SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 3199

VIZUALIZACIJA METEOROLOŠKIH PODATAKA

Ivan Hakštok

Zagreb, lipanj 2013.

Sadržaj

1. Uvod....1
2. Meteorološki podaci....2
2.1. Svojstva meteoroloških čimbenika....2
2.2. Načini prikaza meteoroloških čimbenika....3
2.2.1. Temperatura zraka....3
2.2.2. Vjetar....4
2.2.3. Naoblaka....6
2.2.4. Osunčanost....7
2.2.5. Padaline....9
2.2.6. Vidljivost....10
3. Programska implementacija....12
3.1. Radna okolina....12
3.2. Ulazni podaci....13
3.3. Baza podataka....14
3.4. Grafičko sučelje....15
3.5. Priručni grafovi....17
3.6. Izrada grafičkih modela....18
3.7. Trodimenzionalni prikaz....19
3.8. Korištenje programskog rješenja....23
3.9. Performanse....24
4. Zaključak....26
5. Literatura....27
6. Sažetak....28
7. Abstract....29

1. Uvod

Meteorologija je jedno od najkompliciranijih, ali i najpotrebnijih područja znanosti, te obuhvaća veliki niz čimbenika koji utječu na rezultate dobivene izračunima konačnih stanja u atmosferi. Količina podataka potrebna za takve, procesorski zahtjevne izračune iziskuje određene načine vizualizacije tih podataka radi boljeg prikaza i razumijevanja što oni u stvari predstavljaju te njihovu međusobnu povezanost.

Podaci kao što su primjerice količina sunčevog zračenja (insolacija), postotak oblačnosti i količina padalina moguće je vizualizirati izravno, jer su njihovi čimbenici oku vidljivi, dok je podatke kao što su brzina vjetra, temperatura i tlak zraka nemoguće izravno prikazati na oku vidljiv način te je stoga njihovu vrijednost potrebno prikazati pomoću grafova, ispisanih brojčanih vrijednosti ili pomoću utjecaja koje imaju na okolinu.

U ovom radu biti će istraženi razni načini prikaza dostupnih podataka, bilo preko grafova, raznih dvodimenzionalnih i trodimenzionalnih prikaza ili pomoću vizualizacije njihovog utjecaja na neki prostor. Za praktični dio rada, cilj je napraviti aplikaciju koja koristi istražene metode vizualizacije dobivenih podataka. Aplikacija ima jednostavno korisničko sučelje te je moguće jednostavno mijenjati određene parametre da bi se omogućio širok raspon načina vizualizacije.

2. Meteorološki podaci

2.1. Svojstva meteoroloških čimbenika

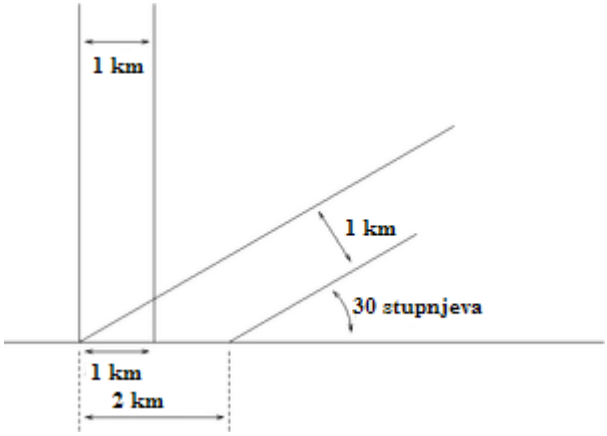
Neki od najbitnijih podataka potrebnih u meteorologiji su temperatura, tlak zraka, relativna vlažnost zraka, smjer i jačina vjetra te količina padalina. Da bi podatke bilo moguće prikazati, potrebno je razumjeti na koji način se oni manifestiraju na okolinu, na temelju čega je moguće zaključiti koje metode bi bile najbolje za prikaz.

Temperatura zraka je fizička mjera količine toplinske energije zraka. Temperatura zraka utječe na temperature kopnenih i vodenih masa, te kod vodenih masa može dovesti do isparavanja, što rezultira povećanjem vlažnosti zraka, padalinama i naoblakom. U slučaju pada temperature zraka iznad tla, može doći i do pojave magle, što rezultira smanjenjem vidljivosti.

Tlak zraka je tlak kojim cjelokupna masa zraka djeluje na površinu zemlje. Standardni tlak zraka na razini mora ima vrijednost 101.325 kPa, no opada prilikom povećanja nadmorske visine, za otprilike 4% na svakih 300 metara.

Vlažnost zraka je količina vodene pare u atmosferi. Vlažnost se može izražavati kao apsolutna i kao relativna. Apsolutna vlažnost je masa vodene pare podijeljena s jediničnim volumenom (jedan kubni metar). Relativna vlažnost je omjer parcijalnog tlaka vodene pare i tlaka zasićene vodene pare na određenoj temperaturi.

Insolacija ili osunčanost je količina energije sunčevog zračenja primljena kroz neko vrijeme. Osunčanost je veća što je kut između tla i sunca bliži 90 stupnjeva. Na slici 1. se vidi da, zbog manjeg kuta, ista količina sunčevog zračenja djeluje na dva puta veću površinu. Na Zemlji, područja s najvećim vrijednostima insolacije nalaze se bliže ekvatoru.



Slika 1: Ovisnost insolacije o kutu upada sunčevih zraka

Vjetar je pojava kretanja zračnih masa iz područja većeg u područje manjeg tlaka zraka. Vjetar ima svoju brzinu i smjer, te se njegovo djelovanje može promatrati na objektima nad kojima vjetar djeluje.

2.2. Načini prikaza meteoroloških čimbenika

2.2.1. Temperatura zraka

Temperatura zraka je možda i najvažniji meteorološki podatak, no prikaz temperature teško je izravno ostvariv. Jedan od načina na koji se može vidjeti utjecaj temperature na zrak je pojava efekta fatamorgane. Uslijed velikih temperatura, može doći do zagrijavanja podloge, što ujedno zagrijava i zrak koji se nalazi uz tu podlogu. Takav zrak je rjeđi, što proizlazi iz formule za idealni plin, uz aproksimaciju zraka kao idealnog plina.

Povećanjem temperature se smanjuje gustoća plina, uz pretpostavku da se radi o izobarnom procesu. Do tog zaključka može se doći iz sljedećih formula:

$$pV = nRT$$

$$n = \frac{m}{M}$$

$$P = \frac{m}{V} = \frac{pM}{RT}$$

gdje je:

P - gustoća zraka

p - tlak zraka

M - molekularna masa

R - plinska konstanta, *R = 8.3144621 J/mol K*

T - temperatura

Budući da je zrak uz podlogu rjeđi, zrake koje dolaze na granično područje između dvije mase zraka različite temperature pod određenim kutom bivaju reflektirane [1]. Vizualizaciju fatamorgane moguće je napraviti jednostavnom reflektirajućom podlogom, no taj način prikaza je pogodan samo za određene podloge pri većim temperaturama.

2.2.2. Vjetar

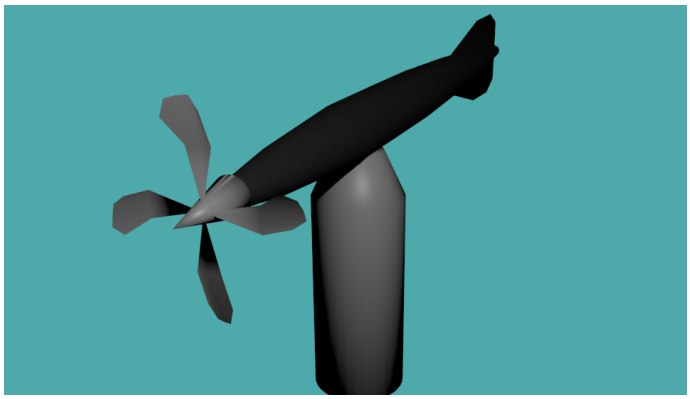
Vjetar kao strujanje zraka nije moguće izravno prikazati, iz razloga što zrak sam po sebi nije vidljiv. Međutim, djelovanje vjetra nad objektima može se vrlo lako vidjeti, a i prikazati. Kod meteoroloških postaja, brzina vjetra se mjeri anemometrom.

