina uzorkuje pet puta za svako teme, performanse ne opadaju značajno. Orijentacija vektora normale se određuje na osnovu nagiba terena i lat/lon pomeraja datog temena od centra bloka/matrice-blokova.

Prelazni region na spoljašnjem obodu svakog nivoa klipmape se standardno koristi kod geometrijskih klipmapa [64] da glatko smeša različite nivoe detalja. Ovaj region prikazan je plavom bojom na Sl. 4.1 (b). Vektori normala moraju biti smešani na isti način kao i visine u

prelaznom regionu, što zahteva dodatna čitanja iz mape visina, ukoliko se normale generišu proceduralno.

Izračunavanje rastojanja odsecanja za temena u okviru bloka je korak vrlo specifičan za EKM. Umesto kreiranja prstena ili dodatnih traka za spajanje blokova, blokovi se iscrtavaju kao uniformne celine, sa unutrašnjim temenima odsečenim površinom bloka/matrice-blokova veće rezolucije (Sl. 4.1 (b)). Odsecanje se definiše u verteks šejderu kroz izračunavanje rastojanja odsecanja, na osnovu pozicije bloka/matrice-blokova više rezolucije, uz preklapanje za polovinu koraka rešetke. Preklopljena temena na unutrašnjem obodu se spuštaju naniže (crvena površina na Sl. 4.1 (b)), čime se popunjavaju pukotine izazvane T-spojevima, a u slučaju rešetki visoke rezolucije time se čak i uklanja potreba za prelaznim regionom.

Osim unutrašnjeg odsecanja, verteks šejder takođe definiše i odsecanje na granici particije, ukoliko blok preseca 45-tu paralelu. U polarnim particijama, izračunavanje rastojanja odsecanja zahteva transformaciju latitude temena, iz blokovskih u globalne lat/lon koordinate. Ovo je jedan od mnogih slučajeva kada je jednostavna među-particiona transformacija koordinata od ključne važnosti za performanse iscrtavanja. Za razliku od unutrašnjeg odsecanja, koje se definiše relativno u odnosu na blok, i kao takvo je vrlo precizno, odsecanje na granici particije je mnogo zahtevnije. Za blokove manjeg nivoa detalja, rastojanje odsecanja se računa u verteks šejderu direktno na osnovu globalne latitude temena. Za detaljnije blokove, rastojanje odsecanja se izračunava na bazi latitudinalnog pomeraja centra bloka od 45-te paralele i uticaja pozicije



Slika 4.1: Koraci u procesu kreiranja bloka: uniformna rešetka oko centra bloka – O_{blk} (a), unutrašnje odsecanje, ispuna pukotina spuštanjem temena na unutrašnjem obodu (crvena oblast) i mešanje (plava oblast) na spoljašnjem obodu (b), odsecanje na granici particije za polarni (c) i ekvatorijalni blok (d) i ispuna pukotina spuštanjem preklopljenih temena na granici particija (zelena oblast). Pozicija temena u koordinatnom sistemu bloka $(\varphi_{blk}, \theta_{blk})$ određena je vrednošću promenljive *gl_VertexID*. Unutrašnje odsecanje je definisano skupom od četiri rastojanja od centra bloka, određenih granicom bloka veće rezolucije $(\eta_l, \eta_r, \eta_t, \eta_b)$ i umanjenih za polovinu koraka rešetke $(\chi/2)$.

temena u lat/lon prostoru bloka na globalnu latitudu tog temena. Ovaj uticaj se izračunava na centralnom procesoru, u bliskoj okolini centra bloka, korišćenjem konačnih razlika. Zbog rotacije polarnih blokova u odnosu na globalnu geografsku rešetku, i longitudinalni i latitudinalni pomeraj temena u lokalnom lat/lon prostoru bloka utiču na njegovu globalnu latitudu. Sa druge strane, za ekvatorijalne blokove koristi se samo latitudinalni pomeraj, jer se ekvatorijalni i globalni lat/lon koordinatni sistemi poklapaju. Rastojanje odsecanja na granici particije se pomera za deo koraka rešetke, a temena na ivici se spuštaju naniže, kako bi se pokrile pukotine nastale zbog spajanja rešetki pod različitim uglovima (Sl. 4.1 (c) i (d)).

Transformacija iz lat/lon prostora bloka u topocentrični koordinatni sistem definisana je projektovanjem razlike vektora položaja tekućeg temena i koordinatnog početka topocentričnog sistema, oba u geocentričnom Dekartovom sistemu, na topocentrične ose. Geocentrična pozicija koordinatnog početka topocentričnog sistema i orijentacije njegovih osa su zajedničke za sva temena u okviru bloka/matrice-blokova, pa se zbog toga izračunavaju na centralnom procesoru, korišćenjem aritmetike dvostruke tačnosti. Međutim, ostale operacije, među kojima su izračunavanje geocentrične pozicije svakog temena i projektovanje na topocentrične ose, moraju biti izvršene u verteks šejderu. Moderni grafički procesori podržavaju izračunavanje korišćenjem dvostruke preciznosti, ali su performanse takvih operacija daleko ispod odgovarajućih operacija u jednostrukoj tačnosti. Dodatno, trigonometrijske funkcije, zajedno sa ostalim transcendentnim funkcijama, su implementirane u hardveru specijalnim funkcionalnim jedinicama, koje još uvek podržavaju samo operande jednostruke tačnosti. Imajući to u vidu, sva izračunavanja predloženog algoritma, koja se izvršavaju na grafičkom procesoru, ograničena su na jednostruku tačnost.

Topocentrična pozicija temena računa se globalno (kao što je prethodno objašnjeno) ili lokalno (korišćenjem interpolacije) zavisno od rastojanja u odnosu na centar bloka/matrice-blokova. Za mala rastojanja, EKM koristi linearnu interpolaciju zakrivljenosti površine elipsoida. Koeficijenti interpolacije se izračunavaju na centralom procesoru za tačku centra bloka/matrice-blokova, u koordinatnom sistemu date particije, a prosleđuju se verteks šejderu korišćenjem uniformnih promenljivih. Koji se od ova dva metoda izračunavanja pozicije koristi odlučuje se za svako teme posebno, a prelaz nastaje kada je projekcija greške na ekran manja od jednog piksela. U poglavlju 7 – *Preciznost implementacije*, pokazano je da takva oblast postoji za slučaj jednostavne linearne interpolacije i WGS84 elipsoida, uz dodatne informacije o preciznosti same implementacije. Iako je ostvarena subpikselska preciznost, ukoliko se elipsoid iscrtava kao rešetka, bez deformacija usled primene mape visina, vertikalni pokreti posmatrača mogu

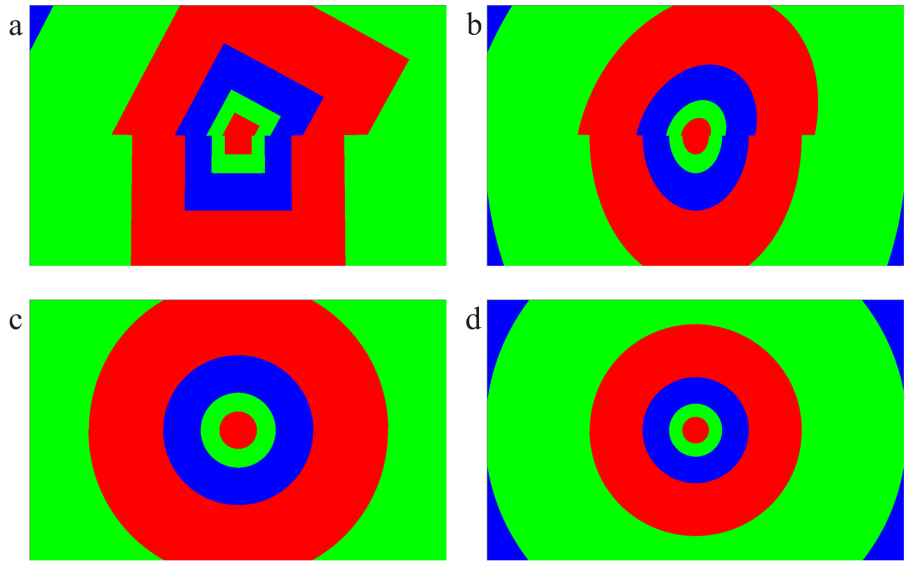
otkriti, iako jedva primetno, postojanje dva metoda pozicioniranja temena. Ovo je u potpunosti eliminisano korišćenjem oblasti u kojoj se mešaju oba pristupa.

Konačno, izračunavanje u okviru verteks šejdera se završava dobijanjem klip-koordinata temena, množenjem topocentričnih koordinata matricom projekcije, modeliranja i pogleda.

4.2 Fragment šejder

Osnovna funkcija fragment šejdera je da omogući primenu teksturnog-pokrivača na blokove kreirane u verteks šejderu. Fragment šejder bira odgovarajući nivo klipmape za svaki fragment ponaosob, zavisno od njegovog rastojanja (l), u lat/lon koordinatnom sistemu bloka ($\varphi_{blk}, \theta_{blk}$), od centra bloka/matrice-blokova. Ovakva metrika je nametnuta korišćenjem različitih izvora podataka za različite razmere, svaki sa svojim koloritom i sadržajem. Pri takvoj konfiguraciji, korišćenje metrike zasnovane na izvodima promene teksturnih koordinata u prostoru ekrana rezultuje uočljivim greškama u iscrtavanju (eng. *artifacts*). Anizotropno filtriranje je uključeno, ali je njegov efekat ograničen na samo jedan nivo klipmape.

Rastojanje l teksela sa teksturnim koordinatama (s,t) računa se po formuli (4.3). Kada bi se koristilo maksimalno rastojanje teksela duž osa geografske rešetke (Sl. 4.2 (a)), teksturni-pokrivač bi odgovarao obliku bloka na koji je primenjen, što bi stvorilo vidljivi diskontinuitet

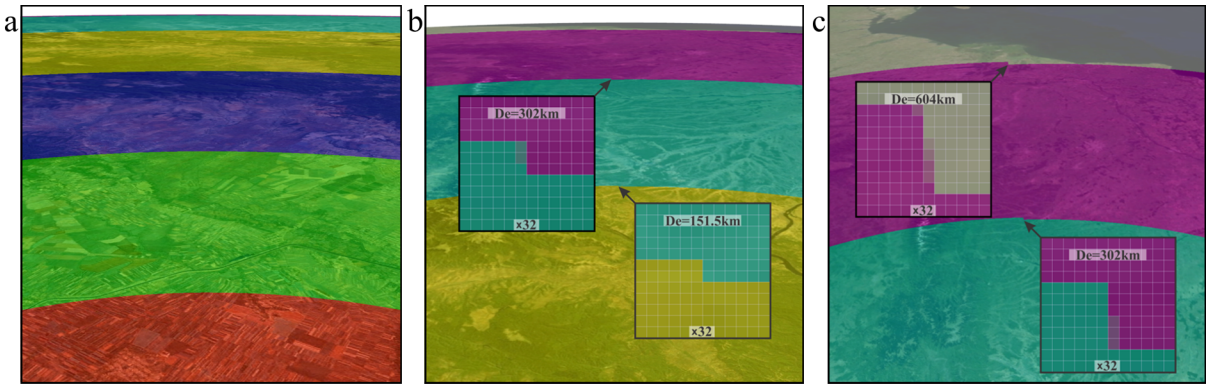


Slika 4.2: Različite metrike rastojanja pri izboru nivoa klipmape teksturnog-pokrivača primenjene na blokove koji presecaju 45° severne geografske širine: $l = max(|s|,|t|)$ (a), $l = \sqrt{s^2+t^2}$ (b), $l = \sqrt{s^2 cos^2\theta + t^2}$ (c) i $l = \sqrt{s^2 cos^2\theta + t^2}/cos\theta_w$ (d).

na $\pm 45^\circ$ geografske širine, usled činjenice da mešanje ne može da pređe granice particije. Izračunavanje rastojanja, korišćenjem Pitagorine teoreme u lat/lon prostoru, rezultuje eliptičnim oblikom nivoa (Sl. 4.2 (b)), sa vidljivim neslaganjem susednih particija na $\pm 45^\circ$ geografske širine, usled različitih orijentacija rešetki. Eliptični oblik je posledica konvergencije rešetke ka polovima. To se može kompenzovati množenjem longitudinalne teksturne koordinate s kosinusom latitude u koordinatnom prostoru tekuće particije (Sl. 4.2 (c)). Ova aproksimacija je dovoljno precizna, za WGS84 elipsoid, da omogući poklapanje na nivou piksela do vrlo velikih rastojanja.

Na Sl. 4.3 prikazano je odstupanja granica nivoa detalja teksturnih-pokrivača za ekvatorijalnu i polarnu particiju, za posmatrača na poziciji ($\varphi = 45^\circ, \theta = 21^\circ$), pogleda usmerenog ka istoku. Na malim visinama (h^v = 11 km) odstupanja ne postoje (Sl. 4.3 (a)). Odstupanja se mogu uočiti tek na visinama preko 60 km, na rastojanjima većim od 150 km, mereno po površini elipsoida (Sl. 4.3 (b)), ali su i tada toliko mala da se mogu otkriti samo ukoliko se zna gde se nalaze i uz značajno uvećanje prikaza. Odstupanja su jasno uočljiva (Sl. 4.3 (c)) tek na većim visinama (h^v = 260 km). Da bi se sprečilo da neravnine utiču na skrivanje prelaza, pri vizuelizaciji je isključen prikaz reljefa.

Prethodna kompenzacija pretvara eliptični oblik teksturnog-pokrivača u krug, ali se on prostire preko granica nivoa klipmape (što je posledica distorzije aspekta piksela, kao što je pomenuto u poglavlju 3). Zbog toga se veličina kruga mora smanjiti (jednačina (4.3)) proporcionalno kosinusu latitudinalnog rastojanja (θ_w) centra bloka/matrice-blokova od ekvatora



Slika 4.3: Odstupanja granica nivoa detalja teksturnih-pokrivača za ekvatorijalnu i polarnu particiju, za posmatrača na poziciji ($\varphi = 45^\circ, \theta = 21^\circ$), pogleda usmerenog ka istoku, pri malim (a), srednjim (b) i velikim visinama (c).

ili pola, zavisno od particije kojoj pripada (jednačina (4.4)).

$$l = \frac{\sqrt{s^2 \cdot \cos^2\theta + t^2}}{\cos\theta_w} \tag{4.3}$$

$$\theta_w = \begin{cases} \theta & za\, ekvatorijalnu\, particiju \\ \frac{\pi}{2} - |\theta| & za\, polarne\, particije \end{cases} \tag{4.4}$$

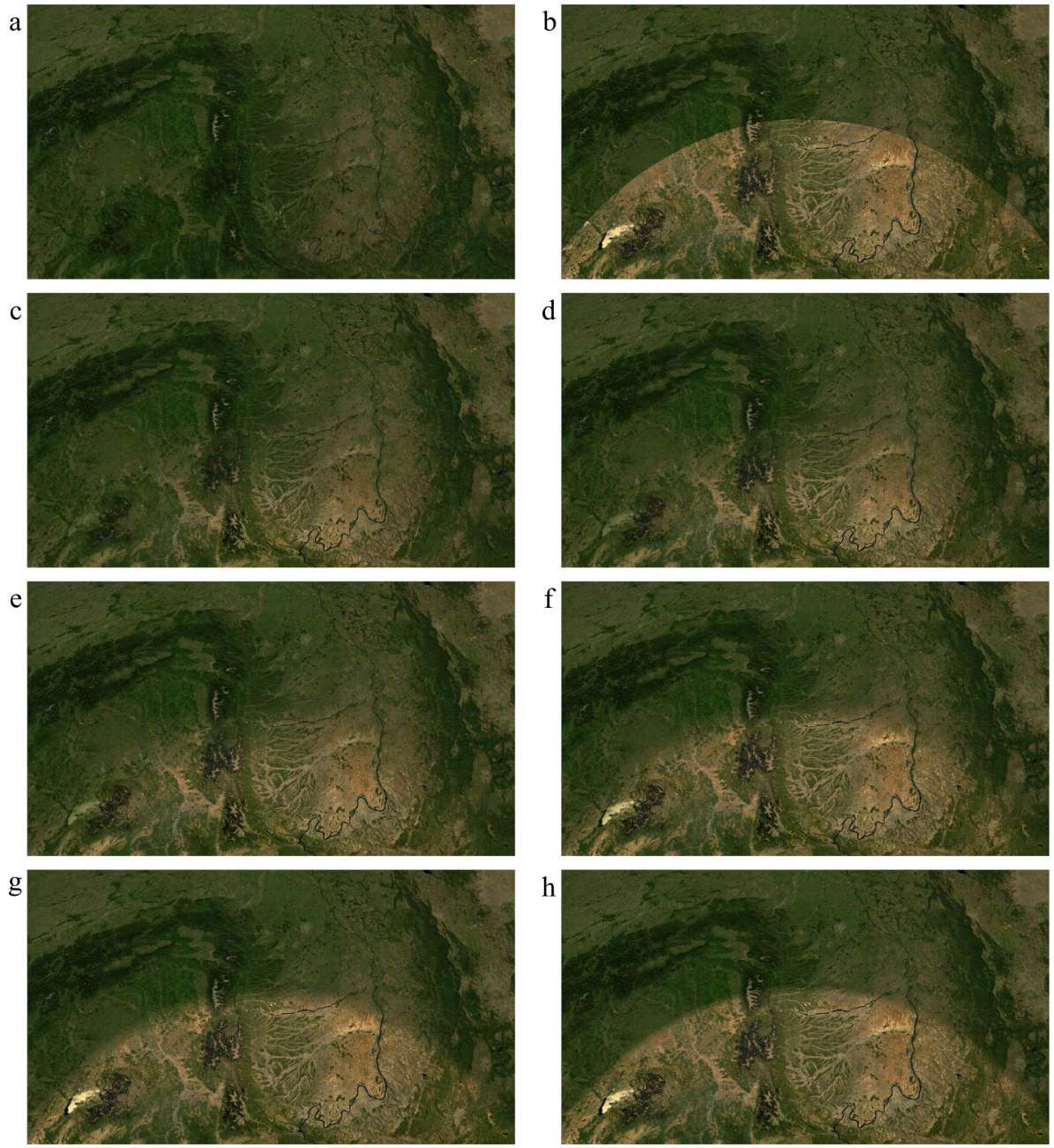
Odgovarajući nivo klipmape (λ) se izračunava korišćenjem jednačine (4.5), na osnovu prethodno izračunatog rastojanja fragmenta (l) i prostorne veličine aktivnog regiona najdetaljnije teksture (D_0). Aktivni-region je efektivna površina nivoa klipmape koja se koristi za iscrtavanje. Komplement do čitave površine nivoa naziva se „nevalidni okvir" i koristi za: keširanje, učitavanje podataka i kontrolu ažuriranja [100].

$$\lambda = \max\left(0, 2 + \log_2\left(l/D_0\right)\right) \tag{4.5}$$

Celobrojni deo λ služi za izbor osnovnog nivoa, dok razlomljeni deo definiše stepen mešanja sa narednim, grubljim nivoom. Razlomljeni deo se može dodatno skalirati, kako bi se smanjio region mešanja. Smanjivanje širine regiona mešanja omogućuje oštriji prikaz teksturnog-pokrivača u slučaju korišćenja tekstura manje rezolucije, jer se udaljava od posmatrača granica od koje se mešaju nivoi. Međutim, uži region mešanja čini prelaz uočljivijim.

Na Sl. 4.4, prikazan je uticaj širine regiona mešanja na uočljivost prelaza između nivoa detalja teksturnih-pokrivača sa različitim koloritom. U prikazu su korišćene teksture iz *Blue Marble Next Generation* [98] skupa podataka za manje detaljan prikaz (Sl. 4.4 (a)) i modifiovanog *Orthorectified Landsat Enhanced Thematic Mapper (ETM+) Compressed Mosaics* [60] skupa podataka za detaljniji prikaz (Sl. 4.4 (b)). Zbog različitog kolorita, prelaz na udaljenosti od 302 km, mereno po površini elipsoida, je jasno uočljiv (Sl. 4.4 (b)). Korišćenjem mešanja duž čitave širine prstena svakog od nivoa čini prelaze potpuno nevidljivim (Sl. 4.4 (c)). Ovo je podrazumevani način primene klipmapa. Međutim, ukoliko se zbog ograničenja sistema na kome se izvršava aplikacija koriste teksture nižih rezolucija od neophodnih za datu veličinu ekrana, oštrina prikaza može se povećati, u izvesnoj meri, smanjivanjem širine regiona mešanja. Eksperimentalno je utvrđeno da i nakon sužavanja regiona mešanja sa faktorom 1.5, prelaz ne postaje uočljiv (Sl. 4.4 (d)). Daljim sužavanjem regiona mešanja prelaz postaje sve uočljiviji. Prikazano je sužavanje regiona mešanja sa faktorom 2 (Sl. 4.4 (e)), 4 (Sl. 4.4 (f)), 8 (Sl. 4.4 (g)) i 16 (Sl. 4.4 (h)).

EKM odvaja rukovanje teksturnim-pokrivačem od rukovanja geometrijom. Prednost je mogućnost da se koriste različite veličine i rezolucije tekstura za teksturni-pokrivač i geometriju.



Slika 4.4: Uticaj širine regiona mešanja na uočljivost prelaza između nivoa detalja teksturnih-pokrivača sa različitim koloritom, u slučaju kada se koriste: teksture istog kolorita (a), teksture različitog kolorita bez mešanja (b), mešanje na čitavoj širini prstena (c), mešanje na 2/3 (d), 1/2 (e), 1/4 (f), 1/8 (g) i 1/16 širine prstena (h).

Ovo je veoma korisno u slučaju EKM, jer omogućuje primenu teksturnih-pokrivača vrlo visoke rezolucije bez uticaja na brzinu iscrtavanja, dok je gustina blokova (tj. rezolucija mape visina) direktno proporcionalna vremenu iscrtavanja scene.

Ukoliko se teren posmatra pod malim uglom, mešanje nivoa (tj. trilinearno filtriranje) može rezultovati mutnim prikazom. Oštrina prikaza teksturnog-pokrivača može se značajno

poboljšati primenom anizotropnog filtriranja [79]. Time se umanjuje i efekat distorzije aspekta teksela, uslovljenog izabranom projekcijom. Međutim, korišćenje anizotropnog filtriranja rezultuje pogrešno obojenim fragmentima na graničnim linijama slojeva teksturnog-pokrivača, zbog nedefinisanih izvoda teksturnih koordinata na ivicama tekstura. To se koriguje eksplicitnim izračunavanjem izvoda i postavljanjem njihovih vrednosti GLSL funkcijom *textureGrad* [56].

U naredna dva poglavlja opisan je deo EKM protočnog sistema za vizuelizaciju terena koji se izvršava na strani centralnog procesora.

Poglavlje 5

Iscrtavanje scene

EKM deli površinu čitave planete u tri particije, svaku sastavljenu od višestrukih koncentričnih nivoa detalja. Svaki nivo čini jedan ili više blokova organizovanih u matricu-blokova. Procesi pripremanja, pribavljanja podataka i iscrtavanja koriste više niti. Ovo poglavlje posvećeno je niti koja se izvršava na centralnom procesoru i vrši iscrtavanje scene, kao i njenim zadacima tokom osvežavanja prikaza.

5.1 Globalno iscrtavanje

Tri osnovna zadatka koja obavlja nit za iscrtavanje scene prilikom svakog osvežavanja prozora, definisana Alg. 1, su:

- obrada ažuriranja na čekanju,
- izvršavanje komandi i
- isctavanje scene.

Obrada ažuriranja na čekanju je faza koja definiše kvotu za ažuriranje i obezbeđuje obilazak svih nivoa klipmapa u sve tri particije, tragajući za ažuriranjima koja još nisu obrađena. Detalji vezani za ažuriranja klipmapa predstavljeni su u poglavlju 6 – *Ažuriranje klipmapa*. Svrha kvote je da ograniči maksimalnu količinu podataka koja se može poslati grafičkoj kartici tokom jednog osvežavanja prikaza. Time se omogućuje tečno iscrtavanje scene i distribucija ažuriranja kroz više uzastopnih osvežavanja.

