UNIVERZA V MARIBORU

FAKULTETA ZA ELEKTROTEHNIKO,
RAČUNALNIŠTVO IN INFORMATIKO

Aleš Zamuda

MODELIRANJE, SIMULACIJA IN UPODABLJANJE DREVESNIH EKOSISTEMOV

Diplomska naloga

Maribor, junij 2006

UNIVERZA V MARIBORU

FAKULTETA ZA ELEKTROTEHNIKO,
RAČUNALNIŠTVO IN INFORMATIKO
2000 Maribor, Smetanova ul. 17

Diplomska naloga univerzitetnega študijskega programa

MODELIRANJE, SIMULACIJA IN UPODABLJANJE DREVESNIH EKOSISTEMOV

Študent: Aleš ZAMUDA
Študijski program: univerzitetni, Računalništvo in informatika
Smer: Programska oprema

Mentor: red. prof. dr. Nikola GUID
Lektorica: Sonja Bošković

Maribor, junij 2006

UNIVERZA V MARIBORU

FAKULTETA ZA ELEKTROTEHNIKO, RAČUNALNIŠTVO IN INFORMATIKO
2000 Maribor, Smetanova ul. 17

Številka: RI-365
Datum: 07. 06. 2006

SKLEP O DIPLOMSKEM DELU

1. **Aleš Zamuda,** študent univerzitetnega študijskega programa Računalnišvo in informatika, smer Programska oprema, izpolnjuje pogoje, zato se mu dovoljuje izdelati diplomsko delo.

2. **Tema diplomskega dela** je s področja Inštituta za računalništvo pri predmetu **RAČUNALNIŠKA ANIMACIJA**

   MENTOR: red. prof. dr. Nikola Guid

3. **Naslov diplomskega dela:**
   MODELIRANJE, SIMULACIJA IN UPODABLJANJE DREVESNIH EKOSISTEMOV

4. **Vsebina diplomskega dela:**
   Izdelajte interaktivni modelirnik dreves, ki temelji na Holtonovem in Weber-Pennovem modelu. Zatem postavite algoritme za izračun ekoloških parametrov in jih skupaj z modelirnikom dreves vgradite v simulator drevesnega ekosistema. Na koncu še sprogramirajte vizualizator in upodobite rezultate simulacije.

5. Diplomsko delo je potrebno izdelati skladno z "Navodili za izdelavo diplomskega dela" in ga oddati v treh izvodih do 07. 06. 2007 v referatu za študentske zadeve.

PREDSTOJNIK INŠTITUTA

red. prof. dr. Viljem Žumer

DEKAN

red. prof. dr. Igor Tičar

UNIVERZA V MARIBORU FAKULTETA ZA ELEKTROTEHNIKO, RAČUNALNIŠTVO IN INFORMATIKO

MENTOR

red. prof. dr. Nikola Guid

ZAHVALA

Za pomoč in vodenje pri opravljanju diplomske naloge se zahvaljujem mentorju red. prof. dr. Nikoli Guidu, asistentoma Damjanu Strnadu in Simonu Kolmaniču ter Andreju Neratu.
Posebna zahvala velja staršem, ki so mi omogočili študij. Zahvaljujem se tudi dekletu Darji, ki me je spodbujala ne le sedaj, pač pa cel čas študija in še naprej.

MODELIRANJE, SIMULACIJA IN UPODABLJANJE DREVESNIH EKOSISTEMOV

Ključne besede: **računalniška animacija, modeliranje dreves, simulacija ekosistemov**

UDK: 004.92 (043.3)

Povzetek

Najprej podamo zgodovino in razvoj modeliranja in vizualizacije ekosistemov do danes. Nato navedemo uporabljene metode pri simulaciji in opišemo nekatere poglede na vizualizacijo.

Za vizualizacijo ekosistemov je bilo potrebno pripraviti vizualno realistične modele rastlin. Pri tem uporabimo leta 2004 pripravljen interaktivni modelirnik in vizualizator naravnih dreves, ki smo ga predstavili tudi na ERK 2004.

Osrednji del diplomskega dela predstavlja izgradnja simulatorja, ki upošteva naravne faktorje za porazdelitev rastlin na omejenem delu pokrajine. Rezultate simulacije predstavimo z vizualizatorjem terena, ki na površju nariše drevesa, poenostavljena glede na razdaljo od gledišča.

Sledi opis implementiranih algoritmov. Delo zaključimo z navodili za delo s programom in z razlago sklopov programskega paketa.

MODELING, SIMULATION AND RENDERING OF TREE ECOSYSTEMS

Key words: **computer animation, tree modelling, ecosystem simulation**

UDK: 004.92 (043.3)

Abstract

First, modelling and simulation of ecosystems state of art are provided. Then, used methods for simulation are listed and some aspects of visualization are discussed.

For visually realistic visualization of ecosystems, plant models must be provided. To accomplish this, we use an interactive modeller and visualizer of natural trees developed in 2004 and presented at ERK 2004.

Construction of simulator considering natural factors to distribute plants across limited landscape plays a central role in this diploma work. Simulation results are visualized on surface using our terrain visualizer, which renders trees progressively lower-detailed with a distance from a point of view.

Specification of implemented algorithms follows. This work concludes with instructions for a user and comments on presented software package.

Kazalo

Poglavje 1

Uvod

Modeliranje je ena od temeljnih dejavnosti na vseh področjih naravnih znanosti, saj nam omogoča formalen opis naravnih sistemov. Razumevanje naravnih zakonitosti nam pomaga aproksimativno določiti model, ki do določene mere sledi naravnemu svetu. V diplomskem delu nas zanimajo ekološki sistemi in njihovi modeli.

Z modeliranjem ekološkega sistema ustvarimo abstrakten model, ki nam skozi simulacijo daje določene odgovore na različne scenarije. Pri tem je ključnega pomena hitrost in cena odvijanja simulacije. Simulacija nam odgovori v precej krajšem času, kot bi na rezultate čakali v realnem okolju, ter po nižji ceni in z manj stranskimi učinki kot sicer.

Pri študiju ekoloških modelov je nepogrešljiva računalniška grafika, ki nam omogoča modelirana okolja realistično vizualizirati. Znanja iz področja računalniške animacije pa nam pomagajo pri animaciji scene skozi določeno časovno obdobje.

V diplomskem delu se osredotočimo na problematiko zaraščanja pokrajine. V ta namen potrebujemo ustrezen postopek za porazdelitev dreves po pokrajini, ki upošteva naravne zakonitosti, znane iz ekologije in biologije. S pretvorbo teh zakonitosti v računalniške algoritme omogočimo simulacijo ekoloških zakonitosti v računalniku. Med ekološke vplive med drugim štejemo vlago, strmino, vetrovnost in osončenost na mestu rasti rastline kot tudi interakcije med samimi rastlinami.