Slika 2: Anemometar s čašicama

Postoje razni oblici anemometara, no za prikaz su najpogodniji anemometar u obliku vjetrenjače i anemometar s čašicama. Oba ta tipa anemometra se sastoje

od statičnog i rotirajućeg dijela. Kod anemometara u obliku vjetrenjače, cijeli anemometar je tipično postavljen horizontalno, dok je anemometar s čašicama postavljen vertikalno. Ovo je bitno kod izrade grafičkog prikaza, jer je moguće jednostavno prikazati rotaciju krilaca anemometra. Os rotacije je kod anemometra u obliku vjetrenjače paralelna s ravninom xy, dok je kod anemometra s čašicama paralelna sa z-osi. Za prikaz će pogodniji biti anemometar u obliku vjetrenjače, jer je njime istovremeno moguće prikazati i smjer vjetra [2].

Budući da vjetar aproksimativno puše paralelno s podlogom, smjer vjetra može se prikazati kao vektor paralelan s ravninom xy. Ako je cijeli anemometar učvršćen na rotirajuću osovinu, moguće ga je rotirati oko z osi tako da mu vektor smjera bude suprotan smjeru vektora vjetra.



Slika 3: Model anemometra

Os rotacije krila je tada pravac kojemu je vektor smjera suprotan vektoru smjera vjetra, a sadrži točku koja je zajednički centar svih krila. Omjer brzine vjetra i brzine rotacije vrha je najčešće jedan, te se kutna brzina krila može izračunati preko formula:

$$v_v = v_k = 2\pi r f_k$$

$$\omega_k = 2\pi f_k = \frac{v_v}{r}$$

gdje je:

v_v – brzina vjetra

v_k – brzina krila

f_k – frekvencija krila

r – polumjer krila

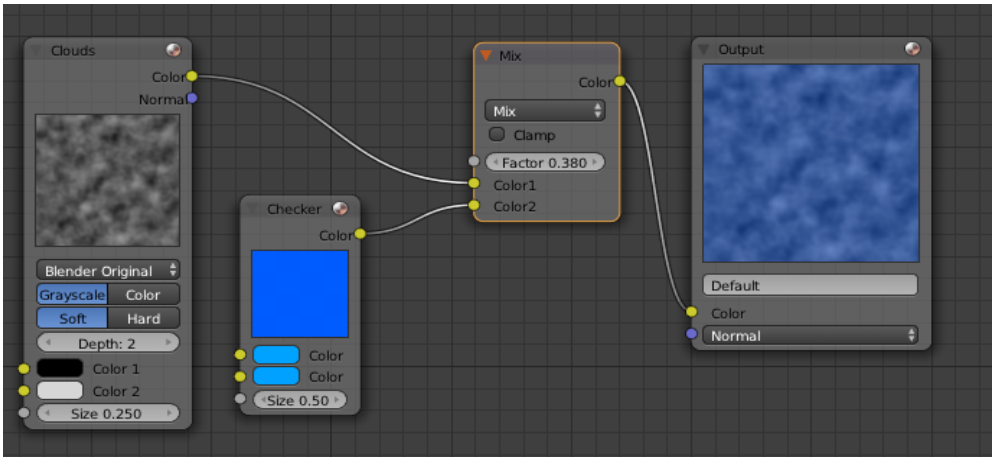
ωk – kutna brzina krila

Smjer vjetra uobičajeno se izražava stupnjevima, gdje je sjever na 0°, istok na 90°, jug na 180° a zapad na 270°.

2.2.3. Naoblaka

Oblaci su nakupine kondenzirane vodene pare u Zemljinoj atmosferi. Pojava naoblake je gotovo svakodnevna, a količinu naoblake moguće je odrediti omjerom površine vidljivog neba prekrivene oblacima i ukupne površine vidljivog neba. Naoblaka se tipično izražava skalama od 0 do 8 ili od 0 do 10.

Postoje razni načini prikaza naoblake. Moguće je koristiti generator oblaka koji bi na temelju danih podataka generirao približnu pokrivenost neba naoblakom. Alternativni način je korištenje drukčije teksture za svaki od stupnjeva pokrivenosti. Učitavanjem podatka za neki sad bi se prikazala tekstura čija pokrivenost odgovara učitanom podatku. Treći način je korištenje interpolacije između dvije teksture, gdje je jedna potpuno čisto nebo, a druga potpuno pokriveno nebo [3].



Slika 4: Primjer interpolacije između dvije teksture

Kako se može uočiti na slici 4., interpolacija između dvije teksture i nije baš korisna jer naoblaka često nije ravnomjerno raspoređena. Budući da je naoblaka u našim podacima zadana skalom, najbolji način za prikaz raznih stupnjeva naoblake bio bi generiranjem teksture neba i naoblake za svaki stupanj, te mijenjanjem teksture prema učitanom podatku.

2.2.4. Osunčanost

Osunčanost tijekom dana može se povezati s naoblakom. Ako u nekom trenutku postoji razina osunčanosti, a u tom istom trenutku postoji i neka vrijednost naoblake, može se zaključiti da dio nebeske sfere u kojem se nalazi sunce nije bio prekriven oblacima, ili da je sloj oblaka bio dovoljno tanak da sunčeve zrake u nekoj mjeri prodiru kroz njega.

Normalno je za očekivati da će tijekom ljetnih dana osunčanost biti veća nego tijekom zimskih, no budući da osunčanost ne govori ništa o jačini sunčevog zračenja kao i o položaju sunca, moguće je čak i tijekom zimskih dana imati veliku vrijednost satne osunčanosti.

Položaj sunca mora biti izračunat na druge načine. Za izračun je potrebno transformirati položaj sunca u horizontalni koordinatni sustav u čijem se središtu nalazi promatrač. Zatim je potrebno izračunati solarni azimut γ i zenitnu udaljenost θ_z [4]. Izračun se može napraviti prema sljedećim formulama:

$$n = 367\mathrm{y} - \left\lfloor 7\frac{(y + \left\lfloor \frac{m+9}{12} \right\rfloor)}{4} \right\rfloor + \left\lfloor \frac{275\mathrm{m}}{9} \right\rfloor + d - 730531.5 + \frac{ut}{24}$$

$$\Omega = 2.1429 - 0.0010394594\mathrm{n}$$

$$L = 4.8950630 + 0.0172027916898\mathrm{n}$$

$$g = 6.24006 + 0.0172019699\mathrm{n}$$

$$l = L + 0.03341607\sin(g) + 0.00034894\sin(2\mathrm{g}) - 0.0001134 - 0.0000203\sin(\Omega)$$

$$ep = 0.4090928 - 6.2140 * 10^{-9} n \cos(\Omega)$$

$$ra = tg^{-1}[\frac{\cos(ep)\sin(l)}{\cos(l)}]$$

$$\delta = \sin^{-1}[\sin(ep)\sin(l)]$$

$$gmst = 6.6974243242 + 0.0657098283\mathrm{n} + ut$$

$$lmst = (gmst * 15 + long) * (\frac{\pi}{180})$$

$$\omega = lmst - ra$$

$$\Theta_z = \cos^{-1}[\cos(\Phi)\cos(\omega)\cos(\delta) + \sin(\delta)\sin(\Phi)]$$

$$\gamma = tg^{-1}[\frac{-\sin(\omega)}{tg(\delta)\cos(\Phi) - \sin(\Phi)\cos(\omega)}]$$

gdje je:

n – julijanski dani protekli od 1. siječnja 2000.