Faza *izvršenja komandi* prethodi iscrtavanju scene, modifikuje položaj i orijentaciju kamere, kao i ostale atribute scene u skladu sa interakcijom korisnika.

```
a ← obradi ažuriranja na čekanju;
k ← izvrši komande;
if a ili k then
    iscrtaj scenu
end
```

Algoritam 1: Osvežavanje prikaza

Faza *iscrtavanja scene* osvežava prikaz ako i samo ako jedna od prethodnih faza to zahteva, jer je došlo do modifikacije scene usled primene nekog od ažuriranja ili komande. Ukoliko je posmatrač visoko iznad površine ($h^v > h_{lim}$), planeta se iscrtava bez klipmapa, korišćenjem jednodelnog proceduralno generisanog elipsoida. Time se sprečava provera složenih uslova i anomalija, a takođe olakšava iscrtavanje više planeta u okviru velikih konstelacija.

```
if h^v > h_lim then
    iscrtaj jednodelni elipsoid;
else
    θ_n ← severna vidljiva latituda(h^v, θ^v);
    θ_s ← južna vidljiva latituda(h^v, θ^v);
    if θ_n > 45° then
        iscrtaj particiju(severna);
    end
    if θ_s < 45° and θ_n > −45° then
        iscrtaj particiju(ekvatorijalna);
    end
    if θ_s < −45° then
        iscrtaj particiju(južna);
    end
end
```

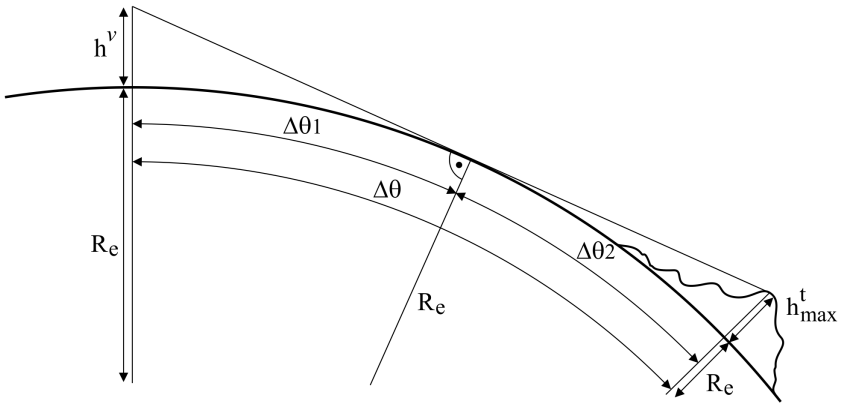
Algoritam 2: Iscrtavanje scene

Ako je posmatrač blizu površine planete, visina na kojoj se nalazi (h^v) i globalna latituda (θ^v) se koriste za izračunavanje opsega geografskih širina vidljivih iz date tačke – θ_n i θ_s, gde su: $\theta_n = \theta^v + \Delta\theta$ i $\theta_s = \theta^v - \Delta\theta$. $\Delta\theta$ izračunava se po formuli (5.1), na osnovu sferne aproksimacije

planete Zemlje, sa ekvivalentnim radijusom R_e.

$$\begin{aligned}\Delta\theta &= \Delta\theta_1 + \Delta\theta_2 \\ &= \arccos\left(\frac{R_e}{R_e + h^v}\right) + \arccos\left(\frac{R_e}{R_e + h^t_{max}}\right)\end{aligned} \tag{5.1}$$

Prethodna jednačina podrazumeva najgori slučaj, kada ne postoje prepreke na putu ka horizontu i vrh najviše planine se nalazi na najvećem rastojanju na kome može da se vidi. Za planetu Zemlju, može se smatrati da ne postoji planina viša od 9000 m (h^t_{max}). Sl. 5.1 ilustruje proces izračunavanja maksimalnog opsega vidljivih geografskih širina.



Slika 5.1: Maksimalni opseg latituda $\Delta\theta$, obuhvaćen pogledom posmatrača, zavisi od visine na kojoj se posmatrač nalazi h^v, u odnosu na površinu sferne planete poluprečnika R_e, i visine najvišeg vrha planine h^t_{max}.

Particija se iscrtava samo ukoliko preseca izračunati opseg. Glavni koraci u procesu iscrtavanja scene predstavljeni su u Alg. 2.

5.2 Iscrtavanje particije

Svaka od particija se iscrtava, na osnovu Alg. 3, primenom sledećih koraka:

- konverzija globalnih lat/lon koordinata posmatrača (φ^v, θ^v) u lat/lon koordinate tekuće particije $(\varphi^v_{part}, \theta^v_{part})$;
- (samo za polarne particije) izračunavanje ugla rotacije rešetke – α;
- izračunavanje minimalne prostorne veličine bloka/matrice-blokova – ρ;
- provera konzistentnosti klipmapa i

- iscrtavanje blokova/matrice-blokova dok je veličina ρ manja od maksimalne vrednosti – ρ_{max}, duplirajući veličinu u svakoj iteraciji.

Sva izračunavanja u okviru particije vrše se u lat/lon koordinatnom sistemu te particije. Zato je prvi korak transformacija pozicije posmatrača iz globalnog (φ^v, θ^v) u koordinatni sistem date particije $(\varphi^v_{part}, \theta^v_{part})$. Za polarne particije potrebno je izračunati i ugao rotacije rešetke, na osnovu jednačina (3.14) do (3.16).

$(\varphi^v_{part}, \theta^v_{part}) \leftarrow (\varphi^v, \theta^v)$;
$\alpha \leftarrow$ izračunaj ugao rotacije rešetke;
proveri konzistentnost klipmapa;
$\rho \leftarrow$ izračunaj minimalnu veličinu bloka/matrice-blokova;
while $\rho < 2 \cdot \rho_{max}$ **do**
    iscrtaj matricu-blokova(ρ, φ^v_{part}, θ^v_{part}, α);
    $\rho \leftarrow 2 \cdot \rho$;
end

Algoritam 3: Iscrtavanje particije

Prostorna veličina najdetaljnijeg bloka/matrice-blokova (tj. minimalna veličina bloka) računa se korišćenjem jednačine (5.2), na osnovu: relativne visine posmatrača – h^v, ugla vidnog polja kamere u horizontalnoj ravni – Φ_{hor} i efektivnog poluprečnika Zemlje – R_e. Obzirom da je veličina bloka diskretna vrednost, nema potrebe za tačnim izračunavanjem vidljivog dela elipsoida. Zbog toga jednačina (5.2) koristi sfernu aproksimaciju, čija je zapremina jednaka zapremini referentnog elipsoida ($R_e = (a \cdot b)^{1/3} \approx (2 \cdot a + b)/3 = 6371km$).

$$\rho = \frac{h^v}{R_e \cdot \tan(0.5 \cdot \Phi_{hor})} \cdot \frac{180}{\pi} \tag{5.2}$$

Broj nivoa klipmapa potrebnih za iscrtavanje particije nije unapred poznat. Nivoi, tj. blokovi/matrice-blokova, se iterativno iscrtavaju dok se ne dostigne maksimalna veličina, definisana opsegom vidljivih geografskih širina. Maksimalna veličina bloka računa se kao: $\rho_{max} = 2 \cdot \Delta\theta / \delta_{aspektmin}$, gde je $\Delta\theta$ definisano u jednačini (5.1), a $\delta_{aspektmin}$ je minimalna vrednost distorzije aspekta teksela (Sl. 3.4).

Pre iscrtavanja, klipmape se proveravaju na konzistentnost. Svaki sloj se proverava u odnosu na tekuću poziciju posmatrača, a zahtevi za ažuriranje se šalju ukoliko je to potrebno. Ovaj korak se koristi i za izbor nivoa sa najvećom rezolucijom koji je na raspolaganju u skladu sa relativnom visinom posmatrača i konzistentnošću (ažuriranošću) nivoa klipmape.

5.3 Iscrtavanje matrice-blokova

Već je pomenuto ranije da je svaki nivo detalja, koji se sastoji od jednog bloka ili grupe blokova organizovanih u matricu, centriran u odnosu na poziciju posmatrača. Međutim, da bi se izbegao efekat „plutanja", rešetka mora da bude poravnata sa uzorcima visina tekućeg *nivoa*. Štaviše, konzistentno iscrtavanje ivica nivoa zahteva poravnanje sa uzorcima sledećeg, manje detaljnog, nivoa (*nivo+1*). Zbog toga se centar matrice-blokova ne poklapa striktno sa pozicijom posmatrača, već sa rešetkom sledećeg grubljeg nivoa. Prva linija u Alg. 4 određuje centar tekuće matrice-blokova $(\varphi^{tm}_{part}, \theta^{tm}_{part})$ u lat/lon koordinatama particije, na osnovu tekućeg položaja posmatrača i poravnanja sa rešetkom sledećeg nivoa.

Ukoliko matrica-blokova preseca tekuću particiju, izračunava se matrica projekcije modeliranja i pogleda (PMV), a svi vidljivi blokovi iscrtavaju.

$(\varphi^{tm}_{part}, \theta^{tm}_{part}) \leftarrow$ poravnanje rešetke$(\varphi^{v}_{part}, \theta^{v}_{part}, level+1)$;
$(\varphi^{tm2}_{part}, \theta^{tm2}_{part}) \leftarrow$ poravnanje rešetke$(\varphi^{v}_{part}, \theta^{v}_{part}, level+2)$;
if *matrica-blokova* $\cap$ *tekuća particija* **then**
  izračunaj PMV matricu$(\varphi^{v}_{part}, \theta^{v}_{part}, \varphi^{tm}_{part}, \theta^{tm}_{part}, type_{part})$;
  forall the *blok* $\in$ *matrica-blokova* **do**
    if *blok je vidljiv* **then**
      iscrtaj blok;
    end
  end
end
postavi ravni odsecanja$(\varphi^{tm}_{part}, \theta^{tm}_{part}, \varphi^{tm2}_{part}, \theta^{tm2}_{part})$;

Algoritam 4: Iscrtavanje matrice-blokova

Dok se matrica projekcije i pogleda (PV) izračunava samo jednom za svaki pokret kamere, matrica modeliranja (M) se menja sa svakom iscrtanom matricom-blokova. Alg. 5 prikazuje glavne korake u procesu izračunavanja PMV matrice.

Topocentrična pozicija (x_t, y_t, z_t) centra matrice-blokova izračunava se na osnovu tipa particije, jer se postupak razlikuje za ekvatorijalnu i polarne particije. Osim toga, polarne particije zahtevaju dodatnu rotaciju blokova, oko topocentrične Z_t-ose za ugao α $(R_z(\alpha))$, definisanu jednačinom (3.16). Centar koordinatnog sistema postavlja se ispod posmatrača $(\varphi^{v}_{part}, \theta^{v}_{part})$ na visini definisanoj površinom terena (h^t). Matrica M predstavlja kumulativnu transformaciju

$PV \leftarrow$ preuzmi matricu projekcije i pogleda;

$M \leftarrow I$;

if $type_{part} = ekvatorijalni$ **then**

  $(x_t, y_t, z_t) \leftarrow WGS2Topocentric(\varphi^v_{part}, \theta^v_{part}, h^t, \varphi^{tm}_{part}, \theta^{tm}_{part}, h^t)$;

else

  $(x_t, y_t, z_t) \leftarrow WGS2TopocentricPolar(\varphi^v_{part}, \theta^v_{part}, h^t, \varphi^{tm}_{part}, \theta^{tm}_{part}, h^t)$;

  $M \leftarrow M \cdot R_z(\alpha)$;

end

$M \leftarrow M \cdot T(x_t, y_t, z_t) \cdot R_y(\varphi^{tm}_{part} - \varphi^v_{part}) \cdot R_x(\theta^v_{part} - \theta^{tm}_{part})$;

$PMV \leftarrow PV \cdot M$;

Algoritam 5: Izračunavanje PMV matrice

sastavljenu od: translacije centra matrice-blokova i dve rotacije oko topocentričnih (lokalnih) osa. Rotacije su definisane longitudinalnim i latitudinalnim pomerajima centra u odnosu na položaj posmatrača.

Iscrtavanje bloka je zapravo samo jedan poziv OpenGL funkcije *glDrawElements*, obzirom da nisu potrebni nikakvi atributi za definisanje temena rešetke i da se isti indeksni bafer koristi za sve blokove planete. Blokovi koji ne presecaju vidno polje mogu se efikasno eliminisati iz procesa iscrtavanja na osnovu orijentacije kamere i njenog vidnog polja u horizontalnoj ravni (Φ_{hor}). Uticaj eliminacije blokova koji nisu vidljivi na performanse, prilikom korišćenja matrice-blokova, prikazan je u poglavlju 8 – *Efikasnost implementacije*.

Pre završetka iscrtavanja tekuće matrice-blokova, moraju se postaviti ravni odsecanja za naredni nivo. Ravni odsecanja se određuju na osnovu prostorne veličine i položaja detaljnije matrice-blokova. Obzirom da su i detaljnija i grublja matrica-blokova poravnate sa rešetkama odgovarajućih nivoa, izračunavanje rastojanja ravni odsecanja mora da uključi poklapanje dva različita nivoa. Zbog toga centar sledećeg nivoa ($\varphi^{tm2}_{part}, \theta^{tm2}_{part}$) takođe mora biti izračunat u tekućem prolazu procedure iscrtavanja matrice-blokova.

Efikasno iscrtavanje u slučaju brzog kretanja posmatrača zahteva efikasno pribavljanje i prosleđivanje podataka. Sledeće poglavlje prikazuje proces ažuriranja klipmapa i višenitnu protočnu organizaciju procesa pribavljanja tekstura.

Poglavlje 6

Ažuriranje klipmapa

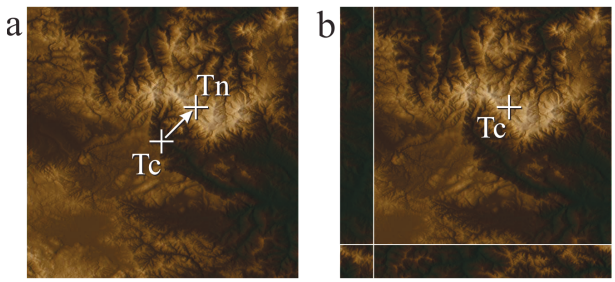
Pribavljanje podataka i ažuriranje odgovarajućih struktura predstavlja vitalnu funkciju mnogih algoritama. Kada je u pitanju vizuelizacija terena velikih razmera, pogotovu ukoliko se posmatrač brzo kreće, pribavljanje podataka je presudan faktor efikasnosti čitavog sistema. Klipmape inherentno pružaju konzistentno keširanje, efikasno pribavljanje i predstavljanje podataka u obliku pogodnom za interpretaciju na grafičkom procesoru. U ovom poglavlju prikazan je postupak ažuriranja klipmapa i dat predlog implementacije softverskog protočnog sistema, koji ga čini efikasnijim, uz minimizaciju uticaja ažuriranja tekstura na vreme iscrtavanja scene.

6.1 Ažuriranje nivoa klipmape

Ažuriranje nivoa klipmape vezano je za horizontalno kretanje posmatrača, tako da centar nivoa prati njegov trenutni položaj. Naivni pristup u ažuriranju zahtevao bi promenu čitavog nivoa za svaki, makar i najmanji, pokret posmatrača. Međutim, kako je kretanje prostorno koherentno, nakon svakog pokreta deo prostora koji pokriva nivo nije promenjen. Koliki je taj prostor zavisi od brzine kretanja i detaljnosti nivoa. Što su brzina posmatrača i detaljnost nivoa veći to je i promena nivoa nakon pokreta veća. Da bi se iskoristila prostorna koherencija i ažurirao samo potreban deo nivoa, bez njegove reorganizacije, koristi se – *toroidno ažuriranje*.

Pri toroidnom ažuriranju, novi podaci se upisuju na mestu gde bi prostorno trebalo da budu, ali po modulu veličine datog nivoa. Na Sl. 6.1, ilustrovan je postupak toroidnog ažuriranja pri kretanju posmatrača iz tačke T_c u tačku T_n.

EKM pri svakom pomeraju posmatrača proverava ažurnost svih nivoa klipmapa, utvrđi-



Slika 6.1: Toroidno ažuriranje nivoa klipmape (b) pri pomeranju posmatrača iz tačke T_c u tačku T_n (a).

vanjem rastojanja posmatrača od njihovih centara. Ukoliko je rastojanje veće od širine okvira nivoa, tj. „praga ažuriranja", šalje se zahtev za ažuriranje. Okvir nivoa služi kao keš i oblast za učitavanje novih podataka pri kretanju, a predstavlja razliku između fizičke veličine nivoa klipmape i efektivne veličine koja se koristi za iscrtavanje. Ažuriranje se obavlja u vrlo uskim trakama, čija je jedna dimenzija manja ili jednaka širini okvira, a druga dimenzija jednaka širini ili dužini nivoa klipmape (npr. 64×4096 ili 4096×64 teksela). Širina trake mora da bude celobrojni umnožak 4 teksela, zbog poravnanja sa veličinom blokova koji se koriste u kompresiji tekstura [13].

Ako je pomeraj posmatrača veći od praga ažuriranja po obe koordinate, vrši se dekompozicija ažuriranja po koordinatnim osama. Umesto jednog zahteva za ažuriranje površine L-oblika, šalju se dva nezavisna zahteva za ažuriranje pravougaonih oblasti (Z1 i Z2 na Sl. 6.2 (a)). Iako se na taj način zahteva i deo teksture koji će već sledećim ažuriranjem biti odbačen (oblast preklapanja dva zahteva), ovakav pristup ima brojne prednosti, kao što su:

- jednostavno formatiranje podataka (ažuriranja su pravougaonog oblika),
- podela ažuriranja u dva koraka, od kojih je svaki nezavisan i svojom samostalnom primenom ne narušava konzistentnost nivoa i
- mogućnost dalje podele ažuriranja na uže segmente, čime se omogućuje finija kontrola količine podataka koja se prenosi ka grafičkom procesoru.

Da bi se očuvala konzistentnost teksture nakon svakog ažuriranja, zahtevi su numerisani i moraju biti obrađeni redom. Po pristizanju prvog ažuriranja, A1 na Sl. 6.2 (b), centar teksture (T_c) se pomera po odgovarajućoj osi za širinu oblasti ažuriranja, a podaci upisuju na odgovarajuće mesto, po modulu veličine teksture. Obzirom da je sada pomeren logički početak teksture, drugi blok podataka za ažuriranje (A2 na Sl. 6.2 (c)) mora biti podeljen na dva segmenta (2.1 i 2.2).

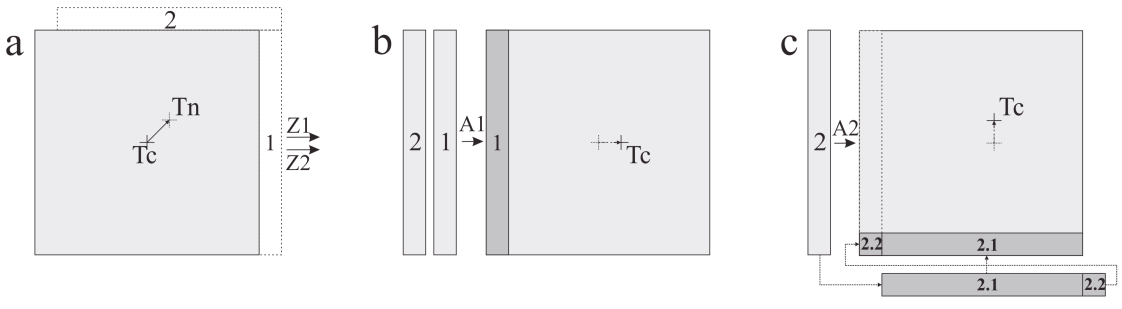
Svaki nivo klipmape ima dva centra:

- aktivni centar (T_c) i
- naredni centar (T_n)

Aktivni centar određuje logičko središte teksture datog nivoa klipmape. Sva izračunavanja, primene i ažuriranja se vrše u odnosu na aktivni centar. *Naredni centar* postaje središte teksture, tj. *aktivni centar*, nakon što pristignu sva ažuriranja za koja su poslati zahtevi. *Naredni centar* se koristi isključivo za definisanje zahteva za ažuriranje. Prilikom kretanja posmatrača, zahtevi se šalju kontinuirano, bez čekanja da se bilo koji od njih realizuje. Svaki poslati zahtev pomera *naredni centar*, kako bi se omogućilo pravilno formiranje sledećeg zahteva, dok svako primljeno ažuriranje pomera *aktivni centar*.

Što je kretanje posmatrača brže, to je širina polja koja se ažuriraju veća, a time i količina podataka koja se mora preneti do grafičkog procesora. Kako je nemoguće paralelno prenositi podatke i vršiti iscrtavanje, osim u posebnim slučajevima i za specifični hardver [106], količina prenetih podataka u jednom osvežavanju prikaza mora biti ograničena. To je razlog zbog koga je uvedena *kvota* u fazi *obrade ažuriranja na čekanju*, kao što je opisano u prethodnom poglavlju. Bez obzira kako je postavljena *kvota*, korak sa kojim je moguće vršiti ažuriranja direktno je određen veličinom oblasti koja se zahteva. Zbog toga se zahtevi dele na uže trake.

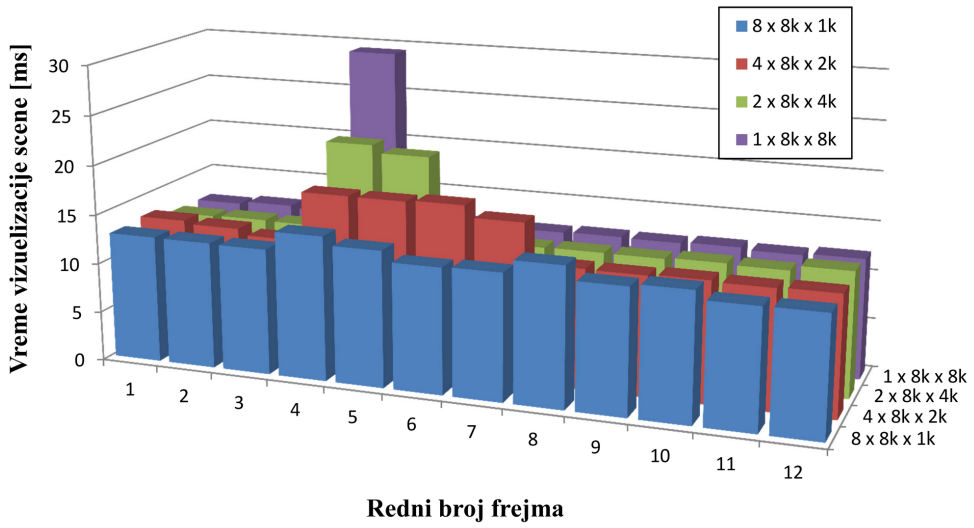
Ako je kretanje toliko brzo da svako sukcesivno ažuriranje zahteva promenu od preko 50% čitavog nivoa, umesto toroidnog inkrementalnog ažuriranja šalju se zahtevi za ažuriranje čitavog nivoa klipmape, smanjujući time količinu podataka koju treba preneti i pojednostavljujući samo ažuriranje. Kod zahteva za ažuriranje čitavog nivoa postavlja se i poseban indikator, kojim



Slika 6.2: EKM ažuriranje nivoa klipmape. Pomeraj posmatrača veći od praga ažuriranja po obe ose formira dva nezavisna zahteva – Z1 i Z2 (a). Po pristizanju prvog ažuriranja – A1, pomera se aktivni centar T_c, a podaci kopiraju na odgovarajuće mesto u teksturi (b). Zbog pomeranja logičkog početka teksture, drugo ažuriranje mora biti podeljeno na dva segmenta – 2.1 i 2.2 (c).

se iz odgovarajućeg reda čekanja uklanjaju prethodno poslati zahtevi za dati nivo. Uklanjanje zastarelih zahteva poboljšava performanse i sprečava pribavljane podataka koji su već zastareli. I kada se zahteva čitav nivo, zahtev se deli u više nezavisnih zahteva, kako bi se sprečio zastoj u iscrtavanju scene pri nailasku takvog ažuriranja, ali i da bi se uopšte omogućilo ažuriranje, obzirom da bi jedinstveni blok podataka premašio postavljenu kvotu. Na Sl. 6.3, prikazano je vreme vizuelizacije scene pri deljenju ažuriranja u više nezavisnih zahteva. Predstavljeni su sukcesivni intervali osvežavanja prikaza, na laptop-računaru Asus N550JK-CN019D, kada se ažurira jedan nivo klipmape veličine 8192×8192 teksela. Ako se ažuriranje ostvari prenosom čitavog nivoa teksture, nastaje zastoj u iscrtavanju ($1 \times 8k \times 8k$). Podelom ažuriranja u dva dela ($2 \times 8k \times 4k$) i smanjivanjem kvote, proces se deli na dva osvežavanja prikaza, ali još uvek stvara zastoj, jer svako pojedinačno osvežavanje traje dugo. Tek podelom ažuriranja nivoa na 8 delova ($8 \times 8k \times 1k$) i daljim smanjenjem kvote, iscrtavanje postaje tečno.

