Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Gal Kos

3D vizualizacija dogajanja na bojišču v okolju Unity

diplomsko delo

univerzitetni študijski program prve stopnje
računalništvo in informatika

Mentor: prof. dr. Saša Divjak
Somentor: doc. dr. Matija Marolt

Ljubljana 2016

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

3D vizualizacija dogajanja na bojišču v okolju Unity

Tematika naloge:

V diplomskem delu razvijte aplikacijo za vizualizacijo dogajanja na bojišču v okolju Unity. Podpira naj prikaz stanja entitet preko protokola DIS v realnem času, in sicer tako sistemov s 3D modeli kot enot s simbologijo zveze Nato. Svet predstavite z 2D zemljevidi, izvedite pa tudi študijo uporabe 3D višinskih slik za generiranje 3D terena.

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Gal Kos sem avtor diplomskega dela z naslovom:

3D vizualizacija dogajanja na bojišču v okolju Unity

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom prof. dr. Saše Divjaka in somentorstvom doc. dr. Matija Marolta,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela na svetovnem spletu preko univerzitetnega spletnega arhiva.

V Ljubljani, dne 5. septembra 2016 Podpis avtorja:

Za začetek bi se želel zahvaliti vsem, ki so na kakršenkoli način pomagali pri nastanku tega dela. Najprej se zahvaljujem prof. dr. Saši Divjaku za zaupanje, ki mi ga je izkazal s svojim mentorstvom. Prav tako bi se rad zahvalil somentorju doc. dr. Matiji Maroltu za usmerjanje in svetovanje pri izdelavi diplomskega dela. Velika in iskrena zahvala gre as. dr. Cirilu Bohaku za vse tehnične nasvete, odlične ideje in pomoč pri pisanju.

Hvala vsem prijateljem in kolegom, ki so mi skozi leta pomagali, me spodbujali, bodrili in nemalokrat zabavali.

Nazadnje se za vso podporo zahvaljujem svoji družini, ki je z mano vsa leta mojega šolskega in študijskega izobraževanja. Hvala.

Svoji družini.

Kazalo

Slike

Tabele

Seznam uporabljenih kratic

kratica	angleško	slovensko
2D	two dimensional	dvodimenzionalna, dvodimenzionalno, dvodimenzionalni
3D	three dimensional	tridimenzionalna, tridimenzionalno, tridimenzionalni
API	application programming interface	vmesnik za programiranje aplikacij
APP-6	Allied Procedural Publication 6	/
BIH	International Time Bureau	/
C2	Command And Control	/
C4I	Command, Control, Communications, Computers, and Intelligence	/
CTP	Conventional Terrestrial Pole	/
DIS	Distributed Interactive Simulation	/
DTED	Digital Terrain Elevation Data	/
EGM	Earth Gravitational Model	/
EPSG	European Petroleum Survey Group	/
FIFO	First In, First Out	prvi-noter-prvi-ven
GDEM	Global Digital Elevation Model	/

GIS	Geographic Information System	geografski informacijski sistem
HLA	High Level Architecture	/
HTTP	Hypertext Transfer Protocol	/
IEEE	Institute of Electrical and Electronics Engineers	/
IERS	International Earth Rotation and Reference Systems Service	/
IO	Information Operations	informacijske operacije
ITRS	International Terrestrial Reference System	/
JCATS	Joint Conflict And Tactical Simulation	/
LOD	Level of detail	/
MOOTW	Military Operations Other Than War	vojaške nevojne operacije
MOVES	The Modeling, Virtual Environment, and Simulation Institute	/
NASA	National Aeronautics and Space Administration	/
NATO	North Atlantic Treaty Organisation	Organizacija severnoatlantske pogodbe
NGA	National Geospatial-Intelligence Agency	/
NPS	Naval Postgraduate School	/
PDD	Personal Detection Device	osebno zaznavno napravo
PDU	protocol data unit	protokolarna podatkovna enota
PNG	Portable Network Graphics	/
SOF	Special Operations Forces	specialne sile
SRTM	Shuttle Radar Topography Mission	/

SV	/	Slovenska vojska
TES	Tactical Engagement Simulation	/
UDP	User Datagram Protocol	/
UI	user interface	uporabniški vmesnik
VBS	Virtual Battlespace	/
WGS	World Geodetic System	svetovni geodetski sistem
WMO	World Meteorological Organization	Svetovna meteorološka organizacija
XML	Extensible Markup Language	/

Povzetek

V tem diplomskem delu smo s pomočjo programskega jezika C# implementirali aplikacijo v pogonu Unity, ki omogoča vizualiziranje stanja entitet preko standarda DIS v realnem času. Aplikacija omogoča vizualiziranje dveh tipov entitet, in sicer entitete tipa enote, ki so predstavljene s simbologijo zveze NATO, ter sisteme, ki so predstavljeni s 3D modeli. V okviru projekta smo realizirali dinamično generiranje okolja v dveh dimenzijah s pomočjo satelitskih posnetkov podjetja Google. Okolje je predstavljeno v spletni Mercatorjevi projekciji in je implementirano po principu nivojev podrobnosti. Za pozicioniranje in premikanje entitet je uporabljena aproksimacijska metoda pretvarjanja kartezičnih koordinat v geografske po Bowringu. Poleg osnovnega okolja smo izdelali prototip dinamičnega generiranje terena, na katerega smo aplicirali satelitske posnetke. Podatke o višinah smo pridobili iz podatkovne baze projekta SRTM.

Ključne besede: standard DIS, spletna Mercatorjeva projekcija, aproksimacijska metoda po Bowringu, simulator JCATS, projekt SRTM, pogon Unity.

Abstract

In this thesis we have implemented an application in Unity that visualizes states of entities via standard DIS in real time. Our programming language of choice was C#. The application supports visualizing of two types of entities, namely entity type unit and entity type system. Units are presented with NATO symbology while systems are presented with 3D models. In the scope of the thesis dynamical generation of environment in two dimensions was implemented using Google satellite imagery. The environment is presented in Web Mercator and implemented with Level of detail algorithm. For positioning and movement of the entities Bowring approximation method for transformation from spatial to geographical coordinates was used. In addition to the basic environment we have developed prototype for the dynamical generation of terrain on which satellite imagery was applied. Height data that were used in the prototype was obtained from the database of the project SRTM.

Keywords: standard DIS, Web Mercator, Bowring approximation method, JCATS, Project SRTM, Unity.

Poglavje 1

Uvod

Človek je od nekdaj opazoval svet in njegovo spreminjanje, pa naj bo to vreme, naravni pojavi, plimovanje ali izmenjevanje dneva in noči. Ob opazovanju spreminjanja opazovanega objekta se vedno postavi vprašanje o tem, ali lahko človek na te spremembe na kakršnikoli način vpliva. Velikokrat je lahko le nemi opazovalec, ki nad opazovanim objektom nima in nikoli ne bo imel nadzora.

Ko ljudje naletimo na tovrstne probleme, po navadi zamenjamo strategijo in skušamo opazovani objekt posnemati oziroma simulirati. To storimo tako, da ustvarimo posnetek opazovanega objekta in prevzamemo nadzor nad spremenljivkami, ki definirajo obnašanje našega simuliranega objekta oziroma modela. Tu se torej srečamo s pojmoma simulacija in model.

Simulacije in modeli so abstrakcije realnosti. Modeli so matematična, logična ali kakšna druga strukturirana predstavitev realnosti, medtem ko so simulacije specifični sistemi, ki uporabljajo modele in s pomočjo njih pridejo do določenih izidov. Oboji namenoma poudarjajo samo del resničnosti, kar je potrebno bodisi zaradi omejene računske moči bodisi zaradi usmeritve pozornosti na pomemben vidik simulacije.

Mnogokrat simulacije suhoparno predstavljajo svoje rezultate, saj pri njih po navadi ni pomemben izgled, ampak matematična pravilnost modela glede na njegov dvojnik iz resničnosti oziroma pravilnost izhodov simulacije glede

na vhode. Takrat pridejo v poštev vizualizacijska orodja, ki preko dogovorjenega načina oziroma protokola komunicirajo s simulacijo in na ustrezen način interpretirajo oziroma predstavljajo njene izhode.

1.1 Cilj diplomskega dela

Cilj tega diplomskega dela je bila implementacija aplikacije za prikaz dogajanja na sodobnem bojišču. Projekt je bil v osnovi zasnovan skupaj z oddelkom Slovenske vojske (SV), ki v vojašnici v Postojni izvaja vojaške vadbe s pomočjo simulatorjev. Vsi simulatorji, ki jih uporablja SV, so predstavljeni v tem diplomskem delu.

Aplikacija je v osnovi namenjena prikazovanju izhodnih podatkov simulatorja JCATS (angl. Joint Conflict And Tactical Simulation) preko protokola s standardom DIS (angl. Distributed Interactive Simulation).

Poleg interpretacije izhodov simulatorja, je bilo potrebno definirati okolje, v katerem so predstavljena stanja simulatorja. Tako je glavni cilj te diplomske naloge vizualizator, ki v okolju prikazuje izhode simulacije, pri čimer je pomembno, da izvajanje doseže v realnem času.

1.2 Realnočasovne vojaške simulacije

Simulacije so poskus modeliranja realne ali hipotetične situacije, ki so namenjene preučevanju delovanja sistema. S spreminjanjem parametrov simulacije je mogoče napovedati vedenje fizičnega sistema, ki ga te simulacije poskušajo posnemati oziroma simulirati. Modeliranje sistemov poteka s pomočjo matematičnih modelov, ki skušajo najti analitične rešitve, s katerimi je mogoče napovedovanje obnašanja sistema iz niza parametrov in začetnih pogojev. Simulacije se pogosto uporabljajo kot nadomestilo sistemskih modelov, za katere preproste oblike analitičnih rešitev niso mogoče. Za vse tipe simulacij je značilno, da poskušajo poustvariti vzorec ponovljivih scenarijev za model, v

katerem bi bil zajem vseh stanj časovno ali računsko prezahteven ali nemogoč.

Modeli so matematična predstavitev poljubnega subjekta ali procesa, zato so ustvarjeni iz množice podatkov, enačb in izračunov, ki posnemajo dejanja originalnih dvojnikov. Lahko so preproste podobe dejanskih dvojnikov ali pa kompleksne strukture, ki nosijo vse značilnosti objektov oziroma procesov, ki jih predstavljajo.

Simulacije lahko prejemajo podatke iz več vhodnih virov, in sicer preko senzorjev in drugih fizičnih naprav povezanih z modelom, preko krmilnikov, ki usmerjajo napredek simulacije, preko trenutnih ali preteklih podatkov, ki so ročno vneseni, preko vrednosti, ki so pridobljene kot stranski produkt drugih procesov, ali preko vrednosti, ki so pridobljene namensko, s pomočjo drugih simulacij, modelov ali procesov.

Simulacije se razlikujejo tudi po času, v katerem so na voljo vhodni podatki. Nespremenljivi podatki so pogosto vgrajeni v programsko kodo modela, bodisi zato, ker je vrednost resnično invariantna, ali pa zato, ker je upoštevano, da je vrednost nespremenljiva za vse mogoče primere. Drug način vnašanja podatkov v simulacijo je predprocesiranje, kjer se podatki preberejo ob zagonu simulacije. Nazadnje se lahko podatki pridobivajo tudi med potekom simulacije.

Realnočasovne simulacije

Realnočasovne simulacije so simulacije, ki potekajo v realnem času (angl. Real-Time). Tovrstna simulacija je računalniški model fizičnega sistema, ki se izvaja enako hitro kot teče realni svetovni čas [7]. Med diskretno časovno simulacijo čas napreduje v enakomernih časovnih intervalih. Tovrstni sistem se v splošnem imenuje simulacija s fiksnim časovnim korakom. Obstajajo tudi druge tehnike, ki probleme rešujejo s spremenljivim časovnim korakom in se uporabljajo za reševanje visokofrekvenčne dinamike in nelinearnih sistemov, vendar za realnočasovne simulacije niso primerne.

V realnočasovnih simulacijah natančnost izračunov ni odvisna samo od na-

tančnosti dinamičnega prikaza sistema, temveč tudi od časa, ki je potreben za pridobivanje rezultatov. Da je realnočasovna simulacija veljavna, mora uporabljeni realnočasovni simulator pridobiti notranje spremenljivke in rezultate simulacije v enakem času, kot bi to storil njegov fizični dvojnik. Dejansko je potrebno, da je čas za izračun rešitve v danem časovnem koraku krajši od trajanja realnosvetovnega v isti sekvenci, kar realnočasovnemu simulatorju omogoča izvajanje vseh operacij, ki so potrebne, da je realnočasovna simulacija relevantna. V danem časovnem koraku ni izvedena nobena operacija predhodnega oziroma naslednjega koraka, saj simulator v primeru, ko izračuna vse operacije dane sekvence, čaka do naslednjega časovnega koraka. Nasprotno pa pospešena simulacija (angl. accelerated simulation) uporablja čas mirovanja za izračun enačb naslednjega časovnega intervala. Če operacije simulatorja niso izvršene v danem fiksnem časovnem koraku, se zgodi prekoračitev, kar pomeni, da simulacija vsebuje napake in posledično ni veljavna.

Na podlagi teh osnovnih opredelitev lahko sklepamo, da realnočasovni simulator deluje pričakovano, če so enačbe in stanja simuliranega sistema rešene pravočasno, s sprejemljivo podobnostjo z njegovim fizičnim dvojnikom in brez pojavov prekoračitev (angl. override).

Naloge simulatorja opravlja simulacijski razreševalnik (angl. solver). Za vsak časovni korak razreševalnik izvede enake serije nalog: 1. prebere vhode in generira izhode, 2. reši modelne enačbe 3. izmenja rezultate z ostalimi simulacijskimi vozlišči, 4. čaka na začetek naslednjega koraka.

Vojaške simulacije

Vojaške simulacije (angl. military simulations) so simulacije, v katerih je mogoče preizkusiti in izpopolniti teorije vojskovanja brez potrebe po dejanskih sovražnostih [20]. So koristen način za razvoj taktičnih, strateških in doktrinarnih rešitev, vendar pa kritiki trdijo, da so ugotovitve iz teh simulacij zaradi narave približkov uporabljenih modelov inherentno pomanjkljive. Vojaške simulacije zajemajo širok spekter dejavnosti od terenskih vaj polnega obsega do

abstraktnih računalniških modelov, ki lahko delujejo z malo sodelovanja živega uporabnika ali brez njega.

Obsežne vojaške vaje ali celo tiste v manjšem merilu, niso vedno izvedljive oziroma niso niti zaželene, prav tako pa je vključno s finančnimi sredstvi pomemben dejavnik razpoložljivost virov. Pogosto je bolj priročno preizkušanje teorij z manjšim številom osebja, obenem pa zmanjšane vaje ohranijo del človeškega vložka in tako še vedno odražajo človeško nepredvidljivost, ki naredi bojno modeliranje zahtevnejše.

Računalniško podprte simulacije so nadgradnja terenskih simulacij. Obstajajo različne vrste vojaških računalniških simulacij. Računalniško simulirani nasprotnik oziroma agent lahko na primer nadomesti katero izmed ekip. Tako lahko sodeluje pri ocenjevanju ali celo popolnoma nadomesti sodnika. V to kategorijo spada tudi serija simulacij VBS (angl. Virtual Battlespace) (glej poglavje 1.2.1). Ko agent zamenja obe uporabniški ekipi, lahko simulacija postane v celoti sistemsko vodena in se z minimalnim nadzorom tudi sama upravlja. Absolutna računalniška simulacija lahko deluje ob praktično vsakem času na skoraj vsakem mestu, edino opremo, ki jo potrebuje, je delovna postaja. Takšna simulacija v nasprotju s terensko v enakem času izvede mnogo več iteracij, kar pomeni, da so statistični podatki, izračunani iz njenega modela, verodostojnejši za nadaljnje načrtovanje.

Po drugi strani lahko vojaške simulacije kategoriziramo v dve veliki področji.

Hevristične simulacije so tiste, ki se izvajajo z namenom spodbujanja raziskav in reševanja problemov ter niso nujno empirično rešljive. Pogosto raziskujejo *Kaj pa če?* scenarije, ki poskušajo zagotoviti vpogled v procese odločanja in kriznega upravljanja, pri čimer je njihov glavni namen predložitev konkretnih zaključkov. Te simulacije dejansko ne zahtevajo sklenitve, saj se bo scenarij končal, ko bo doseženo določeno število potez in se bo iztekel dodeljeni čas, ne glede na to, ali je bila prvotna situacija rešena ali ne.