L – srednja longituda sunca

g – srednja anomalija sunca

l – ekliptična longituda sunca

ep – ukošenost ekliptike

ra – desni uspon

δ – deklinacija

$gmst$ – srednje Greenwich zvjezdano vrijeme

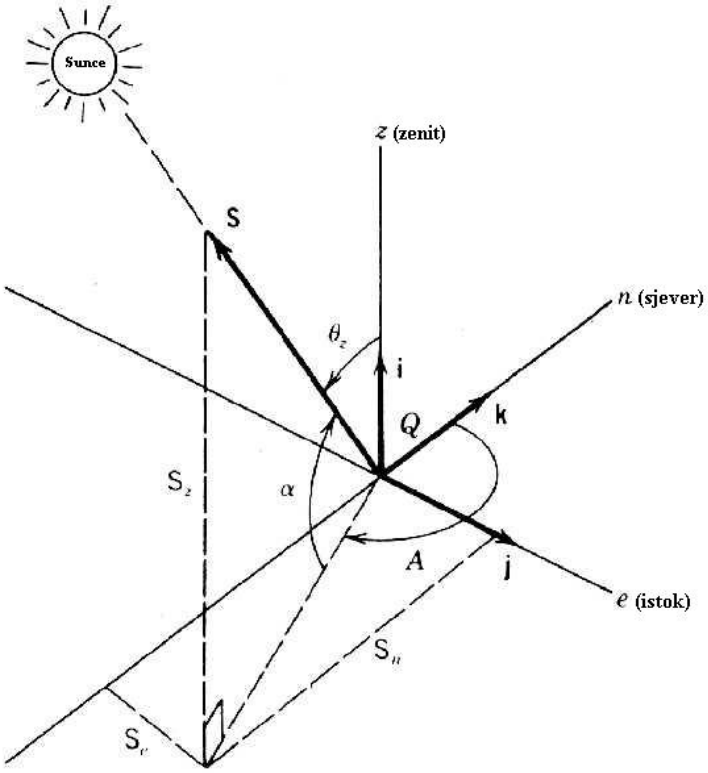
$lmst$ – lokalno srednje zvjezdano vrijeme

$long$ – zemljopisna dužina promatrača

Φ – zemljopisna širina promatrača

ω – satni kut

Uz pomoć horizontalnih koordinata, moguće je postaviti model osvjetljenja koji predstavlja sunce unutar koordinatnog sustava scene. Još je potrebna i udaljenost, no za nju ne treba uzimati stvarnu udaljenost Zemlje i Sunca već je dovoljno izabrati udaljenost takvu da sunce obasjava cijelu scenu, a intenzitet sunca tada je moguće prilagoditi optimalnom osvjetljenju.



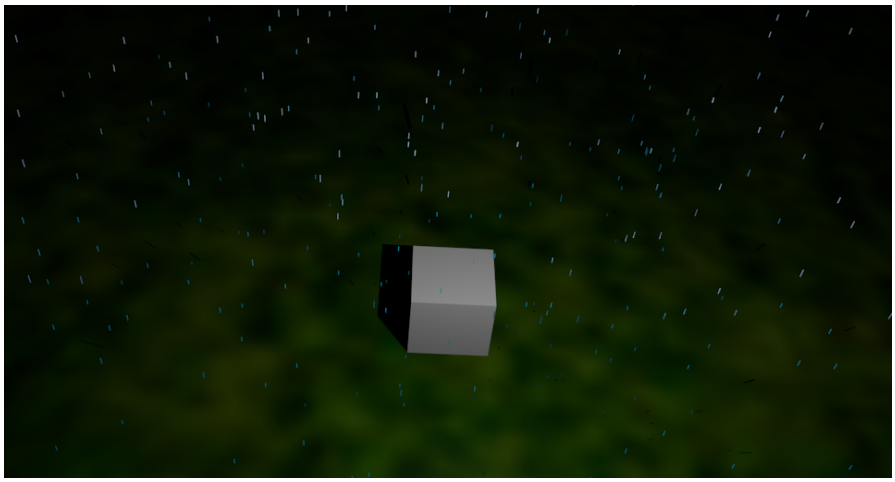
Slika 5: Sunce u horizontalnom koordinatnom sustavu

2.2.5. Padaline

Vizualizaciju padalina najjednostavnije je vizualizirati pomoću sustava čestica. Iz podataka o količini padalina moguće je približno zaključiti gustoću čestica u sustavu potrebnom da bi ta padalina bila vizualizirana. Problem nastaje kod tipa padalina – kiša, snijeg ili tuča. Očito je za svaku padalinu potreban drukčiji način vizualizacije, te se potreban način inicijalizira analizom ostalih podataka.

Iako se tri navedene vrste padalina poprilično razlikuju po brzini i obliku, ne padaju u istim uvjetima, pa je te uvjete za njihov nastanak potrebno procijeniti iz danih podataka. Radi pojednostavljenja vizualizacije, model tuče će biti zanemaren, jer su naleti tuče kratkotrajni te nisu toliko bitni za prikaz. Sada je moguće pretpostaviti da kiša pada u uvjetima kada je temperatura zraka veća od nule, dok snijeg pada kada je temperatura zraka manja od nule. U prirodi, kiša i snijeg ovise o još par faktora, no na temelju danih podataka ionako se ne može uvijek sa sigurnošću odrediti točan tip padaline u danom vremenskom intervalu.

Kod sustava čestica, lako je moguće prilagoditi brzinu ovisno o vrsti padalina, te je također moguće vizualizirati i utjecaj vjetra na padaline. Mogući problem se javlja kod emitera čestica. U slučaju da su za naoblaku korišteni modeli, moguće je iskoristiti taj model, no u većini slučajeva, za oblake se koriste teksture. Jedan od načina za stvaranje emitera u takvom slučaju bio bi model koji prisustvuje u sceni na način da emitira čestice, no zanemaruje ga se prilikom iscrtavanja. Takav primjer može se vidjeti na slici 6.



Slika 6: Kiša kao sustav čestica s nevidljivim emiterom

2.2.6. Vidljivost

Vidljivost je podatak koji govori na kojoj najvećoj udaljenosti promatrač može razlikovati svjetlost ili predmete, najčešće mjerena u kilometrima. Najčešće se definira kao najveća udaljenost na kojoj se crni objekt određenih dimenzija vidi na svijetloj podlozi [5].

Mala vidljivost najčešće je uzrokovana maglom ili izmaglicom, koje bi trebalo vizualizirati za dovoljno male vrijednosti vidljivosti. Budući da je magla pojava kroz koju se moguće kretati, treba koristiti način prikaza takav da je raspodjela magle jednaka bez obzira na to na kojem se mjestu nalazimo. Neki od načina prikaza magle su prikaz česticama i prikaz teksturama.

Kod prikaza česticama, nedostatak je što je kod guste magle potreban veliki broj čestica za prikaz. Međutim, kod složenih grafičkih sustava, ovakav način prikaza je neophodan, jer u kombinaciji s volumetrijskim osvjetljenjem znatno utječe na realizam scene.

Drugi način prikaza je prikaz teksturama. Kod ovakvog prikaza, magla se prikazuje kao niz slojeva prozirnih tekstura. Razina vidljivosti može utjecati na gustoću slojeva i količinu prozirnosti. Prednost ovog načina je relativno jednostavna izrada i integracija u prikaz, no nije pogodno za složene modele jer nema mogućnost dovoljne interakcije s osvjetljenjem.