Na osnovu brzine kretanja i veličine nivoa, može se odrediti i maksimalna rezolucija za koju ima smisla vršiti ažuriranje. Tada se zabranjuje detaljnijim nivoima da šalju zahteve, sve dok brzina ne opadne. Ovim se samo smanjuje opterećenje sistema, ali se ne utiče na sam proces prikaza, jer klipmape inherentno obezbeđuju konzistentnost. Naime, zahtevi za ažuriranje se uvek obrađuju od nivoa sa nižom ka nivoima sa višom rezolucijom. Ukoliko sistem ne stigne da



Slika 6.3: Vreme vizuelizacije scene pri deljenju ažuriranja jednog celog nivoa klipmape, veličine 8192×8192 teksela, u više nezavisnih zahteva. Merenja su izvršena na laptop-računaru Asus N550JK-CN019D.

ažurira sve nivoe koji su poslali zahteve, detaljniji nivoi, koji još nisu ažurirani, biće isključeni iz prikaza, čime se ostvaruje isti efekat kao da oni nisu ni slali zahteve. Međutim, niti koje vrše obradu podataka u pozadini biće zaposlenije, pokušavajući da ažuriraju i detaljnije slojeve. Proces ažuriranja od manje detaljnih ka detaljnijim nivoima je sasvim prirodan, jer se prilikom kretanja velikom brzinom i u realnom svetu javlja efekat zamućenja pogleda.

6.2 Protočni sistem za pribavljanje i ažuriranje tekstura

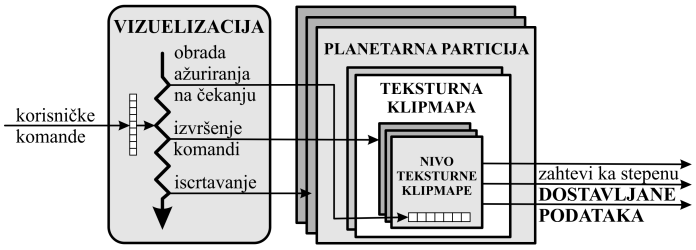
Većina algoritama za vizuelizaciju terena mora da rukuje ogromnom količinom podataka, koja značajno prevazilazi veličinu raspoložive operativne memorije. Takav način rada poznat je pod nazivom „izvršenje van jezgra" (eng. *out-of-core execution*) i zahteva primenu efikasnih metoda za pribavljanje podataka. Efikasno pribavljanje podataka obično kombinuje:

- keširanje i pribavljanje unapred,
- kompresiju, da bi se povećao protok i
- asinhrono čitanje sporih medijuma, skriveno drugim aktivnostima.

Moderni višejezgarni procesori ostvaruju značajno povećanje brzine izvršavanja poslova njihovom podelom i paralelizacijom kroz više niti, kao i iskorišćavanjem paralelizma u procesu pripreme podataka. Kada je u pitanju pribavljanje podataka, niti se često organizuju tako da svaka obavlja jedan korak u odgovarajućem softverskom protočnom sistemu. Višenitni protočni sistemi ne smanjuju kašnjenje, ali značajno povećavaju protok kada postoji stalni tok podataka [105]. Autori u [105] i [23] prepoznaju četiri stepena u protočnom sistemu za pribavljanje tekstura, svaki dodeljen odgovarajućoj niti: pribavljanje tekstura (ili pristup medijumu), transkodiranje, vizuelizacija i grafički drajver.

EKM uvodi dodatni stepen – *dostavljanje podataka*, koji se koristi da upravlja keširanim podacima i predstavlja posrednika, između stepena za *pristup medijumu* i *transkodera* sa jedne strane, i stepena za *vizuelizaciju* i *teksturnih klipmapa* sa druge (Sl. 6.5). Keš je organizovan kao *klipmapa podataka* koja pokriva veću prostornu oblast od odgovarajuće *teksturne klipmape*, a koja se ažurira blokovima podataka učitanim sa odgovarajućeg medijuma.

Stepen za *vizuelizaciju* je aplikacioni kraj protočnog sistema za pribavljanje i ažuriranje tekstura, koji komunikaciju sa grafičkim drajevrom obavlja kroz OpenGL funkcijske pozive. Na Sl. 6.4 prikazana je njegova interakcija sa *teksturnim klipmapama*, sadržanim u okviru *planetarne*

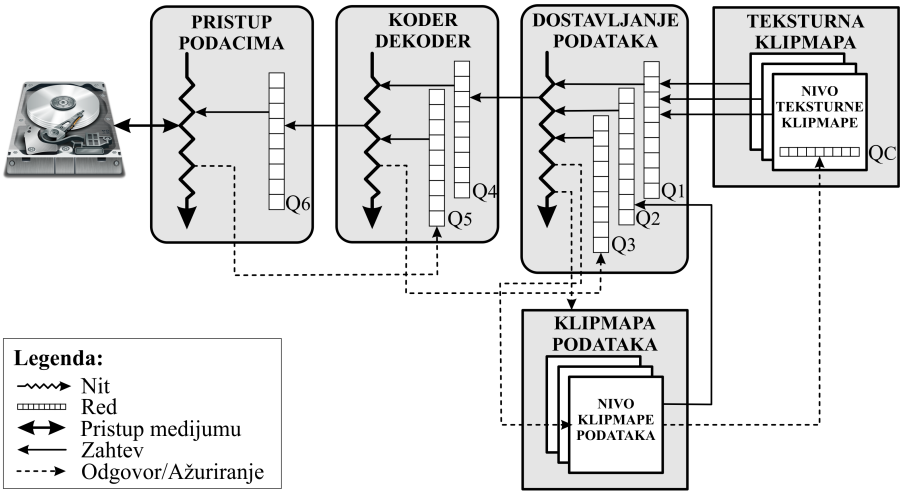


Slika 6.4: Aplikacioni kraj protočnog sistema za pribavljanje i ažuriranje tekstura – stepen za *vizuelizaciju* i *planetarne particije*.

particije. Pre same vizuelizacije scene, vrši se obilazak redova čekanja koji sadrže ažuriranja *teksturne klipmape*, pristigla u međuvremenu. Sve dok ukupna količina podataka ne pređe zadatu kvotu, ažuriranja se prosleđuju grafičkoj kartici. Svaki nivo *teksturne klipmape* ima dodeljeni red čekanja, u koji se smeštaju ažuriranja za dati nivo, čime se sprečavaju zastoji u izvršenju i efikasno upravlja protokom podataka.

Nakon obrade ažuriranja, stepen za *vizuelizaciju* interpretira komande, asinhrono upisane u registar komandi. Ukoliko komande pomeraju kameru, proverava se konzistentnost svih nivoa klipmapa i šalju odgovarajući zahtevi za ažuriranje. Sl. 6.5 prati put zahteva i tok podataka kroz protočni sistem za pribavljanje i ažuriranje tekstura.

Stepen za *dostavljanje podataka* prihvata zahteve za ažuriranje nivoa *teksturnih klipmapa*, kroz red sa prioritetom Q_1. Prioritet je baziran na veličini teksela, čime je omogućen redosled ažuriranja od grubljih ka detaljnijim nivoima. Ukoliko zahtevani podaci postoje u kešu ovog stepena, oni se pakuju u odgovarajući paket odgovora i upisuju u red Q_C, koji odgovara nivou



Slika 6.5: Jezgro EKM softverskog protočnog sistema za pribavljanje i ažuriranje tekstura.

klipmape koja ih je zahtevala. Ako zahtev ne može biti ispunjen, odgovarajući nivo *klipmape podataka* šalje svoj zahtev za ažuriranje u red Q_2, koji se zatim prosleđuje sledećem stepenu – *transkoderu*.

Transkoder je stepen koji vrši prekodiranje podataka u ovom protočnom sistemu. U slučaju zahteva, on ih samo prosleđuje iz svog ulaznog reda poruka Q_4 ka stepenu za *pristup podacima*. Stepen za *pristup podacima* obrađuje zahteve iz svog ulaznog reda poruka (Q_6), čita podatke sa medijuma i prosleđuje ih u red sa podacima Q_5, u okviru *transkodera*, na dalju obradu. Formati za skladištenje podataka i formati tekstura se obično ne podudaraju. Zadatak *transkodera* je da prilagodi ulazne podatke formatu pogodnom za teksture. Prekodirani blokovi podataka se šalju u red Q_3, u okviru stepena za *dostavljanje podataka*, odakle se kasnije koriste za ažuriranje odgovarajućeg nivoa *klipmape podataka*.

Osim stepena za *vizuelizaciju*, svi ostali stepeni su multiplicirani. Postoji posebna grana protočnog sistema za svaku *teksturnu klipmapu*. Obzirom da planeta ima tri globalne particije, svaka sa najmanje dve teksturne klipmape, protočni sistem za pribavljanje i ažuriranje tekstura ima ukupno šest grana. Međutim, sve tri particije nikada nisu vidljive istovremeno, što smanjuje broj aktivnih grana na četiri, ili čak samo na dve, ukoliko je posmatrač daleko od granice dveju particija.

Zahtevi i podaci enkapsulirani su u okviru poruke koja ima jedinstveni format u čitavom protočnom sistemu za pribavljanje i ažuriranje tekstura, a sadrži sledeća polja (u zagradama je data njihova veličina):

- PID – pokazivač na naziv izvorne datoteke (8 B),
- PPZ – pokazivač na onoga ko je zahtev poslao (8 B),
- OP1 – okvirni pravougaonik zahtevane oblasti u lat/lon koordinatama (32 B),
- OP2 – okvirni pravougaonik u celobrojnim koordinatama teksture (16 B),
- PI – polje indikatora (1 B),
- TIP – tip izvora podataka (1 B),
- UTP – ulazni tip podataka za transkoder (1 B),
- ITP – izlazni tip podataka iz transkodera (1 B),
- MID – identifikator poruke poslate u grupi (1 B),

- MMI – maksimalna vrednost identifikatora poruke poslate u grupi (1 B),
- PAZ – pravac ažuriranja (1 B),
- RBZ – redni broj zahteva, jedinstven na nivou onoga ko je zahtev poslao (4 B),
- PBP – pokazivač na blok podataka koji se vraća (8 B) i
- DBP – dužina bloka podataka koji se vraća (4 B).

Zahtev koji kreira *teksturna klipmapa* postavlja polja: PPZ, OP1, OP2, PI, PAZ i RBZ. Polje PPZ omogućuje dostavljanje podataka nivou klipmape koji je zahtev poslao, bez posredovanja ili prolaska kroz više nivoa. OP1 definiše prostornu veličinu zahtevane oblasti, a zajedno sa OP2 i nivo detalja, tj. razmeru koja se koristi kao izvor. Jedan od indikatora u PI služi za pražnjenje dela reda čekanja sa prioritetom, a koristi se pri brzom kretanju posmatrača, kada zahtevi datog nivoa klipmape (tj. prioriteta) koji još čekaju u redu nisu više relevantni. Prioritet, odnosno veličina teksela, nije sastavni deo poruke, već se kao dodatni parametar prosleđuje uz zahtev. Prioritet se može i kasnije izračunati na osnovu odnosa veličina OP1 i OP2, ali za to ne postoji potreba, obzirom da se prioritet rešava smeštanjem zahteva na odgovarajuću poziciju u redu Q_1, stepena za *dostavljanje podataka*. Polje PAZ olakšava ažuriranje određivanjem sa koje strane tekućeg okvirnog pravougaonika treba dodati pristigle podatke, dok RBZ predstavlja brojač poslatih zahteva.

Za razliku od *teksturne klipmape*, koja ne zna ništa o izvorima podataka i formatima, *klipmape podataka* imaju njihov kompletan opis. U zahtevu *klipmape podataka* postavljena su sva polja, osim izlaznih: PBP i DBP. PPZ je u ovim zahtevima adresa nivoa *klipmape podataka* koji je zahtev poslao. Ukoliko se radi o podacima podeljenim u sekcije (na šta ukazuje polje TIP), od kojih je svaka smeštena u posebnu datoteku, izračunava se naziv sekcije i postavlja polje PID. Druga vrlo bitna razlika *klipmape podataka* u odnosu na *teksturne klipmape* je ta što se ažuriranje često ne vrši u trakama koje su dužine čitavog nivoa, već u blokovima definisanim veličinom datoteka koje se učitavaju. Zbog toga *klipmape podataka* šalju grupe zahteva za ažuriranje. Grupu čini jedna vrsta ili kolona matrice blokova, ili čitav nivo. Nivo *klipmape podataka* nije validan sve dok ne pristignu ažuriranja za sve zahteve iz grupe. Zato se koriste polja MID (redni broj poruke u grupi) i MMI (maksimalni broj poruke u grupi). Tek nakon pristizanja ažuriranja čiji je MID jednak MMI, stigla je čitava grupa i nivo je validan. Ukoliko je izvor podataka ECW, MrSID ili neka druga datoteka (tip izvora je određen poljem TIP), koja ne

deli podatke na sekcije i nivoe detalja, polja MID i MMI se ne koriste, tj. imaju istu vrednost u svakom zahtevu. Obzirom da *klipmape podataka* poznaju i način skladištenja podataka, poljima UTP i ITP definišu zadatak *transkodera*.

Polja PBP i DBP se po prvi put postavljaju u stepenu za *pristup podacima*, a kasnije modifikuju u *transkoderu*, ukoliko se razlikuju polja UTP i ITP. U nastavku će biti prikazani izvorni formati za smeštanje podataka i formati tekstura.

6.3 Predstavljanje podataka

Vizuelizacija terena EKM algoritmom zahteva dva suštinski različita tipa klipmapa: geometrijske klipmape i teksturne-pokrivače. Iako su obe klipmape zapravo polja tekstura, njihova primena je različita, što utiče i na format podataka.

6.3.1 Format geometrijske klipmape

Geometrijska klipmapa je polje tekstura u čijim su tekselima smeštene visine terena, na osnovu kojih se formira geometrija blokova u verteks šejderu. Veličina nivoa zavisi od veličine blokova (u koracima rešetke) i broja blokova u matrici-blokova. Nivo je predstavljen 16-bitnom jednokanalnom 2D teksturom. Da bi se omogućilo filtriranje teksture prilikom uzorkovanja, podaci su smešteni u UNORM formatu [72]. UNORM je neoznačeni normalizovani ceo broj čija se vrednost tumači kao realni broj jednostruke tačnosti u opsegu [0.0, 1.0].

Da bi se efikasnije iskoristio 16-bitni prostor za smeštanje visina, mora se prvo razmotriti opseg vrednosti na koje se može naići na planeti koja se vizuelizuje. U slučaju planete Zemlje, najniža tačka na površini nalazi se na obali Mrtvog mora, sa trenutnom nadmorskom visinom od -429 m. Zbog povišene temperature i znatno smanjenog dotoka vode iz reke Jordana, nivo Mrtvog mora se sve više smanjuje. Od 50-tih godina prošlog veka do danas nivo se spustio za čitavih 40 m [48], sa tendencijom daljeg opadanja od oko 0.7 m godišnje [44]. Ovo treba uzeti u obzir prilikom razmatranja donje granice opsega vrednosti koje želimo predstaviti. Kako je najviši planinski vrh na svetu Mont Everest, sa visinom od 8850 m, ukupan opseg visina reljefa planete Zemlje ne prelazi 9300 m. Vrednosti visina u globalnim digitalnim modelima terena (GDEM) odstupaju od stvarnih vrednosti, pa je i to neophodno predvideti u određivanju opsega. U Tab. 6.1 dat je pregled ekstremnih vrednosti visina terena izmerene na površini Zemlje i odgovarajućih vrednosti u trenutno aktuelnim GDEM modelima.

Imajući sve prethodno u vidu, EKM koristi jednačinu (6.1) za prekodiranje visina terena, omogućujući rezoluciju od 14.3 cm pri smeštanju u 16-bitne teksture. Prekodiranje se može ostvariti u *transkoderu*, ali je to preporučljive uraditi u fazi pripreme podataka, ukoliko je moguće.

$$h_t' = (h_t + 500) \cdot 7 \tag{6.1}$$

U fazi pripreme GDEM podataka za vizuelizaciju primenom EKM algoritma, potrebno je:

- izvršiti reprojekciju podataka polarnih particija,
- podeliti podatke na datoteke veličine pogodne za učitavanje,
- prekodirati vrednosti i ispuniti praznine,
- formirati datoteke sa podacima različitih nivoa detalja i
- kompresovati sadržaj datoteka.

Veličina datoteka direktno utiče na veličinu i vreme ažuriranja *klipmapa podataka*. Neki izvori podataka, na primer SRTM3 i SRTM1, mogu sadržati i oblasti bez podataka (takozvane „praznine”). One se moraju ispuniti pre primene u vizuelizaciji. Kako postupak ispune najčešće nije trivijalan (sadrži interpolacione ili statističke metode za određivanje vrednosti koje nedostaju), poželjno je izvršiti ga u fazi pripreme podataka. Tada se podaci mogu i prekodirati, da bi se bolje iskoristio opseg vrednosti koji je na raspolaganju u procesu predstavljanja.

Korak uzorkovanja podataka za potrebe neke aplikacije ne mora se slagati sa korakom definisanim u izvornim podacima. U tom slučaju neophodno je reuzorkovati podatke sa željenom rezolucijom. Da bi se omogućilo direktno popunjavanje svih nivoa *klipmapa podataka*, treba formirati datoteke i sa nižim rezolucijama, uzorkovanjem svakog drugog podatka iz detaljnijeg nivoa (ili mešanjem četiri susedna uzorka), uz zadržavanje veličine datoteke. Smanjenje veličine

Tabela 6.1: Pregled ekstremnih vrednosti visina izmerenih na površini Zemlje i odgovarajućih vrednosti u trenutno aktuelnim GDEM modelima.

Lokacija	Izmerena visina [m]	CGIAR-CSI SRTM [m]	ASTER GDEM2 [m]
Mrtvo more	-429	-428	-459
Mont Everest	8850	8806	8776

datoteka i ubrzavanje učitavanja ostvaruje se kompresovanjem njihovog sadržaja. EKM koristi *miniz* biblioteku [42] za kompresiju izvornih podataka bez gubitaka. Dekompresija se obavlja u *transkoderu*, a vraćanje vrednosti u osnovni opseg u verteks šejderu.

6.3.2 Format teksturnog-pokrivača

Teksturni-pokrivač je polje tekstura koje sadrži satelitske ili avionske ortografske snimke površine planete. Obično ima više nivoa i mnogo veću rezoluciju od geometrijske klipmape. Zbog svoje veličine, ortografski snimci često koriste vrlo visok stepen kompresije prilikom skladištenja podataka. Mogu biti podeljeni u sekcije, od kojih je svaka zapamćena u posebnoj datoteci, ili organizovani u monolitnu datoteku sa više nivoa detalja, kao što to omogućuju ECW i MrSID formati. Koraci pretprocesiranja podataka su isti kao i za geometrijske klipmape, sa izuzetkom nepostojanja koraka prekodiranja vrednosti i ispune praznina, ali uz obaveznu reprojekciju podataka detaljnijih nivoa, bez obzira kojoj particiji pripadaju. Reprojekcija snimaka većih rezolucija je neophodna jer oni najčešće nisu u ekvidistantnoj cilindričnoj projekciji, već u nekoj od transverzalnih Merkatorovih projekcija (UTM).

Iako su JPEG, ECW i MrSID vrlo efikasni formati za skladištenje slika, oni nisu pogodni za kompresiju tekstura jer koriste promenljivu dužinu kodova i sprečavaju proizvoljan pristup i brzo dekodiranje dela slike [9]. Da bi se omogućio direktan pristup bilo kom delu teksture, algoritmi za kompresiju tekstura kodiraju blokove teksela uvek fiksnim brojem bitova. Dekompresija implementirana u hardveru, uz povećanu efikasnost keša zbog smanjenja veličine podataka, ima za posledicu efikasnije korišćenje kompresovanih u odnosu na nekopresovane teksture. Takođe, treba uzeti u obzir i značaj smanjenja količine podataka pri prenosu preko magistrale, tj. iz operativne u memoriju grafičke kartice. Imajući sve prethodno rečeno u vidu, može se zaključiti da kompresija tekstura smanjuje vreme ažuriranja uz istovremeno povećanje efikasnosti same vizuelizacije.

EKM koristi S3TC [13] za kompresiju teksturnog-pokrivača. Tačnije, koristi se DXT1 (tj. BC1) oblik ove kompresije. DXT1 kodira blokove 4×4 teksela kodom dužine 64 bita. Na početku koda su definisane dve osnovne boje (C_0 i C_1), svaka sa po 16 bitova u RGB565 formatu (5 bitova za crvenu komponentu, 6 za zelenu i 5 za plavu). U preostalom delu koda, svaki teksel kodiran je sa 2 bita. Značenje kodova pojedinačnih teksela, odnosno način njihovog dekodiranja, definisan je u Tab. 6.2.

DXT1 format podržava samo jednu transparentnu boju. Ukoliko je boja C_0 veća od boje

Tabela 6.2: Značenje kodova pojedinačnih teksela u bloku kodiranom DXT1 algoritmom. C_0 i C_1 su dve osnovne boje datog bloka u RGB565 formatu.

Kod	$C_0 > C_1$	$C_0 \leq C_1$
0	C_0	C_0
1	C_1	C_1
2	$(2 \cdot C_0 + C_1)/3$	$(C_0 + C_1)/2$
3	$(C_0 + 2 \cdot C_1)/3$	providna

C_1, posmatrana kao neoznačeni 16-bitni celi broj, tada je tekstura potpuno neprovidna i koriste se četiri boje za interpolaciju čitavog bloka (Tab. 6.2). U protivnom, neprovidni tekseli se dobijaju korišćenjem jedne od tri boje, dok je četvrti indeks rezervisan za providne teksele [73]. Za složeniju transparenciju potrebno je koristiti DXT3 ili DXT5 format. Kako ortografski snimci ne sadrže delimičnu transparenciju, a DXT1 ima duplo veći kompresioni faktor od DXT3 i DXT5, u nastavku će biti razmotrena samo DXT1 kompresija.

DXT1 ima fiksni faktor kompresije i on iznosi 6 (16 teksela u RGB formatu zamenjuju se kodom dužine 64 bita). Međutim, realno smanjenje zauzeća memorije grafičke kartice, u odnosu na nekopresovanu teksturu, je čak 8 puta. Razlog za to je nemogućnost drajvera da koriste nekompresovani RGB format direktno, već zbog poravnanja podataka interno smeštaju svaki teksel u lokaciju dužine 32 bita.