Stohastične simulacije so tiste, ki vsaj delno vključujejo element naključja.

Idealne vojaške simulacije naj bi bile čim realnejše, torej oblikovane tako, da zagotavljajo merljive in ponovljive rezultate, ki so lahko potrjeni z opazovanjem resničnih dogodkov. To še posebej velja za simulacije, ki so stohastične narave, saj se uporabljajo na način, ki je namenjen za pridobitev uporabnih in napovedljivih rezultatov.

Večino vojaških simulacij sodi med ti dve opredelitvi, čeprav se terenske simulacije štejejo bolj med hevristične in računalniške bolj med stohastične. Pri tem je potrebno poudariti, da so simulacije kljub vsemu le približek realnosti in zato točne samo toliko, kot je točen pripadajoči model.

Del simulacije je tudi validacija, ki ima pomembno vlogo pri določitvi stopnje realnosti vojaške simulacije. Predstavlja proces testiranja modela s posredovanjem preteklih podatkov in primerjavo izhodnega rezultata s preteklim. Če lahko model zanesljivo reproducira znane rezultate, je s pomočjo validacije potrjen za veljavnega oziroma nasprotno. Za veljavni model je predpostavljeno, da je sposoben zagotoviti izsledke, ki jih je mogoče z razumno stopnjo negotovosti napovedati.

1.2.1 Živa, virtualna in konstruktivna simulacija

Živa, virtualna in konstruktivna simulacija je široko uporabljena taksonomija za razvrščanje modelov in simulacij [17].

Žive simulacije so vojaški dogodki namenjeni usposabljanju in vključujejo resnične (nesimulirane) udeležence. Le-ti operirajo z dejanskimi sistemi in vsebujejo žive entitete oziroma druge podatke, ki so pridobljeni iz realnega sveta. Obravnavane so kot simulacije, ker posredovanje ni vodeno proti pravemu sovražniku. Če v vaji sodeluje terenski sistem bojevanja, se sistem šteje kot živi udeleženec ne glede na to, ali se nahaja v živem, simuliranem ali deljenem živosimuliranem okolju. Primeri živih entitet so dejanske ladje, letala ali poveljevani vojaki.

Virtualne simulacije zajemajo resnične udeležence, ki operirajo s simuliranimi sistemi ter vsebujejo virtualne entitete in podatke, ki izhajajo iz virtualne

ali modelne predstavitve dejanskega oziroma teoretičnega sistema. Primeri virtualnih simulacij so simulatorji letenja, simulatorji vožnje ali modeli virtualnih sistemov, ki temeljijo na terenski opremi.

Konstruktivne simulacije vključujejo simulirane udeležence, ki operirajo s simuliranimi sistemi. Dejanski uporabniki stimulirajo vhode simulaciji, vendar od njih ni odvisen nadaljnji potek in izid simulacije. Če na primer uporabnik enoti ukaže premik in napad na sovražno tarčo, konstruktivna simulacija določi hitrost premikanja, učinek posredovanja in morebitno škodo. Tovrstne simulacije navadno nadzirajo dejanski uporabniki. Po potrebi lahko prepisujejo programirane akcije in izhode, s čimer lahko inicializirajo ali prevzamejo nadzor na obstoječo konstruktivno entiteto oziroma nadzor nad procesom. Prav tako lahko spreminjajo predprogramirane akcije ali odstranjujejo entitete. Tipična konstruktivna simulacija lahko generira na tisoče entitet.

Virtual Battlespace (VBS)

Virtual Battlespace (VBS) je vojaški simulator, ki spada med virtualne simulacije in temelji na sodobnih igralnih tehnologijah [24]. Sistem je namenjen vadbi vojaških taktik manjših enot v interaktivnem 3D okolju. Platforma omogoča upravljanje scenarijev v realnem času, uporabo prirejenih vozil in opreme, generiranje lastnih scenarijev in koriščenje spremenljivih okoljskih razmer.

Simulator VBS nudi realistične bojne simulacije in upravljanje kopnih, morskih in zračnih prevoznih sredstev. Inštruktorji lahko ustvarijo nove scenarije in nato zaženejo simulacijo iz različnih zornih kotov. Sistem vodenja oddelkov omogoča, da uporabniki izdajo ukaze članom oddelkov, obenem pa tudi koordinirajo raznovrstne naloge. Simulator omogoča prosto igro znotraj scenarijev, ki temeljijo na vadbenih misijah in vključuje simulacijo vetra, dežja, megle, oblakov, delov dneva in plimovanja. Simulator VBS lahko deluje preko lokalnega omrežja ali interneta in je lahko nameščen na prenosnih ali namiznih računalnikih. Simulator zagotavlja sintetično okolje za praktično usposabljanje vodstvenih in organizacijskih sposobnosti, ki so potrebne za izvedbo misij

manjših enot.

Simulator VBS podpira velike večigralske omrežne seje, pri čimer omogoča pridruževanje novih igralcev, medtem ko je simulacija v teku. Omogoča destrukcijo stavb in penetracijo streliva skozi stene. Orožne platforme so sposobne toplotnega zaznavanja, simulacijo požarnih nadzornih sistemov in pregrevanja orožnih cevi. Velik poudarek je na pravilnosti orožne balistike.

Modul *After-Action-Review* omogoča podrobno analizo končane misije, saj beleži vse premike entitet in vozil, kakor tudi poti izstrelkov in destrukcijo objektov ali terena.

Prehod DIS/HLA (angl. High Level Architecture) povezuje virtualni simulator z drugimi vojaškimi simulacijami, uporablja pa se tudi za povezovanje med več strežniki VBS.

V sklopu serije so bile do sedaj izdane tri večje verzije simulacije, in sicer VBS1, VBS2 in VBS3. Tudi Slovenska vojska (SV), podobno kot večino držav zveze NATO (angl. North Atlantic Treaty Organisation), uporablja simulator VBS. Trenutno SV izvaja virtualno vadbo z verzijo VBS2, vendar nameravajo pridobiti licenco za uporabo najnovejše različice.

Joint Conflict and Tactical Simulation (JCATS)

Joint Conflict And Tactical Simulation (JCATS) je skupni, večstranski, stohastični, visokoresolucijski in interaktivni simulacijski sistem, ki modelira interakcije med vojaškimi silami [8, 9]. V interakcijah sodelujejo različno velike enote, od delovnih skupin do individualnih oseb. Sistem JCATS je primer konstruktivne simulacije, ki se izvaja na osnovi fizikalnih zakonov in vhodnih podatkov. Sistem natančno posnema bojevanje, ki poteka na ravni entitet. Simulacija uporablja enak teren, kot je uporabljen pri vojaških zemljevidnih sistemih poveljevanja in nadzora (angl. Command And Control - C2) ter izkorišča preverjene vojaške podatke in algoritme za generiranje digitalnega terena in modeliranje vojaškega osvajanja. Sistem JCATS natančno simulira vojaško opazovanje, senzorske sisteme, orožje, strelivo in prevozna sredstva na

kopnem, na morju in v zraku. Model simulacije je sestavljen iz posameznih entitet. Vsaka entiteta ima svoje lastnosti, kot je na primer individualna logistika ali stopnja zdravja, usposobljenosti in utrujenosti. Nadalje so lahko del entitete tudi individualno orožje, različne platforme, senzorje in strelivo. Entitete so kompatibilne s standardi DIS in HLA, kar olajšuje povezovanje z drugimi vojaškimi simulacijskimi modeli.

Sistem je interno razdeljen na tri večje operacijske podskupine, in sicer na kopno, zrak in morje. Na kopnem simulacija podpira manevriranje, posredni in neposredni ogenj, zračno obrambo, inženirske in logistične operacije ter vzdrževanje. V zraku sistem zagotavlja prestrezanje, zračno podporo in spremstvo, delovanje proti zračni obrambi, oskrbo z gorivom, zračni most, spuščanje enot iz zraka in zračni nadzor. Na morju je mogoča ognjena podpora, amfibijske operacije, podmorniško in protipodmorniško bojevanje ter pomorsko letalstvo.

Programska oprema JCATS teče na operacijskem sistemu *Red Hat Linux Enterprise* in se lahko izvaja na namiznih ali prenosnih računalnikih. Sistem omogoča dinamični nadzor zmogljivosti in dejavnosti entitete v bojnem prostoru ter interpretira vplive vseh sistemov in taktik na bojišču. V osnovi je sistem simulacija človek na človeka, kjer živi nasprotniki posnemajo taktike in odzive sovražnika. Vodje nadzorujejo svoje enote na individualnih delovnih postajah in preko komunikacijskega sistema poročajo svojim poveljnikom oziroma poveljstvu. Sistem tekom vaje ustvari podroben prikaz, ki je sestavljen iz simultanih individualnih in agregiranih entitet.

Sistem živih simulacij SAAB

Sistem živih simulacij SAAB je primer žive vojaške simulacije. Sistem zajema testiranje natančnosti orožja, preverjanje psihofizičnih sposobnosti, taktično usposabljaje in trening živega ognja [11].

Živi laserski trenažni sistem oziroma sistem TES (angl. Tactical Engagement Simulation) je sistem, ki se za razliko od konstruktivnih in virtualnih

simulatorjev uporablja za usposabljanje vojaških in častniških enot. Sistem TES je sestavljen iz dveh podsistemov. Prvi podsistem nadzoruje laserski oddajnik, sprejemnik in sistem, ki ocenjuje in določa učinke orožja. Drugi podsistem upravlja sistem identifikacije in sledenja. Vaje lahko potekajo bodisi znotraj območja trenažnega centra, ki ima stalno infrastrukturo komunikacij in inštrumentov, bodisi z uporabo prenosljivega podatkovno-komunikacijskega sistema, ki zagotavlja enake pogoje zunaj območij trenažnih centrov.

Osnovni segment sistema TES je laserski oddajnik, ki je nameščen na orožje. Ko je orožje sproženo, oddajnik odda laserski žarek, ki očem ni nevaren. Če oddani žarek naleti na laserski sprejemnik, ta sproži odziv glede na vrsto prejetega žarka. V laserskem žarku je shranjena informacija o tipu streliva in identiteta strelca, kar sprejemnik prebere in ustrezno obravnava. Ciljane entitete znotraj prizadetega območja ocenijo učinek orožja in z uporabo modela ranljivosti določijo povzročeno škodo.

Vsak vojak nosi osebno zaznavno napravo (angl. Personal Detection Device - PDD), ki je sestavljena iz jopiča in sprejemnika, nameščenega na čelado. Kontrolni modul naprave PDD z zaslonom in zvočnikom sporoča različna opozorila, kot so uboj, ranjenje, bližnjo zgrešitev in topniški ogenj. Med laserskim oddajnikom, nameščenim na orožje, in napravo PDD je vzpostavljena brezžična povezava, ki omogoča sprožitev orožja samo v primeru, ko je vojak v simulaciji živ oziroma poškodovan samo do določene stopnje.

Na bojnih vozilih oziroma plovilih je laserski oddajnik nameščen na glavnem orožju in integriran v sistem za nadzor ognja. Tako so vse pomembne informacije, med katere spadata tudi določitev lastnosti laserja in izbira streliva, poslane sistemu za nadzor.

Preko radijskega omrežja naprava PDD pošilja podatke glavnemu računalniku, s pomočjo katerega poveljujoči nadzoruje gibanje enot.

Poglavje 2

Orodja in standardi

V tem poglavju so predstavljena orodja in standardi, ki so bili uporabljeni pri izdelavi diplomskega dela in pri implementaciji aplikacije za vizualizacijo stanja na sodobnem bojišču. Najprej je predstavljen geografski informacijski sistem, zatem katografske projekcije, standard DIS in simbologija zveze NATO ter na koncu še pogon *Unity*.

2.1 Geografski informacijski sistem

Geografski informacijski sistem (angl. Geographic Information System - GIS) je računalniški sistem za zajem, shranjevanje in prikaz vseh vrst prostorskih oziroma geografskih podatkov [16]. Poleg osnovnih funkcionalnosti omogoča še manipulacijo, analizo in upravljanje podatkov.

V splošnem opisuje poljubni informacijski sistem, ki integrira, shranjuje, ureja, obdeluje, izmenjuje in prikazuje geografske podatke. Aplikacije, ki temeljijo na sistemu GIS, so orodja, ki uporabnikom omogočajo kreiranje interaktivnih poizvedb, analiziranje prostorskih informacij, urejanje podatkov na mapah ter vizualizacijo. Sistem sestavljajo tehnologije in metode, ki predstavljajo osnovo lokacijskih storitev, saj so te odvisne predvsem od analize in vizualizacije.

Sistem omogoča povezovanje nepovezanih informacij, pri čimer za indeksiranje uporablja lokacijo. Lokacije oziroma prostorsko-časovni zapisi so predstavljeni kot strukture in so sestavljeni iz časovnega žiga nastanka zapisa ter x, y in z koordinat, ki predstavljajo geografsko širino, dolžino in nadmorsko višino. V idealnem primeru naj bi bili vsi prostorsko-časovni zapisi, zajeti na Zemlji, medsebojno povezani in naj bi predstavljali dejanske fizične lokacije.

Sistem GIS torej uporablja prostorsko-časovno lokacijo za indeksiranje informacij. S pomočjo indikacijske spremenljivke sistem zgradi relacijsko podatkovno bazo, ki vsebuje nize oziroma numerične vrednosti in združi drugače nepovezane podatke. Spremenljivka je sestavljena iz lokacije in/ali mere v prostorskem času.

Natančnost sistema GIS je pogojena s kvaliteto vhodnih podatkov ter z načinom njihovega referenciranja. Podatki sistema GIS predstavljajo realne subjekte, kot so na primer cestno omrežje, raba tal, nadmorska višina ali vodotoki, pri katerih so digitalni podatki osnova za reprezentacijo. Subjekti so razdeljeni v dve abstraktni množici, in sicer na diskretne objekte (npr. stavbe, ceste) ter opisna področja (npr. količina padavin, gostota prebivalstva). Rastrske slike in vektorski podatki sta dve metodi shranjevanja podatkov v sistemu GIS, namenjeni obema abstraktnima skupinama. Za referenciranje preslikanih lokacijskih atributov se uporabljajo točke, črte in poligoni.

Da so lahko vektorski podatki uporabljeni za napredne raziskave, morajo biti topološko pravilni. Topološka pravilnost je običajno dosežena z urejanjem podatkov, potem ko so ti že vneseni v sistem GIS.

Sistem GIS omogoča prestrukturiranje podatkov v različne formate. Naprednejše obdelave podatkov zajemajo obdelavo slik, lažno barvno upodabljaje, uporabo 2D Fourierovih transformacij in ostalih višje stopenjskih tehnik za manipulacijo podatkov.

Ker se digitalni podatki zbirajo in shranjujejo na različne načine, se lahko zgodi, da dve strukturi podatkov nista povsem združljivi, zato je pomembno, da je sistem sposoben pretvarjanja prostorskih podatkov med različnimi struk-

turami.

2.1.1 Svetovni geodetski sistem WGS 84

Svetovni geodetski sistem (angl. World Geodetic System - WGS) je standard za uporabo v kartografiji, geodeziji in navigaciji [2, 27]. Sistem združuje standardni koordinatni sistem za Zemljo, standardno sferoidno referenčno površino (datum ali referenčni elipsoid) za neobdelane višinske podatke in gravitacijsko izenačeno površino (geoid), ki opredeljuje nominalno morsko površino.

Zadnja različica sistema je WGS 1984, EPSG:4326 (angl. European Petroleum Survey Group), ki je bila uveljavljena leta 1984 ter revidirana leta 2004. Starejše različice sistema so bile WGS 72, WGS 66 in WGS 60 ter so bile uveljavljene v pripadajočih letih.

Center koordinatnega sistema WGS 84 se nahaja v središču mase Zemlje, pri čemer je napaka manjša od 2 cm. V sistemu WGS 84 ničelni poldnevnik ustreza referenčnemu poldnevniku mednarodne storitve za rotacijo Zemlje (angl. International Earth Rotation and Reference Systems Service - IERS), 5.31 kotnih sekund oziroma 102.5 m vzhodno od poldnevnika Greenwich na geografski višini kraljevega observatorija v Greenwichu. Datumska površina WGS 84 je sploščeni sferoid (elipsoid), ki je definiran z ekvatorialnim polmerom (angl. major/equatorial radius) in uravnavanjem (angl. flattening), s pomočjo katerih lahko izračunamo tretji prameter - polarni polmer.