3. Programska implementacija

Vizualizacija meteoroloških podataka je za ovaj rad implementirana pomoću programskog jezika Python i biblioteke Ogre3d, uz korištenje programa Blender za izradu potrebnih grafičkih modela. U idućim poglavljima prikazan je tijek izrade odgovarajućeg programskog rješenja. Ključne faze izrade su:

- postavljanje radne okoline za izradu programskog rješenja pomoću programskog jezika python
- izrada baze i načina učitavanja dostupnih podataka u bazu
- izrada grafičkog sučelja i potrebnih metoda za rad s njim
- izrada modela potrebnih za prikaz podataka
- izrada sustava vizualizacije i njegovih funkcija
- implementacija istraženih načina vizualizacije

3.1. Radna okolina

Programska podrška razvijena je unutar integriranog radnog okruženja *Microsoft Visual Studio 2010*, uz upotrebu dodatka *Python Tools* koji omogućuje pokretanje programa napisanih u programskom jeziku Python unutar navedenog radnog okruženja. Sam interpreter za jezik Python je besplatan i dostupan na http://www.python.org/, dok je *Microsoft Visual Studio 2010* komercijalni program tvrtke Microsoft. Za potrebe ove programske podrške korištena je inačica Python 2.7.4.

Za potrebe razvoja potrebne programske podrške, bilo je potrebno učitati biblioteke za rad s numeričkim vrijednostima, grafičkim sučeljem, grafovima i grafičkim prikazom trodimenzionalnih objekata. S tom svrhom, preuzete su sljedeće biblioteke:

- Numpy, Scipy: Biblioteke koje implementiraju mnogobrojne matematičke funkcije, što je bilo neophodno za rad s dostupnim

podacima.

- wxPython: Biblioteka za stvaranje i rad s grafičkim sučeljem.
- matplotlib: Biblioteka za rad s prikazom grafova. Potrebno zbog jednostavnijeg prikaza dostupnih podataka, pogotovo onih koje nije moguće izravno vizualizirati.
- Ogre3d: Biblioteka za rad s trodimenzionalnim grafičkim prikazima. U ovom radu korištena je inačica za programski jezik Python pod nazivom Python Ogre. Potrebno za konačnu vizualizaciju podataka koje je moguće vizualizirati.

Biblioteke prilikom instalacije same postavljaju potrebne parametre unutar postavki programskog jezika Python, pa ih stoga nije potrebno posebno uključivati unutar radnog okruženja – moguće je odmah u kodu pozivati potrebne biblioteke.

3.2. Ulazni podaci

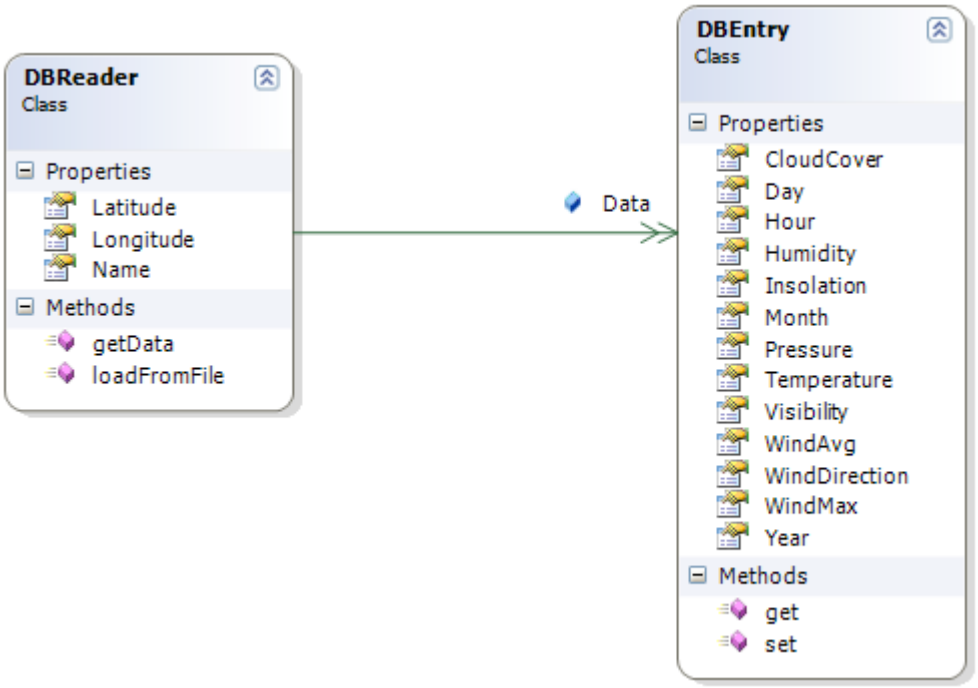
Ulazni podatci su zadani u obliku tekstualne datoteke. Prvi red sadrži informacije o meteorološkoj stanici s koje su podatci dobiveni. Informacije sadrže naziv stanice, nadmorsku visinu te geografsku dužinu i širinu. Drugi red sadrži opis jednog reda koji sadrži konkretne podatke. Treći red sadrži mjerne jedinice podataka.

Svi ostali redovi sadrže podatke koji su sa stanice dobiveni u razmaku od jednog sata. Dobiveni podaci su sat, dan, mjesec, godina, prosječna brzina vjetra, smjer vjetra, maksimalni satni udar vjetra, temperatura, relativna vlažnost, tlak zraka, osunčanost, količina padalina, trajanje padalina, količina naoblake i vidljivost. U ovom radu, korišteni su podaci u rasponu od 1. siječnja 2011. do 31. prosinca 2011.

3.3. Baza podataka

Baza podataka ostvarena je posebnim razredom koji sadrži funkcije za učitavanje podataka iz datoteke te rad s učitanim podatcima. Baza unutar sebe sadrži osnovne podatke o stanici iz koje su dobiveni podatci, te polje koje sadrži pokazivače na pojedine elemente baze.

Pojedini skupovi podataka baze također su implementirani svojim razredom. Kao svojstva razreda navedeni su pojedini podatci. Parsiranjem tekstualne datoteke, za svaki red osim naslovnog se stvara nova instanca razreda skupa podataka baze te joj se dodjeljuju podatci. Prikaz razreda može se vidjeti na slici 7.



Slika 7: Dijagram razreda baze

Prednost ovakve strukture je relativno lak pristup podatcima te mogućnost preglednog korištenja velike količine podataka. Budući da su pojedini skupovi podataka učitani slijedno po datumu i sadu, moguće ih je dohvatiti izračunom indeksa na temelju sata, dana, mjeseca i godine. Nedostatak je nešto veće

zauzeće memorije, te samim time potreba za učitavanjem veće količine podataka iz memorije, što dolazi kod izražaja prilikom učitavanja većeg broja podataka.

Poziv postupka za učitavanje iz datoteke vrši se na samom početku programa, dok se kod bilo kojeg idućeg pristupa bazi pristupa podacima koji su već učitani u memoriju. Takav način rada smanjuje broj potrebnih čitanja s tvrdog diska.

3.4. Grafičko sučelje

Grafičko sučelje ostvareno je korištenjem biblioteke wxPython. Način na koji se pomoću navedene biblioteke stvara grafičko sučelje je sljedeći:

- Svaki prozor je ili razred Frame ili razred naslijeđen iz navedenog razreda. Taj razred predstavlja standardni prozor unutar operacijskog sustava Windows.
- Unutar prozora, vrši se podjela danog prostora pomoću razreda Sizer. Taj razred parametarski dijeli prostor, no pomoću jedne instance razreda, prozor se ne može istovremeno podijeliti horizontalno i vertikalno. Međutim, moguće je unutar svakog pojedinačnog prostora nastalog korištenjem razreda Sizer stvoriti novu podjelu daljnjim korištenjem navedenog razreda.
- Unutar svake instance razreda Sizer, moguće je dodati proizvoljan broj zasebnih elemenata. Prostor prikaza se elementima dodjeljuje ravnopravno osim ako način dodjele nije izričito naveden.