Za upodabljanje dreves uporabimo geometrijski proceduralni model. Posamezno drevo modeliramo s postavljanjem parametrov znotraj proceduralnega modela. Parametre ločimo na vektorske in skalarne, pri čemer lahko vektorske interaktivno oblikujemo z grafi. Med vektorske parametre proceduralnega modela štejemo strukturo vejitev, kompleksnost zgradbe, kote vejitev, razmerja med dolžinami odsekov vej in vpliv gravitacije na geometrijo drevesa. Skalarni parametri v modelu so višina in debelina osnovnega debla, vpliv vetra ter gostota in velikost listov. S pomočjo navedenih vektorskih in skalarnih parametrov rekurzivno zgradimo geometrijski model, ki mu dodamo teksture za končni izgled drevesa.

Da smo realizirali zastavljene cilje, smo najprej zgradili modelirnik naravnih dreves. Drevo lahko določimo interaktivno ali pa z nalaganjem parametrov iz knjižnice že oblikovanih dreves, ki je priložena modelirniku. Pri tem se vizualizacija osveži takoj, ko spremenimo kak parameter modela, kar pripomore pri učenju uporabe modelirnika. Po zbrani bazi parametričnih modelov za drevesa je postalo jasno, da bi bilo mogoče z njimi vizualizirati tudi drevesne ekosisteme. V ta namen smo pripravili vizualizator terena, ki ima možnost povezave z vizualizatorjem za drevesa in na osnovi simulacije določi porazdelitve dreves po terenu ter na mestih rasti upodobi proceduralni model drevesa. Ker smo vizualizirali rast dreves, je imelo vsako upodobljeno drevo v odvisnosti od svoje starosti na novo določeno geometrijo, z drevesi iste vrste pa si je delilo le parametre proceduralnega modela.

Čas, potreben za prikaz velikega števila dreves na sliki, smo zmanjšali z odstranjevanjem objektov izven projekcije v času upodabljanja. Omenjen čas smo še dodatno izboljšali s poenostavljanjem geometrije drevesa na podlagi oddaljenosti od opazovalca. Zmožnost poenostavljanja smo vgradili v proceduralni model drevesa, tako da smo za bolj oddaljena drevesa izrisali manj nivojev podrobnosti.

Samo diplomsko delo je razdeljeno v sedem poglavij. Po uvodu v drugem poglavju predstavimo predhodne modele za drevesa in za simulacijo ekosistemov v računalniški grafiki ter pregled vizualizatorjev. V tretjem poglavju obravnavamo obstoječe metode za proceduralno modeliranje dreves in opišemo zasnovo novega interaktivnega modelirnika dreves ter postopke za animacijo in poenostavljanje geometrije drevesa, zapišemo pa tudi nekaj primerov dejanskih parametrov za modele dreves. V četrtem poglavju podamo modeliranje in simulacijo ekoloških parametrov, kot so višina, naklon, vlaga, osončenost, vetrovnost, način razširjanja dreves in tekmovalnost med rastlinami. V petem poglavju se lotimo vizualizatorja pokrajine in naštejemo tudi različne pristope za njegovo realizacijo. V šestem poglavju podamo rezultate simulacije drevesnih ekosistemov in komentiramo parametre simulacije ob nekaj upodobljenih slikah. Sedmo poglavje poda zaključek k diplomskemu delu. Sledijo dodatek A s seznamom uporabljenih simbolov, dodatek B z izpeljavo povprečnega kota osončenja v poletnem času in v dodatek C z navodili za uporabo izdelane aplikacije, arhitekturno zgradbo izvorne kode ter opisom nekaterih ključnih razredov. Na koncu je navedena še literatura.

Poglavje 2

Pregled modelirnikov dreves, simulacij drevesnih ekosistemov in vizualizatorjev

2.1 Modelirniki dreves

Obstaja več tehnik opisovanja geometrijskih modelov dreves. Ker je ročno modeliranje drevesne strukture in listov pogosto precej zamudno, pri tem raje uporabimo proceduralni model za določitev geometrije.

Proceduralni modeli temeljijo na različnih oblikah gradnje osnovne vejitvene strukture, razlikujejo pa se predvsem v stopnji podrobnosti, za katero je tehnika primerna, okretnosti in zahtevnosti modeliranja, prostorski in časovni zahtevnosti modelov, možnosti animacije in vrsti predstavitve končnega modela.

Enega od prvih specializiranih modelov za tvorbo dreves sta predstavila Aono in Kunii [Aono in Kunii, 1984]. Razvila sta štiri geometrijske modele, ki predstavljajo zaporedne nadgradnje. Skupna so jim pravila, da vsaka vejitev osnovne veje tvori dve podveji, da se dolžina in premer podvej manjšata s konstantnim faktorjem, da so koti vejitev enaki na vseh nivojih drevesa, da je ravnina, ki jo določata podveji, pravokotna na ravnino, ki jo podajata starševska in prastarševska veja, ter da se vejitve hkrati izvedejo na vrhu vseh obstoječih vej. Opisala sta tudi kot osnega zasuka zaporednih stranskih vej vzdolž starševske veje, to lastnost imenujemo filotaksa. V modele sta vključila točkaste atraktorje, ki pritegnejo ali odbijejo rast drevesa in simulirajo učinek vetra, sonca in gravitacije.

Bloomenthalov model [Bloomenthal, 1985] na osnovi ročno podanega skeleta drevesa zgolj doda sedla pri vejitvah vej in skelet obleče s krivuljami NURBS, ki jih opremi s foto-teksturami.

Reeves in Blau sta za predstavitev gozdnih površin pri manjši podrobnosti uporabila sisteme delcev [Reeves, 1985]. Veje dreves sta predstavila z daljicami, liste s točkami (majhnimi krogi), osnovna debla dreves pa s stožci. Parametri njunega modela drevesa so širina spodnjega dela krošnje drevesa, višina drevesa, višina od tal do prve veje na krošnji, srednja dolžina vej in kot vejitve. Iz algoritma dobljena vejitvena struktura se naknadno obdela z ločenimi algoritmi za simuliranje gravitacije, delovanja vetrov in težnje po svetlobi.

Oppenheimer je drevesa modeliral z uporabo fraktalnih tehnik [Oppenheimer, 1986]. Deblo in veje gradi z izvajanjem linearnih transformacij, podanih z matrikami velikosti 3×3. Geometrijo drevesa sestavljajo mnogokotniške prizme ali le daljice vzdolž vej, na katere je nalepljena proceduralna tekstura. Upodobljeni modeli so namenjeni visoki stopnji podrobnosti, vendar ne vsebujejo listov drevesa.

Z biološko motiviranim modelom z L-sistemi [Lindenmayer, 1968] je Prusinkiewicz modeliral drevesa pri visoki stopnji podrobnosti [Prusinkiewicz in Lindenmayer, 1990]. V L-sisteme je vpeljal grafično predstavitev prepisnih nizov, definiranih s kontekstno odvisno gramatiko. Prepisne nize je vizualiziral po principu želve LOGO [Papert s sodelavci, 1979], ki je z geometrijsko interpretacijo niza narisala topologijo drevesa. Ta tehnika ima veliko izrazno moč in je doživela največ dopolnitev, vendar je realizacija modelirnika precej zahtevna, modeliranje drevesa pa zahteva dobro poznavanje uporabljenega domensko specifičnega jezika za definicijo gramatike prepisnega niza. V ta namen je v zadnjih letih večji poudarek na interaktivnosti oblikovanja in upodabljanja [Deussen s sodelavci, 2002; Lintermann in Deussen, 1999] kot pa na novih metodah.