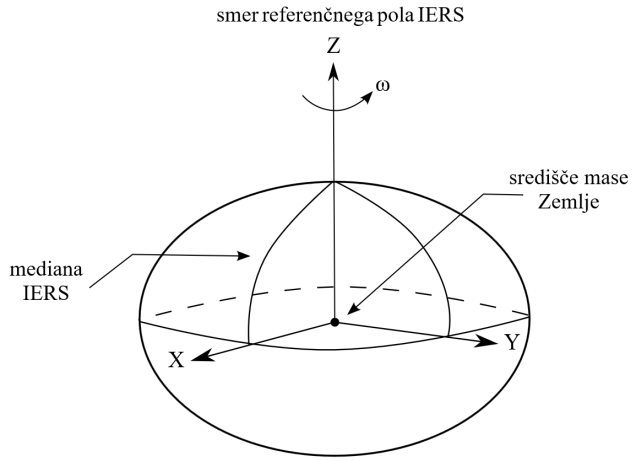
WGS 84 uporablja geoid zemeljskega gravitacijskega modela iz leta 1996 (angl. Earth Gravitational Model 1996 - EGM96), ki je bil revidiran leta 2004. Ta geoid opredeljuje nominalno morsko površino s sferično harmonsko serijo stopnje 360, ki zagotavlja resolucijo v približni vrednosti 100 km zemljepisne širine na ekvatorju. Odstopanja geoida EGM96 od referenčnega elipsoida se nahajajo v območju od -105 m do približno $+85$ m. Zdajšnji geoid EGM96 se razlikuje od originalnega geoida iz leta 1984, saj ima za približno 100 km višjo resolucijo (prvotni 200 km, zdaj 100 km).

V spodnji tabeli 2.1 so navedeni osnovni parametri Zemljinega elipsoida.

	ekvatorialni polmer a	polarni polmer b	obratno uravnavanje *(1/f)*
WGS 84	6378137.0 m	$\approx$ 6356752.314245 m	298.257223563

Tabela 2.1: Osnovni parametri Zemljinega elipsoida

WGS 84 je torej Zemlje-središčen kopenski referenčni sistem in geodetski datum. Bazira na konsistentnem nizu konstant in modelnih parametrov, ki opisujejo Zemljino velikost, obliko ter gravitacijsko in geomagnetsko polje. Izhodišče koordinatnega sistema služi kot geometrični center elipsoida, medtem ko os Z predstavlja rotacijska os taistega elipsoida, s pomočjo katerega so generirane geodetske koordinate. V spodnjem seznamu so našteti elementi, ki definirajo okvir standarda.



Slika 2.1: Svetovni geodetski sistem WGS 84 [26]

Izhodišče

Središče mase Zemlje, ki je opredeljena za celotno Zemljo in vključuje tudi oceane in atmosfero.

Osi

Z - os Smer referenčnega pola IERS, ki ustreza smeri konvencionalnega kopenskega pola (angl. Conventional Terrestrial Pole - CTP) med-

narodnega časovnega urada (franc. Bureau International de 'Heure - BIH).

X - os Presečišče referenčne mediane IERS in ravnine, ki poteka skozi izhodišče in normalo na Z-os.

Y - os Dokončuje desnosučni, Zemeljsko-središčni pravokotni koordinatni sistem.

Obseg

Zajema lokalni Zemeljski okvir, v smislu relativistične teorije gravitacije in je poravnan z ITRS (angl. International Terrestrial Reference System).

Orientacija

Orientacija BIH - desnosučna.

Evolucija časa

Napredovanje časa ne ustvarja odvečne globalne rotacije.

2.1.2 Format DTED

Format DTED (ang. Digital Terrain Elevation Data) je standard digitalnih podatkovnih nizov, ki je sestavljen iz matrike oziroma digitalnega modela reliefa, v katerem so shranjene nadmorske višine terena [15]. Nadmorske višine so opisane kot višine nad geoidom EGM96 in ne kot višine nad referenčnim elipsoidom WGS84.

Standard je predpisala agencija NGA (angl. National Geospatial-Intelligence Agency), ki je del Ministrstva za obrambo Združenih držav Amerike s sedežem v Fort Belvoirju v Virginiji.

Format DTED podpira veliko aplikacij, vključno s profiliranjem terena, 3D vizualizacijo terena, načrtovanjem oziroma vadbo misij, modeliranje in simulacijo. Zagotavlja kvantitativne podatke srednje resolucije, ki so zapisani v digitalnem formatu. Digitalni podatki so namenjeni vojaškim sistemskim aplikacijam, ki za svoje delovanje zahtevajo nadmorske višine površja.

Format DTED je standardiziran za stopnje 0, 1 in 2, ki so opisane v ameriškem vojaškem standardu. Format opisuje več parametrov, med najpomembnejšimi je resolucija. Pri stopnji 0 je razmik med posameznimi nadmorskimi višinami približno 900 m, pri stopnji 1 približno 90 m in pri stopnji 2 približno 30 m. Predlagane so bile tudi stopnje 3, 4 in 5, vendar še niso standardizirane.

Podatki formata DTED so shranjeni s predznačenimi števili po pravilu debelega konca (angl. Big Endian Rule), kjer je najpomembnejši bajt shranjen na najnižjem naslovu.

Projekt SRTM

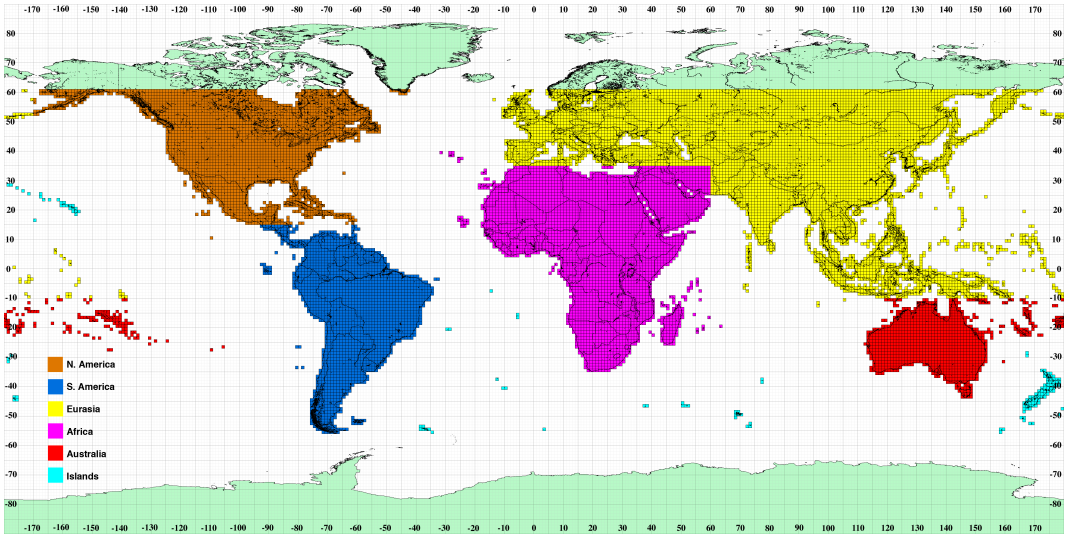
Projekt SRTM (angl. Shuttle Radar Topography Mission) je bil mednarodni raziskovalni projekt [21], ki sta ga vodili agencija NGA in agencija NASA (angl. National Aeronautics and Space Administration). Glavni cilj projekta je bila pridobitev digitalnega modela nadmorskih višin večine Zemljinega površja. Tekom projekta je bila ustvarjena visoko-resolucijska digitalna topografska baza podatkov. Baza zajema vrednosti nadmorskih višin, ki so del Zemljinega plašča v obsegu 56 stopinj južno od ekvatorja do 60 stopinj severno od ekvatorja. Zemeljsko površje je bilo izmerjeno s posebej izdelanim radarskim sistemom, ki je bil nameščen na raketoplan Endeavour.

Modeli nadmorski višin, ki so v bazi SRTM, so razporejeni v ploščice, od katerih vsaka pokriva eno stopinjo geografske širine in eno stopinjo geografske dolžine. Resolucija neobdelanih podatkov je ena kotna sekunda (približno 30 m), kar ustreza drugi stopnji formata DTED. Ta različica je bila izdana samo za ozemlje ZDA, medtem ko so za ostali svet na voljo podatki v resoluciji treh kotnih sekund, kar ustreza prvi stopnji formata DTED. Vsaka ploščica z eno kotno sekundo je sestavljena iz 3601 vrstice, vsaka vrstica vsebuje 3601 16-bitno celico. Dimenzija ploščice z resolucijo treh kotnih sekund ustreza kvadratni matriki z velikostjo stranice 1201. Originalne nadmorske višine SRTM so bile izračunane glede na elipsoid WGS 84 in nato konvertirane v višine nad

geoidom EGM96.

Modeli nadmorskih višin, ki izhajajo iz podatkov SRTM, se uporabljajo v geografskih informacijskih sistemih. Na svetovnem spletu so podatki SRTM prosto dostopni in shranjeni v formatu DTED s končnico *hgt*. Leta 2009 je model zamenjal novejši model ASTER GDEM (angl. Global Digital Elevation Model).

Na sliki 2.2 je prikazano področje projekta SRTM in obseg površja, ki ga projekt zajema.



Slika 2.2: Obseg projekta SRTM [13]

2.2 Kartografske projekcije

Kartografska projekcija je matematični postopek, ki omogoča preslikavo zakrivljene površine v ravnino, pri čimer lahko zakrivljeno površino predstavlja bodisi sfera bodisi rotacijski elipsoid [18]. Uporablja se predvsem za preslikavo Zemlje in drugih nebesnih teles. Cilj študij kartografskih projekcij je kreiranje matematične osnove za izdelavo zemljevidov, ki se nadalje uporabljajo v kartografiji, geodeziji, geografiji, navigaciji, računalništvu in v ostalih sorodnih znanostih. Projekcije se uporabljajo za prikazovanje dela elipsoida oziroma sfere ali pa za prikaz celotne površine telesa. Pri tem je pomembno, da je stopnja deformacije oziroma popačenje čim manjše.

Med pomembnejše lastnosti elipsoida oziroma sfere spadajo razdalja, oblika, smer, azimut in merilo. Kartografska projekcija je zasnovana tako, da ohrani vsaj eno od zgornjih lastnosti, vendar zgolj v omejenem obsegu. Vsaka projekcija ohranja zgornje lastnosti različno.

Glavna naloga kartografske preslikave je vzpostavitev odvisnosti med koordinatami na elipsoidu oziroma sferi in koordinatami njihovih transformacij na ravnini. Ta odvisnost je običajno izražena z enačbama:

$$x = f_1(\varphi, \lambda) \tag{2.1}$$

$$y = f_2(\varphi, \lambda) \tag{2.2}$$

pri čemer φ predstavlja zemljepisno širino, λ zemljepisno dolžino. x in y predstavljata pravokotni koordinati v ravnini projekcije.

Točke na površini elipsoida oziroma sfere so določene s presekom poldnevnikov in vzporednikov ter predstavljajo osnovno kartografsko mrežo v projekcijski ravnini.

Kartografska projekcija je sistematična preslikava zemljepisne širine in dolžine posameznih lokacij iz površine elipsoida oziroma sfere na ravnino oziroma karto. Vse kartografske projekcije s pomočjo algoritma, z določenim sprejemljivim ali nesprejemljivim odstopanjem, izkrivljajo površino krogle. Zaradi

stopnje odstopanja in načina translacije obstaja veliko različnih kartografskih projekcij, ki različno ohranjajo lastnosti krogelnih oziroma elipsoidnih teles.

Kartografske projekcije glede na vrsto deformacije delimo v tri sklope, in sicer na konformne ali enakokotne projekcije, ki ohranjajo velikosti kotov, ekvivalentne projekcije, ki ohranjajo površine in razmerja med površinami, ter ekvidistančne projekcije, ki ohranjajo dolžine in usmerjenost vektorjev dolžin.

Projekcije lahko delimo tudi glede na obliko projekcijske ravnine.

Valjna projekcija (angl. cylindrical projection) je vsaka projekcija, pri kateri so poldnevniki predstavljeni kot enakomerno razporejene navpičnice, medtem ko so vzporedniki preslikani v vodoravne črte. Takšno projekcijo predstavlja valj, katerega os sovpada z osjo vrtenja elipsoida oziroma sfere. Polmer valja je enak polmeru projeciranega telesa.

Stožčna projekcija (angl. conic projection) je vsaka projekcija, pri kateri so poldnevniki preslikani na enakomerno razporejene črte, ki imajo skupno navidezno izhodiščno točko v vrhu stožca. Vzporedniki se preslikajo v krožne loke s središčem v vrhu stožca.

Azimutna projekcija (angl. azimuthal projection) je vsaka projekcija, pri kateri so ohranjene razdalje do osrednje točke. Primer azimutne projekcije je tangentna ravnina, ki se dotika Zemlje v točki T. V tem primeru tangentna točka T predstavlja osrednjo točko projekcije.

2.2.1 Mercatorjeva projekcija

Mercatorjeva projekcija je konformna projekcija, pri kateri je površje elipsoida preneseno na ravnino s pomočjo valja, zaradi česar jo uvrščamo med valjaste projekcije [19].

Tako kot pri vseh valjastih projekcijah so poldnevniki in vzporedniki predstavljeni kot ravne črte, pravokotne druga na drugo. Posledica tega je raztezanje karte v smeri vzhod-zahod. To raztezanje so povečuje sorazmerno s povečanjem oddaljenosti od ekvatorja in povzroči raztezanje na severu oziroma jugu. Z Mercatorjevo projekcijo ni mogoče prikazati polarnih območij, saj je

vrednost linearnega merila na obeh polih neskončna. Ker je projekcija konformna, se velikosti kotov ohranjajo, medtem ko se merilo spreminja glede na lokacijo. Spreminjanje merila izkrivlja velikost geografskih objektov, kar je še posebej opazno na območjih blizu polov. Na geografskih širinah 70° severno ali južno od ekvatorja je Mercatorjeva projekcija praktično neuporabna.

Naslednji enačbi definirata Mercatorjevo projekcijo:

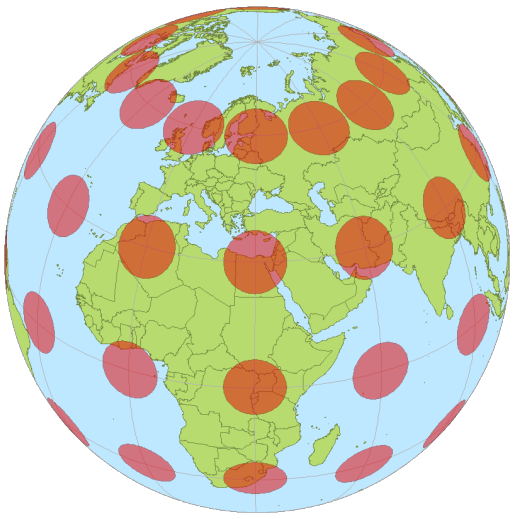
$$x = \lambda_0 - \lambda \tag{2.3}$$

$$y = \ln\left(\tan\left(\frac{\pi}{4} + \frac{\varphi}{2}\right)\right) \tag{2.4}$$

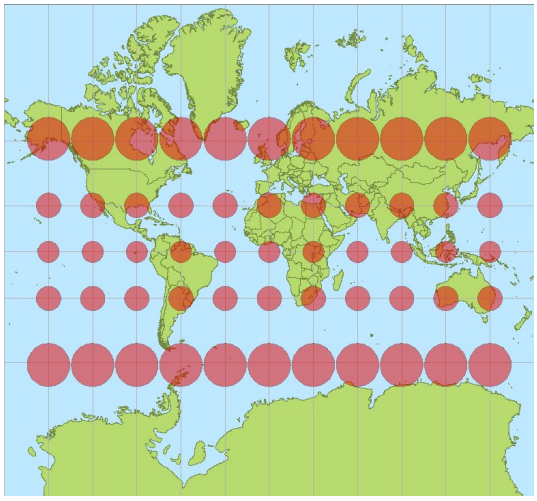
kjer sta x in y koordinati na projekcijski ravnini Mercatorjeve projekcije pri geografski širini φ in geografski dolžini λ. λ_0 označuje geografsko dolžino središča zemljevida.

Kot je razvidno iz enačbe 2.4, se koordinata y približuje $\pm\infty$, ko se geografska širina približuje poloma, kjer je $\varphi = \pm 90°$.