Iako je dekodiranje DXT1 formata vrlo jednostavno i brzo, proces kodiranja je kompleksan. Za datih 16 teksela u bloku treba odabrati dve boje, takve da se njihovim kombinovanjem, kao što je definisano u Tab. 6.2, što vernije predstave originalne boje teksela. Osnovni problem u DXT1 kompresiji je, dakle, nalaženje linije u prostoru boja koja će obezbediti najbolju aproksimaciju i odgovarajućih tačaka na toj liniji, koje predstavljaju osnovne boje bloka [104]. Sam proces kompresije tekstura može se ostvariti u različitim delovima protočnog sistema za pribavljanje i ažuriranje tekstura. Kompresiju može obaviti: OpenGL API, transkoder ili odgovarajući alat u fazi pripreme podataka.

Prepuštanje kompresije OpenGL-u je najjednostavniji, ali istovremeno i najlošiji pristup. Vreme izvršenja je neprihvatljivo dugo. Obzirom da prilikom kretanja posmatrača postoji neprekidni dotok novih podataka koje treba kompresovati, kao i da se taj proces odvija u glavnoj niti za vizuelizaciju, nemoguće je ostvariti tečan prikaz scene.

Prebacivanje procesa kompresije tekstura u posebnu nit, čime se rasterećuje nit za

vizuelizaciju scene, predstavlja mnogo bolji pristup. Prekodiranje podataka je zapravo glavni zadatak *transkodera*. Na raspolaganju su različiti algoritmi i biblioteke koji omogućuju DXT kompresiju u realnom vremenu, bilo na centralnom procesoru [104] ili na grafičkom procesoru [18]. Iako DXT kompresija predstavlja idealan problem za paralelizaciju i izvršenje na grafičkom procesoru, zbog potpune nezavisnosti blokova, u slučaju EKM to nije preporučljivo, jer se troše resursi potrebni za vizuelizaciju, uz značajna dodatna kašnjenja uslovljena promenom konteksta i sinhronizacijom OpenGL i CUDA API-ja. Za implementaciju kompresije na strani centralnog procesora na raspolaganju su i brojne gotove biblioteke otvorenog koda, kao što su: *stb_dxt* (u okviru *Stb* biblioteke [8]), *Squish* [14], *Crunch* [41], *DXFCompressor* [68] itd.

Uporedni prikaz vremena izvršenja i kvaliteta kompresije različitih jednonitnih implementacija DXT1 algoritama na centralnom procesoru dat je u Tab. 6.3. Kompresija je testirana na ortografskom snimku veličine 4096×4096 teksela, prikazanom na Sl. 6.6. Kolona *Metod* definiše korišćeni metod, T_{CPU} predstavlja vreme izvršenja na centralnom procesoru, *RMS* srednje-kvadratnu grešku (eng. *root mean square*), a *MAX* maksimalnu grešku. Greška se izračunava kao Dekartovo rastojanje u RGB prostoru boja, pri čemu je svaka komponenta boje kodirana vrednostima iz intervala [0, 255]. Metod je određen nazivom biblioteke, parametrima kompresije i korišćenim skupom instrukcija. Sve metode napisane su u programskom jeziku C, a one koje u svom nazivu sadrže MMX ili SSE koriste optimizovani asemblerski kod zasnovan na odgovarajućem skupu instrukcija. Ostale skraćenice u nazivima metoda za *Stb*, *Squish* i *Crunch* biblioteke predstavljaju stanje parametara kompresije i imaju sledeća značenja: NO – STB_DXT_NORMA, HQ – STB_DXT_HIGHQUAL, RF – squish::kColourRangeFit, CF – squish::kColourClusterFit, ICF – squish::kColourIterativeClusterFit, MU – squish::kColourMetricUniform, MP – squish::kColourMetricPerceptual, QSF – cCRNDXTQualitySuperFast, QF – cCRNDXTQualityFast, QN – cCRNDXTQualityNormal, QB – cCRNDXTQualityBetter, QU – cCRNDXTQualityUber, FP – cCRNCompFlagPerceptual=true. Jasno je uočljivo da jedino Van Vaverenovi (eng. *van Waveren*) metodi [104] omogućuju kompresiju u realnom vremenu. Vreme kompresije kod ostalih metoda zavisi od sadržaja teksture, a može biti i do četiri reda veličine duže u odnosu na najbrži metod. Zato je ostale biblioteke preporučljivo koristiti isključivo u fazi pripreme podataka. Treba napomenuti da *Squish*, iako je najčešće citirana i korišćena biblioteka u naučnim radovima i tehničkim dokumentacijama za poređenje sa drugim algoritmina, sadrži izvesne greške, koje su morale biti otklonjene da bi rezultati prikazani u Tab. 6.3 bili validni.

Svakako najefikasnije rešenje podrazumeva izvršenje DXT1 kompresije u fazi pripreme

Slika 6.6: Tekstura veličine 4096×4096 teksela, korišćena u proveri brzine i kvaliteta DXT1 kompresije primenom različitih biblioteka otvorenog koda.

podataka, odnosno čuvanje sekcija ortografskih snimaka u DXT1 formatu. Zauzeće prostora na odgovarajućem memorijskom medijumu je veće u odnosu na JPEG, ECW ili MrSID kompresiju, ali se na taj način ne troši procesorska snaga u toku izvršenja aplikacije (inače *transkoder* mora prvo da dekompresuje sliku iz izvornog formata, a zatim kompresuje u DXT1), a moguće je ostvariti i veći kvalitet slika, jer vreme potrebno za kompresiju nije kritično u ovoj fazi. Kvalitet DXT1 kompresije zavisi od primenjenog algoritma, odnosno načina izbora osnovnih boja blokova, a algoritmi sa kvalitetnijom interpolacijom obično zahtevaju mnogo više vremena za izvršenje, kao što se može videti u Tab. 6.3.

Klipmape podataka predstavljaju keš, smešten u operativnoj memoriji, koji se koristi za direktno prosleđivanje podataka klipmapama smeštenim u memoriji grafičkog podsistema. Zato se u slučaju teksturnog-pokrivača podaci u njima čuvaju u DXT1 formatu. Zbog velikih dimenzija primenjenih tekstura i velikog broja slojeva, one zauzimaju najveći deo operativne memorije alocirane od strane EKM aplikacije.

Osim polja tekstura, za potrebe teksturnog-pokrivača koristi se još jedna standardna 2D tekstura, koja sadrži satelitski snimak čitave planete Zemlje u ekvidistantnoj cilindričnoj projekciji. Ona služi za teksturisanje monolitnog elipsoida pri jako velikim rastojanjima posmatrača od površine planete, i takođe je kompresovana korišćenjem DXT1 algoritma.

Ovim poglavljem završen je opis EKM algoritma vizuelizacije. U nastavku sledi njegova evaluacija, koja započinje potvrđivanjem preciznosti implementacije elipsoidne rešetke u okviru sledećeg poglavlja.

Tabela 6.3: Poređenje vremena izvršenja i kvaliteta kompresije različitih jednonitnih implementacija DXT1 algoritama na centralnom procesoru. Test je izvršen na laptop-računaru Asus N550JK-CN019D. Za svaki od metoda prikazani su: vreme izvršenja na centralnom procesoru (T_{CPU}), srednje-kvadratna greška (RMS) i maksimalna greška (MAX) u RGB prostoru boja.

Metod	T_{CPU}[ms]	RMS	MAX
NVIDIA OpenGL	989.6	11.786	80.387
Stb NO	719.1	10.469	77.052
Stb HQ	818.0	10.304	76.085
Squish RF MU	1244.8	12.283	102.157
Squish RF MP	1247.7	13.021	114.359
Squish CF MU	291669.1	10.003	83.241
Squish CF MU SSE	20626.6	10.003	83.241
Squish CF MP	294735.1	10.137	81.639
Squish CF MP SSE	20596.4	10.137	81.639
Squish ICF MU	387663.5	10.001	82.626
Squish ICF MU SSE	25928.9	10.001	82.626
Squish ICF MP	382316.7	10.136	81.639
Squish ICF MP SSE	26009.0	10.136	81.639
Crunch QSF	10942.0	9.949	84.415
Crunch QSF FP	11015.3	10.158	91.673
Crunch QF	17178.0	9.906	83.940
Crunch QF FP	17143.4	10.130	78.803
Crunch QN	35241.9	9.862	81.425
Crunch QN FP	35914.3	10.017	81.639
Crunch QB	123192.5	9.841	83.767
Crunch QB FP	83593.3	9.995	81.639
Crunch QU	271714.2	9.839	82.626
Crunch QU FP	192534.4	9.989	81.639
vanWaveren	317.5	11.512	106.611
vanWaveren MMX	80.5	11.512	106.611
vanWaveren SSE	47.1	11.512	106.611

Poglavlje 7

Preciznost implementacije

Jedna od glavnih prednosti EKM algoritma je mogućnost vizuelizacije WGS84 referentnog elipsoida sa subpikselskom tačnošću, korišćenjem dvostruke preciznosti aritmetike centralnog procesora i jednostruke preciznosti grafičkog procesora. Pre procene tačnosti predloženog modela, moramo uvesti metriku i zavisnost greške u prostoru ekrana od greške u svetskom prostoru.

7.1 Predlog metrike

Za perspektivnu projekciju, greška u prostoru ekrana δ može biti izražena, korišćenjem jednačine (7.1), kao funkcija: greške u svetskom prostoru – ε, rastojanja u svetskom prostoru od posmatrača do datog temena – d, vidnog polja – Φ i rezolucije prozora – w.

$$\delta = \frac{\varepsilon \cdot w}{2 \cdot d \cdot \tan\left(\frac{\Phi}{2}\right)} [pix] \tag{7.1}$$

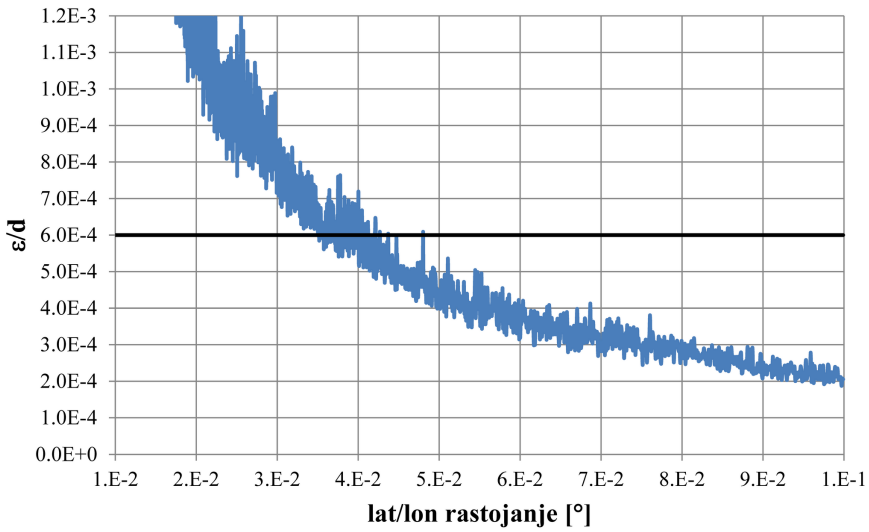
Vidno polje i rezolucija prozora zavise od pravca, u prostoru ekrana, u kome se mere. Međutim, jednačina (7.1) sadrži količnik $w/tan(\Phi/2)$ koji ne zavisi od pravca. Indeksi *hor*, *vert* i *diag*, u jednačini (7.2), odnose se na horizontalni, vertikalni i dijagonalni pravac u prostoru ekrana, respektivno.

$$\frac{w_{hor}}{\tan\left(\frac{\Phi_{hor}}{2}\right)} = \frac{w_{vert}}{\tan\left(\frac{\Phi_{vert}}{2}\right)} = \frac{w_{diag}}{\tan\left(\frac{\Phi_{diag}}{2}\right)} \tag{7.2}$$

Ako uvedemo t kao gornju granicu greške u prostoru ekrana, odnosno ako važi da je $t \geq \delta$, tada se zahtevana preciznost modela može izraziti kao:

$$\frac{\varepsilon}{d} \leq t \cdot \frac{2 \cdot \tan\left(\frac{\Phi}{2}\right)}{w} \tag{7.3}$$

Prema jednačini (7.3), količnik ε/d mora biti manji od 6E-4, da bi se ostvarila piksel-ska preciznost vizuelizacije za ekran visoke rezolucije (1080p HD) pri vidnom polju od 60° u horizontalnoj ravni. Preciznost je potvrđena kroz eksperimente vršene na različitim NVIDIA grafičkim procesorima (G80, GF100 i GM107). Vrednosti jednostruke tačnosti, izračunate na grafičkom procesoru i preuzete mehanizmom povratne sprege nakon transformacija (eng. *transform feedback*), upoređene su sa odgovarajućim vrednostima proširene dvostruke preci-znosti, izračunatim na centralnom procesoru [51]. Euklidova metrika se koristi za određivanje ε, kao razlika dve trodimenzionalne pozicije iste tačke korišćenjem različitih preciznosti, dok se d izračunava kao rastojanje tačnije pozicije od centra topocentričnog koordinatnog sistema. Pravo rastojanje do temena je zapravo veće od d (a greška manja od izračunate), obzirom da se posmatrač uvek nalazi iznad površine. Međutim, da bi izračunavanja bila nezavisna od visine posmatrača, u svim narednim formulama biće korišćeno topocentrično rastojanje.

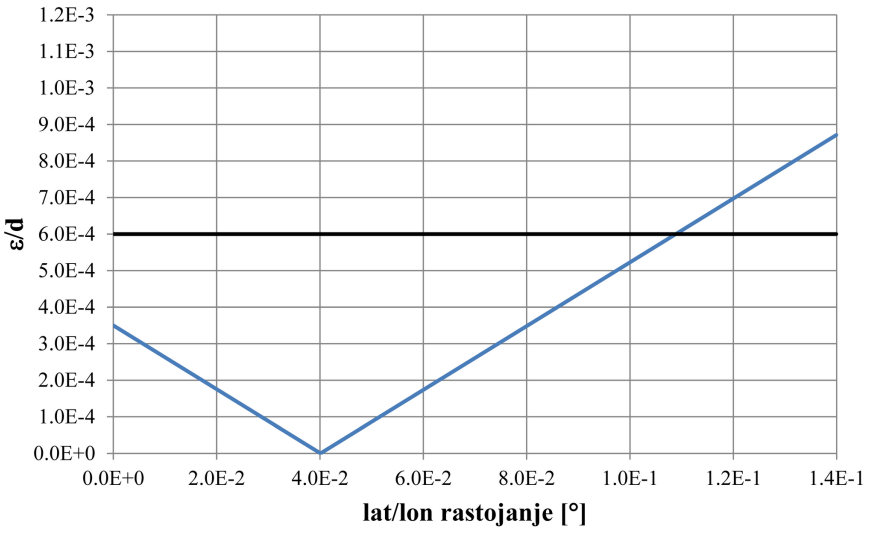


Slika 7.1: Maksimalne vrednosti količnika ε/d za globalno izračunavanje WGS84 elipsoida korišćenjem jednostruke preciznosti aritmetike grafičkog procesora.

Korišćenje jednostruke preciznosti za globalno izračunavanje elipsoida, kao što je definisano u jednačinama (3.5) do (3.13), zadovoljava zahtevanu tačnost samo za lat/lon rastojanja od topocentričnog koordinatnog početka koja su veća od izvesne vrednosti. Za WGS84 referentni elipsoid, to rastojanje je aproksimativno 5E-2°. Sl. 7.1 prikazuje maksimalne vrednosti količnika ε/d, na relativno malom rastojanju od topocentričnog koordinatnog početka, kada se on kreće po čitavoj globalnoj particiji. Očigledno je da je za mala rastojanja potrebna neka aproksimacija.

7.2 Linearna aproksimacija

Jedna od najjednostavnijih aproksimacija, koja je još uvek dovoljno precizna da ispuni postavljena ograničenja, jeste linearna aproksimacija. Ona se može ostvariti izračunavanjem priraštaja geocentričnih Dekartovih koordinata elipsoidne površine $P_{xyz}(\varphi_{part}, \theta_{part})$ duž lat/lon osa tekuće particije (jednačine (7.4) i (7.5)). Zbog bolje aproksimacije na većim rastojanjima, koriste se prednje konačne razlike $((F(x+\Delta x)-F(x))/\Delta x)$ umesto izvoda $(\partial F/\partial x)$. Rezultujući vektori, v_θ i v_φ, se zatim projektuju na ose topocentričnog koordinatnog sistema (X_t, Y_t i Z_t) korišćenjem skalarnih proizvoda (jednačine (7.6) i (7.7)).



Slika 7.2: ε/d za linearnu aproksimaciju WGS84 referentnog elipsoida duž ekvatora, pri korišćenju jednostruke preciznosti i $\Delta = 4\text{E-}2°$.

$$v_\theta = \frac{P_{xyz}(\varphi_{part}, \theta_{part}+\Delta) - P_{xyz}(\varphi_{part}, \theta_{part})}{\Delta} \tag{7.4}$$

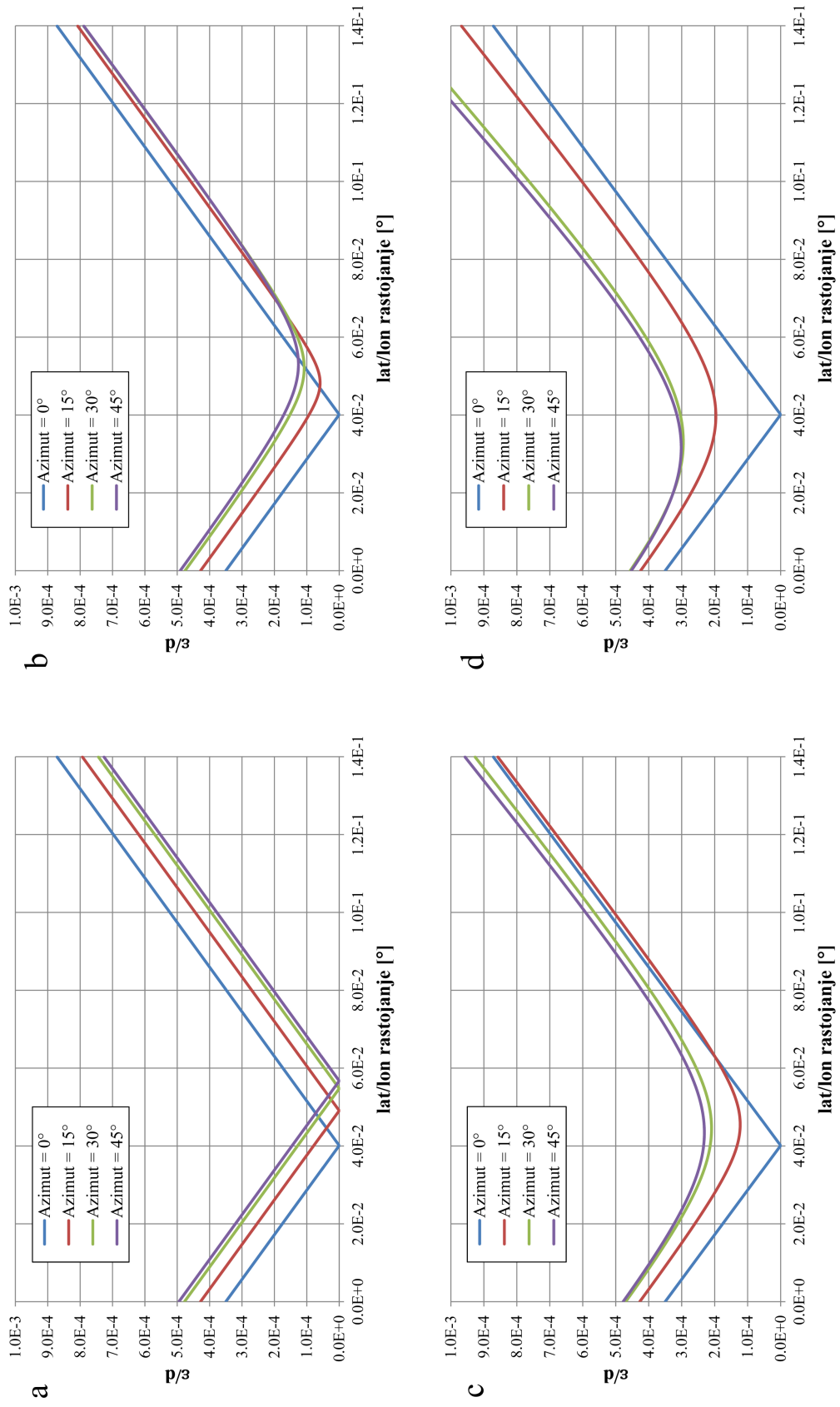
$$v_\varphi = \frac{P_{xyz}(\varphi_{part}+\Delta, \theta_{part}) - P_{xyz}(\varphi_{part}, \theta_{part})}{\Delta} \tag{7.5}$$

$$\tilde{v}_\theta = [v_\theta \cdot X_t, v_\theta \cdot Y_t, v_\theta \cdot Z_t]^{\mathrm{T}} \tag{7.6}$$

$$\tilde{v}_\varphi = \left[v_\varphi \cdot X_t,\ v_\varphi \cdot Y_t,\ v_\varphi \cdot Z_t\right]^{\mathrm{T}} \tag{7.7}$$

Vektori $\tilde{v}_\varphi$ i $\tilde{v}_\theta$ računaju se korišćenjem aritmetike proširene dvostruke preciznosti na centralnom procesoru i prosleđuju verteks šejderu, gde se pozicija temena aproksimira, na osnovu njegovog lat/lon rastojanja u koordinatnom sistemu date particije $(d\varphi, d\theta)$ od centra bloka/matrice-blokova, korišćenjem jednačine (7.8).

$$P = \tilde{v}_\varphi \cdot [d\varphi, d\varphi, |d\varphi|]^{\mathrm{T}} + \tilde{v}_\theta \cdot [d\theta, d\theta, |d\theta|]^{\mathrm{T}} \tag{7.8}$$



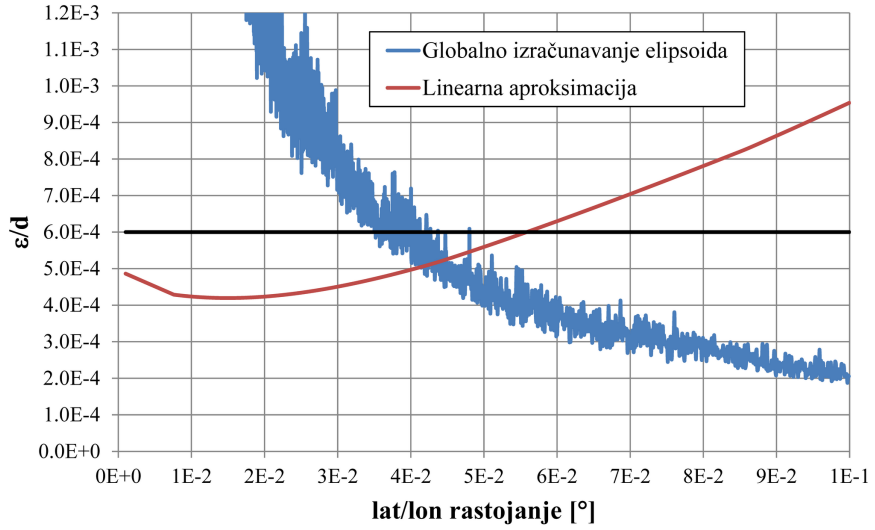
Slika 7.3: ε/d za linearnu aproksimaciju WGS84 referentnog elipsoida duž ekvatora, pri korišćenju jednostruke preciznosti i $\Delta = 4\text{E-}2°$ na različitim geografskim širinama: $\theta = 0°$ (a), $\theta = 15°$ (b), $\theta = 30°$ (c) i $\theta = 45°$ (d).