Holtonov žilni model [Holton, 1994] je prav tako biološko motiviran, saj debeline vej in razmerja vejitev od osnovne veje določa žilni model. Porazdelitev žil v drevesu določa debelino in posledično dolžino veje, saj se žile celoštevilsko delijo v podveje, veje z eno žilo pa imajo liste. Modelu poleg števila vseh žil v drevesu podamo še razmerja med dolžinami vej in kote vejitev med izhajajočimi vejami. Na kote vejitve vplivajo tudi elementi okolja, ki določajo težnje drevesa, kot so pokončnost debla oz. uravnoteženost krošnje, upogibanje vej k tlom zaradi vpliva gravitacije, rast v smeri svetlobe, rast v navpični smeri, rast v vodoravni smeri in rast v ravnini, ki je pravokotna na ravnino med osrednjo osjo drevesa in starševsko vejo. Dobra stran tega modela je omenjeni samodejni izračun debeline in razmerja med delitvami vej, vendar pa mora uporabnik preostale parametre še vedno vnesti tabelarično, kar zmanjša okretnost modeliranja.

Weber in Penn sta drevo predstavila s preprosto geometrijo [Weber in Penn, 1995] brez razvoja vejitvene strukture. Za vse veje na enakih nivojih sta podala kot vejitve, razmerje dolžin glede na osnovno vejo in debeline vej. Predstavila sta

tudi možnost animacije zibanja drevesa v vetru, obrezovanje vej drevesa na določen volumen in poenostavljanje geometrije drevesa.

Strnad je drevesa za upodobitev pri srednji podrobnosti predstavil s fraktalnimi hiperteksturami [Strnad in Guid, 2004]. Drevesa je definiral fraktalno z iterativnimi funkcijskimi sistemi, ki jih je upodobil s prostorskim upodabljanjem. Model zaradi časovno zamudne tehnike upodabljanja ni primeren za uporabo v interaktivni animaciji, vendar pa omogoča zvezen prehod med stopnjami podrobnosti.

2.2 Simulatorji drevesnih ekosistemov

Pri simulatorjih drevesnih ekosistemov se omejimo predvsem na simulatorje, povezane z možnostjo upodabljanja v računalniški grafiki. Ker je ročno nameščanje rastlin na teren zamudno, naključno pa dokaj nerealistično, za namestitev dreves v sceni uporabimo simulacijo.

2.2.1 Simulator dreves, temelječ na L-sistemih

S simulacijo so Deussen in sodelavci [Deussen s sodelavci, 1998] porazdelitev dreves določili na individualni ravni z obravnavo **ekološke sosednosti** (*ecologic neighborhood*) posameznih rastlin. Izhajali so iz osnovnega fenomena porazdelitve rastlin, ki mu pravimo **samo-redčenje** (*self-thinning* [Rickefs, 1990]). Ta pove, da rastline na začetku rastejo brez medsebojnega oviranja, ob večji zgostitvi pa začnejo manjše rastline odmirati.

Simulacijo so začeli z začetno populacijo rastlin, ki so jim dodelili naključno pozicijo in naključni polmer ekološke sosednosti. Vsaka rastlina je pripadala neki rastlinski vrsti, ki je definirala naslednje parametre:

- število dodanih novih rastlin na pokrajino ob vsakem simulacijskem koraku,
- največjo velikost rastlin,
- povprečno hitrost rasti,
- verjetnost preživetja ob dominiranosti in
- potrebo po vlagi.

Moč rastline so določili z zmnožkom starosti rastline in parametra **živosti** (*vigour*). Živost rastline je bila izračunana za vsako rastlino posebej in je predstavljala zmnožek potrebe po vlagi in vsebnosti vlage na mestu rasti.

Z naraščanjem simulacijskega časa so večali ekološke polmere rastlin in odstranjevali odmrle rastline. Rastline so lahko odmrle od starosti ali v primeru, ko so bile nadvladane. S simulacijo so dosegli samo-redčenje rastlin na terenu in porazdelitev vrst rastlin na mesta terena z ugodnimi pogoji.

Rastline so vizualizirali z **izdelavo primerkov** (*instancing*) rastlinskih vrst. Vsako rastlinsko vrsto so predstavili le z enim modelom za primerek in le-tega so uporabili večkrat. Posamezne rastline so dobili tako, da so model primerka naključno zasukali okoli vertikalne osi. S tem so dosegli različno usmerjenost rastlin in posledično manj zaznavno podobnost med rastlinami pri upodobitvi. Njihov sistem za vizualizacijo je omogočal naravna okolja tudi do sto tisoč rastlin, vendar je za tako upodobitev potreboval do osem ur za eno sliko.

Dopolnitve modela z L-sistemi

Model so kasneje dopolnili z **rastjo rastlin blizu starševskih rastlin** (*clustering*, *clumping*, *underdispersion*) [Lane in Prusinkiewicz, 2002]. Hitro so namreč ugotovili, da je prejšnji model tvoril preveč enakomerno porazdeljene rastline, v naravi pa te rastejo precej po skupinah.

Rast v skupinah so dosegli s **Hopkinsonovim indeksom** (*Hopkinson index*) [Hopkins, 1954]. Ta je definiran kot trenutna zgostitev rastlin na nekem mestu. Na osnovi tega indeksa so zgradili matriko verjetnosti postavitve rastline na določeno krpo površja. Rastline so postavili na površje glede na te verjetnosti, ob dodajanju nove rastline pa vrednosti v matriki ustrezno posodobili.

Čeprav so z opisanim modelom dobili nekoliko boljše rezultate kot prej, pa so rezultati še vedno odsevali le malo dejanskih vplivov okolja.

2.3 Vizualizatorji

2.3.1 Vizualizator, temelječ na žilnem modelu

Z vizualizacijo pokrajin, na katerih so bili prikazani gozdovi in grmovje, se je prvi ukvarjal Holton [Holton, 1994]. Drevesa je predstavil z geometrijskim modelom in jih upodobil z algoritmom sledenja žarku. Ker je njegov model za predstavitev dreves omogočal poenostavitev geometrije, je lahko v pokrajini prikazal več tisoč dreves, ki so bila nameščena ročno.

2.3.2 Modeliranje dreves s 3D teksturami

S 3D teksturami so drevesa predstavili Chiba, Muraoka in Hosokawa [Chiba s sodelavci, 1997]. Iz prostorske teksture so z algoritmom sledenja žarku določili gostoto in barvo za voksle.

Drevesa so na pokrajino porazdelili s **krožnim Poissonovim vzorčenjem** (*Poisson disc sampling*) [Cook, 1986]. Vsako drevo so tako skušali postaviti na naključno mesto v sceni, preverili pa so le, ali to morda ni preblizu kakšnemu že postavljenemu drevesu. S tem so dobili vizualno zadovoljive rezultate za do dvajset tisoč dreves. Upodabljanje je trajalo več ur za eno sliko.