Distorzijo projekcije prikazujemo s pomočjo Tissotove elipse izkrivljanja, ki prikazuje popačenost lokalnih območij dane projekcije. Na spodnjih slikah sta prikazani Tissotovi elipsi izkrivljanja sfere (2.3) in Mercatorjeve projekcije (2.4).



Slika 2.3: Tissotova elipsa izkrivljanja sfere [23]



Slika 2.4: Tissotova elipsa izkrivljanja Merkatorjeve projekcije [22]

Spletna Mercatorjeva projekcija

Spletna Mercatorjeva projekcija je različica Mercatorjeve projekcije, ki se uporablja predvsem v spletnih programih za kartiranje [25]. Uporablja enake formule kot standardna Mercatorjeva projekcija pri zemljevidih malega merila. Standardno Mercatorjeva projekcija uporablja elipsoidno obliko projekcije, medtem ko se pri spletni Mercatorjevi uporabljajo sferične formule. V globalnem merilu je razlika med projekcijama neopazna, vendar povzroči, da zemljevidi lokalnih območij nekoliko odstopajo od resničnih elipsoidnih Mercatorjevih zemljevidov pri enakem merilu. Odstopanje postane izrazitejše dlje od ekvatorja in lahko doseže tudi do 35 km.

Spletno Mercatorjevo projekcijo uporabljajo skoraj vsi večji spletni ponudniki zemljevidov, vključno z *Google Maps*, *Bing Maps*, *OpenStreetMap*, *MapQuest* in *ESRI*.

Ker so formule spletne Mercatorjeve projekcije v sferični obliki, morajo biti geografske koordinate v elipsoidnem datumu WGS 84. Ta razlika povzroča, da je projekcija nekoliko nekonformna, kar pomeni, da se velikosti kotov ne ohranjajo popolnoma.

Formule za spletno Mercatorjevo projekcijo so v osnovi enake kot za standardno sferično Mercatorjevo, vendar so pred upoštevanjem povečave koordinate sveta prilagojene tako, da je zgornji levi kot enak (0, 0) in spodnji desni enak (256, 256):

$$x = \frac{128}{\pi} 2^z (\lambda + \pi) \text{ pikslov} \tag{2.5}$$

$$z = \frac{128}{\pi} 2^z \left(\pi - \ln\left[\tan\left(\frac{\pi}{4} + \frac{\varphi}{2}\right)\right]\right) \text{ pikslov} \tag{2.6}$$

kjer λ predstavlja zemljepisno dolžino v radianih, φ zemljepisno širino in z stopnjo povečave.

Ker projekcija projektira pola v neskončnost, prikaz polov ni mogoč. Tako je možno prikazati samo območje do 85.051129° severno in južno. Vrednost 85.051129° je zemljepisna širina, pri kateri karta v celoti postane kvadrat in je izračunana kot φ pri katerem je $y = 0$.

$$\varphi_{max} = \left[2 \arctan(e^{\pi}) - \frac{\pi}{2}\right] \tag{2.7}$$

Spletna Mercatorjeva projekcija si s standardno deli kar nekaj lastnosti; sever je povsod zgoraj, poldnevniki so enakomerno razporejene navpičnice in območja v bližini polov so zelo povečana.

Za razliko od elipsoidne in sferične Mercatorjeve projekcije, spletna Mercatorjeva ni čisto konformna, ker uporablja elipsoidne geografske koordinate s sferično projekcijo. Loksodrome oziroma krivulje, ki sekajo vse poldnevnike pod istim kotom pri spletni Mercatorjevi, niso ravne. Glavna prednost projekcije je optimalnost in računska preprostost.

2.3 Standard DIS

Distributed Interactive Simulation (DIS) je IEEE (angl. Institute of Electrical and Electronics Engineers) standard namenjen izvajanju realnočasovnih vojaških simulacij [1]. Standard DIS je časovno-prostorsko skladna sintetična predstavitev svetovnih okolij. Namenjen je povezovanju interaktivnih prosto-igralnih dejavnosti udeležencev v operativne vaje. Sintetično okolje je ustvarjeno skozi izmenjavo protokolnih podatkovnih enot med porazdeljenimi, računsko samostojnimi simulacijskimi aplikacijami, simulacijami in merilnimi inštrumenti, pri čimer izmenjava podatkovnih enot poteka v realnem času. Entitete simulacije so lahko prisotne na enem mestu ali geografsko porazdeljene.

Arhitektura standarda DIS običajno nima centralnega računalnika, ki bi nadzoroval celotno simulacijsko vajo, vendar nekateri simulacijski sistemi imajo centralni računalnik. V tovrstnih sistemih centralni računalnik ohranja stanje sveta in sprotno izračunava učinke vsake akcije posamezne entitete na ostale subjekte in okolje. Standard DIS uporablja porazdeljen simulacijski pristop, kar pomeni, da za simuliranje stanja vsake entitete skrbijo ločene simulacijske aplikacije, ki se nahajajo na omrežno-povezanih, gostiteljskih računalnikih.

Avtonomne simulacijske aplikacije so odgovorne za ohranjanje stanja ene ali več entitet. Ko uporabnik upravlja s kontrolami simulacijske ali dejanske opreme, je simulacija odgovorna za modeliranje posledic dejanj entitete, ki jih interpretira s pomočjo simulacijskega modela. Prav tako pošilja sporočila ostalim simulacijam, v katerih obvešča o pomembnih aktivnostih, ko je to potrebno. Vse simulacije tolmačijo in so odzivajo na sporočila drugih simulacij ter vzdržujejo modele stanj entitet, ki so zastopane v simulacijski vaji.

Vsaka simulacijska aplikacija preko protokola posreduje stanje entitete, ki jo nadzoruje oziroma meri (lega, orientacija, hitrost, ...) ostalim simulacijam v omrežju. Simulacija, ki je v vlogi odjemalca, je dolžna uporabiti prejeto stanje in izračunati, če lahko entiteto zazna z vizualnimi oziroma elektronskimi sredstvi.

Za zmanjševanje komunikacijskega procesiranja je uporabljen algoritem dead reckoning, ki je postopek ocenjevanja položaja in usmerjenosti subjekta. Standard algoritem uporablja za optimizacijo hitrosti osveževanja stanja entitete. Vsaka simulacija za entiteto, ki jo upravlja, vzdržuje notranji model in model algoritma dead reckoning. Model algoritma dead reckoning predstavlja položaj entitete v drugih simulacijah v omrežju in je dejansko preslikava položaja in orientacije entitete. Simulacija redno primerja oba modela, in če razlika med njima preseže v naprej določeno mejo, simulacija s pomočjo podatkov notranjega modela, posodobi model algoritma dead reckoning. Ko je model posodobljen, simulacija pošlje nove informacije ostalim simulacijam v omrežju, ki ob prejemu posodobijo svoj model algoritma dead reckoning za pripadajočo entiteto. Z uporabo algoritma dead reckoning simulaciji ni potrebno tako pogosto poročati o statusu svojih entitet in osveževati svojih modelov za tuje entitete. Tako simulacija prihrani na procesorski moči.

2.3.1 Družine PDU

Standard DIS opredeljujejo podatkovne enote PDU (angl. protocol data unit), ki se izmenjujejo v omrežju med simulacijskimi aplikacijami. Enote so razdeljene na področja oziroma protokolarne družine.

Informacije/interakcije entitete

V funkcionalnem področju informacije/interakcije entitete (angl. Entity Information/Interaction) so navedene enote PDU, ki nosijo osnovne informacije o entitetah in trkih entitet. Entiteta je fizični objekt v sintetičnem okolju, ki ga ustvari in nadzira simulacija. Entiteta je lahko živi, virtualni ali konstruktivni subjekt. Primeri entitet so na primer tanki, ladje, letala, stavbe in življenjske oblike (ljudje in živali). Informacije o entiteti so posredovane ob inicializaciji, spremembi stanja in v časovnih intervalih. Namenjene so podpori simulacij z različnimi nivoji resničnosti ter vizualnim, zvočnim in senzorskim

modelom. Trenutno podprte interakcije trka zajemajo elastične in neelastične trke. Ob trku dveh entitet mora biti simulacija, ki ju nadzira, obveščena o trku. Obveščanje je implementirano tako, da vsaka simulacija pošlje sporočilo o trku, ko zazna, da je njena entiteta trčila z drugo. Prav tako vsaka simulacija določi škodo lastne entitete, pri čemer uporabi podatke iz sporočila o trku.

Vojskovanje

V protokularno družino vojskovanje (angl. Warfare) spadajo enote PDU, ki nosijo osnovne informacije o bojevanju. Enote PDU podpirajo streljanje orožja, detonacijo streliva, simulacijo eksplozij, ki niso posledica streliva ter izračun in širjenje škodljivih učinkov. Ko entiteta sproži orožje, simulacija, ki jo nadzoruje, posreduje informacije o eksploziji ostalim simulacijam, ki bi podatek morda potrebovale. Tudi detonacijo streliva in ostalih eksplozij posreduje simulacija, ki zgornje detonacije nadzira. Ostale simulacije, ki nadzirajo prizadete entitete, s pomočjo podatkov v sporočilu detonacije ocenijo škodo lastnih entitet. Posledice streljanja z orožjem, trkov in ostalih virov škode so posredovane preko sporočil o statusu škode entitete.

Logistika

Enote PDU, ki v standardu predstavljajo logistiko (angl. Logistics), v simulacijski vaji modelirajo popravilne, oskrbovalne in logistične storitve. Enote predstavljajo zahtevke za storitve in prenos zalog ter se izmenjujejo med simulacijami, ki so ponudniki zgornjih storitev, ter med simuliranimi entitetami, ki tovrstne storitve potrebujejo.

Upravljanje simulacije

Družina, ki pokriva upravljanje simulacije (angl. Simulation Management), vsebuje enote PDU, ki so koriščene za upravljanje vaje in razbremenjujejo delovanje vadbenega omrežja. Funkcije protokolarne družine so razdeljene na

upravljanje omrežja in upravljanje simulacije.

Porazdeljena regeneracija oddajnikov

Funkcionalno področje porazdeljena regeneracija oddajnikov (angl. Distributed Emission Regeneration) pokriva simulacijo laserskih sistemov, aktivnih elektromagnetnih oddajnikov in aktivnih akustičnih oddajnikov. Med aktivne elektromagnetne oddajnike spadajo radarji in sistemi elektronske identifikacije in nadzora, medtem ko so primer aktivnih akustičnih oddajnikov sonarni sistemi.

Radijske komunikacije

Avdio in digitalnosporočilne komunikacije, ki spadajo v funkcionalno področje radijske komunikacije (angl. Radio Communications), imajo pomembno vlogo pri vajah DIS. Entiteta, ki oddaja, najprej pošlje podrobnosti o komunikacijski napravi, in nato sporočilo, ki vsebuje glasovne ali digitalne podatke. Entiteta, ki sprejema, s pomočjo prvega sporočila določi, ali je zmožna interpretirati prejete podatke. Če je entiteta zmožna interpretacije podatkov, s pomočjo prvega sporočila določi način procesiranja in drugo sporočilo prebere.

Upravljanje entitet

Funkcionalno področje upravljanje entitet (angl. Entity Management) je namenjeno podpori večjih vaj DIS, saj zagotavlja mehanizme, ki omogočajo agregacijo oziroma grupiranje entitet med vajo ter poročajo o stanju grup namesto o stanju posameznih entitet. V standardu so vsebovani posebni protokoli, ki podpirajo agregacijo, prav tako pa lahko prenašajo lastništvo entitete med simulacijami v omrežju.

Minska polja

V protokolarni družini minska polja (angl. Minefield) so zajete enote PDU, ki podpirajo simulacijo min in minskih polj. Izmenjava informacij o minah in minskih poljih se izvaja bodisi v časovnih intervalih ali kot odziv na poizvedbo.

Sintetično okolje

V funkcionalnem področju sintetično okolje (angl. Synthetic Environment) so navedene enote PDU, ki podpirajo simulacijo neentitetnih objektov sintetičnega okolja in simulacijo okolja povezanega s terenom, vesoljem ali vodo. Med ne-entitetne objekte spadajo vreme, dnevni učinki ter naravne in človeške motnje (tj. izbruhi vulkanov, potresi, prah in dimni oblaki iz vozil ali eksplozij). Enote PDU nosijo informacije o spremembi objektov sintetičnega okolja.

Upravljanje simulacij z zanesljivostjo

Enote PDU znotraj funkcionalnega področja upravljanje simulacij z zanesljivostjo (angl. Simulation Management with Reliability) opravljajo enako naloge kot enote PDU družine Simulation Management, vendar poleg tega ta družina določa mehanizme za zanesljivo komunikacijo. Zanesljivost komunikacije je nujna za izvajanje kritičnih nalog upravljanja, ki ne smejo biti odvisne od izgub enot PDU.

Informacijske operacije

Informacijske operacije (angl. Information Operations - IO) podpirajo interoperabilnost simuliranega elektronskega bojevanja, operacije računalniškega omrežja (angl. Computer Network Operations - CNO) ter vojaške prevare oziroma podobne postopke, ki lahko vplivajo na odločanje sovražnika. Standard DIS posreduje podrobnosti o napadih IO v sporočilih IO action, ki lahko vsebujejo tudi pričakovane posledice napada. Dejanske posledice napada so posredovane v sporočilih IO report.

Nerealnočasovni protokoli

Večina vaj DIS poteka s sodelovanjem uporabnika, oziroma simulacije zaradi katerega drugega kriterija zahtevajo, da čas simulacije napreduje enako hitro kot realni svetovni čas. Standard DIS podpira tudi druge časovne metode (angl. Non-Real Time Protocol), s pomočjo katerih lahko čas simulacije napreduje drugače kot realno-svetovni. Te časovne metode opisujejo, kako se lahko obstoječa sporočila DIS uporabijo za podporo vaj in poskusov, ki ne potekajo v realnem času.

Informacije/interakcije živih enitet

Zadnja protokolarni družina so informacije/interakcije živih enitet (angl. Live Entity Information/Interaction). Dodatno optimizirane enote PDU so namenjene podpori živih udeležencev na vadbenih območjih z omejeno pasovno širino. Vadbeno območje je površina, kjer je zahtevana polna zmogljivost sistema. Žive entitete vključujejo življenjske oblike (vojaki), vozila in plovila.

2.3.2 Open-DIS

Projekt *Open-DIS* je odprtokodna implementacija standarda DIS. Implementacija standarda je realizirana v programskih jezikih Java, C++ in C#. V sklopu projekta je izdana tudi dokumentacija, ki je v osnovi namenjena implementaciji v programskem jeziku Java, vendar imajo ostale implementacije podobno zgradbo vmesnika.

Projekt *Open-DIS* je razvil Inštitut za modeliranje, virtualna okolja in simulacije (angl. The Modeling, Virtual Environment, and Simulation Institute - MOVES), ki se nahaja na univerzi za pomorstvo v Montereyu v Kaliforniji (angl. Naval Postgraduate School - NPS). Pri razvoju projekta je bilo spodbujeno sodelovanje zunanjih posameznikov in skupin.

Projekt gosti domena *sourceforge.net*, ki v svetu velja za vodilni repozitorij odprtokodnih projektov.

2.4 Simbologija zveze NATO

Skupna vojaška simbologija zveze NATO je standard za označevanje simbolov na vojaških zemljevidih. Dokument APP-6 (angl. Allied Procedural Publication 6) definira standardni nabor skupnih simbolov, ki zagotavljajo interoperabilnost zveze NATO. Interoperabilnost je sposobnost sistemov, enot ali sil, da zagotavljajo storitve drugim sistemom, enotam ali silam in jih od njih sprejemajo. Tako izmenjane storitve omogočajo učinkovito skupno delovanje. Standard predstavlja enoten sistem vojaške simbologije za kopenske, zračne, vesoljske in mornariške formacije oziroma enote, ki so lahko bodisi prikazane na avtomatiziranih sistemih za prikazovanje terena bodisi ročno označene na zemljevidu.

2.4.1 Zbirke simbolov

Standard APP-6 zagotavlja enotno operativno simbologijo, pri čemer natančno definira, kako naj bodo simboli sestavljeni in izrisani. Tako je standard v največji možni meri kompatibilen in interoperiabilen s sistemom zveze NATO, kot so razvoj, operacije, vadba in sistem poveljevanja, kontrole, zvez, informatike in obveščevalne dejavnosti (angl. Command, Control, Communications, Computers, and Intelligence - C4I). Standard omogoča učinkovit prenos informacij o oznakah z uporabo standardnih metodologij za taksonomijo informacij ter hierarhijo in identifikacijo simbolov.