Preciznost aproksimacije zavisi od izabranog Δ. Veća vrednost rezultuje manjom preciznošću u topocentričnom koordinatnom početku, ali i većim rastojanjem do koga se aproksimacija precizno može primeniti. Greška linearne aproksimacije WGS84 referentnog elipsoida duž ekvatora, za $\Delta = 4\text{E-}2^\circ$, ilustrovana je na Sl. 7.2. Ta specifična vrednost za Δ je proizvoljno odabrana, dovoljno daleko da obezbedi primenu na rastojanjima na kojima globalno izračunavanje elipsoida nije precizno, ali takođe i dovoljno blizu da garantuje zahtevanu preciznost za infinitezimalna rastojanja oko koordinatnog početka.

Osim od rastojanja u lat/lon prostoru date particije, greška aproksimacije takođe zavisi od azimuta i latitude u lat/lon prostoru tekuće particije (Sl. 7.3). Kvalitet aproksimacije opada sa povećanjem latitude i približavanjem azimuta vrednosti $45^\circ \pm 90^\circ$.

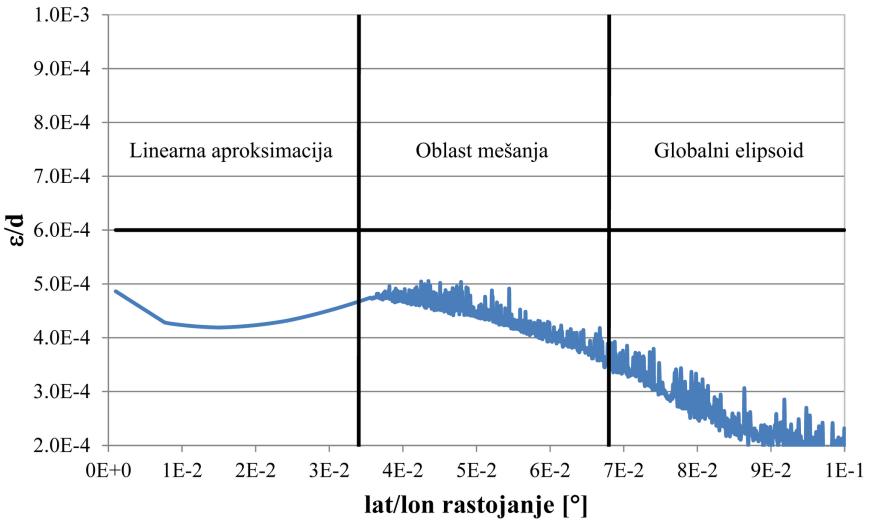
7.3 Kombinovanje metoda izračunavanja

Upoređivanjem envelopa maksimalnih vrednosti greške na celoj površini particije, za globalno izračunavanje elipsoida i lokalnu linearnu aproksimaciju, kao što je prikazano na Sl. 7.4, otkriva se da jedinstveno granično lat/lon rastojanje može biti odabrano za čitav WGS84 elipsoid.



Slika 7.4: Poređenje envelopa maksimalnih grešaka za globalno izračunavanje elipsoida i lokalnu linearnu aproksimaciju za $\Delta = 4\text{E-}2^\circ$.

Međutim, i pored postignute subpikselske tačnosti, ako se elipsoid iscrtava kao žičani model, bez mape visina, vertikalni pokreti posmatrača mogu otkriti dva različita načina razmeštanja temena. Ovo se potpuno eliminiše korišćenjem zone mešanja (Sl. 7.5).



Slika 7.5: Envelopa maksimalne greške pri vizuelizaciji korišćenjem kombinovanog metoda.

Osim visoke tačnosti, predloženi metod omogućuje konzistentnu vizuelizaciju WGS84 referentnog elipsoida do ekstremno malih rastojanja od površine. Izobličenja počinju da se javljaju tek pri rastojanjima manjim od 6E-20 m. U narednom poglavlju biće pokazano da je vizuelizacija korišćenjem EKM istovremeno i vrlo efikasna.

Poglavlje 8

Efikasnost implementacije

Iako EKM nameće značajno opterećenje grafičkom procesoru, kako bi se precizno izračunala pozicija temena u topocentričnom koordinatnom sitemu, implementirani algoritam postiže veliku brzinu vizuelizacije scene. Rezultati prikazani u ovom poglavlju dobijeni su na laptop-računaru Asus N550JK-CN019D, sa *Intel Core i7 4700HQ* centralnim procesorom i *NVIDIA GTX 850M* grafičkom karticom sa 2 GB DDR3 memorije, na kome je instaliran *Windows* 8.1 64-bitni operativni sistem. Vremena izvršenja na centralnom i grafičkom procesoru prikupljena su korišćenjem specijalizovane, u tu svrhu razvijene biblioteke za profilisanje grafičkih aplikazija – *GLProfiler*. U ovom poglavlju biće prikazani uslovi eksperimenata, struktura i način rada *GLProfiler* biblioteke, rezultati eksperimenata, kao i poređenje sa drugim trenutno aktuelnim i izuzetno efikasnim algoritmima za vizuelizaciju terena.

8.1 Uslovi eksperimenata

U eksperimentima je korišćeno nekoliko izvora podataka. Kao digitalni model terena korišćen je SRTM v4.1 [53], sa rezolucijom 3 lučne sekunde po uzorku, u WGS84 ekvidistantnoj cilindričnoj projekciji. Uzorci visina su skalirani i translirani (jednačina (6.1)), kako bi se bolje iskoristio 16-bitni memorijski prostor. Podaci su reprojektovani samo za polarne particije, a sve sekcije su kompresovane bez gubitaka, korišćenjem *miniz* biblioteke [42], da bi se smanjila veličina zauzetog prostora na disku i skratilo vreme učitavanja. Svi detalji vezani za pripremu i predstavljanje podataka dati su u poglavlju 6 – *Ažuriranje klipmapa*.

Teksturni-pokrivač objedinjuje podatke iz vrlo širokog skupa izvora. Za manje detaljne nivoe (15 lučnih sekundi po uzorku i grublje) korišćen je *Blue Marble Next Generation* skup

podataka [98]. Za nivoe srednje detaljnosti, između 0.5 i 15 lučnih sekundi po uzorku, korišćen je *GeoCover Orthorectified Landsat 7 Thematic Mapper Mosaics* [60]. Obzirom da su podaci iz ovog skupa u WGS84 univerzalnoj transverzalnoj Merkatorovoj projekciji (eng. *WGS84 Universal Transverse Mercator* – UTM) i dva od tri kanala nisu u vidljivom spektru, podaci su reprojektovani, a kolorit je promenjen kako bi se uklopio u čitav sistem. Najdetaljniji nivoi kreirani su na bazi različitih ortografskih avio-snimaka, sa preciznošću od 10 cm po uzorku ili većom, ali samo za ograničena područja.

Eksperimenti su vršeni sa 16 nivoa u teksturnom-pokrivaču i 8 nivoa u klipmapi sa visinama terena (tj. u geometrijskoj klipmapi). Da bi se postigla oštrina slike pri vizuelizaciji na ekranu 1080p HD rezolucije, uz primenu umekšavanja slike $4\times$ i anizotropnog filtriranja $16\times$, veličina nivoa klipmape teksturnog-pokrivača postavljena je na 5376×5376 teksela. Veličina klipmape visina postavljena je na 1024×1024 uzoraka (ili na 2048×2048 uzoraka, pri vizuelizaciji sa vrlo visokom gustinom rešetke).

8.2 Profilisanje grafičkih aplikacija

Paralelno sa razvojem algoritma za vizuelizaciju terena, razvijana je i biblioteka za profilisanje i otkrivanje grešaka u grafičkim aplikacijama, koje koriste OpenGL – *GLProfiler* [33].

Profilisanje (eng. *profiling*) je oblik dinamičke analize programa, koja omogućuje prikupljanje informacija o toku njegovog izvršenja. Prikupljene informacije mogu se iskoristiti za optimizaciju, poboljšanje performansi ili za proces evaluacije kvaliteta. Profilisanje se ostvaruje kroz merenje vremena izvršenja pojedinih programskih segmenata, merenje zauzeća resursa, prikupljanje statistike i otkrivanje vremenskih zavisnosti pojedinih događaja.

Postoje dva generalna pristupa u profilisanju: instrumentacija koda i presretanje funkcijskih poziva. Alati zasnovani na instrumentaciji koda poseduju svoj API, čiji se funkcijski pozivi umeću u program koji se analizira, u cilju signalizacije odgovarajućih događaja. Zbog uključivanja u druge programe, često se nazivaju i „ugrađeni alati za profilisanje". U ovu grupu spadaju: *NVIDIA PerfKit* [83], *AMD GPUPerfAPI* [4], *VSProfileLib* [89] i *OpenGL Timestamp Profiler* [40].

Alati zasnovani na presretanju funkcijskih poziva ne zahtevaju nikakve promene u aplikaciji koja se ispituje. Štaviše, obično nije ni potreban izvorni kod. Svi pozivi se presreću pre nego što stignu do odgovarajućeg API-ja, beleže se i meri se vreme njihovog izvršenja. Najveći

problem ovih alata je njihovo održavanje, obzirom da mora da prati dinamiku razvoja API-ja koji presreću. U proteklim godinama, razvoj OpenGL API-ja je bio toliko dinamičan, da su čak i najbolja rešenja kasnila za tekućom specifikacijom. Najpoznatiji predstavnici ove grupe alata su: *NVIDIA Nsight* [81], *AMD GPU PerfStudio* [3], *gDEBugger* [47] i *AMD gDEBugger* [1].

Svako merenje inherentno unosi grešku u izmerenu vrednost. U slučaju instrumentacije koda, dodatne instrukcije produžavaju vreme izvršenja aplikacije, a ponekad mogu i potpuno da zaustave protočni sistem, čineći očitavanja netačnim. Dodatni problem stvaraju sistemi koji dinamički menjaju stanje svojih performansi (tzv. *P-stanje*). Da bi izmerena vrednost mogla da se vrednuje, mora da se zna pod kakvim uslovima je dobijena.

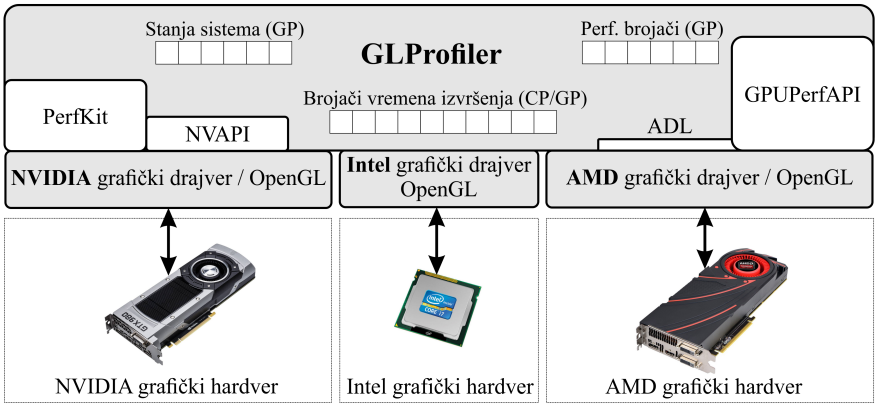
GLProfiler spada u grupu alata za profilisanje zasnovanih na instrumentaciji koda, a implementiran je u obliku biblioteke za *MS Windows* operativne sisteme, koja se statički ili dinamički povezuje sa programskim kodom koji se analizira. Obzirom da mora očitavati stanja grafičkog podsistema i vrednosti izabranih brojača, *GLProfiler* interno koristi druge biblioteke, implementirane od strane odgovarajućih proizvođača grafičkog hardvera, i to:

- NVAPI – omogućuje praćenje stanja performansi, kao i očitavanje i promenu postavki drajvera NVIDIA grafičkih kartica [82],
- NvPmApi – deo je NVIDIA PerfKit SDK paketa koji omogućuje očitavanje hardverskih brojača na NVIDIA grafičkim karticama i softverskih brojača u okviru drajvera [83],
- ADL – omogućuje pristup funkcionalnostima drajvera AMD Radeon i AMD FirePro grafičkih kartica [2] i
- GPUPerfAPI – omogućuje očitavanje hardverskih brojača na AMD Radeon grafičkim karticama i APU [4].

Globalna arhitektura *GLProfiler* biblioteke prikazana je na Sl. 8.1, a sama biblioteka omogućuje:

- merenje vremena izvršenja programskog segmenta na centralnom i grafičkom procesoru,
- praćenje stanja performansi u kome se nalazi grafička kartica (radnih frekvencija procesora i memorije),
- praćenje zauzeća resursa (grafičkog procesora i grafičke memorije),

- grafički prikaz svih izmerenih vrednosti u realnom vremenu, u polutransparentnim prozorima preko osnovnog prikaza aplikacije,
- pozivanje korisnički definisane funkcije u slučaju nastanka greške,
- beleženje svih izmerenih vrednosti i nastalih grešaka u posebnim datotekama za kasniju analizu, kao i
- niz pomoćnih funkcija vezanih za kreiranje OpenGL konteksta, isključivanje vertikalne sinhronizacije, itd.



Slika 8.1: Arhitektura *GLProfiler* biblioteke.

Prikupljanje informacija o vremenu izvršenja pojedinih programskih segmenata vrši se brojačima, organizovanim u polje u okviru GLProfiler biblioteke. Brojači očitavaju proteklo vreme i na centralnom (CP) i grafičkom procesoru (GP), ili samo na jednom od njih, zavisno od načina instanciranja. Za merenje vremena na centralnom procesoru koriste se *Windows* API funkcije *QueryPerformanceCounter* [69] i *QueryPerformanceFrequency* [70], dok se za grafički procesor koriste OpenGL vremenski upiti (eng. *timer query*) [24]. OpenGL vremenski upiti mogućuju tri metoda očitavanja vremena, i to:

1. sinhrono očitavanje vremena (u ns), proteklog od kreiranja OpenGL konteksta do trenutka kada je prethodno izdata komanda stigla do OpenGL servera, ali još nije izvršena,
2. asinhrono očitavanje vremena (u ns), proteklog od kreiranja OpenGL konteksta do trenutka kada su sve prethodno izdate komande izvršene u potpunosti i njihov rezultat je već smešten u bafer slike (eng. *frame-bafer*) i

3. asinhrono očitavanje vremena (u ns), potrebnog za izvršenje skupa komandi uokvirenih *glBeginQuery/glEndQuery* OpenGL funkcijskim pozivima.

GLProfiler koristi drugi metod (asinhrono očitavanje proteklog vremena od kreiranja OpenGL konteksta), jer dozvoljava neograničeno ugnježdavanje upita, za razliku od trećeg metoda, koji ne dozvoljava preklapanje merenih intervala. Vremenski upiti na grafičkom procesoru su mnogo složeniji od merenja vremena na centralnom procesoru i zahtevaju posebnu pažnju kako se ne bi zaustavio protočni sistem. Ukoliko u trenutku početka novog merenja prethodna vrednost nije dostupna (kašnjenje može biti i više perioda osvežavanja ekrana), GP brojači aktiviraju instance *podbrojača*. Podbrojači su u okviru GP brojača organizovani u dve lančane liste, od kojih jedna sadrži one koji još čekaju na očitavanje vrednosti, dok su u drugoj neaktivni (tj. „slobodni"). Ovakva organizacija minimizuje vreme na centralnom procesoru potrebno za započinjanje novog merenja na grafičkom procesoru. Implementacija zasnovana na listi slobodnih podbrojača omogućuje merenje vremenskih intervala i u okviru petlji, sa proizvoljnim brojem iteracija. Broj iteracija utiče jedino na maksimalni broj elemenata u listama.

Osim vremena izvršenja programskih segmenata, *GLProfiler* prati i zauzeće resursa. Zauzeće memorije grafičkog podsistema prati se korišćenjem specijalizovanih OpenGL ekstenzija za svakog od proizvođača hardvera [31]. Za očitavanje tekućeg stanja memorije na NVIDIA grafičkim karticama koristi se eksperimentalna *NVX gpu memory info* [99] ekstenzija. Njome se pribavljaju informacija o: količini ukupne posvećene (eng. *dedicated*) memorije, količini slobodne video memorije, broju oslobođenih (eng. *evicted*) blokova i njihovoj veličini. Detektovanje oslobađanja memorijskih blokova je izuzetno korisna informacija pri profilisanju, jer ukazuje na prepunjenost grafičke memorije i gubitak performansi zbog potrebe za realokacijom.

Praćenje tekućeg stanja memorije na AMD grafičkim karticama vrši se pomoću *ATI meminfo* [10] ekstenzije. Ona omogućuje očitavanje: količine ukupne posvećene grafičke memorije, količine ukupne deljene sistemske memorije, kao i veličine najvećih blokova u svakom od ova dva tipa memorije. AMD deli memoriju u tri segmenta: *vbo*, *texture* i *renderbuffer memory pool*. Ovi segmenti se mogu preklapati ili biti potpuno disjunktni. Tekuća implementacija u potpunosti preklapa sva tri segmenta, tako da se iste vrednosti dobijaju očitavanjem bilo kojeg od njih.

Smanjenje potrošnje energije i disipacije je vrlo značajan zadatak u projektovanju savremenih integrisanih kola. Osim optimizacija tokom projektovanja, svi proizvođači centralnih i grafičkih procesora implementiraju i različite dinamičke metode redukcije potrošnje, koje

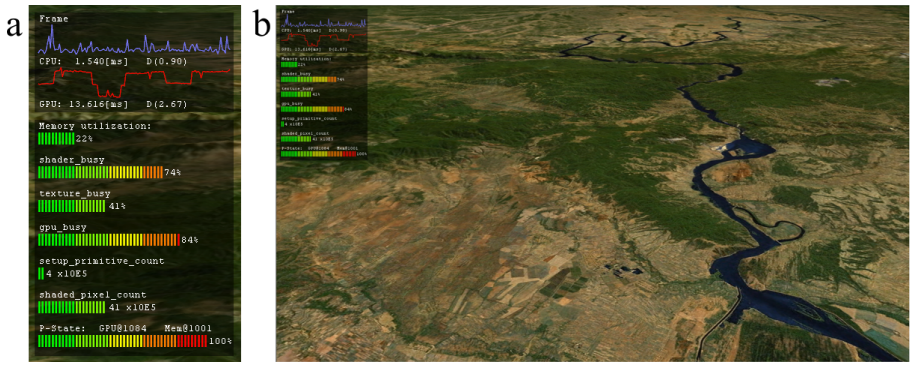
tokom izvršenja menjaju radne frekvencije ili potpuno isključuju pojedine procesne elemente. Određivanje vremena izvršenja programskih segmenata, bez znanja u kom stanju se nalazi odgovarajući procesor ili podsistem, može dovesti do pogrešnog tumačenja izmerenih rezultata [32]. Grafički procesori najnovije generacije čak više i nemaju fiksne frekvencije na kojima rade, već se frekvencija trenutno usklađuje sa njihovim opterećenjem, što dodatno povećava značaj praćenja njihovog stanja. Obzirom da ne postoji standardni način za očitavanje tekućeg stanja grafičke kartice, *GLProfiler* u tu svrhu koristi biblioteke odgovarajućih proizvođača, i to: NVAPI za NVIDIA i ADL za AMD grafičke kartice.

Dodatne informacije o radu grafičkih kartica mogu se dobiti pribavljanjem vrednosti hardverskih i softverskih brojača, implementiranih na samim uređajima i u drajverima, respektivno. Dok su softverski brojači vezani za odgovarajući grafički API (Direct3D ili OpenGL) i omogućavaju očitavanje karakterističnih vremena u izvršenju drajvera ili broj generisanih primitiva, hardverski brojači zavise od konkretnog grafičkog procesora (tačnije od proizvođača i generacije procesora) i pružaju uvid u zauzeće pojedinih funkcionalnih elemenata, broj izvršenih instrukcija i sl. Za pristup NVIDIA grafičkim karticama koristi se *NvPmApi* (tj. *PerfKit*), dok se za pristup AMD grafičkim karticama koristi *GPUPerfAPI* biblioteka.

Intel je kao proizvođač grafičkog hardvera daleko iza kompanija kao što su NVIDIA i ATI, sa daleko slabijom podrškom za očitavanje stanja ili upravljanje njihovim drajverima. Zbog toga, *GLProfiler* nema mogućnosti da prikaže više informacija o izvršenju grafičkih aplikacija na Intelovim procesorima. Postoji jedino podrška za utvrđivanje vremena izvršenja programskog segmenta, korišćenjem OpenGL vremenskih upita, ali samo za HD 2000 (*Core i3/i5/i7* procesori II generacije) ili noviji grafički hardver.

Sve izmerene vrednosti *GLProfiler* beleži u izlazne datoteke, prilikom gašenja aplikacije. Po jedna datoteka pridružena je svakom brojaču, a za svaki interval merenja beleže se vremena izvršenja na centralnom i grafičkom procesoru (ili samo jednog od njih, u zavisnosti od tipa brojača), stanje performansi, zauzetost grafičkog procesora i memorije, zauzetost deljene sistemske memorije (samo za AMD kartice), količina oslobođenje video memorije (samo za NVIDIA kartice) i celobrojna vrednost (*hint*) koja se kao parametar prosleđuje pri zaustavljanju brojača i identifikuje programske uslove pri kojima je izvršeno merenje. Osim toga, na nivou čitave datoteke, izračunava se i zapisuje statistika (minimalne, maksimalne i srednje vrednosti). Datoteke se formiraju prilikom gašenja aplikacije, kako pristup disku i dodatne operacije ne bi uticale na proces merenja.

Da bi stanje moglo da se prati i u toku rada aplikacije, *GLProfiler* prikazuje izmerene vremenske intervale i trenutne vredosti pojedinih parametara u realnom vremenu u okviru prozora aplikacije (Sl. 8.2). Trajanje vremenskih intervala prikazuje se linijskim grafikonima (gornji deo Sl. 8.2 (a)), pri čemu plava linija predstavlja centralni, a crvena grafički procesor. Broj uzastopno izmerenih vrednosti, prikazanih na grafikonu, može se podešavati, a u datom primeru iznosi 128. Trenutno izmerena vrednost ispisuje se i numerički, izražena u ms, kao i apsolutna vrednost razlike minimalne i maksimalne vrednosti (D) prikazane na grafikonu. Grafikon dinamički menja razmeru po Y-osi, u skladu sa vrednostima koje prikazuje. Iscrtavanje se vrši korišćenjem OpenGL-a bez prosleđivanja atributa. Grafikon se formira u verteks šejderu, na osnovu ugrađenog atributa *gl_VertexID* (slično kao kod terena), vrednosti upisanih u 1D teksturu i nekoliko uniformnih promenljivih.

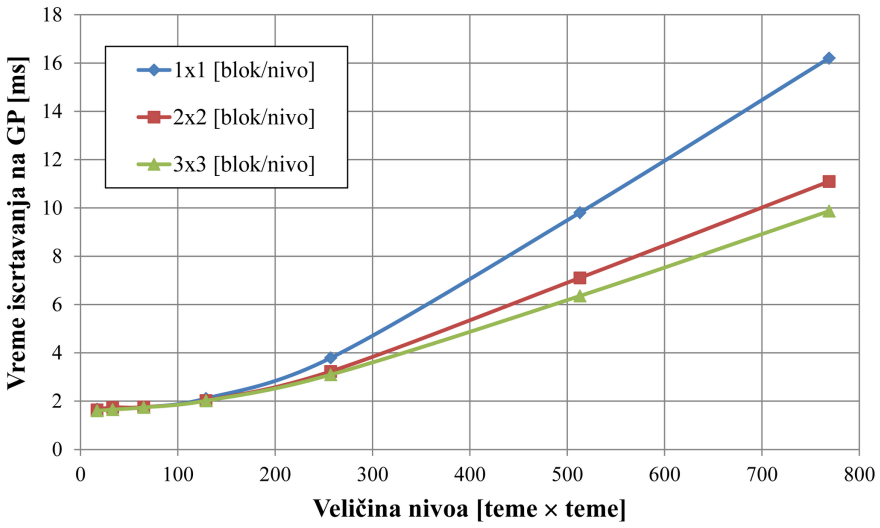


Slika 8.2: Primer izgleda grafičkog interfejsa *GLProfiler* biblioteke.