2.3.3 Vizualizator ekosistemov dreves z L-sistemi

Deussen in sodelavci so prišli do spoznanj, da je za vizualizacijo mnogih rastlin potrebno [Deussen s sodelavci, 1998]:

- rastline realistično postaviti po terenu,
- za rastline uporabiti kompakten predstavitveni model zaradi omejitev pomnilniških kapacitet in
- zmanjšati količino geometrije za upodobitev (z izdelavo primerkov).

Realističnost postavitve rastlin na teren so zagotovili ročno ali s simulacijo, opisano v podrazdelku 2.2.1. Ročno so rastline porazdelili na globalni ravni s sliko v velikosti ploskve pokrajine, ki je določila gostote rastlin. Iz gostot so dejanske položaje rastlin dobili s poltonskim algoritmom. Pred dokončno postavitvijo položajev rastlin so te prilagodili tako, da so rastline pomaknili v središča Voronojevih mnogokotnikov, ki so oklepali rastline.

Za predstavitev geometrije so uporabili L-sisteme [Prusinkiewicz s sodelavci, 1997], rastline pa upodobili z algoritmom metanja žarka (*ray casting*) ali z algoritmom sledenja žarku (*ray tracing*). Pokrajino so sestavili s pomočjo lokalnih operatorjev nad ploskvijo površja, ki je vsebovala griče in potoke.

Predogled scene je bil možen z OpenGL, vendar je realistično upodabljanje pokrajine z rastlinami trajalo več ur in ni bilo primerno za animacijo v realnem času.

2.3.4 Vizualizatorji terena

Na spletu je možno najti več vizualizatorjev za hiter izris terena z OpenGL [Pressman, 2001; DigiBen, 2005], večina pa uporablja kakšno od razširitev OpenGL. Mrežo s 100×100 točkami ponavadi izrisujejo v realnem času, s približno 100 ali več

sličicami na sekundo. Kvaliteta sicer ni tako dobra kot pri upodabljanju s sledenjem žarka, vendar je hitrost izrisa precej hitrejša. Tudi poraba pomnilnika je manjša pri upodabljanju z OpenGL, saj podano geometrijsko strukturo v programu lahko takoj zavržemo, ko smo jo prenesli na strežnik OpenGL, ta pa zaradi uporabe vmesnega pomnilnika geometrijsko strukturo po izrisu prav tako zavrže.

Poglavje 3

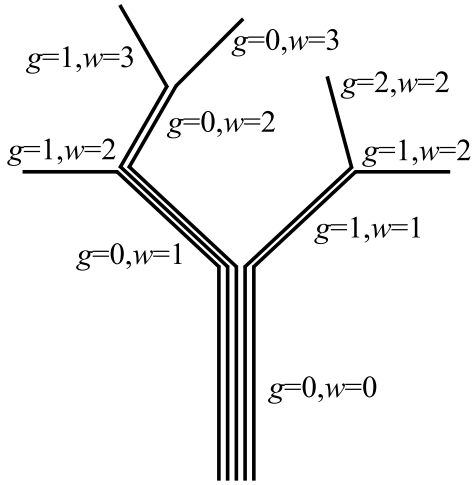
Modeliranje dreves

Za realistično računalniško modeliranje naravnih okolij je obvezno vanje vključiti vegetacijo (travo, drevesa, grmovje). Za verodostojen tridimenzionalni izgled je potrebno uporabiti geometrijsko predstavitev, še posebej, če želimo izvajati animacijo rastlin. V nadaljevanju se bomo pri obravnavi vegetacije omejili na drevesa, ki so med rastlinjem najbolj opazen element v naravnih okoljih.

Pri vseh naštetih tehnikah v razdelku 2.1 je potrebno doseči kompromis med okretnostjo in kompleksnostjo modeliranja. Nekateri 3D modelirniki danes že vključujejo specialna orodja za izdelavo dreves. Problem teh orodij je v tem, da so premalo okretna in uporabniku omogočajo le omejen nabor tipičnih predstavnikov drevesnih vrst ali pa zaradi želje po čim večji oblikovalski svobodi definirajo veliko število numeričnih parametrov, katerih nastavljanje je zamudno in ne-intuitivno, pomen pa pogosto nejasen. Zato je bil naš cilj izdelati geometrijski model drevesa z minimalnim številom potrebnih parametrov, katerih nastavljanje bo intuitivno in interaktivno. Model naj bi bil dinamičen in tako omogočal enostavno animacijo. To nam je uspelo s kombinacijo obstoječih tehnik, premišljenim izborom disjunktivnih numeričnih značilnosti dreves in grafičnim vmesnikom za podajanje vrednosti parametrov, katerih spreminjanje se interaktivno odraža na senčenem modelu drevesa. Rezultat dela je modelirnik dreves, ki vključuje tudi sistem za simulacijo rasti in animacijo dreves. Naš modelirnik temelji predvsem na Holtonovem žilnem modelu dreves in uporablja nekaj idej iz Weber-Pennovega modela. Zato bomo najprej podrobneje predstavili oba modela.

3.1 Holtonov žilni model dreves

Kot smo že zapisali v razdelku 2.1, Holtonov model [Holton, 1994] temelji na notranji žilni strukturi botaničnih dreves. Deblo definiramo kot prvi del drevesa, ki prihaja



Slika 3.1: Graveliusov in Weibullov red veje.

iz zemlje, do prve vejitve ne glede na debelino vejitve. Glede na podano začetno število žil S v deblu drevesa se te pri vsaki vejitvi delijo na dve podveji, tako da se skupno število žil ohrani. Število odsekov vej B, ki jih dobimo zaradi vejitev, je enako:

$$B = 2S - 1, \tag{3.1}$$

Na sliki 3.1 je $S = 5$ in $B = 9$.

Število žil S neposredno določa ostale lastnosti veje, kot sta npr. njena debelina in dolžina. S številom žil je omejeno tudi število nadaljnjih vejitev, ki se končajo pri veji z eno žilo, iz katere rastejo le še listi.

Drug pomemben podatek, ki ga beležimo pri vsaki veji, je njen *red*. Vendar tudi pri tem ločimo dve različni obliki štetja, ki določata Graveliusov oz. Weibullov red. Graveliusov red veje označimo z g, Weibullov red pa z w. Pri Graveliusovem štetju se ob vsaki cepitvi veje t. i. *stranski podveji* red poveča, medtem ko druga, t. i. *glavna podveja*, šteje kot nadaljevanje prvotne ali *osnovne veje* in s tem ohrani njen red. Pri Weibullovem štetju se obema podvejama red poveča (slika 3.1).

Holton za izračun geometrije drevesa uporablja biološko osnovane enačbe, s pomočjo katerih določa ostale lastnosti veje s pomočjo (v odvisnosti od) števila žil v njej. V enačbah pogosto nastopajo uporabniško nastavljivi parametri, ki so definirani ločeno za vsak red veje (posebej Graveliusovega in Weibullovega).