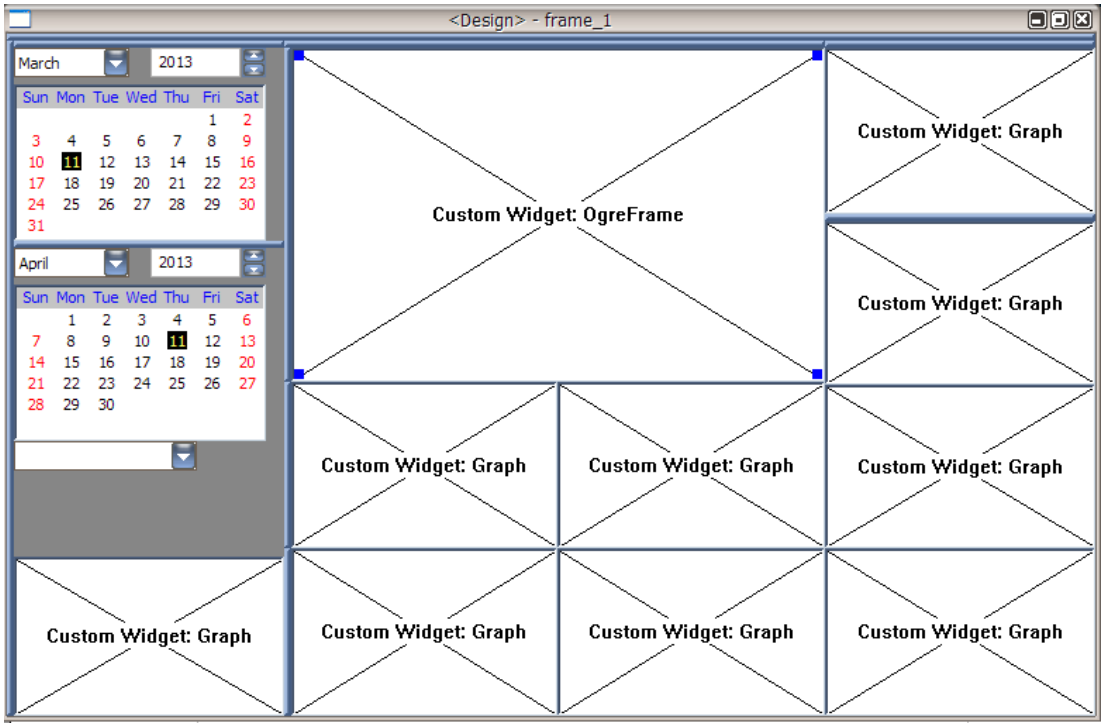
Za potrebe programske podrške ovog rada, stvorena su dva prozora, pomoću dvije instance razreda MyFrame. Glavni prozor sadrži podjelu na 13 podprostora pomoću instanci razreda Sizer, a unutar svakog podprozora se nalazi jedan element prikaza. Potrebni elementi prikaza za program su:

- Dvije instance razreda CalendarControl, koji služi za prikaz

kalendara s mogućnošću odabira datuma. Korištene su za postavljanje početnog i krajnjeg datuma prikaza.

- Devet instanci različitih razreda izvedenih iz razreda Frame koji služe za prikaz priručnih grafova. Više o njima bit će rečeno u poglavlju o grafovima.
- Instanca razreda ComboBox, koja služi za odabir točnog vremena koje se nalazi između datuma zadanih razredima CalendarControl.
- Instanca razreda OgreFrame, izvedenog iz razreda Frame, koja služi za ostvarivanje trodimenzionalnog grafičkog prikaza. Više o tome će biti rečeno u poglavlju o grafičkom prikazu.

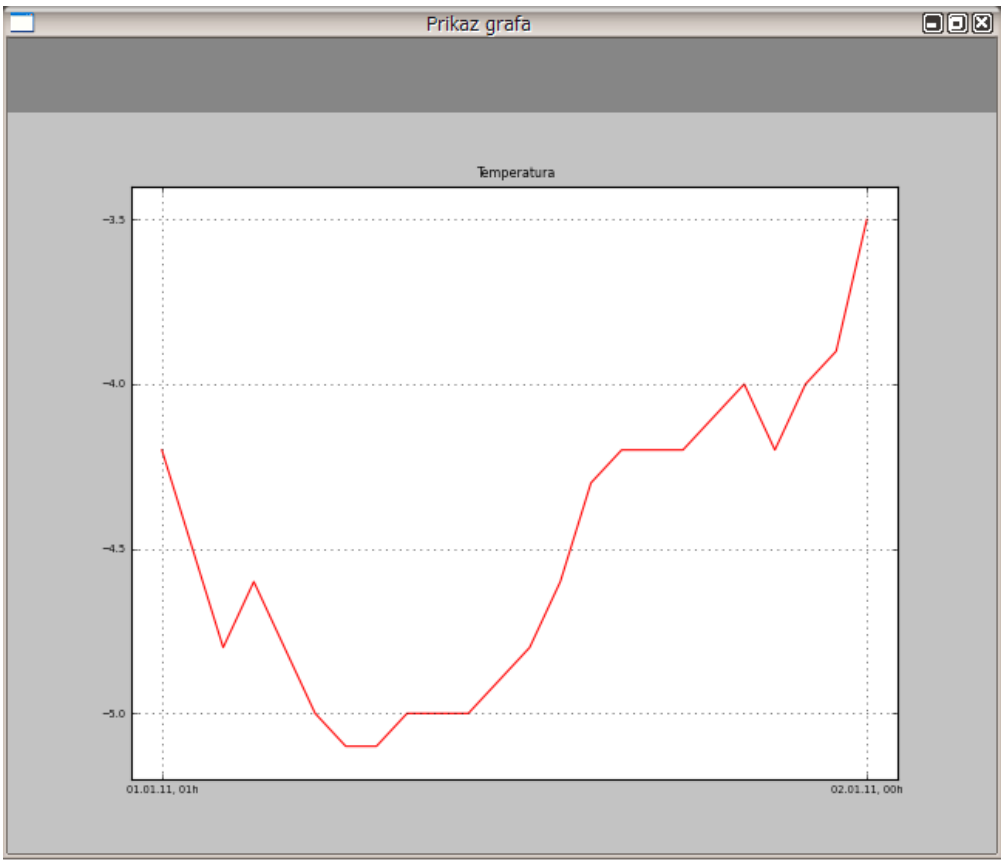
Grafičko sučelje dizajnirano je unutar programa wxGlade, međutim Microsoft Visual Studio 2010 nema podršku za datoteke napravljene u tom programu te je bilo potrebno izvesti Python kod iz programa wxGlade te taj kod otvoriti u obliku .py datoteke. Izrada grafičkog sučelja prikazana je na slici 7.



Slika 8: Izrada grafičkog sučelja

3.5. Priručni grafovi

Budući da nije moguće sve dostupne podatke vizualizirati izravno, a i budući da podatke koje je moguće vizualizirati najčešće nije moguće izravno iščitati iz prikaza, u program je potrebno dodati način prikaza vrijednosti podataka. U ovom radu, to je napravljeno uz pomoć grafova.



Slika 9: Primjer grafa

Na slici 9. može se vidjeti primjer grafa za temperaturu. Na x-osi su prikazani datumi, dok su na y-osi prikazane vrijednosti podatka u danom trenutku. Prikaz grafa ostvaren je pomoću biblioteke matplotlib, unutar koje je moguće jednostavnim zadavanjem skupa podataka iscrtati te iste podatke. Iznimka od ovog načina je graf za prikaz smjera vjetra, koji je zbog veće ilustrativnosti napravljen u polarnom koordinatnom sustavu. Unutar tog sustava, kut predstavlja smjer vjetra dok udaljenost od središta predstavlja broj sati unutar kojih je vjetar dolazio iz navedenog smjera.