Ostali parametri prikazuju se horizontalnim linijama progresa i numeričkim vrednostima (donji deo Sl. 8.2 (a)). I ovaj prikaz implementiran je na sličan način kao i linijski grafikoni, bez prosleđivanja atributa, ali i bez korišćenja tekstura. *GLProfiler* automatski raspoređuje prikaz izmerenih vremenskih intervala i parametara sistema, sleva udesno i odozgo naniže. Na Sl. 8.2 (b) dat je tipičan prozor aplikacije sa uključenim grafičkim prikazom *GLProfilera* i jednim aktiviranim brojačem, čije je ime *Frame*. Brojačima se mogu dodeljivati proizvoljna imena, koja služe za njihovo razlikovanje tokom prikaza i definisanje naziva datoteke u koju brojač upisuje vrednosti pri gašenju aplikacije. Ostali detalji vezani za implementaciju *GLProfiler* biblioteke mogu se naći u [33].

8.3 Rezultati eksperimenata

Da bi se ostvarila procena performansi EKM algoritma, izvršen je veliki broj testova. U nastavku će biti prikazani samo najkarakterističniji. Prvi od njih ilustruje vreme iscrtavanja statičnih scena, tj. scena u kojima se posmatrač ne pomera, već samo rotira svoj pogled. Sl. 8.3 prikazuje vreme iscrtavanja na grafičkom procesoru u zavisnosti od veličine bloka/matrice-blokova. Pri iscrtavanju, kamera se nalazi na 10000 m visine i 43.3° severne geografske širine. Kamera rotira oko vertikalne ose i nagnuta je za -30°, tako da se horizont poklapa sa vrhom ekrana. Ovaj eksperiment eliminiše ažuriranja klipmapa, ali maksimalizuje opterećenje grafičkog procesora, zahtevajući da čitav HD ekran bude ispunjen pikselima koji potiču od terena. Takođe, maksimalizovan je i broj vidljivih nivoa klipmapa.



Slika 8.3: Vreme iscrtavanja scene za različite veličine blokova i matrica-blokova.

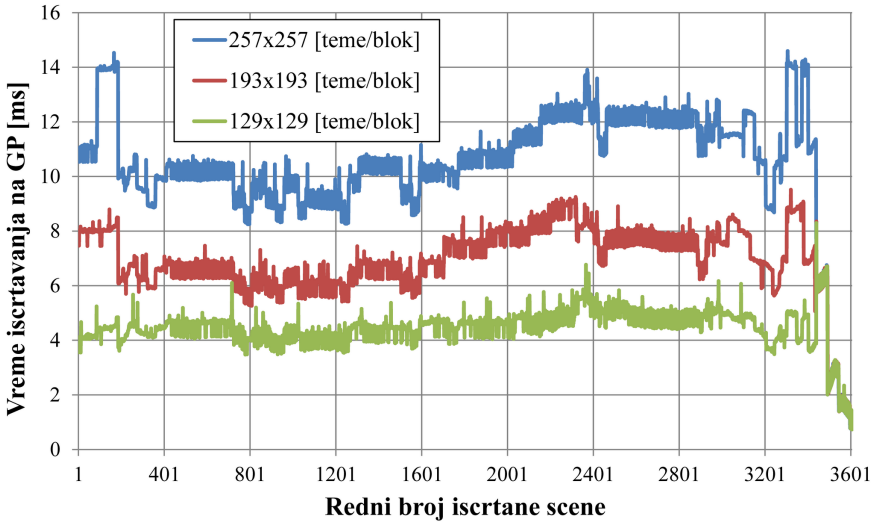
Prema metrici predloženoj u [64], prikaz na 1080p HD ekranu, pri vidnom polju od 60° u horizontalnoj ravni, zahteva 831^2 temena po nivou da bi se ostvarila aproksimativna veličina trouglova u prostoru ekrana od 5 piksela. Bez odbacivanja blokova koji nisu u vidnom polju (eng. *view frustum culling*) i uz korišćenje samo jednog bloka po nivou, teško je ostvariti prethodno postavljeni uslov na hardveru koji je korišćen u testu, uz interaktivno osvežavanje ekrana. Međutim, više performanse mogu se ostvariti korišćenjem više blokova za predstavljanje nivoa (tj. korišćenjem matrice-blokova), uz odbacivanje blokova koji nisu u vidnom polju. Odbacivanje blokova van vidnog polja ostvaruje se na centralnom procesoru, na osnovu njihove pozicije u okviru matrice-blokova i orijentacije kamere (azimut i nagib). Vreme izvršenja na grafičkom procesoru se značajno smanjuje ako se koriste matrice veličine 2×2 bloka, za svaki

od nivoa. Dobitak je veći što je veća veličina bloka, tj. broj temena u bloku. Korišćenje 3×3 matrice-blokova obezbeđuje još malo bolje rezultate, kao što se može videti na Sl. 8.3. Dalje povećanje broja blokova u okviru nivoa nameće mnogo složeniji algoritam za odbacivanje, dok su dobici u brzini zanemarljivi. Zbog toga, u svim narednim eksperimentima korišćene su matrice blokova veličine 3×3.

Obzirom da EKM iscrtava svaku particiju planete odvojeno, zavisno od njene vidljivosti, brzina iscrtavanja se blago menja sa geografskom širinom posmatača. Podrazumevajući iste uslove i manje visine posmatrača, vreme iscrtavanja na grafičkom procesoru je oko 18% duže na 45° geografske širine nego na Ekvatoru.

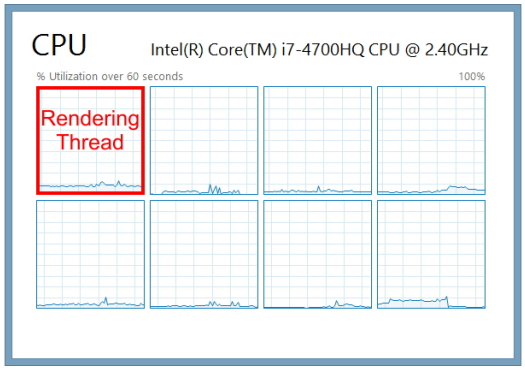
Korišćenje ortografskih avionskih i satelitskih snimaka za teksturisanje podrazumeva da su senke već sadržane u teksturi, obzirom da se radi o pravim fotografijama terena. Međutim, ukoliko su ipak potrebni dodatni efekti osvetljenja, moraju se izračunati normale, kao što je opisano u poglavlju 4 – *Formiranje terena na grafičkom procesoru*, što izaziva pad performansi. Na *Fermi* grafičkim procesorima [78, 45], povećanje vremena izvršenja veteks šejdera je oko 48% za nivoe sa $3\times3\times257^2$ temena, dok je na *Maxwell* grafičkim procesorima [80] to povećanje oko 21%. Očigledno je da je nova arhitektura mnogo otpornija na dodatna čitanja iz tekstura u okviru verteks šejdera.

Sl. 8.4 prikazuje rezultate eksperimenta koji testira sve aspekte implementacije kroz slobodan let preko terena. Tokom leta, posmatrač je na različitim visinama, od 200 m iznad površine, pa sve do 18500 km (kada je čitava planeta u vidnom polju).



Slika 8.4: Vreme iscrtavanja scene pri preletu, korišćenjem tri različite veličine blokova.

Zauzeće centralnog procesora je veoma malo. U prethodnom eksperimentu, nit za iscrtavanje koristi tek oko 10% procesorskog vremena jednog jezgra (1.5 ms po osvežavanju scene), dok ukupno zauzeće centralng procesora nikada ne prelazi 6% (Sl. 8.5). Čak iako nit za iscrtavanje ažurira teksture sinhrono, bez korišćenja PBO (eng. *Pixel Buffer Object*) [57] ili dualnog mehanizma za kopiranje [106], zauzeće je ekstremno nisko, što daje prostora za implementaciju naprednijih i procesorski zahtevnijih algoritama. Sl. 8.5 otkriva i da ostale niti, koje su deo softverskog protočnog sistema za pribavljanje i ažuriranje tekstura, takođe veoma malo angažuju centralni procesor. Malo zauzeće centralnog procesora je jedna od najjačih strana predloženog algoritma.

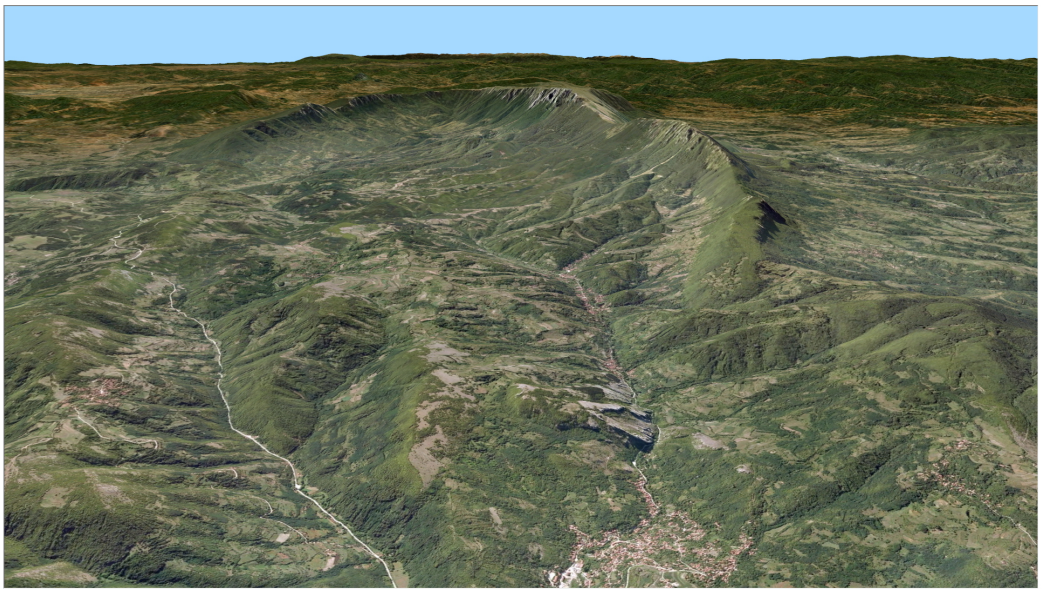


Slika 8.5: Zauzetost centralnog procesora tokom testa preleta, prikazana korišćenjem WTM 8.1 (eng. *Windows 8.1 Task Manager*).

Vrlo efikasno pribavljanje i ažuriranje tekstura čini EKM otpornim na opadanje frekvencije osvežavanja scene, čak i pri korišćenju tekstura vrlo visokih rezolucija za teksturni-pokrivač. Isto vreme osvežavanja je zabeleženo za širok opseg različitih rezolucija teksturnog-pokrivača, od 1024×1024 do 8192×8192 teksela po nivou klipmape. Glavni ograničavajući faktor je raspoloživa grafička memorija. Polje tekstura sa šesnaest 5376×537 DXT1 kompresovanih nivoa zauzima oko 220.5 MB.

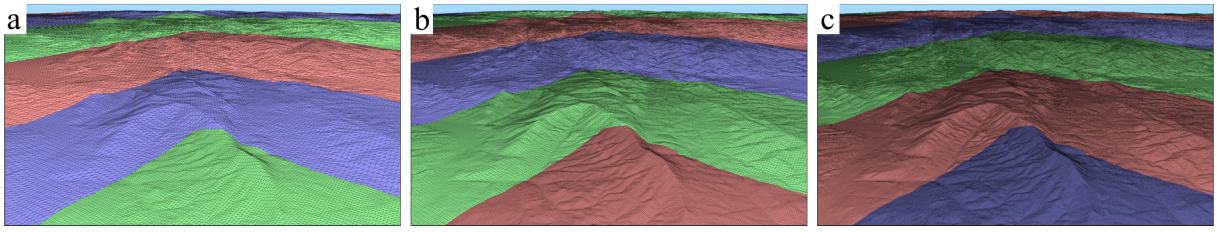
Sl. 8.6 ilustruje oštrinu vizuelizacije scene pri korišćenju tekstura rezolucije 5376×5376 teksela za teksturni pokrivač. Ovako visoka rezolucija je preporučljiva za prikaz na 1080p HD ekranima, obzirom da klasično trilinearno filtriranje zahteva primenu tekstura dva puta većih od rezolucije prozora u kome se prikazuju, dok se dodatni faktor množenja od 1.4 koristi za kompenzaciju distorzije aspekta teksela (vidi Sl. 3.4). Otpornost na korišćenje teksturnog-pokrivača vrlo visoke rezolucije je druga prednost predloženog algoritma.

Treća prednost EKM je stabilna frekvencija osvežavanja scene. Kao i svi drugi pristupi zasnovani na klipmapama, vreme iscrtavanja ne zavisi od neravnina terena, već od broja blokova



Slika 8.6: Scena iscrtana korišćenjem tekstura visoke rezolucije.

i njihove rezolucije. Sl. 8.7 prikazuje uniformnu teselaciju karakterističnu za klipmape, sa svakim nivoom detalja obojenim drugom bojom. Aproksimativne veličine trouglova u prostoru ekrana za teren prikazan na slici, a prema metrici predloženoj u [64], su: 21.5 (a), 10.8 (b) i 5.4 piksela (c). Korišćenje uniformne rešetke daje slabu kontrolu teselacije, tako da gotovo vertikalne površine mogu proizvesti vrlo velike trouglove, a jako oštri oblici reljefa se relativno kasno pojavljuju na sceni (tek kada gustina mreže odgovarajućeg nivoa detalja bude dovoljna da ih prikaže). Međutim, globalni digitalni modeli terena su obično definisani upravo korišćenjem uniformne rešetke, što savršeno odgovara klipmapama. Ekstremni oblici terena su vrlo retka pojava, i kao takvi bi trebalo da budu predstavljeni zasebnim 3D objektima.



Slika 8.7: Vizuelizacija zasnovana na klipmapama koristi uniformnu teselaciju, nezavisnu od neravnina na terenu. Scena je iscrtana korišćenjem 3×3 matrice-blokova sa: 65×65 (a), 129×129 (b) i 257×257 (c) temena po bloku.

8.4 Poređenje sa drugim algoritmina i implementacijama

Vrlo je teško porediti različite algoritme vizuelizacije, posebno kada su rezultati dobijeni na različitim hardverskim konfiguracijama, pod različitim uslovima, korišćenjem različitih skupova podataka i kada su rezultati raspoloživi samo preko objavljenih članaka. Ipak, u nastavku sledi poređenje sa GAH (eng. *GPU-Aware Hybrid*) [26] i *Outerra* [55] algoritmima za vizuelizaciju terena. Dok je GAH bio naučni projekat, sa ciljem da unapredi performanse vizuelizacije terena na ekranima visoke rezolucije, *Outerra* je okvir koji se stalno unapređuje i ima demo verziju (*Outerra Anteworld*) dostupnu za besplatno preuzimanje i instaliranje. Takođe, mora se napomenuti da nigde nije objavljeno da je GAH ikada bio korišćen za vizuelizaciju terena planetarnih razmera. Bez obzira na to, biće smatran adekvatnim konkurentom visokih performansi. *Outerra*, sa druge strane, koristi sfernu aproksimaciju Zemlje. Postoje i drugi algoritmi, slični EKM po mnogim aspektima, kao što je ECM (eng. *Ellipsoidal Cube Maps*) [59]. Ali, na žalost, ECM je fokusiran na preciznost, a ne na performanse iscrtavanja, pa zbog toga ne predstavlja validnog konkurenta za poređenje.

Razmatrajući vreme iscrtavanja scene na 1080p HD ekranu, GAH postiže od 60 do preko 160 osvežavanja u sekundi (od 16.7 ms manje od 6 ms) na NVIDIA GTX 480 grafičkoj kartici. EKM ima daleko stabilniju frekvenciju osvežavanja. Na nešto slabijoj grafičkoj kartici (NVIDIA GTX 470), EKM postiže približno isto vreme osvežavanja (8 ms) sa 8 nivoa detalja u klipmapi visina sa po 875^2 temena u svakom nivou. Sa datom gustinom rešetke, EKM ima veću maksimalnu grešku u prostoru ekrana u odnosu na GAH, ali tokom testiranja nisu primećena vidna pomeranja pojedinih temena na terenu.

Outerra se značajno oslanja na proceduralno generisane detalje, pa je poređenje ostvareno na većim visinama. Uzimajući u obzir samo *Outerra wc:global* brojač (vreme provedeno samo u iscrtavanju globalnog terena), brzina iscrtavanja scene je oko 78 MΔ/s (3.08 ms za scenu sa 241 KΔ) na Asus N550JK laptopu, mada je ukupno vreme osvežavanja scene mnogo veće (12.9 ms). EKM prikazuje oko 54 MΔ/s (10.8 ms za scenu sa 580 KΔ), ali je ova vrednost izračunata deljenjem ukupnog vremena osvežavanja scene sa brojem rasterizovanih primitiva, na osnovu vrednosti koju vraća NVIDIA PerfKit API. Broj odbačenih primitiva je mnogo veći.

Razmatrajući alokaciju memorije na strani grafičkog procesora za podatke o visinama terena, GAH zavisi od skupa podataka i tekuće scene. Za *Vorarlberg* skup podataka, prosečna alokacija memorije je 128 MB, dok je za skup podataka *Utah* oko 32 MB. EKM ima konstantno zauzeće memorije i zahteva 2 MB za 1024×1024 16-bitni nivo mape visina (preciznost smeštenih

visina je oko 14 cm). Dakle, čitav skup podataka *Utah*, koji je korišćen za testiranje GAH algoritma, može biti smešten u 8 nivoa zauzimajući 16 MB (oblast od 460×600 km uzorkovana na 5 m). Pored većeg zauzeća memorije, GAH nameće i promenljivi transfer između centralnog i grafičkog procesora, zavisan od podataka.

Outerra Anteworld nije pogodan za poređenje zauzeća memorije, obzirom da je većina efekata generisana proceduralno, bez pokrivača visoke rezolucije generisanih na osnovu realnog terena.

Zaključak

U ovoj disertaciji predstavljen je originalni algoritam za trodimenzionalnu vizuelizaciju terena planetarnih razmera i kompletna softverska arhitektura koja omogućuje njegovo efikasno izvršenje. Obzirom da je implementacija bazirana na korišćenju klipmapa, na samom početku dat je kratak istorijski osvrt na razvoj algoritama za vizuelizaciju terena koji koriste klipmape, uz ukazivanje na nedostatke koji su uslovili dalji razvoj i pojavu novih algoritama. Elipsoidne klipmape, prikazane u ovoj disertaciji, predstavljaju logičan nastavak tog razvoja i adaptaciju osnovnog koncepta za primenu na čitave elipsoidne planete.

Jedna od premisa, koje su upravljale razvojem predloženog algoritma, je geodetski tačan i vrlo precizan prikaz planete Zemlje. Zbog toga je u nastavku objašnjen postupak modeliranja planete Zemlje, projektovanja njene površine na ravan u cilju organizacije podataka, a prikazani su i neki od najznačajnijih besplatnih izvora podataka.

Sama implementacija algoritma podrazumeva podelu planete na particije, od kojih svaka ima nezavisan skup podataka, na osnovu koga se vrši njena vizuelizacija. U disertaciji je prikazana podela planete na tri particije (ekvatorijalnu i dve polarne), mada algoritam nije ograničen samo na takvu podelu. Troparticiona podela je izabrana jer omogućuje jednostavno uključivanje podataka i smanjuje složenost njihove pripreme. Naime, preko 70% celokupne površine planete čini ekvatorijalnu particiju, koja za vizuelizaciju koristi podatke u WGS84 ekvidistantnoj cilindričnoj projekciji. Kako su svi podaci o reljefu (visine terena) kao i satelitski snimci sitnijih razmera dostupni upravo u ovoj projekciji, nije potrebna njihova reprojekcija i posebna priprema. Lako uključivanje novih podataka bila je još jedna premisa u projektovanju predloženog algoritma. Polarne particije zahtevaju reprojekciju, ali je ona prilično jednostavna i svodi se na rotaciju. Polarne particije, dakle, takođe koriste ekvidistantnu cilindričnu projekciju, ali je rešetka zarotirana za 90°. Ovakva podela na particije zahteva nešto složenije odsecanje ivica i spajanje particija, ali omogućuje očuvanje niske-do-srednje vrednosti distorzije primenjenih tekstura.

Proces vizuelizacije distribuiran je između centralnog i grafičkog procesora. Centralni procesor upravlja iscrtavanjem, pribavlja i prosleđuje podatke i vrši sva izračunavanja koja zahtevaju visoku preciznost, dok grafički procesor kreira geometriju terena i prekriva ga odgovarajućom teksturom na osnovu dostavljenih podataka. Za razliku od prethodnih algoritama zasnovanih na klipmapama, osnovna rešeka nad kojom se superponira reljef se generiše u letu, u verteks šejderu. Ova rešetka odgovara površini WGS84 elipsoida i subpikselski je precizna, bez obzira na primenu aritmetike jednostruke tačnosti na grafičkom procesoru.

Osim preciznosti i geodetske tačnosti, predloženo rešenje odlikuje se i vrlo dobrim performansama, uporedivim sa trenutno vrhunskim algoritmima. Zasluga za to pripada pažljivo organizovanoj softverskoj arhitekturi, a pre svega protočnom sistemu za pribavljanje i ažuriranje tekstura. Ovaj podsistem pruža mogućnost primene tekstura visoke rezolucije bez uticaja na performanse čitavog sistema. Dodatna prednost predložene arhitekture je i vrlo malo zauzeće centralnog procesora, čime se otvara mogućnost implemetacije mnogo zahtevnijih operacija.

Za potrebe praćenja performansi implementiranog algoritma i profilisanje čitave aplikacije, razvijena je posebna biblioteka. Ona je, pre svega, korišćena za prikupljanje podataka o vremenu izvršenja pojedinih programskih segmenata na centralnom i grafičkom procesoru, ali i za praćenje zauzeća grafičke memorije, aktivnosti pojedinih stepena u protočnom sistemu, broja generisanih primitiva itd. Razvijena biblioteka predstavlja potpuno nezavisnu celinu i može se koristiti za profilisanje bilo koje grafičke aplikacije koja koristi OpenGL API.

Doprinosi ove doktorske disertacije su brojni. U njoj je, pre svega, dat predlog novog algoritma i odgovarajuće softverske arhitekture za trodimenzionalnu vizuelizaciju terena planetarnih razmera. Vizuelizacija planete zasnovana je na geodetski precizno definisanom referentnom elipsoidu. Zatim, predložena je podela površine planete na particije i njihova kartografska projekcija koja omogućuje lako uključivanje novih podataka i održava nizak-do-srednji nivo distorzije primenjenih tekstura. Prikazan je metod za generisanje osnovne elipsoidne rešetke u letu, korišćenjem aritmetike jednostruke preciznosti na grafičkom procesoru. Pokazano je da dati metod ima subpikselsku tačnost u slučaju vizuelizacije WGS84 elipsoida. Svi predloženi koncepti su potvrđeni kroz implementaciju aplikacije za trodimenzionalnu vizuelizaciju planete Zemlje. Aplikacija je razvijena u programskom jeziku C++, koristi OpenGL API i namenjena je *Windows* operativnim sistemima. Korišćeni su pomenuti besplatni izvori podataka, a maksimalna rezolucija primenjenog teksturnog-pokrivača je bolja od 10 cm. Predložen je i način profilisanja grafičkih aplikacija, zasnovanih na OpenGL API-ju i bibliotekama proizvođača grafičkog

hardvera, razvijena je odgovarajuća biblioteka i primenjena u validaciji predloženog algoritma vizuelizacije.