Vsi parametri Holtonovega modela so:

- S - število žil v deblu drevesa, ki določa njegovo kompleksnost in s tem tudi starost,
- $k_s^{g,w}$ - razmerje porazdelitve žil na podveji pri vejitvah,
- $\alpha^{g,w}$ - kot med izhajajočima podvejama pri delitvi,

- $l_0^{0,0}$ - višina osnovnega debla,
- k_d - koeficient debeline veje,
- $M_t^{g,w}$ in $m_t^{g,w}$ - spodnja in zgornja meja relativne dolžine podvej glede na dolžino osnovne veje, ločeno za tip podveje ($t \in \{major, minor\}$),
- $N_t^{g,w}$ in $n_t^{g,w}$ - spodnja in zgornja meja dejanske dolžine podvej glede na dolžino osnovne veje, ločeno za tip podveje ($t \in \{major, minor\}$),
- k_c^g - gravicentralizem (t. j. težnja debla po navpični rasti), skalirni faktor med 0 in 1 za vektor v smeri popravka rasti proti centru (če je vrednost prevelika, drevo začne nihati v levo in spet desno okoli središčne osi debla),
- $\alpha_m^{g,w}$ - gravimorfizem (t. j. upogibanje vej zaradi gravitacije),
- k_p - fototropizem (tendenca rasti v smeri proti svetlobi),
- $k_o^{g,w}$ - ortotropizem (t. j. tendenca po rasti vej navpično navzgor),
- $k_p^{g,w}$ - plagiotropizem (t. j. tendenca po rasti vej horizontalno navzven od debla),
- α_p - kot filotakse (t. j. tendenca po rasti veje izven ravnine sestrske veje); s tem je določen tudi planartropizem,
- k_w - Writhov koeficient za naključnost rasti in
- $\overline{\mathbf{k}}$, $\overline{\mathbf{p}}$, $\overline{\mathbf{h}}$, $\overline{\mathbf{g}}$, $\overline{\mathbf{w}}$, $\mathbf{b_0}$, $\mathbf{q}$, $\mathbf{b}$ - enotski vektorji v smeri ortotropizma, plagiotropizma, fototropizma, gravimorfizma, Writhov vektor, začetni vektor veje, smer veje ter dejanski vektor veje.

Smer glavne podveje se tako izračuna s prištevanjem tendenc k smeri osnovne veje:

$$\mathbf{q} = \mathbf{b_0} + k_s^{g,w}\overline{\mathbf{k}} + k_p^{g,w}\overline{\mathbf{p}} + \alpha^{g,w}\overline{\mathbf{h}} + \alpha_m^{g,w}\overline{\mathbf{g}} + k_w\overline{\mathbf{w}}. \tag{3.2}$$

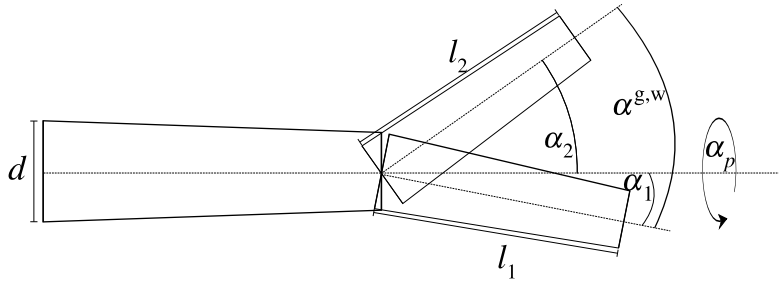
Smer glavne podveje in njeno dolžino določa naslednji vektor:

$$\mathbf{b} = \frac{l_1\mathbf{q}}{\|\mathbf{q}\|}, \tag{3.3}$$

kjer je l_1 dolžina glavne podveje (glej enačbo 3.15).

Debelina veje d se izračuna iz števila žil S_0 po naslednji enačbi:

$$d = k_d\sqrt{S_0}, \tag{3.4}$$



Slika 3.2: Izračun geometrije pri vejitvi.

kjer je parameter $k_d \in [0, 1]$ koeficient debeline, ki je konstanten za celotno drevo.

Ob vsaki vejitvi se žile osnovne veje razdelijo med nastali podveji po naslednjih enačbah:

$$S_1 = \lceil 1 + k_s^{g,w} (S_0 - 2) \rceil , \tag{3.5}$$

$$S_2 = S_0 - S_1. \tag{3.6}$$

Parameter $k_s^{g,w} \in \left[\frac{1}{2}, 1\right]$ določa delež žil, ki gredo v glavno podvejo. Pri tem je S_0 število žil osnovne veje, S_1 število žil v glavni podveji in S_2 število žil v stranski podveji. Povezavo med debelinami izhajajočih vej je preučeval že Leonardo da Vinci, formaliziral pa jo je Mandelbrot [Mandelbrot, 1982], ki je zapisal Pitagorov izrek med debelino osnovne veje in debelinami obeh stranskih vej.

Po vejitvi se podveji odklonita vsaka na svojo stran v isti ravnini, tako da oklepata uporabniško definirani kot vejitve $\alpha^{g,w} \in [0°, 180°]$ (slika 3.2). Zatem se obe podveji zasučeta okoli vzdolžne osi osnovne veje za kot $\alpha_p \in [0°, 180°]$. Pri tem se upošteva še koeficient gravicentralizma $k_c^{g,w} \in [0, 1]$, ki je različen od 1 samo za veje Graveliusovega reda 0.

Odklonska kota α_1 za glavno podvejo in α_2 za stransko podvejo se izračunata po enačbah:

$$\alpha_1 = k_c^{g,w} \sqrt{\frac{S_2}{S_0}} \alpha^{g,w}, \tag{3.7}$$

$$\alpha_2 = \alpha^{g,w} - \alpha_1. \tag{3.8}$$

Gravimorfizem je vključen z dodatnim zasukom veje v smeri tal, pri čemer je kot tega zasuka podan s parametrom $\alpha_m^{g,w} \in [0°, 180°]$.

Za dolžine vej je Holton najprej izračunal razmerji žil med osnovno vejo in podvejama:

$$R_1 = r_1 = \sqrt{\frac{S_1}{S_0}}, \qquad R_2 = r_2 = \sqrt{\frac{S_2}{S_0}}. \tag{3.9}$$

Razmerji je nato omejil in dolžine mej izračunal po naslednjih enačbah:

$$m_{major}^{g,w} \leq R_1 \leq M_{major}^{g,w}, \tag{3.10}$$

L_0, L_1, L_2	relativne dolžine starševske veje in obeh podvej
R_1, R_2	razmerji relativnih dolžin podvej glede na starševsko vejo
l_1, l_2	dejanski dolžini podvej
r_1, r_2	razmerji dejanskih dolžin podvej glede na starševsko vejo
$m^{g,w}_{minor}$, $M^{g,w}_{minor}$	spodnja in zgornja meja relativne dolžine stranske podveje
$m^{g,w}_{major}$, $M^{g,w}_{major}$	spodnja in zgornja meja relativne dolžine glavne podveje
$n^{g,w}_{minor}$, $N^{g,w}_{minor}$	spodnja in zgornja meja dejanske dolžine stranske podveje
$n^{g,w}_{major}$, $N^{g,w}_{major}$	spodnja in zgornja meja dejanske dolžine glavne podveje

Preglednica 3.1: Strukturni parametri žilnega modela.

$$m^{g,w}_{minor} \leq R_2 \leq M^{g,w}_{minor}, \tag{3.11}$$

$$L_1 = R_1 L_0, \qquad L_2 = R_2 L_0, \tag{3.12}$$

$$n^{g,w}_{major} \leq r_1 \leq N^{g,w}_{major}, \tag{3.13}$$

$$n^{g,w}_{minor} \leq r_2 \leq N^{g,w}_{minor}, \tag{3.14}$$

$$l_1 = r_1 L_0, \tag{3.15}$$

$$l_2 = r_2 L_0. \tag{3.16}$$

Pomen uporabljenih oznak je zbran v preglednici 3.1.