Standard simbole deli v pet sklopov, od katerih vsak uporablja lastno shemo identificiranja simbolov:

- enote, oprema in naprave,
- taktična grafika vojaških operacij,
- meteorologija in oceanografija,
- obveščevalna dejavnost pri zaznavanju signalov,
- vojaške nevojne operacije (angl. Military Operations Other Than War - MOOTW).

Enote, oprema in naprave so sestavljeni iz okvirjenih ikon, ki so povezane s pripadajočo točko na zemljevidu. Ikone lahko obkrožajo grafična in tekstovna določila, ki specificirajo kategorijo, količino, smer gibanja itd.

Taktično grafiko vojaških operacij predstavljajo operativne informacije, ki jih ni mogoče predstaviti samo z uporabo simbolov. V tej kategoriji lahko simbol predstavljajo točka, črta ali ploskev. Kategorija zajema več sklopov, kot so posebne oznake področja, meje obsega posamezne enote in ostale edinstvene oznake, ki so povezane z geometrijo terena ter nujne za načrtovanje in upravljanje bojišča.

Meteorološka in oceanografska simbologija je edini sklop, ki ni neposredno definiran v standardu, ampak je uvožen iz simbologije Svetovne meteorološke organizacije (angl. World Meteorological Organization - WMO).

Sklopa obveščevalna dejavnost pri zaznavanju signalov in vojaške nevojne operacije sta ločena od enot, opreme in naprav, čeprav upravljata enake konvencije. Tudi ta dva sklopa sta tako predstavljena z okvirjenimi simboli, povezanimi s pripadajočimi točkami na zemljevidu.

2.4.2 Sestavljanje simbolov

Večina simbolov označuje specifične točke in je sestavljena iz okvirja, polnila, konstitutivne ikone in neobveznih določiteljev simbola. Določitelji simbola zajemajo opcijska tekstovna polja ali grafične indikatorje, ki zagotavljajo dodatne informacije.

Polnilo je območje znotraj simbola, omejeno z okvirjem. Če je polnilo barva, potem zagotavlja izboljšano oziroma redundantno informacijo o izvoru objekta. V primeru, ko barva ni uporabljena, je polnilo transparentno. Obstaja majhno število ikon, ki imajo lastno polnilo. Nanje polnilo, ki predstavlja izvor, ne vpliva.

Okvir

Okvir je geometrijska meja simbola, ki nosi informacije o obsegu bojišča ter informacije o izvoru in statusu operativnega objekta. Ker je uporaba oblike in barve redundantna, je lahko simbologija uporabljena v neidealnih pogojih, kjer prikaz ene izmed informacij ni mogoč (npr. črno-beli zaslon). Večina simbolov je enostavnih oblik, ki ne zahtevajo umetniških spretnosti. Okvir služi kot osnova, na katero so dodane druge komponente in določitelji simbola.

Izvor

Izvor se nanaša na odnos med uporabnikom in predstavljenim operativnim objektom. Osnovne kategorije so neznan, prijateljski, nevtralen in sovražen izvor. Rumeni štiriperesni okvir predstavlja neznani izvor, modri pravokotni okvir prijateljski izvor, zeleni kvadratni okvir nevtralni izvor in okvir v obliki rdečega diamanta sovražni izvor.

Celoten nabor izvorov:

- neopredeljen (še neidentificirana sled ali stik),
- neznan (sled ali stik, katerega značilnosti, vedenje, izvor ali nacionalnost ne kažejo niti podpore niti sovražnosti do lastnih sil),
- domnevno prijateljski (sled ali stik, za katerega domnevamo, da je zaradi svojih lastnosti, vedenja ali izvora prijateljski),
- prijateljski (sled ali stik, ki pripada državi, priznani za prijateljsko),
- nevtralen (sled ali stik, katerega značilnosti, vedenje, izvor ali nacionalnost ne kažejo niti podpore niti sovražnosti do lastnih sil),
- sumljiv (sled ali stik, ki je potencialno sovražen zaradi svojih lastnosti, vedenja, izvora ali nacionalnosti),
- sovražen (stik, ki je nedvoumno spoznan za sovražnika),

- šaljivec (prijateljska sled ali stik, ki za vajo deluje kot sumljiva sled),

- slepar (lasten operativni objekt, ki na vaji simulira sovražnega).

Dimenzije bitke

Dimenzija bitke opredeljuje primarno funkcionalno območje operativnega objekta na bojišču. Operativni objekt lahko ima funkcionalno območje nad površjem Zemlje, torej v zraku ali vesolju, na površju ali pod površjem. Če je funkcionalno območje objekta na površju Zemlje, je lahko bodisi na kopnem ali na morju. Dimenzija pod površjem vključuje objekte, katerih funkcionalno območje je pod morsko gladino (npr. podmornice ali morske mine).

Celoten nabor dimenzij bitke:

- vesolje,
- zrak,
- kopno,
- morska gladina,
- pod morsko gladino,
- specialne sile (angl. Special Operations Forces - SOF),
- ostalo,
- neznana dimenzija.

Dimenziji vesolje in zrak si delita enotno obliko okvirja. Na kopnem sta pri prijateljskem in domnevno prijateljskem izvoru uporabljena različna okvirja za enote in opremo. Silam SOF je dodeljena posebna dimenzija bitke, ker njihovo funkcionalno območje v isti misiji običajno obsega več dimenzij. Okvirji sil SOF so enaki kot v kopenski domeni. Dimenzija "ostalo" je rezervirana za uporabo v prihodnosti in trenutno ni definirana.

Tabela 2.2 predstavlja možne kombinacije izvorov in dimenzij bitke.

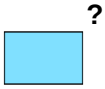
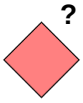
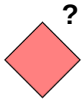
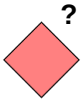
DIMENZIJA BITKE / IZVOR	ZRAK/ VESOLJE	POVRŠJE			POD MORSKO GLADINO
		KOPNO		MORSKA GLADINA	
		ENOTE	OPREMA		
neopredeljen					
neznan					
domnevno prijateljski					
prijateljski					
nevtralen					
sumljiv					
sovražen					
šaljivec					
slepar					

Tabela 2.2: Možne kombinacije izvorov in dimenzij bitke [10]

Položaj ikone

Standard APP-6 opredeljuje enotno mejo v obliki osmerokotnika znotraj vsakega tipa okvirja. Osmerokotnik ob izrisu oziroma upodabljanju dejansko ni prikazan, vendar se mora vsaka ikona znotraj okvirja prilegati tudi osmerokotniku (z določenimi izjemami).

V tabeli 2.3 so prikazane relativne velikosti okvira simbola glede na osmerokotnik.

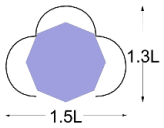
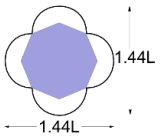
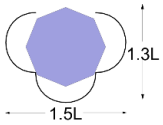
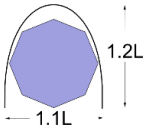
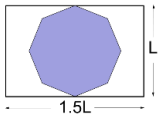
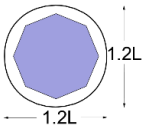
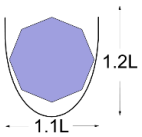
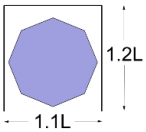
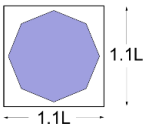
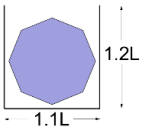
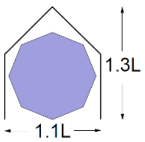
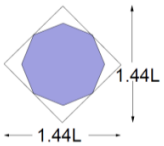
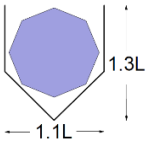
DIMENZIJA BITKE / IZVOR	ZRAK/ VESOLJE	POVRŠJE: ENOTE/OBJEKTI	POVRŠJE: OPREMA	POD MORSKO GLADINO
neznan				
prijateljski				
nevtralen				
sovražen				

Tabela 2.3: Relativne velikosti okvira simbola glede na osmerokotnik [10]

Velikosti enot

Velikost enote je prikazana nad simbolom za vrsto enote.

Celoten nabor dimenzij bitke po velikosti od najmanjše do največje:

- skupina/posadka (3 – 5 članov),

- oddelek (5 – 10 članov),
- sekcija (7 – 13 članov),
- vod (25 – 40 članov),
- četa (60 – 250 članov),
- bataljon (300 – 1,000 članov),
- polk/skupina (500 – 2,000 članov),
- brigada (2,000 – 5,000 članov),
- divizija (10,000 – 20,000 članov),
- korpus (30,000 – 60,000 članov),
- armada (100,000 članov),
- armadna skupina (120,000 – 500,000 članov),
- vojskovališče (250,000 – 1,000,000 članov).

2.5 Pogon Unity

Pogon *Unity* [14] je pogon, ki ga razvija podjetje *Unity Technologies* in je primarno namenjen razvoju računalniških iger. Pogon se uporablja za razvoj aplikacij za različne platforme, med katere spadajo *Android*, *iOS*, *Xbox*, *Playstation*, *Windows*, *Linux*, *Mac* in spletne storitve. Osnovna različica pogona *Personal edition* je na voljo brezplačno in ima v primerjavi s plačljivo okrnjen nabor funkcionalnosti.

Pogon Unity je okolje, ki olajša razvoj iger tako, da lahko uporabnik veliko stvari naredi preko uporabniškega vmesnika. Poleg tega dopušča uporabo obstoječih virov, kot so slike, modeli, ostale datoteke.

Poglavje 3

Implementacija aplikacije za 3D vizualizacijo dogajanja na bojišču

Aplikacija za 3D vizualizacijo dogajanja na bojišču je implementiranja v pogonu Unity v programskem jeziku C#. Poleg pogona *Unity* je bil uporabljen še program *Houdini FX* [12], s katerim je bilo opravljeno večino modeliranja in prilagajanja modelov. Za manipulacijo slik je bil uporabljeno programsko orodje *Paint.NET* [4].

Aplikacija je v osnovi namenjena vizualiziranju poteka simulacije v programu JCATS v realnem času, vendar je možen prikaz entitet tudi iz simulatorja *Virtual Battlespace 3*.

Najprej je opisano delovanje aplikacije, nato so predstavljene še posamezne komponente vizualizatorja.

3.1 Delovanje aplikacije

Ob zagonu aplikacije se uporabniku prikaže uvodna scena, ki omogoča uvoz datotek XML (angl. Extensible Markup Language). S pomočjo izbirnika da-

totek (angl. file chooser) uporabnik določi tri datoteke XML, ki so predhodno izvožene iz simulatorja JCATS. Vsaka izmed datotek ustreza posameznim udeležencem vaje; prva vsebuje informacije o entitetah prijateljskih sil, druga informacije o entitetah nevtralnih sil in tretja informacije o entitetah sovražnih sil. V kolikor se kasneje v vizualizatorju pojavi entiteta, ki ni vsebovana v datotekah XML, jo aplikacija avtomatično dodeli k neznanim silam.

Datoteke XML vsebujejo podatke o entitetah iz simulatorja JCATS. Vizualizator podpira dva tipa entitet, in sicer enote (angl. units) in sisteme (angl. systems).

Ko uporabnik izbere željene datoteke in potrdi izbiro, aplikacija prične s pripravo okolja za naslednjo sceno, ki predstavlja vizualizacijo sodobnega bojišča.

Ob prehodu med scenama program s pomočjo datotek XML zgradi seznam objektov. Vsak objekt vsebuje več parametrov, kot so identifikacijska številka, ime, zdravje, referenco na igralni objekt (angl. game object) in referenco enote, katere član je dani objekt. Objekti tipa enota poleg zgornjega vsebujejo še seznam referenc njihovih podenot oziroma podsistemov.

Če je objekt generiran iz datotek XML, tipa enota, potem mu program dodeli ustrezen simbol zveze NATO. Če je objekt tipa sistem, mu program dodeli ustrezen 3D model. Ko je seznam objektov zgrajen, se prične glavna programska zanka druge scene.

Glavni element prikaza je zemljevid oziroma satelitski posnetek sveta, ki je dinamično pridobljen s strežnikov podjetja *Google*. Zemljevid je prikazan v spletni Mercatorjevi projekciji in omogoča prikaz pri različnih nivojih podrobnosti (angl. Level of detail - LOD), kar pomeni, da je podrobnost prikaza premo sorazmerno odvisna od stopnje povečave. Kasneje je v diplomski nalogi opisana tudi nadgradnja prikaza okolja, kjer so deli zemljevida nalepljeni na dejanski teren. V tem primeru so podatki o višinah pridobljeni iz podatkovne baze projekta SRTM (glej poglavje 2.1.2).

Pomemben del vizualizatorja predstavlja uporabniški vmesnik, ki ga sesta-

vlja drevesna struktura entitet, dnevnik (angl. log), okno s podrobnostmi o entiteti in kompas. Drevesna struktura prikazuje hierarhični seznam entitet, ki so vsebovane v datotekah XML, uvoženih v prvi sceni. V programskem dnevniku so zabeležene vse informacije o dogodkih, ki so se zgodili v vizualizatorju. Okno o podrobnostih enote oziroma sistema prikazuje lastnosti entitete. Kompas se nahaja v desnem spodnjem kotu in uporabniku pomaga predvsem pri navigaciji. V okviru uporabniškega vmesnika je implementirana možnost premikanja kamere oziroma pogleda.

Ob prehodu iz prve v drugo sceno program zažene ločeno korutino, ki v neskončni programski zanki posluša in prestreza pakete protokola UDP (angl. User Datagram Protocol). Protokol UDP v aplikaciji predstavlja zabojnik, ki enkapsulira enote PDU standarda DIS (glej poglavje 2.3).

Ob prejemu paketa tipa UDP program najprej preveri, če paket vsebuje enoto PDU in jo v pozitivnem primeru uvrsti v vrsto (angl. queue). Glavna programska zanka iz te vrste prebira nove enote PDU po sistemu prvi-noter-prvi-ven (angl. First In, First Out - FIFO) in jih glede na vrsto ustrezno obdela.

Enote so v okolju predstavljene s simboli zveze NATO. Simboli so nalepljeni na grafični vmesnik, njihova lokacija je določena glede na povprečno lokacijo njenih neposrednih podenot oziroma podsistemov. Enota postane vidna, ko se v vizualizatorju pojavi vsaj eden izmed njenih podrejenih članov. Iz simbola lahko uporabnik razbere velikost in vrsto enote, njegova barva in oblika pa predstavljata pripadnost enote k določeni sili.

Sistemi so v okolju predstavljeni kot 3D modeli, katerih lokacijo, zdravje in premikanje določajo enote PDU. Barva modela predstavlja pripadnost sistema k določeni sili.

Modeli in simboli zveze NATO so dodeljeni glede na knjižnico, ki je v programu predstavljena kot datoteka XML. V njej so tipi sistemov in enot povezani z ustreznimi modeli oziroma simboli.

3.2 Okolje

Okolje oziroma podlaga predstavlja najpomembnejši element vizualizacije, saj je osnova za postavitev enot, sistemov in ostalih subjektov.

Podlago sestavljajo satelitski posnetki oziroma deli zemljevida (v nadalj. ploščice), ki so asinhrono pridobljeni iz strežnika oziroma naloženi iz diska lokalne delovne postaje. Zemljevid je predstavljen s spletno Mercatorjevo projekcijo, saj so tako shranjene posamezne ploščice zemljevida na strežniku.

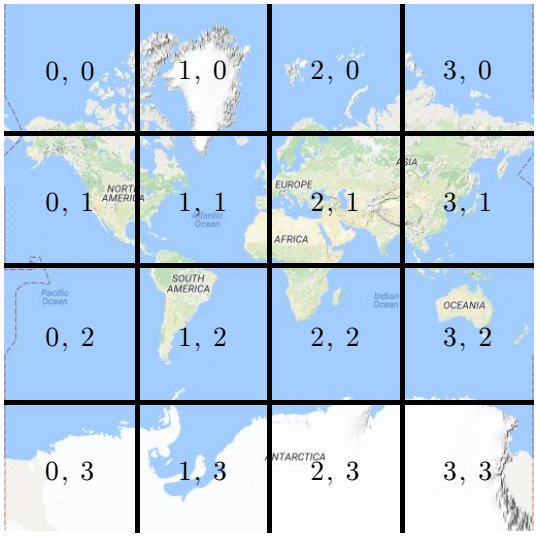
Ploščice so prikazane s principom nivojev podobnosti, ki definira več stopenj natančnosti prikaza in je uporabljen predvsem zaradi optimizacije, saj v realnem času zagotavlja zadovoljive rezultate.