Zbog toga što detaljan prikaz grafa zahtijeva dosta prostora na ekranu, omogućeno je odabirom nekog grafa unutar glavnog prikaza otvoriti taj graf u novom, zasebnom prozoru. Također je moguće aktivirati praćenje položaja miša, što je iskorišteno na način da se u gornjem lijevom kutu prikažu vrijednosti za x- i y-os na koje kursor trenutno pokazuje [6].

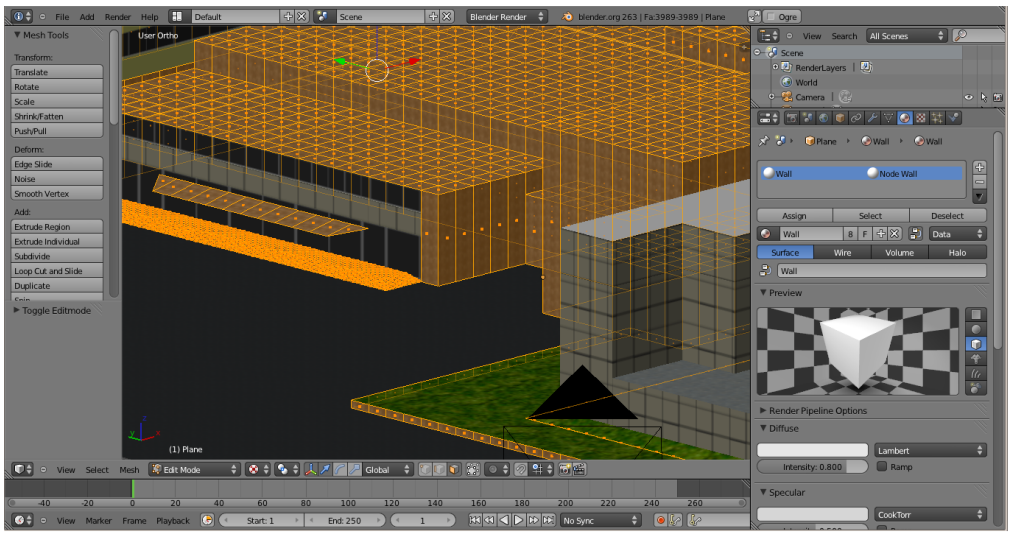
Prikaz grafovima je jednostavan za implementaciju i iz njega je moguće na lak način doći do vrijednosti podataka. To je pogotovo bitno ako je potrebno pokrenuti glavnu vizualizaciju s ciljem da se prikaže neki konkretni podatak.

3.6. Izrada grafičkih modela

Kako bi neki podatci mogli biti prikazani, potrebno je bilo napraviti odgovarajuće trodimenzionalne grafičke modele. Konkretno u ovom radu, potrebna je bila cijela scena nad kojom se prikazuju dani podatci. Korišten je model zgrade Fakulteta elektrotehnike i računarstva, te model anemometra za prikaz podataka o vjetru.

Grafički modeli napravljeni su pomoću programa Blender, koji je besplatan i otvorenog koda (*eng. Open-source*). Model zgrade sadrži otprilike 11000 vrhova, no to nema znatnog utjecaja na performanse programa. Model zgrade je zatim obojen teksturama zidova i raznih podloga napravljenih u programu GIMP.

Samo modeliranje koristi više različitih tehnika modeliranja. Uglavnom je zgrada napravljena korištenjem metode izvlačenja (*eng. Extrude*) iz podloge podijeljene na dovoljan broj manjih segmenata. Potom su dodani neki manji detalji. Bitno je napomenuti da zbog učitavanja modela pomoću biblioteke Ogre, model je potrebno podijeliti na više manjih podmodela, da bi se mogli uspješno učitati.



Slika 10: Modeliranje scene

3.7. Trodimenzionalni prikaz

Prikaz podataka u konkretnoj sceni unutar trodimenzionalnog prostora je krajnji cilj programa. Prikaz je ostvaren korištenjem biblioteke Ogre, točnije njene inačice za jezik Python, Python-Ogre. Iako unutar prikaza nema puno izračuna, biblioteka je pokazala dobre performanse.

Za razliku od ostalih biblioteka korištenih u ovom radu, biblioteka Ogre nije jednostavna za korištenje. Najprije je potrebno učitati sustav za iscrtavanje. U ovom programu, odabran je sustav *Direct3D9,* jer ga je lakše intergirati u standardni prikaz prozora od sustava *OpenGL*. Interakcija sa sustavom vrši se pomoću ugrađenih naredbi biblioteke Ogre.

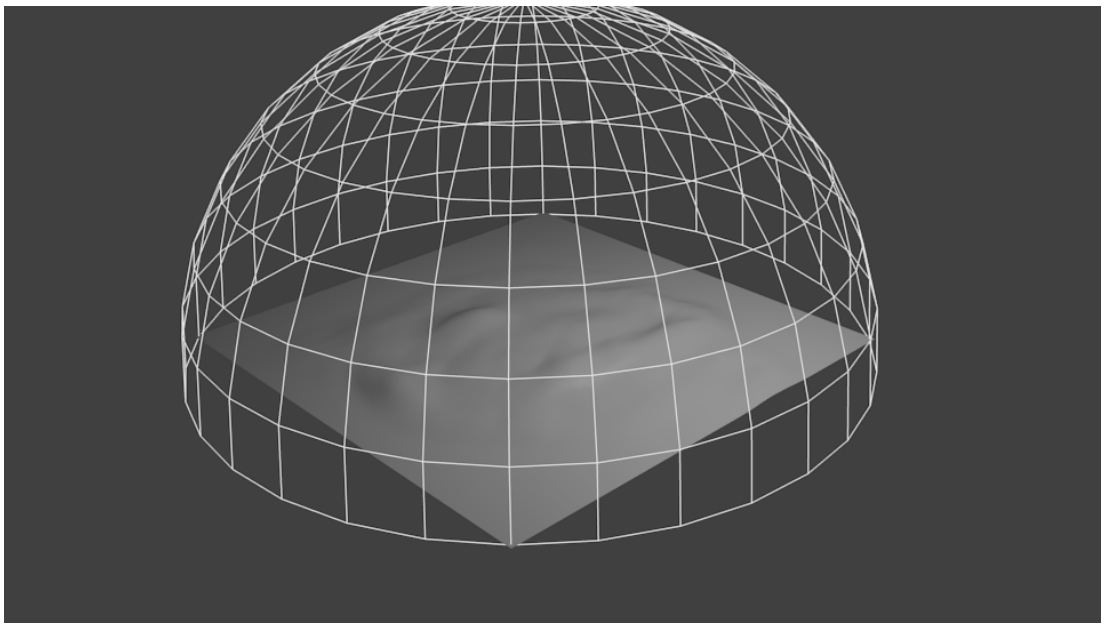
Da bi se prikazala scena, potrebno je stvoriti upravitelja scene (*eng. Scene Manager*) i upravitelja resursa (*eng. Resource Manager*). Unutar upravitelja resursa učitavaju se svi materijali, teksture i postavke sustava čestica koji su korišteni u programu. Upravitelj scenom učitava kamere, objekte i osvjetljenje, te realizira sustav čestica [7].

Unutar upravitelja scene, jedna od ključnih stvari za ovaj rad je inicijalizacija osluškivača okvira (*eng. Frame Listener*). Osluškivač okvira prati promjene nad nekim skupom podataka te osvježava scenu koristeći te podatke. U pravilu se pozivanje osluškivača vrši jednom za svaku sliku, no moguće je unutar njega definirati funkcije koje se pozivaju više ili manje puta. Unutar programa za ovaj rad, samo se jedna funkcija osluškivača obavlja jednom za svaku sliku, a to je rotacija anemometra. Sve ostale funkcije se obavljaju u trenutku kada se promijene ulazni podatci.