Holton poda intervale uporabnih vrednosti za posamezne parametre drevesa in ugotavlja, da začetno število žil med 2000 in 8000 daje najboljše rezultate. Kot primer podajmo, da je jelke (slika 3.3) modeliral z naslednjimi parametri: $S = 2500$, $k_s^{0,w} = 0,985$, $k_s^{i,w} = 0,6$, $\alpha^{0,w} = 115°$, $\alpha^{i,w} = 35°$, $M^{0,w}_{major} = 1$, $m^{0,w}_{major} = 0,98$, $M^{i,w}_{major} = 0,98$, $m^{i,w}_{major} = 0,85$, $M^{0,w}_{minor} = 0,76$, $m^{0,w}_{minor} = 0,7$, $N^{0,w}_{major} = 0,15$, $n^{0,w}_{major} = 0,1$, $N^{i,w}_{major} = 0,9$, $n^{i,w}_{major} = 0,8$, $i = 1,2,3...$ in $w = 0,1,2....$ Ostali parametri niso podani (α_p, $M^{i,w}_{minor}$, $m^{i,w}_{minor}$, $N^{g,w}_{minor}$, $n^{g,w}_{minor}$) in zato pri gradnji geometrijskega modela niso upoštevani.

Za vse tri drevesa je uporabil enake parametre, vsaka pa ima drugo seme za generator psevdo-naključnih števil za Writhove vektorje. Dobljene skelete dreves je oblekel in upodabljal s krivuljami NURBS.

3.2 Weber-Pennov model

Podoben pristop kot Holton sta uporabila Weber in Penn, ki sta definirala 40 različnih numeričnih parametrov drevesa [Weber in Penn, 1995]. Pri tem so nekateri (npr. kot vejitve) definirani ločeno za veje reda 1, 2 in 3 ali več, tako da je njihovo dejansko število preko 80.

Veje sta upodabljala s stožci. Definirala sta tudi nekaj več funkcionalnosti kot

Slika 3.3: Holtonova upodobitev parametričnega modela jelke.

Holton, kot je zibanje drevesa v vetru, stopnjo podrobnosti in obrezovanje drevesa na podan obris. Pozibavanje v vetru sta definirala kot nagib veje za naključen vektor v smeri vetra, ki ga s časom spreminjamo. Vsaka veja je imela svoje seme za generator psevdo-naključnih števil za določitev začetnega kota zibanja. Poenostavljanje geometrije sta realizirala tako, da sta odvisno od razdalje od gledišča izrisala manj redov vej.

V našem modelirniku smo iz Weber-Pennovega modela vzeli idejo o predstavitvi vej s stožci, animacijo drevesa pod vplivom vetra in možnost poenostavljanja geometrije na nivoju proceduralnega modela. Weber in Penn sta red veje omejila na 5, kar nam je dalo idejo, da bi parametrizacijo Holtonovega modela opravili prav tako z omejenim številom nivojev, ki bi jih bilo možno interaktivno načrtovati z grafi.

3.3 Zasnova novega interaktivnega modelirnika dreves

Pri določanju nabora parametrov dreves, ki jih bomo v modelirniku podprli, smo upoštevali naslednja načela:

- pomen vseh parametrov mora biti jasen, njihov učinek pa čim bolj predvidljiv,
- nabor parametrov naj bo dovolj raznolik, da omogoča modeliranje večine v naravi najpogosteje zastopanih drevesnih oblik,
- parametri naj nimajo podvojenega ali konfliktnega delovanja in
- nabor parametrov naj vključuje lastnosti drevesa, ki so pomembne pri animaciji.

Iz Holtonovega modela smo tako odstranili manj pomembne parametre in dodali nekatere ideje iz Weber-Pennovega modela za podporo stopnji podrobnosti in animaciji pozibavanja v vetru. V modelirniku so vgrajeni vsi parametri našteti v razdelku 3.1 razen ortotropizma $k_o^{g,w}$ (modeliran z negativnim gravimorfizmom), fototropizma k_p, plagiotropizma $k_p^{g,w}$ in Writhovega koeficienta k_w (pri upodabljanju ekosistema nadomeščen s sukanjem drevesa okoli osi debla), saj smo želeli parametre čimbolj poenostaviti. Tudi relativne dolžine vej $M_t^{g,w}$ in $m_t^{g,w}$ smo modelirali skupno za oba tipa vej, dejanske dolžine $N_t^{g,w}$ in $n_t^{g,w}$ pa smo zamenjali s skalirnim faktorjem dolžine veje $k_l^{g,w}$. Način porazdelitve listov l_{type} in gostoto listov ρ_l smo prenesli iz programa Corel Bryce. Po uvedenih spremembah smo dodali naslednje parametre:

- $M^{g,w}$ - zgornja meja relativne dolžine podvej glede na dolžino osnovne veje (enaka za oba tipa podvej, $t \in \{minor, major\}$),
- $m^{g,w}$ - spodnja meja relativne dolžine podvej glede na dolžino osnovne veje (enaka za oba tipa podvej, $t \in \{minor, major\}$),
- $k_l^{g,w}$ - skalirni faktor dolžine veje,
- l_{type} - način porazdelitve listov,
- ρ_l - gostota listov,
- l_l - velikost listov,
- t_{bark} (t_{leaf}) - identifikatorja teksture debla (listov),
- l_{LOD} - stopnja poenostavitve geometrijske strukture drevesa,
- w_s - hitrost (jakost) neusmerjenega vetra na drevo,
- k_f - prožnost vej drevesa pri vetru,
- w_g - hitrost (jakost) usmerjenega vetra na drevo in
- $\mathbf{w}$ - smer usmerjenega vetra na drevo.

Glede na to, ali parameter opisuje lastnost drevesa, ki je enaka za celotno drevo, ali pa lastnost, ki se spreminja z lego v drevesu, ločimo dve vrsti parametrov. Prvi parametri so globalni in od reda veje (g, w) neodvisni, drugi parametri so lokalni in od reda veje (g, w) odvisni. Poudariti moramo, da globalnost tu velja za eno drevo.

Globalne lastnosti (parametre) drevesa (npr. velikost listov) in začetne vrednosti za tvorbo modela (npr. začetno število žil) interaktivno določimo s pomočjo pogovornega okna (slika 3.4), katerih vrednosti lahko vnesemo z vpisom števila ali s pomočjo drsnikov. Drsniki omogočajo vnos vrednosti naslednjih parametrov:

Slika 3.4: Pogovorno okno za interaktivno nastavljanje globalnih parametrov parametričnega modela drevesa. Ob spreminjanju parametrov se takoj osveži tudi prikaz drevesa.