3.2.1 Pridobivanje satelitskih posnetkov

Satelitski posnetki so pridobljeni s pomočjo vmesnika *Google Maps* API (angl. application programming interface). Vmesnik API deli satelitske posnetke v niz ploščic, ki so urejene v mrežo. Ploščice so indeksirane s parametrom *zoom*, ki določuje velikost mreže, ter koordinatama (x, y), ki označujeta lokacijo ploščice na izbrani mreži. Ploščica s koordinatama $(0, 0)$ je vedno locirana v severozahodnem kotu mreže oziroma zemljevida. Vrednosti koordinate x naraščajo od zahoda proti vzhodu, vrednosti koordinate y pa od severa proti jugu. Vsaka ploščica je v bazi referencirana z edinstveno kombinacijo parametrov x, y in *zoom*.

Na stopnji povečave 0 (angl. zoom level) je celoten svet upodobljen v eni ploščici. Vsaka nadaljnja stopnja povečave razširi zalogo vrednosti parametrov x in y za faktor 2. Tako je na stopnji povečave 1 zemljevid upodobljen kot mreža ploščic v velikosti 2×2. Na stopnji povečave 2 je mreža velika 4×4, na stopnji povečave 3 8×8 in tako naprej.

Na sliki 3.1 je prikazana razporeditev ploščic na stopnji povečave 2, kjer mreža vsebuje 16 ploščic.



Slika 3.1: Razporeditev ploščic na stopnji povečave 2 [6]

Maksimalno vrednost koordinat x, y pri dani stopnji povečave (3.1) in število ploščic pri dani stopnji povečave (3.2) izračunamo s pomočjo naslednjih enačb:

$$M_{x|y} = 2^z - 1 \tag{3.1}$$

$$n = 2^{2z} \tag{3.2}$$

kjer z predstavlja stopnjo povečave.

Vmesnik API omogoča, da s pomočjo parametra *hl* nastavimo jezik zemljevida.

Zadnji parameter, $lyrs$, določuje sloj oziroma vrsto zemljevida. Zajema spodnje možnosti:

- m - standardni zemljevid,
- h - samo oznake (države, mesta, reke, ceste...),
- s - satelitski posnetki,
- y - satelitski posnetki z oznakami,
- t - teren brez oznak,
- p - teren z oznakami,

- r - predrugačen standardni zemljevid.

Do ploščic na strežniku *Google* je mogoče dostopati s pomočjo vmesnika API preko zahtevkov protokola HTTP (angl. Hypertext Transfer Protocol), pri čimer je poizvedba sestavljena iz zgoraj predstavljenih parametrov. Vsaka uspešna poizvedba vrne sliko tipa PNG (angl. Portable Network Graphics) v velikosti 256×256 pikslov. Za uravnoteženje obremenitve so na razpolago štirje strežniki, ki vračajo enake odgovore.

Zemljevidi Google uporabljajo spletno Mercatorjevo projekcijo za prikaz posameznih ploščic. V teoriji bi se morala geografska širina raztezati od $-90°$ do $90°$, vendar je pokritost zaradi tipa projekcije manjša, saj projekcija pošlje oba pola proti neskončnosti. Robni geografski širini sta enaki širinam, ki vrneta število π kot rezultat spletne Mercatorjeve projekcije:

$$\arctan\left(\sinh\left(\pm\pi\right)\right)\frac{180}{\pi} \approx \pm 85.0511287798° \tag{3.3}$$

3.2.2 Prikazovanje satelitskih posnetkov

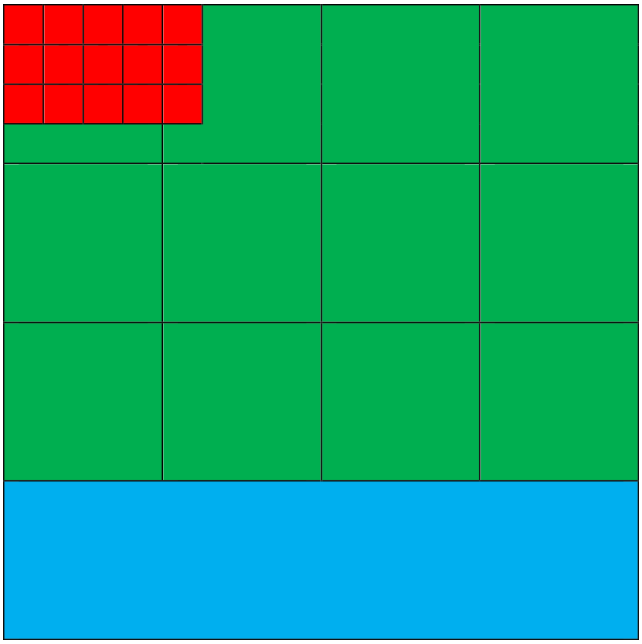
Podlaga je dizajnirana po principu različnih nivojev podrobnosti (LOD), vsak nivo prikazuje določeno stopnjo povečave vmesnika *Google Maps* API. Uporabljene so samo sode stopnje povečave (0, 2, 4, ...), kar pomeni, da vsaka stopnja zajema štirikrat več ploščic na strežniku kot prejšnja, torej 1×1, 4×4, 16×16 in tako dalje.

Za prikaz ploščic iz strežnika so uporabljene ravnine (angl. planes), ki v pogonu *Unity* predstavljajo dvodimenzionalne objekte. Na prvem nivoju podobnosti je celoten svet upodobljen na eni ploščici in tako tudi na eni ravnini. Na drugem nivoju celoten svet predstavlja 16 ploščic, vizualizator jih na enkrat prikazuje 12 (4×3) na 12 ravninah. Vsi ostali nivoji podrobnosti imajo za prikaz svojih ploščic na voljo 15 ravnin (5×3). Skupno število ploščic posameznih nivojev izračunamo po enačbi 3.2.

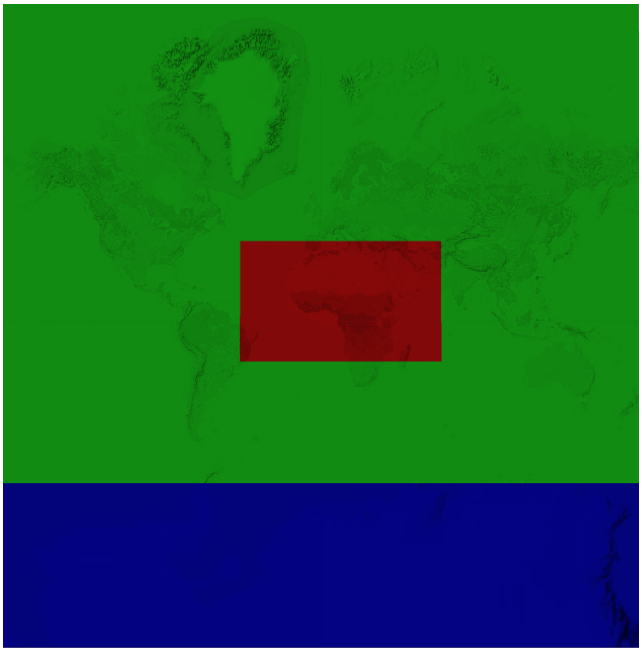
Velikost ravnine je obratno sorazmerna s stopnjo povečave. Na prvem nivoju podrobnosti, kjer je stopnja povečave enaka 0, je velikost ravnine enaka

8192×8192 enot. Na drugem nivoju podrobnosti, kjer je stopnja povečave enaka 2, je velikost ravnine enaka 2048×2048 enot. Na zadnjem, devetem nivoju podrobnosti, kjer je stopnja povečave enaka 16, je velikost ravnine enaka $0,125 \times 0,125$ enot.

Na sliki 3.2 je predstavljeno razmerje velikosti med prvim, drugim in tretjim nivojem podrobnosti. Slika 3.3 prikazuje, kako je razmerje med nivoji predstavljeno v vizualizatorju.



Slika 3.2: Razmerje velikosti med prvim, drugim in tretjim nivojem podrobnosti



Slika 3.3: Razmerje velikosti med nivoji podrobnosti v vizualizatorju

Ploščice so pridobljene preko vmesnika API z uporabo parametrov, opisanih v 3.2.1. Implementirano je shranjevanje ploščic na disk, tako da aplikacija najprej preveri, če ploščica že obstaja na disku in samo v negativnem primeru ploščico pridobi iz strežnika.

Algoritem prikazovanja in skrivanja delov zemljevida

Algoritem 1 definira prikazovanje, skrivanje in posodabljanje ravnin, ki prikazujejo satelitske posnetke, pri čimer uporablja različne nivoje podrobnosti.

Pomembno je, da algoritem zagotavlja izvajanje v realnem času.

function *Initialize()*

1: set which LOD (Level of detail) levels are seen on which heights;
2: create plane for LOD level 1;
3: show plane of LOD level 1 with satellite image;
4: create 12 planes for LOD level 2;
5: create 15 planes for each one of the rest LOD levels;

end

function *Update()*

1: get camera angle and location;
2: hide all planes of all LOD levels except plane of LOD level 1;
3: based on camera height set which LOD levels will be seen;
4: based on step 3 show all planes that must be seen;
if *needed based on step 1* **then**
    5: update satellite images on planes that are seen;
end

end

Algoritem 1: Prikazovanje, skrivanje in posodabljanje ravnin pri različnih nivojih podrobnosti v realnem času

3.2.3 Nadgradnja okolja s terenom

Ker okolje v vizualizatorju predstavljajo ploščice, ki so pripete na ravnine, se okolje razteza samo v dve dimenziji. Da bi lahko dodali še tretjo, moramo ravnine preoblikovati v trodimenzionalen teren. To v praksi pomeni, da je potrebno geografski dolžini, ki v okolju vizualizatorja predstavlja X-os, in geografski širini, ki v okolju predstavlja Z-os, dodati še nadmorsko višino, ki bi v okolju predstavljala Y-os.

V okviru te diplomske naloge je bil za zgornji problem realiziran prototip, s katerim smo vizualizacijo terena realizirali za izbrano manjše območje.

Za tretjo dimenzijo okolja so potrebni podatki o nadmorskih višinah, ki so bili pridobljeni iz baze podatkov projekta SRTM (glej poglavje 2.1.2). Za področje problema je bilo zastavljeno širše območje Slovenije, ki se razteza od 12° do 18° geografske dolžine in od 44° do 47° geografske širine.

Osnovna rešitev problema je lepljenje ploščic s satelitskimi slikami na dele terena na podoben način, kot pri ravninah, s to razliko, da je potrebno pri tej implementaciji poleg ploščic zamenjati še podatke o višinah.

Izbrano območje ne predstavlja kvadratne matrike, saj spletna Mercatorjeva projekcija deformira območja v smeri naraščanja geografske širine. Dimenziji matrike 8407×4804 sta bili izračunani s pomočjo specifikacije standarda DTED (glej poglavje 2.1.2). Ker ploščice delujejo na osnovi kvadratov, je bila zgornja matrika višin s pomočjo programskega orodja *Matlab* preoblikovana v matriko velikosti 8192×8192. Rezultat preoblikovanja je kvadratna matrika, ki ima število vrstic in stolpcev enako potenci števila dva (2), kar je še posebej pomembno pri implementaciji algoritma različnih nivojev podrobnosti za teren.

Algoritem 2 predstavlja rešitev za izbiranje višin določenega dela terena na

podanem nivoju podrobnosti.

```
function GetHeightsForTile(posX, posZ, heightmapOffsetX,
 heightmapOffsetZ, level)
    /* size is size of terrain tile in world units for given level          */
    size ← levelSizes[level];
    /* heightmapResolution is resolution of terrain tile for given level */
    heightmapResolution ← levelHeightmapSizes[level];
    /* stride is skip stride of terrain tile for given level               */
    stride ← levelHeightmapsStrides[level];
    heights[heightmapResolution + 1][heightmapResolution + 1];
    for i ← 0 to heightmapResolution do
        for j ← 0 to heightmapResolution do
            hi ← heightmapOffsetX * heightmapResolution * stride;
            hi ← hi + i * stride;
            if hi >= length(heightmapData) then
                hi ← length(heightmapData) - 1;
            end
            hj ← heightmapOffsetZ * heightmapResolution * stride;
            hj ← hi + j * stride;
            if hj >= length(heightmapData[0]) then
                hj ← length(heightmapData[0]) - 1;
            end
            /* heigthData contains all height values and serves as
               database                                                     */
            heights[i][j] = heightmapData[hi][hj];
        end
    end
    return heights;
end
```

Algoritem 2: Izbiranje višin določenega dela terena na podanem nivoju podrobnosti

3.3 Bralnik datotek XML

Bralnik datotek XML je komponenta vizualizatorja, ki bere in razčlenjuje dokumente XML. Instanca bralnika se zažene ob prehodu iz uvodne scene v sceno vizualizacije, ko uporabnik izbere željene datoteke XML. Te so predhodno izvožene iz simulatorja JCATS in vsebujejo podatke o enotah in sistemih.

Preden lahko bralnik začne z obdelavo datotek XML, mora zgraditi knjižnico oziroma slovar tipov entitet. Bralnik zgradi knjižnico s pomočjo druge konfiguracijske datoteke XML, v kateri so tipi enot in sistemov povezani z ustreznimi simboli oziroma 3D modeli. V konfiguracijski datoteki sta na glavnem nivoju definirani dve različni strukturi. Struktura *UNIT* opredeljuje tipe enot in vsebuje elemente *NAME* (vrsta enote), *NATO_SymbolFile* (relativna pot do simbola) in *Origin* (izvor simbola). Struktura *SYSTEM* opredeljuje tipe sistemov in zajema elementa *NAME* (vrsta enote) in *ModelFile* (relativna pot do 3D modela). Konfiguracijska datoteka je skalabilna, kar pomeni, da se lahko vanjo dodajajo nove definicije za nove vrste enot oziroma sistemov. V programski kodi 3.1 je prikazan izsek iz konfiguracijske datoteke XML, v katerem je predstavljena zgradba obeh struktur.

Programska koda 3.1: Izsek kode iz konfiguracijske datoteke XML

```
<?xml version="1.0" encoding="UTF-8"?>
<EntityTypes>
  ...
  <UNIT>
    <NAME>LANDUNIT</NAME>
    <NATO_SymbolFile>NATO_Symbols\LandUnit_U.png</NATO_SymbolFile>
    <Origin>UNKNOWN</Origin>
  </UNIT>
  <UNIT>
    <NAME>LANDUNIT</NAME>
    <NATO_SymbolFile>NATO_Symbols\LandUnit_F.png</NATO_SymbolFile>
```

```
    <Origin>FRIEND</Origin>
  </UNIT>
  ...
  <SYSTEM>
    <NAME>TEREN VOZILO</NAME>
    <ModelFile>Models/terenskoVozilo</ModelFile>
  </SYSTEM>
  <SYSTEM>
    <NAME>APC8X8POVELJ</NAME>
    <ModelFile>Models/APC8x8</ModelFile>
  </SYSTEM>
  ...
</EntityTypes>
```

Ko je knjižnica entitet zgrajena, bralnik prične z obdelavo datotek XML izvoženih iz simulatorja JCATS. Te datoteke predstavljajo seznam enot in sistemov posamezne sile v vizualizatorju. Tudi te datoteke XML imajo na glavnem nivoju definirani dve različni strukturi.

Struktura *UNIT* opredeljuje entitete tipa enota in med pomembnejšimi vsebuje parametre *JCATS_ID*, *NAME* in *JCATS_ToeSuperID*. Parameter *JCATS_ID* določa identifikacijsko številko enote in ustreza identifikacijski številki enote v standardu DIS. Parameter *NAME* določa vrsto enote, s pomočjo katere je enoti dodeljen ustrezni simbol zveze NATO iz zgoraj omenjene knjižnice. Parameter *JCATS_ToeSuperID* je neobvezen in določa identifikacijsko številko neposredne nadenote dane enote. V strukturi so vsebovani še nekateri drugi parametri, ki za vizualizator, predstavljen v tem diplomskem delu, niso pomembni.