Upravitelj scenom unutar sebe sadrži korijenski čvor (*eng. Root Node*), koji je u stvari prikaz točke (0, 0, 0) unutar koordinatnog sustava scene. Dodavanjem čvorova na korijenski čvor stvara se stablo scene, gdje se promjene izvršene na čvoru roditelja odražavaju na promjene izvršene na čvorovima djece.

Na svaki čvor scene je moguće staviti proizvoljan broj objekata, međutim nad samim objektom nije moguće vršiti geometrijske transformacije. Transformacije se obavljaju nad čvorovima, što rezultira transformacijom svih objekata na čvoru. Zato je preporučeno koristiti poseban čvor za svaki objekt. U programu za ovaj rad korišteno je, ne uzimajući u obzir korijenski čvor, ukupno dvanaest čvorova. Dva čvora služe za objekt kamere, jedan za sustav čestica kiše, jedan za prikaz magle prozirnom teksturom, dva za prikaz svjetla, dok ostalih šest služi za objekte unutar scene.

Također je unutar upravitelja scenom moguće dodati i nebesku plohu (*eng. Sky Plane*), nebesku kocku (*eng. Sky Cube*) ili nebesku kupolu (*eng. Sky Dome*). Za prikaz neba u ovom radu korištena je nebeska kupola, što omogućuje okruženost cijele scene teksturom naoblake.

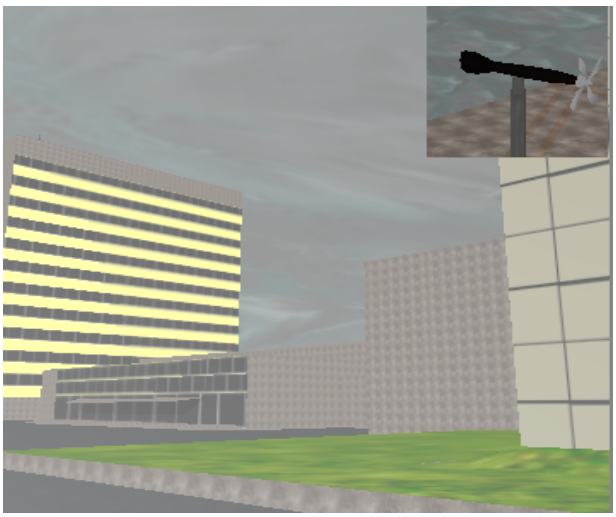


Slika 11: Shema nebeske kupole nad terenom

Unutar osluškivača okvira, moguće je dinamički mijenjati teksturu naoblake ovisno o učitanom podatku o trenutnoj količini naoblake. Podatci o naoblaci se kreću od 0 do 10, te je stoga potrebno 11 tekstura s različitim stupnjevima naoblake, koje se dozivaju iz upravitelja resursa.

U programskom rješenju ovog rada su, radi preglednosti, unutar scene postavljene dvije kamere. Kamera je objekt koji daje lokaciju s koje scena biva promatrana, kao i neke ostale parametre kao što su smjer gledanja i vidno polje (*eng. Field of view*). Jedna kamera služi za cjelokupni prikaz scene, dok druga služi za približeni prikaz anemometra koji se nalazi na većoj udaljenosti od prve kamere.

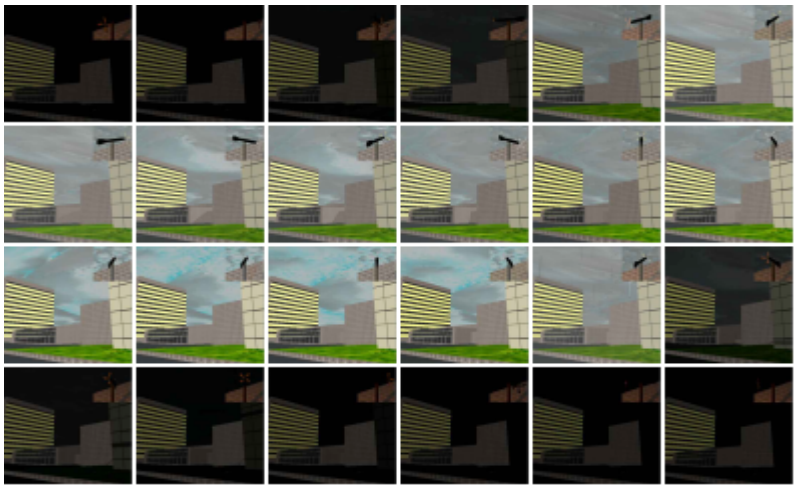
Iscrtavanje pogleda svake kamere iscrtava se unutar takozvanog vidnog prikaza (*eng. Viewport*). Svaki takav prikaz sadrži jednu aktivnu kameru, a njihov razmještaj je moguće programski odrediti. Za ovaj program korištena su dva prikaza, svaki za jednu kameru, koji su razmješteni na način da kamera za glavni prikaz scene pokriva cijeli prozor, a nad njom se u gornjem desnom kutu iscrtava sporedna kamera koja prikazuje anemometar. Na taj način ostvareno je pregledno iscrtavanje elemenata scene koji su vizualizirani iz dobivenih podataka.



Slika 12: Sustav s dvije kamere unutar dva prikaza pogleda

Još jedan bitan čimbenik pri iscrtavanju scene je osvjetljenje. U ovom programu, osvjetljenje se računa dinamički na temelju položaja sunca na nebu. Koordinate sunca računaju se prema trenutno odabranom satu. Kada je izračunata y-koordinata (u biblioteci Ogre, y-os je vertikalna os) manja od nule, to predstavlja podatak da je nastupilo noćno vrijeme, te se vrijednost ambijentne komponente osvjetljenja smanjuje na RGB vrijednost 0.1, 0.1, 0.1.

Tijekom dnevnih sati, intenzitet sunca ovisi i o količini naoblake. Pretpostavljeno je da je ovisnost linearna, s time da je minimum vrijednosti ambijentnog osvjetljenja jednaka 0.8, 0.8, 0.8. Ta pretpostavka je zapravo vrlo dobra aproksimacija stvarnog ponašanja ambijentnog osvjetljenja.



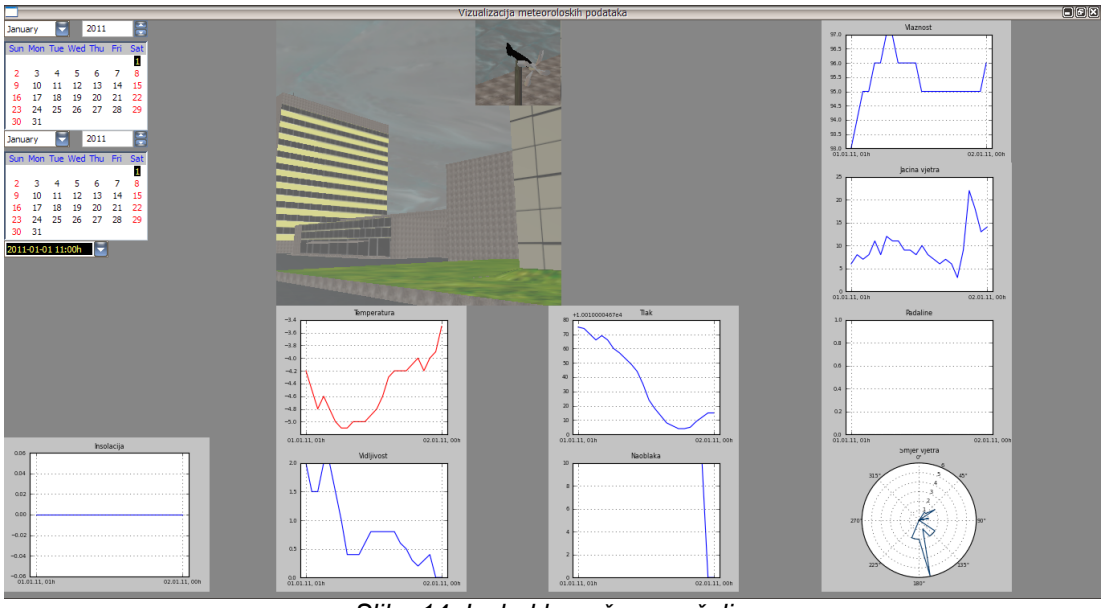
Slika 13: Tijek vizualizacije tijekom 24 sata

Još jedna komponenta osvjetljenja koja se javlja tijekom dana je usmjereno osvjetljenje od sunca prema sceni. Smjer tog osvjetljenja je vektor koji se kreće od izračunate lokacije sunca do središta koordinatnog sustava, a intenzitet je od jutarnjih do poslijepodnevnih stvari aproksimiran kao konstantan, dok u predvečerje, zbog loma svjetlosti u zemljinoj atmosferi, do blagog izražaja dolazi crvena komponenta osvjetljenja. Za scenu je aktivirano ugrađeno sjenčanje biblioteke Ogre, da bi se položaj sunca mogao pratiti putem sjena.