Bez obzira na brojne dobre aspekte predloženog algoritma i softverske arhitekture za vizuelizaciju terena planetarnih razmera, sigurno da postoji još dosta prostora za unapređenje. Ono što će verovatno biti razmatrano u bliskoj budućnosti je rešavanje problema nedostajućih sekcija podataka. To je čest slučaj u GIS aplikacijama, a posledica je diskontinuiteta u izvornim podacima. Razmatra se i drugačija podela planete na particije. Primena SK pristupa može učiniti spajanje particija jednostavnijim i omogućiti ravnomernije prostiranje blokova na granicama particija. U toku je rad na novoj SK projekciji kod koje je moguće menjati karakteristike promenom samo jednog parametra. Takođe, razmatra se i primena *Yin-Yang* projekcije, a moguće je i njeno kombinovanje sa SK projekcijama. *Yin-Yang* projekcija je posebno interesantna zbog podele planete na samo dve particije, što teorijski smanjuje zauzeće memorije za 33%. U slučaju planete Zemlje, realni dobitak je daleko manji, obzirom da južna polarna particija sadrži daleko manje podataka od ostale dve. Tačnije, oni su mnogo manje detaljni, što uslovljava manji broj nivoa u odgovarajućim klipmapama, pa time i manje zauzeće memorije. Međutim, u slučaju neke druge planete, korišćenje dve umesto tri particije može predstavljati značajan dobitak.

Naravno, sva predložena unapređenja mogu dovesti do značajnih promena u samom algoritmu elipsoidalnih klipmapa. Kakav će biti dalji tok razvoja ostaje da se vidi, jer je budućnost nemoguće predvideti.

Bibliografija

[1] Advanced Micro Devices (AMD), "gDEBugger 6.2," April 2012. [URL] http://developer.amd.com/tools-and-sdks/archive/amd-gdebugger/

[2] ——, "AMD Display Library (ADL) SDK 8.0," March 2015. [URL] http://developer.amd.com/tools-and-sdks/graphics-development/display-library-adl-sdk/

[3] ——, "GPU PerfStudio 3.2.18.0," April 2015. [URL] http://developer.amd.com/tools-and-sdks/graphics-development/gpu-perfstudio/

[4] ——, "GPUPerfAPI 2.15,1320.0," January 2015. [URL] http://developer.amd.com/tools-and-sdks/graphics-development/gpuperfapi/

[5] A. A. Apodaca and L. Gritz, *Advanced RenderMan: Creating CGI for Motion Pictures*, 1st ed., ser. The Morgan Kaufmann Series in Computer Graphics. Morgan Kaufmann, December 1999.

[6] A. Asirvatham and H. Hoppe, "Terrain rendering using gpu-based geometry clipmaps," in *GPU Gems 2*, M. Pharr, Ed. Addison-Wesley Professional, March 2005, ch. 2, pp. 27–45.

[7] M. Barnes and D. Salvage, "1800-1988. Why are there so many Geodetic Datums?" APSG 29 Spring Meeting, May 2013. [URL] http://www.apsg.info/Resources/Documents/Presentations/APSG29/2013.05.10_APSG29_Michael_Barnes_Derek_Salvage_Why_are_there_so_many_Geodetic_Datums.pdf

[8] S. Barrett, "stb – single-file public domain libraries for C/C++," June 2015. [URL] https://github.com/nothings/stb/

[9] A. C. Beers, M. Agrawala, and N. Chaddha, "Rendering from compressed textures," in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive*

Techniques, ser. SIGGRAPH '96. New York, NY, USA: ACM, 1996, pp. 373–378. [URL] http://doi.acm.org/10.1145/237170.237276

[10] R. Blackmer, B. Stefanizzi, A. Wolf, and E. Hart, *ATI meminfo*, ATI Technologies, March 2009. [URL] https://www.opengl.org/registry/specs/ATI/meminfo.txt

[11] J. Böttger, M. Preiser, M. Balzer, and O. Deussen, "Detail-in-context visualization for satellite imagery," *Comput. Graph. Forum*, vol. 27, no. 2, pp. 587–596, September 2008. [URL] http://dblp.uni-trier.de/db/journals/cgf/cgf27.html

[12] P. Brown, J. Leech, and M. Kilgard, *EXT texture array*, NVIDIA Corporation, September 2008. [URL] http://www.opengl.org/registry/specs/EXT/texture_array.txt

[13] P. Brown, I. Stewart, N. Haemel, A. Pooley, A. Rasmus, and M. Shah, *EXT texture compression S3TC*, NVIDIA Corporation, July 2013. [URL] https://www.opengl.org/registry/specs/EXT/texture_compression_s3tc.txt

[14] S. Brown, "libsquish – Open source DXT compression library," September 2011. [URL] https://code.google.com/p/libsquish/

[15] M. Calabretta and E. Greisen, "Representations of celestial coordinates in FITS," *Astronomy & Astrophysics*, vol. 395, no. 3, pp. 1077–1122, 2002.

[16] I. Cantlay, "Directx 11 terrain tessellation," NVIDIA, January 2011.

[17] J. Carter, K. Schmid, K. Waters, L. Betzhold, B. Hadley, R. Mataosky, and J. Halleran, "Lidar 101: An Introduction to Lidar Technology, Data, and Applications," National Oceanic and Atmospheric Administration (NOAA) Coastal Services Center, Tech. Rep., November 2012. [URL] http://coast.noaa.gov/digitalcoast/_/pdf/lidar101.pdf

[18] I. Castañ̃o, "High Quality DXT Compression using CUDA," NVIDIA Corporation, February 2007. [URL] http://developer.download.nvidia.com/compute/cuda/1.1-Beta/x86_website/projects/dxtc/doc/cuda_dxtc.pdf

[19] F. Chan and E. O'Neill, "Feasibility study of a quadrilateralized spherical cube earth data base," Environmental Prediction Research Facility, Tech. Rep. EPRF 2-75 (CSC), April 1975.

[20] P. Cignoni, F. Ganovelli, E. Gobbetti, F. Marton, F. Ponchio, and R. Scopigno, "BDAM – batched dynamic adaptive meshes for high performance terrain visualization," *Computer Graphics Forum*, vol. 22, no. 3, pp. 505–514, September 2003.

[21] ——, "Planet–sized batched dynamic adaptive meshes (P-BDAM)," in *Proceedings IEEE Visualization*. Conference held in Seattle, WA, USA: IEEE Computer Society Press, October 2003, pp. 147–155.

[22] M. Clasen and H.-C. Hege, "Terrain rendering using spherical clipmaps," in *Proceedings of the Eighth Joint Eurographics / IEEE VGTC Conference on Visualization*, ser. EUROVIS'06. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2006, pp. 91–98. [URL] http://dx.doi.org/10.2312/VisSym/EuroVis06/091-098

[23] P. Cozzi and K. Ring, *3D Engine Design for Virtual Globes*, 1st ed. CRC Press, June 2011. [URL] http://www.virtualglobebook.com

[24] P. Daniell, A. Mamode, B. Paul, B. Merry, J. Jones, P. Brown, and R. Arnaud, *ARB timer query*, Khronos Group Inc., August 2014. [URL] https://www.opengl.org/registry/specs/ARB/timer_query.txt

[25] C. Dick, J. Krüger, and R. Westermann, "GPU ray-casting for scalable terrain rendering," in *Proceedings of Eurographics 2009 - Areas Papers*, 2009, pp. 43–50.

[26] ——, "GPU-aware hybrid terrain rendering," in *Proceedings of IADIS Computer Graphics, Visualization, Computer Vision and Image Processing 2010*, 2010, pp. 3–10.

[27] A. Diller, "The Ancient Measurements of the Earth," *Isis*, vol. 40, no. 1, 1949. [URL] http://www.jstor.org/stable/227414

[28] A. Dimitrijević, "Arhitektura i implementacija Web servera mapa," magistarski rad, Elektronski fakultet Univerziteta u Nišu, septembar 2003.

[29] A. Dimitrijević, S. Đorđević-Kajan, and D. Rančić, "Eksperimentalna provera ECW kompresije DEM podataka," *YU INFO 2005*, mart 2005.

[30] A. Dimitrijević and D. Rančić, "Efficiency of RINGO Algorithm for Large Terrain Rendering," in *Proceedings of the 11th WSEAS International Conference*

on COMPUTERS, July 2007, pp. 463–467. [URL] http://www.wseas.us/e-library/conferences/2007cscc/papers/561-594.pdf

[31] A. M. Dimitrijević, "Monitoring Graphics Memory Usage," in *OpenGL Insights*, P. Cozzi and C. Riccio, Eds. AK Peters/CRC Press, July 2012, ch. 38, pp. 535–540.

[32] ——, "Performance State Tracking," in *OpenGL Insights*, P. Cozzi and C. Riccio, Eds. AK Peters/CRC Press, July 2012, ch. 37, pp. 527–534.

[33] A. M. Dimitrijević and D. D. Rančić, "GLProfiler – Biblioteka za profilisanje OpenGL grafičkih aplikacija," *YU INFO 2012*, pp. 571–576, mart 2012. [URL] http://www.e-drustvo.org/proceedings/YuInfo2012/html/pdf/578.pdf

[34] ——, "Ellipsoidal Clipmaps – A Planet-Sized Terrain Rendering Algorithm," *Computers & Graphics*, pp. 43–61, 2015. [URL] http://www.sciencedirect.com/science/article/pii/S0097849315000916

[35] M. Duchaineau, M. Wolinsky, D. E. Sigeti, M. C. Miller, C. Aldrich, and M. B. Mineev-Weinstein, "Roaming terrain: Real-time optimally adapting meshes," in *Proceedings of the 8th Conference on Visualization '97*, ser. VIS '97. Los Alamitos, CA, USA: IEEE Computer Society Press, 1997, pp. 81–88.

[36] G. Eckel and K. Jones, *OpenGL Performer Programmer's Guide, Version 3.2*, Silicon Graphics, Inc., July 2004, 007-1680-100. [URL] http://techpubs.sgi.com/library/manuals/1000/007-1680-100/pdf/007-1680-100.pdf

[37] J. S. Falby, M. J. Zyda, D. R. Pratt, R. L. Mackey, and Y. L. Mackey, "NPSNET: Hierarchical data structures for real-time three-dimensional visual simulation," *Computers & Graphics*, vol. 17, no. 1, pp. 65–69, 1993.

[38] D. Feldmann and K. H. Hinrichs, "Gpu based single-pass ray casting of large heightfields using clipmaps," *Proceedings of Computer Graphics International (CGI)*, 2012. [URL] http://viscg.uni-muenster.de/publications/2012/FH12

[39] D. Feldmann, F. Steinicke, and K. H. Hinrichs, "Flexible clipmaps for managing growing textures," in *International Conference on Computer Graphics Theory and Applications (GRAPP)*, P. Richard and J. Braz, Eds. SciTePress, 2011, pp. 173–180. [URL] http://viscg.uni-muenster.de/publications/2011/FSH11

[40] L. Fuentes and R. Menzel, “OpenGL Timestamp Profiler,” August 2012. [URL] https://github.com/Funto/OpenGL-Timestamp-Profiler/

[41] R. Geldreich, “crunch – Advanced DXT texture compression and real-time transcoding library,” April 2012. [URL] https://code.google.com/p/crunch/

[42] ——, “miniz – single c source file deflate/inflate compression library with zlib compatible api,” Google Project Hosting, 2013. [URL] http://code.google.com/p/miniz/

[43] T. Gerstner, “Multiresolution visualization and compression of global topographic data,” GeoInformatica, Tech. Rep., 1999. [URL] http://wissrech.iam.uni-bonn.de/research/pub/gerstner/globe.pdf

[44] S. A. Ghazleh, J. Hartmann, N. Jansen, and S. Kempe, “Water input requirements of the rapidly shrinking dead sea,” *Naturwissenschaften*, vol. 96, pp. 637–643, May 2009.

[45] P. Glaskowsky, “NVIDIA’s Fermi: The First Complete GPU Computing Architecture,” NVIDIA Corporation, Tech. Rep., September 2009. [URL] http://www.nvidia.com/content/PDF/fermi_white_papers/P.Glaskowsky_NVIDIAFermi-TheFirstCompleteGPUComputingArchitecture.pdf

[46] Google, “Google Earth,” 2015. [URL] https://www.google.com/earth/

[47] Graphic Remedy, “gDEBugger 5.8.1,” December 2010. [URL] http://www.gremedy.com/

[48] S. Griffiths, “Slow death of the dead sea: Levels of salt water are dropping by one metre every year,” DailyMail Online, January 2015. [URL] http://www.dailymail.co.uk/sciencetech/article-2897538/Slow-death-Dead-Sea-Levels-salt-water-dropping-one-metre-year.html

[49] M. Hamilton, “Quadrilateralized spherical cubes (quadspheres) in eve flight sim for rendering planets,” July 2013. [URL] http://www.markhamilton.info/2013/07/quadrilateralized-spherical-cubes-cubespheres-in-eve-flight-sim-for-rendering-planets/

[50] D. A. Hastings and P. K. Dunbar, “Global land one-kilometer base elevation (globe) digital elevation model,” National Oceanic and Atmospheric Administration, 1999.

[51] Intel Corporation, *Intel 64 and IA-32 Architectures Software Developer's Manual, Vol.1: Basic Architecture*, 2014. [URL] http://www.intel.com/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-software-developer-vol-1-manual.pdf

[52] Intergraph Corporation, "ERDAS ECW JPEG2000 Software Development Kit (SDK) v5.1," Intergraph Corporation, March 2014. [URL] http://download.intergraph.com/downloads/erdas-ecw-jp2-sdk-v5.1

[53] A. Jarvis, H. I. Reuter, A. Nelson, and E. Guevara, "Hole-filled srtm for the globe version 4," CGIAR-CSI Consortium for Spatial Information, 2008. [URL] http://srtm.csi.cgiar.org

[54] A. Kageyama, "Dissection of a sphere and Yin-Yang grids," *Journal of the Earth Simulator*, vol. 3, pp. 20–28, September 2005. [URL] http://www.jamstec.go.jp/esc/publication/journal/jes_vol.3/pdf/JES3-31Kageyama.pdf

[55] B. Kemen and L. Hrabcak, "Outera," 2014. [URL] http://www.outerra.com

[56] J. Kessenich, *The OpenGL Shading Language, Language Version: 4.50*, January 2015. [URL] https://www.opengl.org/registry/doc/GLSLangSpec.4.50.pdf

[57] M. J. Kilgard, R. Biermann, D. Cornish, N. Carter, M. Craighead, D. Kirkland, and J. Leech, *ARB pixel buffer object*, Khronos Group Inc., October 2013. [URL] https://www.opengl.org/registry/specs/ARB/pixel_buffer_object.txt

[58] R. Kooima, "Planetary-Scale Terrain Composition," doctoral dissertation, Graduate College of the University of Illinois at Chicago, November 2008.

[59] M. Lambers and A. Kolb, "Ellipsoidal cube maps for accurate rendering of planetary-scale terrain data," in *Proceedings Pacific Graphics (Short Papers)*, 2012, pp. 5–10. [URL] http://www.cse.ust.hk/~psander/pg2012proc/pgshort/pdf/005-010.pdf

[60] LandInfo Worldwide Mapping LLC, "GeoCover orthorectified landsat 7 thematic mapper mosaics," NASA Applied Science & Technology Project Office, 2000.

[61] R. Lerbour, J.-E. Marvie, and P. Gautron, "Adaptive real-time rendering of planetary terrains," in *Full Paper Proc. Int. Conf. Computer Graphics, Visualization and Computer Vision (WSCG)*, February 2010, pp. 89–96.

[62] P. Lindstrom, D. Koller, L. F. Hodges, W. Ribarsky, N. Faust, and G. Turner, "Level-of-detail management for real-time rendering of phototextured terrain," Graphics, Visualization, and Usability Center, Georgia Tech, Tech. Rep., 1995.

[63] LizardTech, "MrSID Decode SDK 9," LizardTech, October 2013. [URL] https://www.lizardtech.com/developer/

[64] F. Losasso and H. Hoppe, "Geometry clipmaps: Terrain rendering using nested regular grids," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 769–776, August 2004. [URL] http://doi.acm.org/10.1145/1015706.1015799

[65] E. Makarov, "Clipmats," NVIDIA Corporation, March 2008. [URL] http://developer.download.nvidia.com/SDK/10/Samples/Clipmaps.zip

[66] Microsoft, "Microsoft Virtual Earth 3D 4.0 (Bing Maps 3D)," Microsoft Corporation, April 2014. [URL] https://2ra5-downloads.phpnuke.org/en/c61019/microsoft-virtual-earth-3d

[67] J. R. Miller and T. Gaskins, "Computations on an Ellipsoid for GIS," *Computer-Aided Design and Applications*, vol. 6, pp. 575–583, 2009. [URL] http://people.eecs.ku.edu/~miller/Papers/CAD_6_4__575-583.pdf

[68] J. Mott, "DXT Compression - Retired," Intel Corporation, August 2012. [URL] https://software.intel.com/en-us/articles/dxt-compression-sample

[69] MSDN, *QueryPerformanceCounter function*, Microsoft Corporation, 2014. [URL] http://msdn.microsoft.com/en-us/library/windows/desktop/ms644904(v=vs.85).aspx

[70] ——, *QueryPerformanceFrequency function*, Microsoft Corporation, 2014. [URL] http://msdn.microsoft.com/en-us/library/windows/desktop/ms644905(v=vs.85).aspx

[71] ——, "Bing Maps Tile System," Microsoft Corporation, 2015. [URL] https://msdn.microsoft.com/en-us/library/bb259689.aspx

[72] ——, *Data Conversion Rules*, Microsoft Corporation, 2015. [URL] https://msdn.microsoft.com/en-us/library/windows/desktop/dd607323(v=vs.85).aspx

[73] ——, *Opaque and 1-Bit Alpha Textures (Direct3D 9)*, Microsoft Corporation, 2015. [URL] https://msdn.microsoft.com/en-us/library/windows/desktop/bb147243(v=vs.85).aspx

[74] National Geophysical Data Center, “Digital relief of the surface of the Earth, NOAA National Geophysical Data Center Data Announcement 88-MGG-02,” National Oceanic and Atmospheric Administration, 1988.

[75] National Geospatial-Intelligence Agency (NGA), “EGM2008 - WGS 84 Version,” National Geospatial-Intelligence Agency (NGA), 2013. [URL] http://earth-info.nga.mil/GandG/wgs84/gravitymod/egm2008/egm08_wgs84.html

[76] ——, “Department of Defense (DoD) World Geodetic System (WGS) 1984 – Its Definition and Relationships with Local Geodetic Systems,” National Geospatial-Intelligence Agency (NGA), July 2014. [URL] http://earth-info.nga.mil/GandG/publications/NGA_STND_0036_1_0_0_WGS84/NGA.STND.0036_1.0.0_WGS84.pdf

[77] National Imagery and Mapping Agency (NIMA), “MIL-PRF-89020B, performance specification digital terrain elevation data (DTED),” National Imagery and Mapping Agency (NIMA), May 2000. [URL] http://everyspec.com/MIL-PRF/MIL-PRF-080000-99999/MIL-PRF-89020B_25316/

[78] NVIDIA Corporation, “NVIDIA’s Next Generation CUDA^{TM} Compute Architecture: Fermi^{TM},” NVIDIA Corporation, Tech. Rep., 2009. [URL] http://www.nvidia.com/content/pdf/fermi_white_papers/nvidiafermicomputearchitecturewhitepaper.pdf

[79] ——, *EXT texture filter anisotropic*, November 2014. [URL] https://www.opengl.org/registry/specs/EXT/texture_filter_anisotropic.txt

[80] ——, “NVIDIA GeForce GTX 750 Ti,” NVIDIA Corporation, Tech. Rep., 2014. [URL] http://international.download.nvidia.com/geforce-com/international/pdfs/GeForce-GTX-750-Ti-Whitepaper.pdf

[81] ——, “Nsight Visual Studio Edition 4.7.0,” July 2015. [URL] https://developer.nvidia.com/nvidia-nsight-visual-studio-edition

[82] ——, “NVAPI R352,” July 2015. [URL] https://developer.nvidia.com/gameworksdownload\#?dn=nvapi-r352

[83] ——, “PerfKit 4.4.0,” March 2015. [URL] https://developer.nvidia.com/sites/default/files/akamai/tools/files/PerfKit-4.4.0-windows-desktop.zip

[84] E. O'Neill and R. Laubscher, "Extended studies of a quadrilateralized spherical cube earth data base," Naval Environmental Prediction Research Facility, Tech. Rep. NEPRF 3-76 (CSC), May 1976.

[85] R. Pajarola, "Large scale terrain visualization using the restricted quadtree triangulation," Dept. of Computer Science, ETH Zürich, Tech. Rep. Tech. Rep. 292, 1998.

[86] N. K. Pavlis, S. A. Holmes, S. C. Kenyon, and J. K. Factor, "The development and evaluation of the earth gravitational model 2008 (egm2008)," *Journal of Geophysical Research: Solid Earth*, vol. 117, no. B4, 2012. [URL] http://dx.doi.org/10.1029/2011JB008916

[87] X. Peng, F. Xiao, and K. Takahashi, "Application of the CSLR on the "Yin-Yang"Grid in Spherical Geometry," July 2004, the 2004 Workshop on the Solution of Partial Differential Equations on the Sphere. [URL] http://www-personal.umich.edu/~cjablono/peng.ppt

[88] K. Proudfoot, W. R. Mark, S. Tzvetkov, and P. Hanrahan, "A Real-time Procedural Shading System for Programmable Graphics Hardware," in *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '01. New York, NY, USA: ACM, 2001, pp. 159–170. [URL] http://doi.acm.org/10.1145/383259.383275

[89] A. Ramires, "VSProfileLib – Very Simple Profile Library," April 2015. [URL] http://www.lighthouse3d.com/very-simple-libs/vspl/

[90] D. Rančić, A. Dimitrijević, and B. Predić, "RINGO – Block Based Algorithm for Large Terrain Rendering," in *Proceedings of the 6th WSEAS International Conference on Signal Processing, Computational Geometry & Artificial Vision*, August 2006, pp. 16–20. [URL] http://www.wseas.us/e-library/conferences/2006elounda1/papers/537-210.pdf

[91] ——, "Spatial Coherency and Parallelism in Blocks Reorganization of RINGO Algorithm for Large Terrain Rendering," *WSEAS TRANSACTION on COMPUTERS*, vol. 5, pp. 3073–3079, December 2006.

[92] Republički geodetski zavod Srbije, "Initial geoportal geoSerbia," Republički geodetski zavod Srbije, 2015. [URL] http://www.geosrbija.rs/rga/default.aspx?gui=1&lang=1

[93] M. Rexer and C. Hirt, "Comparison of free high-resolution digital elevation data sets (ASTER 1 GDEM2, SRTM v2.1/v4.1) and validation against accurate heights from the Australian National 2 Gravity Database," *Australian Journal of Earth Sciences*, pp. 1–15, 2014. [URL] http://doi.acm.org/10.1145/965103.807444

[94] M. A. Richards, "A Beginners Guide to Interferometric SAR Concepts and Signal Processing," *IEEE Aerospace and Electronics Systems Magazine*.

[95] L. Row and D. Hastings, "Terrainbase worldwide digital terrain data on cd-rom, release 1.0," National Oceanic and Atmospheric Administration, 1994.

[96] M. Segal and K. Akeley, *The OpenGL Graphics System: A Specification, Version 4.5 (Core Profile)*, May 2015. [URL] https://www.opengl.org/registry/doc/glspec45.core.pdf

[97] J. Snyder, *Map projections—a working manual*, ser. Professional Paper. US Geological Survey, 1987, vol. 1395.

[98] R. Stöckli, E. Vermote, N. Saleous, R. Simmon, and D. Herring, "The blue marble next generation - a true color earth dataset including seasonal dynamics from modis," NASA Earth Observatory, 2005. [URL] http://visibleearth.nasa.gov/view.php?id=74418

[99] H. Stroyan, E. Werness, E. Hart, and M. Kilgard, *NVX gpu memory info*, NVIDIA Corporation, October 2013. [URL] https://www.opengl.org/registry/specs/NVX/gpu_memory_info.txt

[100] C. C. Tanner, C. J. Migdal, and M. T. Jones, "The clipmap: A virtual mipmap," in *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '98. New York, NY, USA: ACM, 1998, pp. 151–158. [URL] http://doi.acm.org/10.1145/280814.280855

[101] C. Thorne, "Using a floating origin to improve fidelity and performance of large, distributed virtual worlds," in *4th International Conference on Cyberworlds (CW 2005), Singapore*, November 2005, pp. 263–270. [URL] http://doi.ieeecomputersociety.org/10.1109/CW.2005.94

[102] T. Ulrich, "Rendering massive terrains using chunked level of detail control," 2002.