Struktura *SYSTEM* opredeljuje entitete tipa sistem in med pomembnejšimi vsebuje parametre *JCATS_ID*, *JCATS_SystemCharName* in *JCATS_ToeSuperID*. Parametra *JCATS_ID* in *JCATS_ToeSuperID* pri strukturi *SYSTEM* pomenita enako, kot pri zgornji, s to razliko, da je identifikacijska številka nadenote pri entitetah tipa sistem obvezna. Parameter *JCATS_SystemCharName* pred-

stavlja vrsto sistema, ki sistemu dodeli ustrezni 3D model iz zgoraj omenjene knjižnice.

S pomočjo parametra *JCATS_ToeSuperID* je zgrajen hierarhično povezan seznam objektov, preko katerega kasneje dostopamo do željenih entitet. Seznam je priročen tudi pri uporabi rekurzivnih metod. V programski kodi 3.2 je prikazan izsek iz datoteke XML izvožene iz simulatorja JCATS, ki prikazuje pomembnejše parametre obeh struktur glavnega nivoja datoteke XML.

Programska koda 3.2: Izsek kode iz datoteke XML izvožene iz simulatorja JCATS

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<ForceOrg>
  ...
  <UNIT>
    ...
    <JCATS_ID>1613</JCATS_ID>
    <NAME>POVTANKC</NAME>
    <JCATS_ToeSuperID>1612</JCATS_ToeSuperID>
    ...
  </UNIT>
  <SYSTEM>
    <JCATS_ID>13703</JCATS_ID>
    <NAME>M84_13703</NAME>
    <JCATS_SystemCharName>M84</JCATS_SystemCharName>
    <JCATS_ToeSuperID>1613</JCATS_ToeSuperID>
    ...
  </SYSTEM>
  ...
</ForceOrg>
```

3.4 Prejemnik enot PDU standarda DIS

Posodabljanje stanj simulacije v vizualizatorju poteka s pomočjo standarda DIS (glej poglavje 2.3).

Ob prehodu iz uvodne scene v sceno vizualizacije program zažene ločeno korutino, ki v neskončni programski zanki posluša in prestreza pakete protokola UDP. V paketih protokola UDP so enkapsulirane enote PDU standarda DIS. Ob prejemu paketa tipa UDP program najprej preveri, če paket vsebuje enoto PDU in jo v pozitivnem primeru uvrsti v vrsto. Glavna programska zanka iz te vrste prebira nove enote PDU po sistemu FIFO in jih, glede na tip enote, ustrezno obdela.

Vizualizator podpira tri tipe enot PDU, in sicer *Entity State*, *Fire* in *Detonation*.

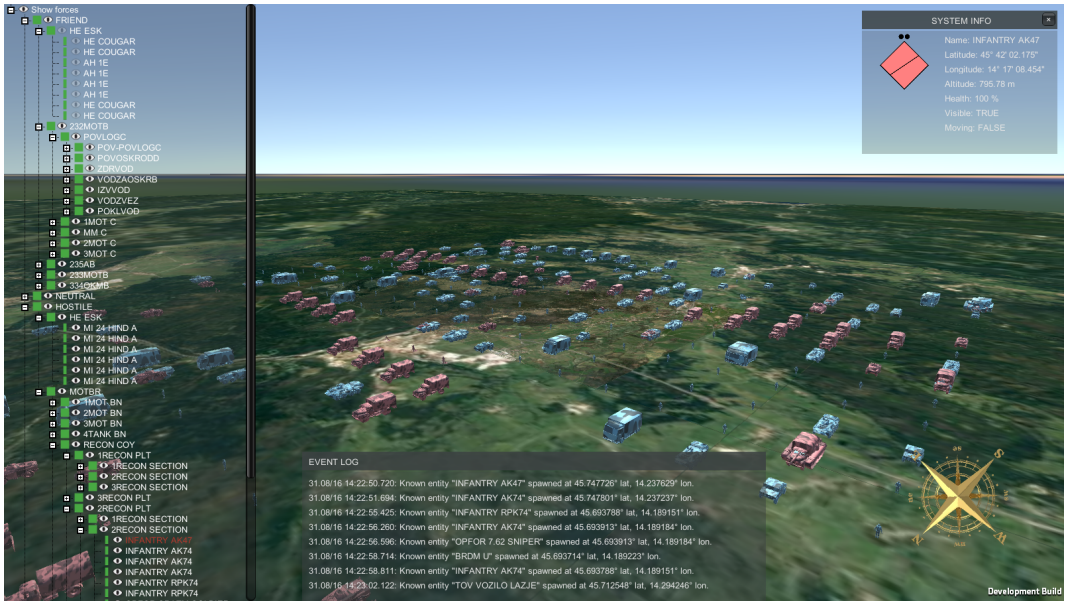
Enota PDU tipa *Entity State* nosi informacije o stanju entitete, torej podatke o lokaciji, zdravju in orientaciji entitete. V primeru, ko je prejeta enota PDU tipa *Entity State*, progam ustreznemu objektu na sceni priredi novo lokacijo in zdravje. V kolikor vektor hitrosti ni enak nič, program objektu nastavi parametre, ki definirajo njegovo premikanje. Vsakokrat, ko vizualizator posodobi lokacijo sistema, hkrati sproži rekurzivno funkcijo, ki posodobi lokacije vseh nadenot danega objekta.

Enota PDU tipa *Fire* nosi informacije o sprožitvi orožja in informacije o obeh udeleženih entitetah. Ob prejemu tovrstne enote PDU se v programski dnevnik doda nov zapis o akciji sprožitve orožja.

Enota PDU tipa *Detonation* nosi informacije o vrsti, obsegu in posledicah detonacije ter informacije o entiteti, ki je detonacijo povzročila. Ob prejemu tovrstne enote PDU se v programski dnevnik doda nov zapis o lokaciji detonacije in entiteti, ki je detonacijo povzročila.

3.5 Uporabniški vmesnik

Uporabniški vmesnik (angl. user interface - UI) v vizualizatorju sestavljajo štiri komponente, in sicer drevesna struktura entitet, okno s podrobnostmi o entiteti, okno, ki prikazuje programski dnevnik, in kompas. Postavitev posameznih komponent je vidna na zaslonskem posnetku vizualizatorja v sliki 3.4. Uporabniški vmesnik je implementiran s pomočjo sistema Unity UI (angl. Unity User Interface), ki omogoča generiranje komponent namenjenih interakciji z uporabnikom. Sistem preko urejevalnika omogoča tudi ročno dodajanje komponent. Poleg zgoraj omenjenih komponent uporabniški vmesnik definira tudi upravljanje vizualizatorja.

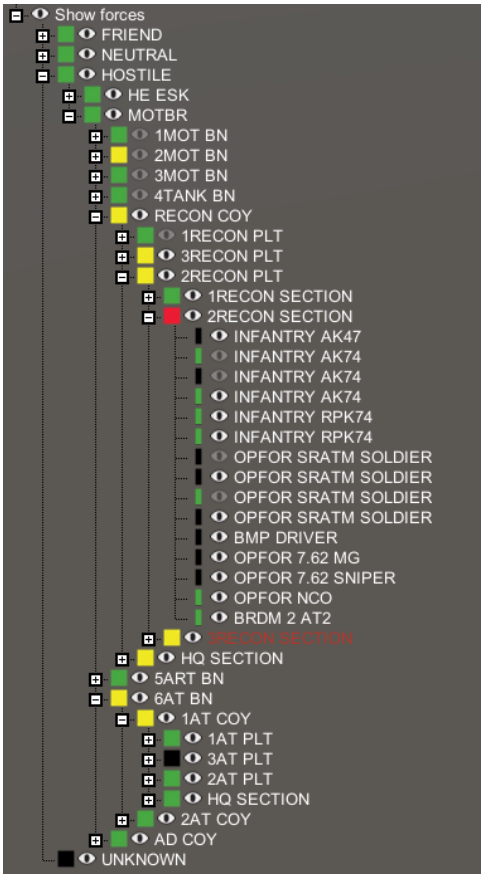


Slika 3.4: Zaslonski posnetek vizualizatorja, ki prikazuje komponente UI

Drevesna struktura predstavlja glavni element uporabniškega vmesnika, saj so v njej predstavljene vse entitete, ki sodelujejo v vizualizirani simulaciji. Drevesna struktura je generirana s pomočjo seznama objektov, ki se kreira ob branju dokumentov XML v uvodni sceni aplikacije. Iz strukture je vidna hierarhija entitet, saj so enote in sistemi razvrščeni enako kot v dokumentih XML

oziroma simulatorju JCATS. Na prvi ravni drevesne strukture so predstavljene štiri sile; prijateljske, nevtralne, sovražne in neznane. Ta prva tri poddrevesa so definirana s pomočjo ustreznih dokumentov XML, izbranih v uvodni sceni, medtem ko je med neznane sile dodeljena vsaka entiteta, ki sodeluje v simulaciji in ni zajeta v dokumentih XML. Med neznane sile so dodeljene tudi entitete drugih simulatorjev, npr. entitete simulatorja VBS.

Drevesna struktura za vsak element definira več funkcionalnosti. Ob vsaki entiteti je barvno prikazano njeno trenutno stanje zdravja in lociran gumb, ki omogoča skrivanje entitete. Gumb za skrivanje deluje tako, da pri entitetah tipa sistem skrije ustrezni model, medtem ko pri entitetah tipa enota skrije vse modele podsistemov in simbole zveze NATO, ki pripadajo vsem podenotam dane enote. Poleg tega se ob kliku na izbrani element prikaže okno s podrobnostmi o entiteti, ob dvokliku pa se kamera premakne nad izbrani objekt in usmeri svoj pogled proti njemu. S pritiskom na ustrezni gumb na tipkovnici je omogočeno tudi skrivanje oziroma prikazovanje drevesne strukture (glej tabelo 3.1). Na sliki 3.5 je prikazan primer drevesne strukture iz vizualizatorja.



Slika 3.5: Primer drevesne strukture entitet

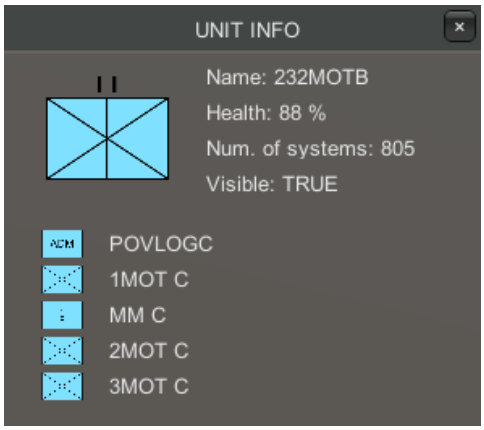
Drugo komponento uporabniškega vmesnika predstavlja okno s podrobnostmi o entiteti. Kot že omenjeno, je prikaz podrobnosti o določeni entiteti omogočen preko klika na ustrezni element v drevesni strukturi. Ker sta v vizualizatorju definirana dva tipa entitet, obstajata tudi dva tipa oken s podrobnostmi, eno za enote in drugo za sisteme.

Okno s podrobnostmi o enoti vsebuje naslednje elemente: ime, zdravje, rekurzivno število podsistemov in simbol zveze NATO. Poleg tega je iz okna razvidno, ali je enota prikazana ali ne. V oknu so dodane povezave do neposrednih podenot oziroma podsistemov dane enote.

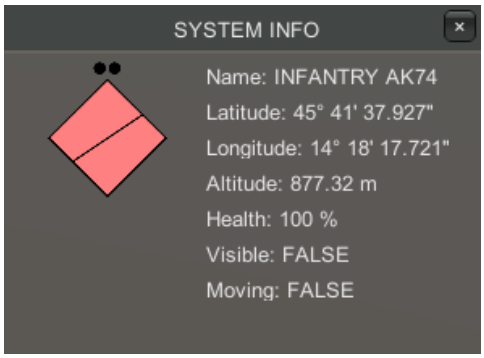
Okno s podrobnostmi o sistemu vsebuje naslednje elemente: ime, zdravje,

geografsko širino in dolžino, nadmorsko višino in vidnost sistema. V oknu je označeno, če se dani sistem premika in v pozitivnem primeru tudi njegova hitrost.

Na sliki 3.6 je prikazan primer okna s podrobnostmi o enoti in na sliki 3.7 primer okna s podrobnostmi o sistemu iz vizualizatorja.



Slika 3.6: Primer okna s podrobnostmi o enoti



Slika 3.7: Primer okna s podrobnostmi o sistemu

Tretja komponenta uporabniškega vmesnika je okno, ki vsebuje zapise programskega dnevnika. Programski dnevnik je namenjen opisovanju simulacije. V njem zapisani dogodki, ki so se zgodili v simulaciji, in sicer prva pojavitev entitete, streljanje in detonacija. Podobno kot drevesno strukturo lahko tudi

to komponento prikažemo oziroma skrijemo s pritiskom na ustrezni gumb na tipkovnici (glej tabelo 3.1). Primer zapisa programskega dnevnika je razviden iz slike 3.8.

Slika 3.8: Primer okna z zapisi programskega dnevnika

Zadnjo komponento uporabniškega vmesnika predstavlja kompas, ki je namenjen pomoči ob navigaciji ob premikanju po okolju. Tudi prikaz te komponente je poljuben (glej tabelo 3.1). Izgled in lokacija kompasa v vizualizatorju je prikazana na sliki 3.4.

Poleg komponent uporabniški vmesnik definira akcije, s katerimi uporabnik nadzira vizualizator. S klasično navigacijsko tehniko gumbov *WASD* je omogočeno premikanje po okolju, pri čimer je kot kamere določen s premikanjem računalniške miške ob pritisku na desni miškin gumb. Če uporabnik ob premikanju drži tipko desni *shift*, potem se premika z dvojno hitrostjo. Poleg premikanja kamere lahko uporabnik s pomočjo drsnega koleščka na miški povečuje zoom. Omogočena je ponastavitev lokacije in pogleda kamere.

V tabeli 3.1 so predstavljene bližnjice z akcijami uporabniškega vmesnika.

bližnjica	akcija
w,a,s,d	navigacijski gumbi za premikanje po okolju
desni *shift*	poveča hitrost premikanja za faktor 2
drsni koleščck miške	povečuje/zmanjšuje povečavo
r	ponastavi pogled kamere
t	skrij/prikaži drevesno strukturo entitet
l	skrij/prikaži programski dnevnik
c	skrij/prikaži kompas
m	spremeni tip mape

Tabela 3.1: Akcije uporabniškega vmesnika

3.6 Prikaz enot in sistemov

Kot je že bilo omenjeno vizualizator trenutno podpira prikaz dveh tipov entitet, in sicer enote in sisteme. Enote so prikazane kot simboli zveze NATO, medtem ko so sistemi predstavljeni kot predhodno generirani 3D modeli.

Simboli zveze NATO so sestavljeni po standardu simbologije zveze NATO, predstavljene v poglavju 2.4.2. Generirani so s pomočjo spletne aplikacije *Joint military symbology explorer* [5], ki glede na vnesene parametre generira simbol v slikovnem formatu PNG.

Modeli za sisteme so bili vzeti iz spleta in obdelani v orodju *Houdini FX*, kjer so bili predvsem optimizirani. Vsi modeli, ki so vsebovani v vizualizatorju, imajo odprtokodno licenco.

Simboli in modeli so referencirani s pomočjo bralnika datotek XML, predstavljenega v poglavju 3.3.

3.6.1 Pozicioniranje enot in sistemov

Pri validaciji prikaza ima pomembno vlogo pravilnost pozicioniranja enot in sistemov. Najprej je razloženo pozicioniranje sistemov in nato še pozicioniranje

enot.

Sistemom določajo lokacijo enote PDU tipa *Entity State* v polju *Entity Location* (lokacija entitete). Polje *Entity Location* vsebuje komponente x, y in z, ki so zapisane v 64-bitni plavajoči vejici. Te komponente so kartezične koordinate, kodirane po standardu WGS 84.

Da lahko sistem pravilno pozicioniramo v okolje, moramo kartezične koordinate najprej prevesti v geografske. Nato geografske koordinate prevedemo v koordinati spletne Mercatorjeve projekcije in nazadnje koordinati projekcije v koordinate sveta.

Za pretvorbo iz kartezičnih koordinat v geografske smo uporabili aproksimacijsko metodo, ki jo je leta 1976 predstavil B. R. Bowring v svojem članku Transformacija iz prostorskih v geografske koordinate (angl. Transformation from spatial to geographical coordinates) [3]. Pretvorba ni popolnoma točna, vendar za višine manjše od 1000 km zagotavlja centimetrsko natančnost.

Spodnje enačbe definirajo transformacijo iz kartezičnih koordinat v geografske po Bowringu:

$$\phi = \operatorname{atan}\left(\frac{Z + e_s^2 b \sin^3\theta}{p - e^2 a \cos^3\theta}\right) \tag{3.4}$$

$$\lambda = \operatorname{atan2}(Y, X) \tag{3.5}$$

$$h = \frac{p}{\cos\phi} - N(\phi) \tag{3.6}$$

kjer ϕ predstavlja geografsko širino, λ geografsko dolžino in h višino nad elipsoidom. X, Y in Z so kartezične koordinate Zemlje. *atan* in *atan2* sta metodi, ki v programskih jezikih implementirata krožno funkcijo arkus tangens. Razlika med metodama je v tem, da metoda *atan* sprejme en parameter, medtem ko metoda *atan2* sprejme dva. Spodaj so navedene pomožne enačbe transformacije:

$$p = \sqrt{X^2 + Y^2} \qquad \theta = \operatorname{atan}\left(\frac{Za}{pb}\right) \qquad e_{II}^2 = \frac{a^2 - b^2}{b^2} \tag{3.7}$$

$$N(\phi) = \frac{a}{\sqrt{1 - e^s \sin^2\phi}} \tag{3.8}$$

$$a = 6378137.0\ m \tag{3.9}$$

$$b \approx 6356752.314245\ m \tag{3.10}$$

$$f = \frac{a-b}{a} \tag{3.11}$$

$$e^2 = 2f - f^2 \tag{3.12}$$

kjer a predstavlja ekvatorialni polmer, b polarni polmer, p je razdalja od polarne osi do dane točke, f uravnavanje, e prvo ekscentričnost in e_s drugo ekscentričnost. Funkcija $N(\phi)$ vrača polmer krivine primarne vertikale ob dani geografski širini. θ je aproksimacija geografske širine geocentra za podani Z.

Ko so kartezične koordinate prevedene v geografske, lahko izračunamo koordinati spletne Mercatorjeve projekcije x in y. Namesto pretvorbe v položaj piksla, koordinati neposredno prevedemo v koordinati sveta x in z. Za pretvorbo smo uporabili prirejeni enačbi 2.5 in 2.6:

$$x = \frac{\frac{a_0}{2}}{\pi} 2^0 (\lambda + \pi) = \frac{a_0}{2\pi} (\lambda + \pi)\ \text{enot} \tag{3.13}$$

$$y = a_0 - \frac{\frac{a_0}{2}}{\pi} 2^0 \left(\pi - \ln\left[\tan\left(\frac{\pi}{4} + \frac{\varphi}{2}\right)\right]\right) = a_0 - \frac{a_0}{2\pi}\left(\pi - \ln\left[\tan\left(\frac{\pi}{4} + \frac{\varphi}{2}\right)\right]\right)\ \text{enot} \tag{3.14}$$