3.8. Korištenje programskog rješenja

Korištenje dobivenog programskog rješenja je zbog načina implementacije izrazito jednostavno. Konačni izgled sučelja može se vidjeti na slici 14. Na sučelju su vidljivi dio za odabir vremena, dio za prikaz i dio za grafove.

Moguće je u lijevom dijelu ekrana odabrati raspon datuma za koje je potrebno dohvatiti podatke. Tada se padajući izbornik koji se nalazi ispod kalendara popunjava datumima i satima. Iz padajućeg izbornika se zatim odabire točan dan i sat za koji se želi ostvariti prikaz. Tada slijedi iscrtavanje grafova i grafičkog prikaza. Grafove je moguće otvoriti u zasebnom prozoru radi bolje preglednosti.



Slika 14: Izgled konačnog sučelja

3.9. Performanse

Unatoč velikom broju podataka i potrebnim izračunima nad tim podatcima, performanse su zadovoljavajuće. Jedino primijećeno usporavanje rada programa javlja se prilikom iscrtavanja grafova, jer je za svaki graf potrebno obraditi podatke iz zadanog raspona datuma. U tablici 1. može se vidjeti koliko odabrani raspon datuma utječe na brzinu iscrtavanja grafova. Zbog relativno malog povećanja vremena za puno veći broj podataka, može se zaključiti da se veći dio vremena izgubi na iscrtavanju, a manji na operacije nad podacima.

Tablica 1: Brzina iscrtavanja grafova

Raspon podataka	Vrijeme iscrtavanja grafova
1 dan	4.14 s
1 tjedan	4.80 s
1 mjesec	5.20 s
1 godina	8.49 s

4. Zaključak

Iako je dobiveno programsko rješenje daleko od idealnog prikaza vremena kao stanja atmosfere, i dalje omogućava zorni prikaz vremenskih prilika čitanjem dobivenih podataka.

U stvarnom svijetu, postoji velik stupanj isprepletenosti i međusobne ovisnosti između raznih atmosferskih čimbenika. Iz tog razloga nije ni moguće stvoriti detaljnu vizualizaciju na temelju malog skupa podataka. Međutim, istraživanjem jednostavnih načina prikaza navedenih u ovom radu moguće je uočiti neke od navedenih povezanosti. Implementacija nekih složenijih metoda zahtijevala bi pomnije proučavanje podataka i njihovog značenja u kontekstu meteorologije.

Implementacija ovih osnovnih načina prikaza je vrlo jednostavna čak i za velike količine podataka. Najveći problemi javljaju se prilikom implementacije trodimenzionalnih elemenata unutar grafičkog prikaza, jer je potrebno simulirati njihovu interakciju s meteorološkim modelom dobivenim iz podataka. Zbog gotovo linearne ovisnosti brzine iscrtavanja i količine podataka, performanse programa ostaju zadovoljavajuće tijekom izvođenja. Izvođenje programa nad puno većim skupovima podataka kao što su desetominutni podaci ponajviše bi se moglo ubrzati korištenjem čisto prevedenih programskih jezika kao što je C++, za razliku od interpretiranog jezika kao što je Python. Također bi bilo moguće optimizirati strukturu baze i dohvaćanja podatka ili skupa podataka.

Nakon upoznavanja osnovnih načina obrade i vizualizacije meteoroloških podataka, moguće je doći do novih, složenijih načina za tu vizualizaciju, te povećanjem složenosti doći sve bliže stvarnom prikazu vremena i meteoroloških čimbenika, što bi uvelike pomoglo razumijevaju općenitog utjecaja tih čimbenika na stanje i promjenu stanja atmosfere.

5. Literatura

[1] A. T. Young, An Introduction to Mirages , http://mintaka.sdsu.edu/GF/mirages/mirintro.html , 2013.

[2] Anemometer, http://www.saylor.org/site/wp-content/uploads/2011/04/Anemometer.pdf , 2013.

[3] R. Ruiters, R. Schnabel, R. Klein, Patch-based Texture Interpolation, http://cg.cs.uni-bonn.de/aigaion2root/attachments/egsr10-texture-interpolation-homepage.pdf , 2013.

[4] M. Blanco-Muriel, D. C. Alarcon-Padilla, T. Lopez-Mortalla, M. Lara-Coira, Computing The Solar Vector http://www.researchgate.net/publication/224807292_Computing_the_Solar_Vector , 2013.

[5] W. C. Malm, Introduction to Visibility, http://www.epa.gov/air/visibility/pdfs/introvis.pdf, 2013.

[6] The Matplotlib API, http://matplotlib.org/api/index.html , 2013.

[7] Ogre API Reference, http://www.ogre3d.org/docs/api/html/ , 2013.

6. Sažetak

U ovom radu opisan je postupak stvaranja programskog sustava za učitavanje i vizualizaciju meteoroloških podataka, te su istraženi neki načini stvaranja grafičkog prikaza. Korišten je programski jezik Python i biblioteke Matplotlib i Ogre3d.

Najprije su opisana svojstva meteoroloških čimbenika za koje je moguće dobiti podatke, a zatim načini prikaza tih istih podataka. Odlučeno je da će se za ovaj rad koristiti jednostavniji načini prikaza, te su oni pobliže opisani.

Zatim je opisan konkretan način izrade programskog rješenja. Prvo su objašnjene komponente programskog rješenja kao što su interna baza podataka i sustav grafova, kao i njihove funkcionalnosti. Konačno je opisan način stvaranja trodimenzionalnog grafičkog prikaza dobiven obradom podataka za dano vrijeme. Dobiveni model stanja atmosfere prikazan je nad modelom zgrade Fakulteta elektrotehnike i računarstva u Zagrebu. Također su razmotrene performanse i prednosti, odnosno nedostatci dobivenog programskog rješenja.

Ključne riječi: vizualizacija meteoroloških podataka, grafički prikaz, obrada podataka, sustavi čestica, grafičko modeliranje, Matplotlib, Ogre3d, Python

7. Abstract

In this paper, methods of creating a program for importing and visualizing meteorological data were described, and several ways of creating a graphical output have been examined. The programming language used was Python, along with the libraries Matplotlib and Ogre3d.

First the properties of several meteorological factors the data was given for were described, and after that the ways of visualizing given data were examined. More simple ways of visualizing have been selected, and those were described in more detail.

Then the exact way of creating a program solution was described. The components like the internal database and the graph drawing system were described first, along with their functionalities. Next, the method of generating a three-dimensional graphical display from the acquired data by processing the database was presented. The resulting atmosphere model was displayed over a model of the building of the Faculty of electrical engineering and computing in Zagreb. Finally, the performance was analysed along with the advantages and disadvantages of the resulting program solution.

Keywords: visualization of meteorological data, graphical display, data processing, particle systems, graphic modelling, Matplotlib, Ogre3d, Python