- število žil S med 0 in 5000,
- višino prvega odseka debla $l_0^{0,0}$ med 0 m in 10 m,
- koeficient debeline veje k_d med 0 in $0,05$,
- gravicentralizem k_c med 0 in 1,
- kot filotakse α_p med $0°$ in $360°$,
- gostoto listov ρ_l med 0 in 30,
- velikost listov l_l med 0 in $0,3$ ter
- hitrost vetra w_g med 0 in 10.

V istem pogovornem oknu iz padajočega menija izberemo tudi tip porazdelitve listov l_{type} z vrednostmi *Spiral*, *Stacked*, *Staggered*, *Bunched* in *Coniferous*.

Od reda veje (g, w) odvisni parametri so: porazdelitev žil $k_s^{g,w}$, kot vejitve $\alpha^{g,w}$, dejanske omejitve dolžin vej $M^{g,w}$ in $m^{g,w}$, skalirni faktor vej $k_l^{g,w}$ ter vpliv gravimorfizma $\alpha_m^{g,w}$. Za opis od reda veje (g, w) odvisnega parametra uporabimo dva pomožna parametra, ki ju podamo grafično. Prvi parameter je odvisen od Graveliusovega reda ($g \in [0, 15]$), drugi pa od Weibullovega reda ($w \in [0, 50]$). Mejni vrednosti $g = 15$ in $w = 50$ sta izkustveno določeni in zadostujeta za načrtovanje dreves z deset in

več tisoč žilami. Od reda (g, w) odvisne parametre določamo samo za celoštevilske vrednosti g in w, čeprav sta oba pomožna parametra podana z zvezno lomljenko.

Od reda (g, w) odvisne parametre računamo po naslednjih enačbah:

$$k_s^{g,w} = \max\left\{\min\left\{k_s^g k_s^w, 1\right\}, \frac{1}{2}\right\}, \tag{3.17}$$

pri čemer so $k_s^{g,w} \in [\frac{1}{2}, 1]$ ter pomožna parametra $k_s^g \in [\frac{1}{2},\ 1]$ in $k_s^w \in [0, 2]$,

$$\alpha^{g,w} = \min\left\{\alpha^g \alpha^w, 180^\circ\right\}, \tag{3.18}$$

pri čemer so $\alpha^{g,w} \in [0^\circ, 180^\circ]$ ter pomožna parametra $\alpha^g \in [0^\circ, 180^\circ]$ in $\alpha^w \in [0, 2]$,

$$\alpha_m^{g,w} = \max\left\{\min\left\{\alpha_m^g \alpha_m^w, 180^\circ\right\}, -180^\circ\right\}, \tag{3.19}$$

pri čemer so $\alpha_m^{g,w} \in [-180^\circ, 180^\circ]$ ter pomožna parametra $\alpha_m^g \in [-180^\circ, 180^\circ]$ in $\alpha_m^w \in [0, 2]$,

$$M^{g,w} = M^g M^w, \tag{3.20}$$

pri čemer so $M^{g,w} \in [0, 20]$ ter pomožna parametra $M^g \in [0, 10]$ in $M^w \in [0, 2]$,

$$m^{g,w} = m^g m^w, \tag{3.21}$$

pri čemer so $m^{g,w} \in [0, 20]$ ter pomožna parametra $m^g \in [0, 10]$ in $m^w \in [0, 2]$,

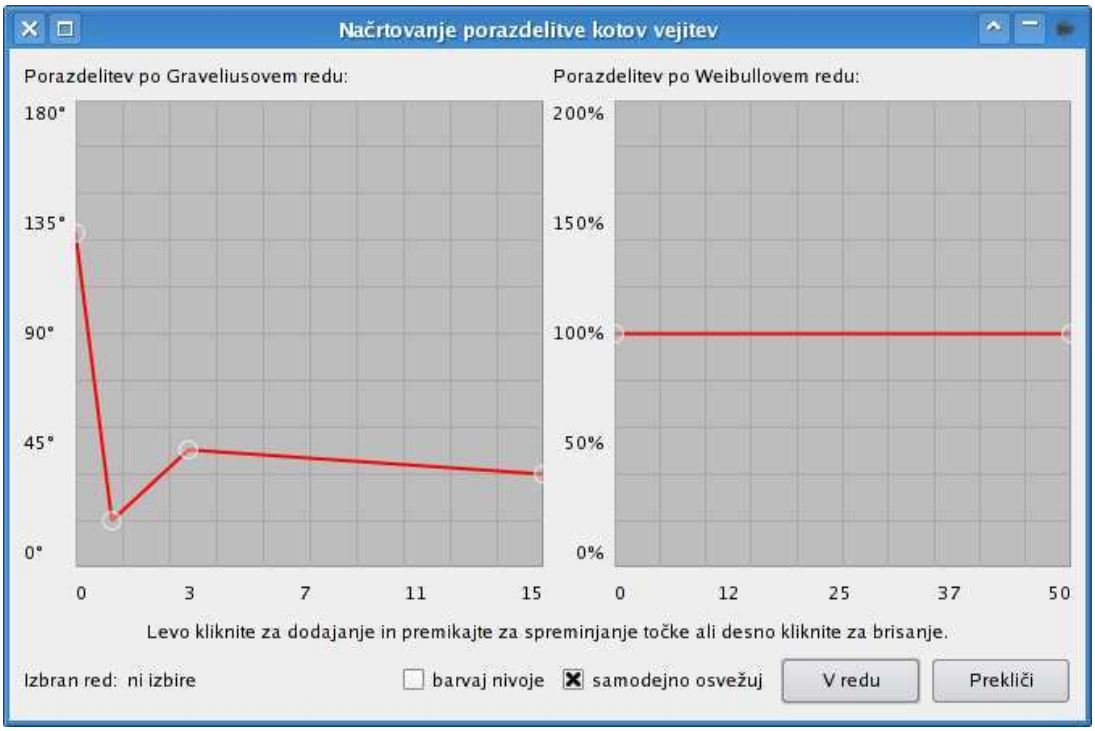
$$k_l^{g,w} = k_l^g k_l^w, \tag{3.22}$$

pri čemer so $k_l^{g,w} \in [0, 20]$ ter pomožna parametra $k_l^g \in [0, 10]$ in $k_l^w \in [0, 2]$.

Vrednosti omenjenih lokalnih pomožnih parametrov podamo s poljubno oblikovano kontrolno lomljenko, njihovo nastavljanje pa je grafično in poteka preko pogovornega okna (slika 3.5).

Lomljenki lahko na poljubnem mestu dodamo novo kontrolno točko, s čemer pripadajoči linearni odsek razpade na dva nova. S premikanjem kontrolnih točk določimo potek vrednosti parametra na ordinatni osi v odvisnosti od Graveliusovega in Weibullovega reda (na abcisni osi). Čeprav sta oba reda celoštevilski vrednosti, nismo omejili položajev kontrolnih točk na abcisni osi, ker bi s tem zmanjšali okretnost modeliranja. Rezultat takega modeliranja je, da dobimo določene lokalne parametre za vse celoštevilske vrednosti redov g in w.