kjer x in y ustrezata komponentam x in z vektorja lokacije objekta, ki predstavlja dani sistem. a_0 predstavlja dolžino stranice na stopnji povečave 0 v enotah sveta.

S tem je definiran položaj sistema v dveh dimenzijah. Komponenta y, ki predstavlja višino, je nastavljena na višino ploščic največje stopnje povečave. To velja za vse sisteme razen tistih, katerih osnovna dimenzija bitke je zrak. Tovrstni sistemi imajo prednastavljeno višino oziroma komponento y, ki se skozi vizualizacijo ne spreminja.

Pozicioniranje enot ni tako zapleteno, saj simulator JCATS trenutno ne podpira prenosa enot PDU, ki bi vsebovale informacije o entitetah tipa enota. Prenos informacij o enotah bo podprt v naslednji verziji simulatorja JCATS.

Pozicioniranje enot je implementirano s povprečenjem lokacij neposrednih članov enote. Algoritem posodabljanja lokacije enote deluje tako, da ob spremembi lokacije sistema sproži funkcijo, ki rekurzivno posodobi lokacije vseh enot, katerih član je dani objekt. Rekurzija lokacije posodablja od spodnje do vrhnje enote.

3.6.2 Premikanje enot in sistemov

Kot že omenjeno, simulator JCATS ne podpira enot PDU za entitete tipa enota, zato je podobno kot pri pozicioniranju, premikanje enot implementirano na podlagi premikanja sistemov.

Tako kot pozicioniranje tudi premikanje sistemov določajo enote PDU tipa *Entity State* preko polja *Entity Linear Velocity* (linearna hitrost entitete). To polje vsebuje komponente x, y in z, ki so zapisane v 32-bitni plavajoči vejici in predstavljajo vektor hitrosti dane entitete. Vektor hitrosti je podan v kartezičnih koordinatah, kodiranih po standardu WGS 84, v m/s. V kolikor vektor hitrosti ni enak 0, se entiteta premika.

Za premikanje sistema moramo igralnemu objektu, ki predstavlja sistem, nastaviti dva parametra, in sicer vektor smeri gibanja ter hitrost v m/s. Vrednosti parametrov izračunamo s pomočjo vektorja hitrosti in kartezičnih koordinat lokacije entitete. S pomočjo postopka, predstavljenega v poglavju 3.6.1, najprej izračunamo koordinate sveta za lokacijo entitete in nato še koordinate sveta vektorja, ki ga dobimo kot rezultat vsote vektorja hitrosti in kartezičnih koordinat lokacije entitete. Tako dobimo dve točki, s pomočjo katerih izračunamo smerni vektor in hitrost premikanja, saj vemo, da se bo objekt v času 1 s premaknil iz točke 1 v točko 2.

Premikanje enot je zajeto v algoritmu posodabljanja lokacije enote, ki je opisan v poglavju 3.6.1.

3.6.3 Orientacija sistemov

Ker sisteme predstavljajo 3D modeli, je pomembna tudi orientacija igralnih objektov.

Orientacija je določena na dva načina. Če se sistem premika, orientacijo predstavlja smerni vektor gibanja. V nasprotnem primeru, ko sistem miruje, orientacijo predstavlja smerni vektor, ki se izračuna ob posodobitvi lokacije sistema (glej poglavje 3.6.2).

Poglavje 4

Zaključek

V tem diplomskem delu smo predstavili način implementacije aplikacije za 3D vizualizacijo dogajanja na sodobnem bojišču. Za razvoj vizualizatorja smo uporabili pogon *Unity* in njegove komponente. Komunikacijo med simulatorjem in vizualizatorjem smo implementirali s pomočjo standarda DIS enkapsuliranega v protokol UDP. Aplikacija je primarno namenjena vizualiziranju simulatorja JCATS, vendar lahko prikazuje tudi stanja entitet, ki so bila preko omrežja poslana iz drugega simulatorja v skladu s standardom DIS.

V vizualizatorju smo preko izbirnika datotek uporabniku omogočili izbiro lastnih dokumentov XML, s čimer uporabnika nismo omejili na končni seznam možnih scenarijev. Pri implementaciji smo se osredotočili predvsem na pravilnost prikaza in izgled grafičnega vmesnika. Za implementacijo okolja smo izbrali spletno Mercatorjevo projekcijo in realizirali sistem nivojev podobnosti, kar je izdatno pripomoglo k uporabniški izkušnji. Za prikaz okolja smo uporabili satelitske ploščice pridobljene iz strežnikov podjetja *Google*. Za pridobivanje ploščic iz strežnika smo definirali storitev HTTP, poleg tega smo nove ploščice shranjevali na disk, s čimer smo strežnik razbremenili. Dodali smo uporabniški vmesnik, ki je razširil uporabniško izkušnjo vizualizatorja. V sklopu uporabniškega vmesnika smo implementirali izris drevesne strukture, okna s podrobnostmi o entiteti, programskega dnevnika in navigacijskega kom-

pasa. S pomočjo tipkovnice in miške smo definirali vhode za povezavo med uporabnikom in vizualizatorjem.

Vizualizator je sposoben interpretiranja dveh vrst entitet, in sicer enot in sistemov. Enote so v sistemu prikazane kot ikone, ki so generirane na podlagi standarda za simbologijo zveze NATO. Ikone smo pridobili s pomočjo zunanje spletne aplikacije. Sistemi so v vizualizatorju predstavljeni kot 3D modeli, ki so bili pridobljeni iz spleta in optimizirani v programu *Houdini FX*. Za povezovanje med ustreznimi vrstami enot in simbolov smo uporabili poseben sistem, ki uporablja dokument XML kot knjižnico oziroma slovar tipov enot. Enako smo implementirali povezovanje sistemov z ustreznimi modeli.

Za pozicioniranje entitet smo uporabili Bowringovo aproksimacijsko metodo, s katero smo kartezične koordinate, pridobljene s pomočjo standarda DIS, prevedli v geografske. Da smo lahko entiteto pravilno pozicionirali, smo morali geografske koordinate prevesti v koordinate spletne Mercatorjeve projekcije, pri čimer smo uporabili lastni modificirani formuli, ki sta bili izpeljani iz standardnih enačb za spletno Mercatorjevo projekcijo. S pomočjo Bowringove metode smo implementirali izračun nadmorske višine dane entitete.

Poleg osnovnega okolja smo izdelali prototip, v katerem smo predstavili način dinamične generacije terena, na katerega smo pripeli satelitske ploščice pridobljene iz strežnika podjetja *Google*. Tudi pri terenu smo implementirali algoritem nivojev podrobnosti, pri čimer smo podatke o višinah pridobili iz podatkovne baze projekta SRTM.

4.1 Nadaljnje delo

Ker smo se pri implementacij osredotočili predvsem na pravilnost prikaza in izgled okolja, smo naleteli na nekaj performančnih problemov. Ob velikem številu sodelujočih entitet se zaradi prepodrobnih modelov izvajanje zelo upočasni, kar pomeni, da število generiranih sličic na sekundo zelo upade. Da bi rešili problem s prepodrobnimi modeli, bi bilo potrebno modeliranje namen-

skih 3D objektov, ki bi bili sestavljeni iz minimalnega števila poligonov.

Vizualizator bi lahko nadgradili z uporabo terenskega prikaza, ki je kot prototip predstavljen v tem diplomskem delu.

Pri produkcijski rabi bi bil nujen razvoj lastnega strežnika za pridobivanje satelitskih ploščic, saj omrežje, na katerem v SV izvajajo simulacijsko vadbo, nima dostopa do globalnega interneta in tako posledično do strežnikov podjetja *Google*. Implementacija strežnika bi bila koristna tudi iz varnostnega vidika.

Namesto standarda DIS bi lahko kot komunikacijski protokol med vizualizatorjem in simulatorjem uporabili standard HLA, ki razširja funkcionalnosti standarda DIS.

4.2 Sklep

V tem diplomskem delu smo razvili aplikacijo za 3D vizualizacijo dogajanja na bojišču, pri čimer smo za komunikacijski protokol uporabili standard DIS. S tem smo zadovoljili osnovne cilje tega diplomskega dela. Z uporabo Bowringove aproksimacijske metode smo zagotovili minimalno napako pozicioniranja in posodabljanja stanj entitet.

Z implementacijo nivojev podobnosti smo okolje vizualizatorja nadgradili z naprednim prikazom, ki omogoča sprejemljive rezultate v realnem času. Poleg tega smo raziskali alternativne načine prikaza terena, s čimer smo izpolnili zastavljene cilje tega diplomskega dela.

Literatura

[1] IEEE Standard for Distributed Interactive Simulation–Application Protocols. *IEEE Std 1278.1-2012 (Revision of IEEE Std 1278.1-1995)*, strani 1–747, Dec 2012.

[2] WGS 84. `http://www.unoosa.org/pdf/icg/2012/template/WGS_84.pdf`, 2016. [Dostopno na spletu; dostopano 26.8.2016].

[3] B. R. Bowring. Transformation from spatial to geographical coordinates. *Survey Review*, 23(181):323–327, 1976. Dostopno na `http://dx.doi.org/10.1179/sre.1976.23.181.323`.

[4] Rick Brewster. paint.net. `http://www.getpaint.net/index.html`, 2016. [Dostopno na spletu; dostopano 1.9.2016].

[5] Kjell Magne Fauske. Joint military symbology explorer. `http://explorer.milsymb.net/`, 2016. [Dostopno na spletu; dostopano 21.7.2016].

[6] Google. Google static maps. `https://maps.googleapis.com/maps/api/staticmap?latutide=0&zoom=1&size=600x600&maptype=terrain`, 2016. [Dostopno na spletu; dostopano 27.08.2016].

[7] J.-N. Paquin J. Bélanger, P. Venne. The What, Where and Why of Real-Time Simulation. 2010. Dostopno na `http://www.opal-rt.com/sites/default/files/technical_papers/PES-GM-Tutorial_04%20-%20Real%20Time%20Simulation.pdf`.

[8] JCATS. JCATS Capabilities Brief Update. https://csl.llnl.gov/content/assets/docs/JCATS-Capabilities-Brief-Update-20DEC2014.pdf, 2014. [Dostopno na spletu; dostopano 10.8.2016].

[9] JCATS. JCATS LLNL Brochure 2015. https://csl.llnl.gov/content/assets/docs/JCATS-LLNL-Brochure-2015.pdf, 2015. [Dostopno na spletu; dostopano 10.8.2016].

[10] NATO. *NATO Joint Military Symbology APP-6(C)*. NATO, Boulevard Leopold III, 1110 Brussels, Belgium, 2011.

[11] SAAB. TES training. http://saab.com/globalassets/commercial/land/training-and-simulation/live-training/manpack300/tes-takes-training-to-the-next-level.pdf, 2016. [Dostopno na spletu; dostopano 15.8.2016].

[12] Side Effects Software. Houdini FX. https://www.sidefx.com/, 2016. [Dostopno na spletu; dostopano 1.9.2016].

[13] SRTM. Continent def. http://dds.cr.usgs.gov/srtm/version2_1/Documentation/Continent_def.gif, 2016. [Dostopno na spletu; dostopano 28.08.2016].

[14] Unity Technologies. Unity (game engine). https://unity3d.com/, 2016. [Dostopno na spletu; dostopano 1.9.2016].

[15] Wikipedia. DTED — wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/DTED, 2016. [Dostopno na spletu; dostopano 2.8.2016].

[16] Wikipedia. Geographic information system — wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Geographic_information_system, 2016. [Dostopno na spletu; dostopano 25.7.2016].

[17] Wikipedia. Live, virtual and constructive — wikipedia, the free encyclopedia. `https://en.wikipedia.org/wiki/Live,_virtual,_and_constructive`, 2016. [Dostopno na spletu; dostopano 22.8.2016].

[18] Wikipedia. Map projection — wikipedia, the free encyclopedia. `https://en.wikipedia.org/wiki/Map_projection`, 2016. [Dostopno na spletu; dostopano 10.8.2016].

[19] Wikipedia. Mercator projection — wikipedia, the free encyclopedia. `https://en.wikipedia.org/wiki/Mercator_projection`, 2016. [Dostopno na spletu; dostopano 12.8.2016].

[20] Wikipedia. Military simulation — wikipedia, the free encyclopedia. `https://en.wikipedia.org/wiki/Military_simulation`, 2016. [Dostopno na spletu; dostopano 21.8.2016].

[21] Wikipedia. Shuttle Radar Topography Mission — wikipedia, the free encyclopedia. `https://en.wikipedia.org/wiki/Shuttle_Radar_Topography_Mission`, 2016. [Dostopno na spletu; dostopano 2.8.2016].

[22] Wikipedia. Tissot Mercator Projection — wikipedia, the free encyclopedia. `https://upload.wikimedia.org/wikipedia/commons/8/87/Tissot_mercator.png`, 2016. [Dostopno na spletu; dostopano 25.08.2016].

[23] Wikipedia. Tissot world from space — wikipedia, the free encyclopedia. `https://upload.wikimedia.org/wikipedia/commons/0/0e/Tissot_world_from_space.png`, 2016. [Dostopno na spletu; dostopano 23.08.2016].

[24] Wikipedia. VBS1. `https://en.wikipedia.org/wiki/VBS1`, 2016. [Dostopno na spletu; dostopano 22.8.2016].

[25] Wikipedia. Web Mercator — wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Web_Mercator, 2016. [Dostopno na spletu; dostopano 12.8.2016].

[26] Wikipedia. WGS 84 reference frame vector graphic — wikipedia, the free encyclopedia. https://upload.wikimedia.org/wikipedia/commons/5/56/WGS_84_reference_frame_%28vector_graphic%29.svg, 2016. [Dostopno na spletu; dostopano 26.08.2016].

[27] Wikipedia. World Geodetic System — wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/World_Geodetic_System, 2016. [Dostopno na spletu; dostopano 26.8.2016].