[103] USGS, "Earthexplorer," USGS, 2015. [URL] http://earthexplorer.usgs.gov

[104] J. M. P. van Waveren, “Real-Time DXT Compression,” Id Software, Inc., May 2006. [URL] http:www.gamedev.no/projects/MegatextureCompression/324337_324337.pdf

[105] ——, “Real-Time Texture Streaming & Decompression,” Id Software, Inc., November 2006. [URL] http://mrelusive.com/publications/papers/Real-Time-Texture-Streaming-&-Decompression.pdf

[106] S. Venkataraman, *NVIDIA Quadro Dual Copy Engines, WP-05462-001 v01*, NVIDIA Corporation, October 2010. [URL] https://www.opengl.org/registry/specs/EXT/texture_compression_s3tc.txt

[107] L. Williams, “Pyramidal parametrics,” in *Computer Graphics*, ser. SIGGRAPH ’83, P. Tanner, Ed., vol. 17. ACM, July 1983, pp. 1–11.

Dodatak A

GLSL programski kod

A.1 Verteks šejder elipsoidnih klipmapa

```
#version 330
uniform sampler2DArray heightMap;

uniform mat4    matPMV;   // Proj*ModelView matrica
uniform int iSize;    // Velicina bloka [257 sempla]
uniform int iType;    // Particija
uniform float   fTeta;    // Latituda  centra bloka u [rad]
uniform float   fPhi;   // Longituda centra [rad]
uniform vec2  fGrid;    // Korak resetke u [rad]
uniform vec2  blockOffset;  // Pomeraj u [velicinama blokova] kod viseblokovskih nivoa
uniform int gridSize; // Velicina grida: 1-1x1, 2-2x2, 3-3x3
uniform vec4    vClipPlanes;  // 4 klip-rastojanja Xmin, Xmax, Ymin, Ymax

// Parametri elipsoid
uniform float a2; // a^2 za elipsoid
uniform float b2; // b^2 za elipsoid
uniform float ba2;  // b^2/a^2

uniform vec3  pos; // Pozicija centra bloka u dekartovim koordinatama

// Ortovi lokalnog topocentricnog koord. sistema
uniform vec3  Ox;
uniform vec3  Oy;
uniform vec3  Oz;

uniform vec3  Pf; // Faktori linearne promene razlike sa promenom Fi
uniform vec3  Pt; // Faktori linearne promene razlike sa promenom Teta
uniform float hc; // Visina lokalnog koordinatnog pocetka
```

```
uniform int demLevel;
uniform int demMaxLevel; // Najgrublji sloj, posle njega se postavljaju 0.0 za visinu
uniform float demMinSize; // Fizicka dimenzija najdetaljnije (prostorno najmanje) teksture
uniform vec2 startDem[20];
uniform vec2 centDemOffset[20];   // Pomeraji centra tekuceg bloka u odnosu na centar ↩
    odgovarajuce teksture
uniform vec2 centDemOffsetNext[20];   // Pomeraji narednog DEM centra u odnosu na tekuci

// Odsecanje
uniform float eqTeta;     // Lat centra bloka u equatorial koordinatama
uniform vec2 clipLinApp;  // Linearna aproksimacija promene Lat u eq koord. po lokalnim Fi/↩
    Teta

// Izlazni parametri
out  vec2 ex_TexCoord;
out  vec2 ex_TexCoordStat;

// Prevodjenje globalnog Lat/Lon u Dekartov koord. sistem za ekvatorijalnu particiju.
vec3 WGS2Cartesian(float dLon, float dLat, float h)
{
  // dLon  [rad], dLat [rad], h [m]
  float cosTdT = cos(fTeta+dLat);
  float sinTdT = sin(fTeta+dLat);
  float cosFdF = cos(fPhi+dLon);
  float sinFdF = sin(fPhi+dLon);

  float N = a2/sqrt(a2*cosTdT*cosTdT + b2*sinTdT*sinTdT);
  float x = (N+h)*cosTdT*sinFdF;
  float y = (ba2*N+h)*sinTdT;
  float z = (N+h)*cosTdT*cosFdF;
  return vec3(x, y, z);
}

// Prevodjenje globalnog Lat/Lon u Dekartov koord. sistem za polarne particije.
vec3 WGS2CartesianNorth(float dLon, float dLat, float h)
{
  // dLon  [rad], dLat [rad], h [m]
  float cosTdT = cos(fTeta+dLat);
  float sinTdT = sin(fTeta+dLat);
  float cosFdF = cos(fPhi+dLon);
  float sinFdF = sin(fPhi+dLon);

  float T2 = asin(cosTdT*cosFdF);
  float cosFdFD = cos(T2);
  float sinFdFD = sin(T2);

  float N = a2/sqrt(a2*cosFdFD*cosFdFD + b2*sinFdFD*sinFdFD);
  float x = (N+h)*cosTdT*sinFdF;
```

```
  float y = (N+h)*sinTdT;
  float z = (ba2*N+h)*cosTdT*cosFdF;
  return vec3(x, y, z);
}

// Vraca dekartove koordinate tacke u lokalnom topocentricnom sistemu.
vec3 WGSVect(float dLon, float dLat, float h)
{
  vec3 P;
  if(iType == 0)
    P = WGS2Cartesian(dLon, dLat, h);
  else
    P = WGS2CartesianNorth(dLon, dLat, h);

  vec3 L = P - pos;
  return vec3(dot(L, Ox), dot(L, Oy), dot(L, Oz));
}

// Vraca visinu na zadatom nivou.
float GetHeight(int level)
{
  float tSize = demMinSize * exp2(float(level));// velicina datog sloja
  vec3 demCoor = vec3(startDem[level] + vec2(0.5, 0.5) + ((ex_TexCoord + centDemOffset[level↩
      ]) / tSize), level);
  float h = texture(heightMap, demCoor).r;
  h = h * 65535.0;
  h = (h / 7.0) - 500.0;
  return h;
}

// Vraca visinu na sledecem (grubljem) nivou.
float GetHeightNext(int level)
{
  float tSize = demMinSize * exp2(float(level+1));// velicina datog sloja
  vec3 demCoor = vec3(startDem[level+1] + vec2(0.5, 0.5) + ((ex_TexCoord + centDemOffset[↩
      level] - centDemOffsetNext[level+1]) / tSize), level+1);
  float h = texture(heightMap, demCoor).r;
  h = h * 65535.0;
  h = (h / 7.0) - 500.0;
  return h;
}

// Konverzija iz blokovskih Lat/Lon u topocentricne koordinate
vec3 LocalCoords(float dTx, float dTy, float h)
{
  vec3 v  = WGSVect(dTx, dTy, h); //WGS u datoj tacki
  vec3 v2 = Pf * vec3(dTx, dTx, abs(dTx)) + Pt * vec3(dTy, dTy, abs(dTy));
  v2.z += (h-hc);
```

```
  float th = 7e-4;
  float dD = max(abs(dTx), abs(dTy));
  if(dD < th)
    return v2;
  else if(dD < 2.0*th)
    return mix(v2, v, (dD-th)/th);
  else
    return v;
}

// Unutrasnje odsecanje.
float InnerClipDist(float dTx, float dTy)
{
  vec2 halfCell = fGrid / 2.0;
  if(dTx <= vClipPlanes.x+halfCell.x || dTx >= vClipPlanes.y-halfCell.x || dTy <= ↩
      vClipPlanes.z+halfCell.y || dTy >= vClipPlanes.w-halfCell.y)
    return 1.0;
  else
    return -1.0; // odsecanje
}

// Odsecanje na granici particije.
float PartitionClipDist(float dTx, float dTy)
{
  float clipDist = 1.0;
  float TetaCurrent = fTeta+dTy;
  float PhiCurrent = fPhi+dTx;
  float preklop = 0.0;
  float fact1 = 0.75;
  float fact2 = 0.85;
  float _45 = 0.78539819;
  if(iType == 0) // Ekvatorijalna particija
  {
    if(fGrid.x > 1e-4)
    {
      float Lat45 = _45 + preklop + fact2*fGrid.y;
      if(TetaCurrent > Lat45)
      {
        float delta1 = (Lat45-TetaCurrent)/fGrid.y;
        clipDist =  delta1;
      }
      else if(TetaCurrent < -Lat45)
      {
        float delta1 = (Lat45+TetaCurrent)/fGrid.y;
        clipDist =  delta1;
      }
    }
    else
    {
```

```
    if(fTeta > 0.0)
      clipDist = fact1*fGrid.y - (clipLinApp.x + dTy);
    else if(fTeta < 0.0)
      clipDist = fact1*fGrid.y + (clipLinApp.y + dTy);
  }
}
else
{
  float Lat45 = _45 - preklop - fact2*fGrid.y;
  if(iType == 1) // Severna polarna particija
  {
    float teta;
    if(fGrid.x > 1e-4)
    {
      // Racunanje globalne latitude - teta
      teta = asin(cos(TetaCurrent)*cos(PhiCurrent));
      if(teta < Lat45)
      {
        float delta1 = (teta-Lat45)/fGrid.y;
        clipDist = delta1;
      }
    }
    else
    {
      teta = eqTeta + clipLinApp.x * dTx + clipLinApp.y * dTy + fact1*fGrid.y;
      float delta1 = teta/fGrid.y;
      clipDist = delta1;
    }
  }
  else // Juzna polarna particija
  {
    float teta;
    if(fGrid.x > 1e-4)
    {
      // Racunanje globalne latitude - teta
      teta = asin(-cos(TetaCurrent)*cos(PhiCurrent));
      if(teta > -Lat45)
      {
        float delta1 = (teta+Lat45)/fGrid.y;
        clipDist = -delta1;
      }
    }
    else
    {
      teta = eqTeta + clipLinApp.x * dTx + clipLinApp.y * dTy - fact1*fGrid.y;
      float delta1 = teta/fGrid.y;
      clipDist = -delta1;
    }
  }
```

```
  }
  return clipDist;
}

void main(void)
{
  // Izracunavanje poyicije temena u okviru bloka
  int i = gl_VertexID / iSize;
  int j = gl_VertexID % iSize;
  int s2 = iSize/2;
  int i_ex = int((j-s2) + (blockOffset.x * (iSize-1)));
  int j_ex = int((s2-i) + (blockOffset.y * (iSize-1)));

  float dTx = i_ex * fGrid.x;
  float dTy = j_ex * fGrid.y;

  float toDeg = 57.295780;//57.295779513;
  ex_TexCoord = vec2(dTx*toDeg, dTy*toDeg);

  float pi = 3.1415927;
  float pi2= 6.2831855;
  float deg100 = 1.7453293;

  if(iType == 0)
    ex_TexCoordStat = vec2(0.5 + (fPhi+dTx)/pi2, 0.5 + (fTeta+dTy)/pi);
  else
    ex_TexCoordStat = vec2(0.5 + (fPhi+dTx)/deg100, 0.5 + (fTeta+dTy)/deg100);

  // Odredjivanje visine
  int level = demLevel; // Tekuci DEM level. level=O je najfiniji.
  float h = 0.0;
  vec3 hv;
  int lvl = max(level, 0);
  if(lvl < demMaxLevel)
    h = GetHeight(lvl);

  // Mesanje susednih nivoa na obodu
  if(level >= 0)
  {
    float h2 = 0.0;
    if(level+1 < demMaxLevel)
      h2 = GetHeightNext(level); //Neg. Osvetljenje

    int s_ex = gridSize * (iSize-1) / 2;
    int w = s_ex / 5;
    float ax = clamp( float(abs(i_ex) + w - s_ex) / float(w), 0.0, 1.0);
    float ay = clamp( float(abs(j_ex) + w - s_ex) / float(w), 0.0, 1.0);

    float maxalpha = max(ax, ay);
```

```
    h = mix(h, h2, maxalpha);
  }
  float dh = 2.0*3189068.5 * fGrid.y;
  gl_ClipDistance[0] = 1.0;

  // Unutrasnje odsecanje
  float c1 = InnerClipDist(dTx, dTy);

  // Odsecanje na granici particije
  float c2 = 1.0;
  if(c1 < 0.0)
  {
    h -= dh;
    gl_ClipDistance[0] = c1;
  }
  else
  {
    c2 = PartitionClipDist(dTx, dTy);
    if(c2 < 0.0)
    {
      h -= dh;
      gl_ClipDistance[0] = c2;
    }
  }

  // Izracunavanje lokalnih topocentricnih koordinata
  vec3 v = LocalCoords(dTx, dTy, h);

  // Izracunavanje klip-koordinata temena
  gl_Position = matPMV * vec4(v.x,v.y,v.z,1.0);
}
```

A.2 Fragment šejder elipsoidnih klipmapa

```
#version 330

uniform sampler2DArray texID; // Polje tekstura - stub klipmape
uniform sampler2D texIDStat;  // Staticka tekstura - piramida klipmape
uniform float Dmin; // Fizicka dimenzija najdetaljnije (prostorno najmanje) teksture
uniform int MaxLevel; // Maksimalni nivo
uniform int levelCount; // Broj nivoa
uniform vec2 startTex[18];
uniform vec2 centTexOffset[18]; // Pomeraji centra teksture (nivoa)

uniform int iType;    // Globalna particija
uniform float   fTeta;    // Latituda  centra bloka u [rad]
```

```
uniform float   fPhi;   // Longituda centra [rad]

// Ulazni parametri
in  vec2 ex_TexCoord;
in  vec2 ex_TexCoordStat;

// Izlazni parametar
out vec4 out_Color;

void main(void)
{
  // Rastojanje u teksturnim koordinatama uz skaliranje po x-koordinati u zavisnosti od ↩
      latitude
  float teta = fTeta;
  float cor = cos(teta);
  float cor2 = cor * cor;
  float mL = sqrt( ex_TexCoord.x*ex_TexCoord.x * cor2 + ex_TexCoord.y * ex_TexCoord.y);
  float alpha = cos(teta);
  float cosfTeta = cos(fTeta);
  float cosfPhi = cos(fPhi);

  if(iType == 1) // Severna polarna particija
    teta = asin(cosfTeta*cosfPhi); // Globalna latituda
  else if(iType == 2) // Juzna polarna particija
    teta = asin(-cosfTeta*cosfPhi); // Globalna latituda

  if(iType > 0) // Za obe polarne particije
    alpha = cos(1.57079638 - abs(teta));

  mL = mL / alpha;


  float tSize; // Velicina datog nivoa
  float Dopt = Dmin*0.9; // Aktivni region

  // Odredjivanje sloja i faktora mesanja
  float fLayer = max(MaxLevel, MaxLevel + 2.0 + log2( mL / Dopt ));
  int layer   = int(floor(fLayer));
  float fact  = fract(fLayer);

  tSize = Dmin * exp2(layer-MaxLevel);// velicina datog sloja

  // Racunanje koordinata
  if(layer >= levelCount)
  {
    out_Color = texture(texIDStat, ex_TexCoordStat);
  }
  else if(layer == (levelCount-1))
  {
```

```
      vec3 texCoor = vec3(startTex[layer] + vec2(0.5, 0.5) + ((ex_TexCoord + centTexOffset[↩
          layer]) / tSize), layer);
      out_Color = mix( textureGrad(texID, texCoor, dFdx(ex_TexCoord) / tSize, dFdy(↩
          ex_TexCoord) / tSize),
              texture(texIDStat, ex_TexCoordStat), fact);
    }
    else
    {
      vec3 texCoor = vec3(startTex[layer] + vec2(0.5, 0.5) + ((ex_TexCoord + centTexOffset[↩
          layer]) / tSize), layer);
      vec3 texCoor2= vec3(startTex[layer+1] + vec2(0.5, 0.5) + ((ex_TexCoord + centTexOffset↩
          [layer+1]) / (2.0*tSize)), layer+1);

      vec2 dx = dFdx(ex_TexCoord) / tSize;
      vec2 dy = dFdy(ex_TexCoord) / tSize;

      out_Color = mix(textureGrad(texID, texCoor, dx, dy), textureGrad(texID, texCoor2, 0.5 ↩
          * dx, 0.5 * dy), fact);
    }
}
```

Biografija

Aleksandar Dimitrijević rođen je 25. maja 1973. godine u Nišu, gde i trenutno živi i radi.

Osnovnu i srednju školu završio je u Nišu sa odličnim uspehom, za šta je nagrađen Vukovim diplomama. Elektronski fakultet u Nišu upisao je školske 1992/93. godine. Za izuzetne rezultate tokom studija, nagrađen je pohvalnicama na prvoj i drugoj godini studija, poveljama na trećoj i četvrtoj godini studija, kao i poveljom „Dušan Mitrović" za najbolje izrađeni diplomski rad u školskoj 1996/97. godini. Diplomirao je 20.10.1997. godine na profilu Računarska tehnika i informatika sa prosečnom ocenom 9,76 i ocenom 10 na diplomskom radu.

Magistarske studije upisao je školske 1997/98. godine, na profilu Računarska tehnika i informatika, i sve ispite predviđene programom položio sa ocenom 10. Dana 24.09.2003. godine odbranio je magistarsku tezu pod nazivom: „Arhitektura i implementacija Web servera mapa", i time ispunio uslov za sticanje akademskog naziva magistra elektrotehničkih nauka.

Od završetka redovnih studija, angažovan je na Elektronskom fakultetu u Nišu u svojstvu istraživača-stipendiste Ministarstva za nauku i tehnologiju Republike Srbije, na projektu „Razvoj telekomunikacionih uređaja i softvera za radiodifuzne kablovske i satelitske sisteme" (Ev.br.s.1.04.12.0153). U istom periodu angažovan je i u nastavnom procesu, na poslovima izvođenja vežbi pri katedri za Računarsku tehniku i informatiku.

Dana 14.02.2000. godine izabran je u zvanje asistenta-pripravnika za predmete „Strukture i baze podataka" i „Računarske mreže".

Dana 11.03.2004. izabran je u zvanje asistent za užu naučnu oblast Računarstvo i informatika. Trenutno je angažovan na izvođenju auditivnih i laboratorijskih vežbi iz predmeta: Strukture podataka, Računarske mreže, Računarska grafika, Interakcija čovek-računar, Projektovanje računarskih mreža, Računarska animacija i 3D grafički protočni sistemi.

IZJAVE AUTORA

Izjava 1.

IZJAVA O AUTORSTVU

Izjavljujem da je doktorska disertacija, pod naslovom

RINGO – ARHITEKTURA ZA TRODIMENZIONALNU VIZUELIZACIJU TERENA VELIKIH RAZMERA

koja je odbranjena na Elektronskom fakultetu Univerziteta u Nišu:

- rezultat sopstvenog istraživačkog rada;
- da ovu disertaciju, ni u celini, niti u delovima, nisam prijavljivao na drugim fakultetima, niti univerzitetima;
- da nisam povredio autorska prava, niti zloupotrebio intelektualnu svojinu drugih lica.

Dozvoljavam da se objave moji lični podaci, koji su u vezi sa autorstvom i dobijanjem akademskog zvanja doktora nauka, kao što su ime i prezime, godina i mesto rođenja i datum odbrane rada, i to u katalogu Biblioteke, Digitalnom repozitorijumu Univerziteta u Nišu, kao i u publikacijama Univerziteta u Nišu.

U Nišu, _____

Autor disertacije: **Aleksandar Dimitrijević**

Potpis autora disertacije:

Izjava 2.

IZJAVA O ISTOVETNOSTI ŠTAMPANOG I ELEKTRONSKOG OBLIKA DOKTORSKE DISERTACIJE

Ime i prezime autora: **Aleksandar Dimitrijević**

Naslov disertacije: **RINGO – ARHITEKTURA ZA TRODIMENZIONALNU VIZUELIZACIJU TERENA VELIKIH RAZMERA**

Mentor: **prof. dr Dejan Rančić**

Izjavljujem da je štampani oblik moje doktorske disertacije istovetan elektronskom obliku, koji sam predao za unošenje u **Digitalni repozitorijum Univerziteta u Nišu**.

U Nišu, _____

Potpis autora disertacije:

Izjava 3.

IZJAVA O KORIŠĆENJU

Ovlašćuje Univerzitetsku biblioteku „Nikola Tesla“ da, u digitalnom repozitorijumu Univerziteta u Nišu, unese unese moju doktorsku disertaciju, pod naslovom:

RINGO – ARHITEKTURA ZA TRODIMENZIONALNU VIZUELIZACIJU TERENA VELIKIH RAZMERA

Disertaciju sa svim prilozima predao sam u elektronskom obliku, pogodnom za trajno arhiviranje.

Moju doktorsku disertaciju, unetu u Digitalni repozitorijum Univerziteta u Nišu, mogu koristiti svi koji poštuju odredbe sadržane u odabranom tipu licence Kreativne zajednice (Creative Commons), za koju sam se odlučio.

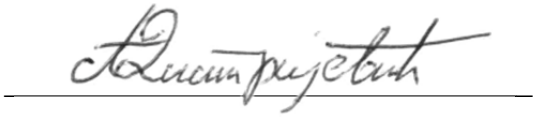
1. Autorstvo **(CC BY)**
2. Autorstvo – nekomercijalno **(CC BY-NC)**
3. Autorstvo – nekomercijalno – bez prerade **(CC BY-NC-ND)**
4. Autorstvo – nekomercijalno – deliti pod istim uslovima **(CC BY-NC-SA)**
5. Autorstvo – bez prerade **(CC BY-ND)**
6. Autorstvo – deliti pod istim uslovima **(CC BY-SA)**

(Molimo da podvučete samo jednu od ponuđenih šest licenci; opis licenci dat je u nastavku teksta).

U Nišu, _____

Autor disertacije: **Aleksandar Dimitrijević**

Potpis autora disertacije:



TIPOVI LICENCI KREATIVNE ZAJEDNICE*

1. Autorstvo (CC BY)

Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, i prerade, ako se navede ime autora, na način određen od strane autora ili davaoca licence, čak i u komercijalne svrhe. Ovo je najslobodnija od svih licenci.

2. Autorstvo – nekomercijalno (CC BY-NC)

Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, i prerade, ako se navede ime autora, na način određen od strane autora ili davaoca licence. Ova licenca ne dozvoljava komercijalnu upotrebu dela.

3. Autorstvo – nekomercijalno – bez prerade (CC BY-NC-ND)

Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, bez promena, preoblikovanja ili upotrebe dela u svom delu, ako se navede ime autora, na način određen od strane autora ili davaoca licence. Ova licenca ne dozvoljava komercijalnu upotrebu dela. U odnosu na sve ostale licence, ovom licencom se ograničava najveći obim prava korišćenja dela.

4. Ауторство – nekomercijalno – deliti pod istim uslovima (CC BY-NC-SA)

Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, i prerade, ako se navede ime autora, na način određen od strane autora ili davaoca licence, i ako se prerada distribuira pod istom ili sličnom licencom. Ova licenca ne dozvoljava komercijalnu upotrebu dela i prerada.

5. Autorstvo – bez prerade (CC BY-ND)

Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, bez promena, preoblikovanja ili upotrebe dela u svom delu, ako se navede ime autora, na način određen od strane autora ili davaoca licence. Ova licenca dozvoljava komercijalnu upotrebu dela.

6. Autorstvo – deliti pod istim uslovima (CC BY-SA)

Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, i prerade, ako se navede ime autora, na način određen od autora ili davaoca licence, i ako se prerada distribuira pod istom ili sličnom licencom. Ova licenca dozvoljava komercijalnu upotrebu dela i prerada. Slična je softverskim licencama, odnosno licencama otvorenog koda.

*

* Više o licencama Kreativne zajednice na adresi: **http://creativecommons.org.rs/?page_id=74CC**