V dodatno pomoč pri oblikovanju kontrolne lomljenke parametra je vizualni namig, pri katerem se del drevesne strukture, na katerega bo vplivala sprememba



Slika 3.5: Interaktivno načrtovanje pomožnih parametrov α^g in α^w za izračun kotov vejitev $\alpha^{g,w}$.

izbrane kontrolne točke oz. odseka lomljenke, pobarva z drugačno barvo (slika 3.6).

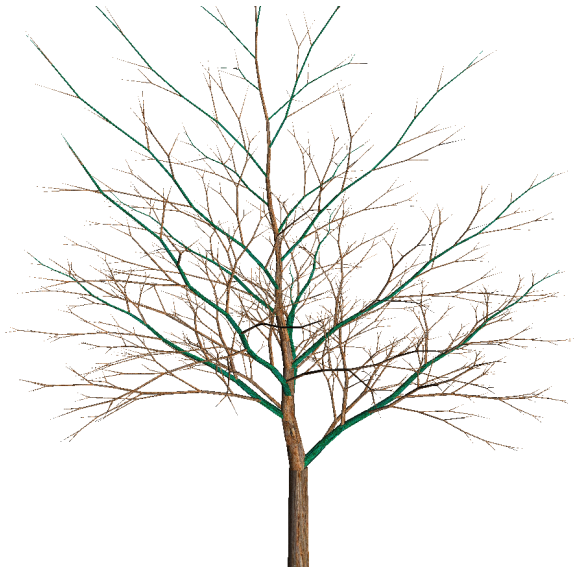
Vsaka sprememba oblike kontrolne lomljenke se takoj odraža na vizualiziranem modelu, zato je mogoče v relativno kratkem času doseči skoraj poljubno drevesno obliko (slike 3.7, 3.8, 3.9, 3.10). V pogovornem oknu na slikah smo spreminjali grafa pomožnih parametrov za parameter gravimorfizma $\alpha_m^{g,w}$.

3.3.1 Gradnja geometrijske strukture dreves

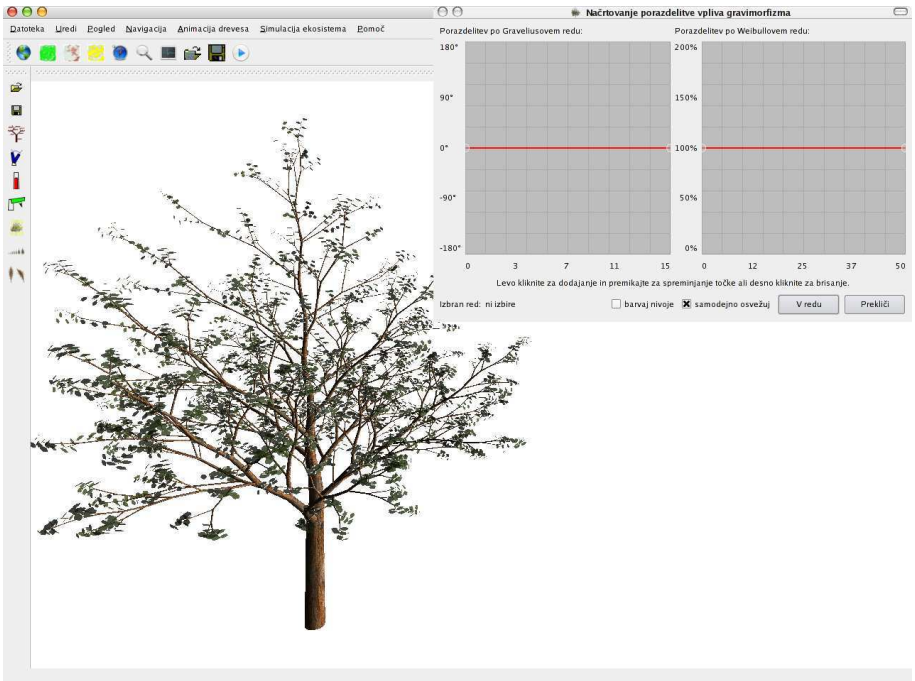
Modelirnik na osnovi podanih oz. z grafi oblikovanih vrednosti parametrov zgradi žilni model in sproti tvori geometrijsko strukturo drevesa. Medtem ko Holton kot geometrijske gradnike uporablja ploskve NURBS, smo mi deblo in veje predstavili kot zaporedje povezanih prisekanih piramid (slika 3.11), podobno kot v modelu Weber-Penn. Razlog za to je v možnosti interaktivne obdelave in veliko enostavnejši animaciji. Listi oz. iglice so štirikotniki s teksturo, ki lahko vsebuje prozorna območja in tako določa poljubno obliko lista.

V nadaljevanju bomo na kratko opisali razlike v matematičnem modelu za gradnjo geometrijske strukture drevesa od Holtonovega modela.

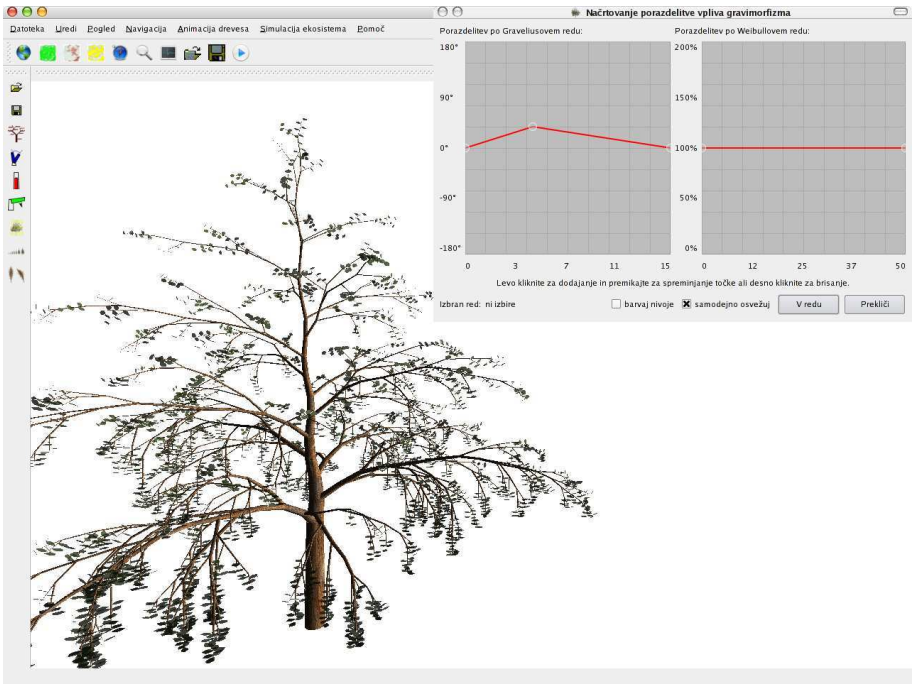
Gravimorfizem smo definirali s kotom $\alpha_m^{g,w} \in [-180°, 180°]$ in s tem združili gravimorfizem s tendenco po rasti navzgor. Ker je vektor zasuka za gravimorfizem definiran v globalnem koordinatnem sistemu (t. j. prvi odsek debla), za ta ko-



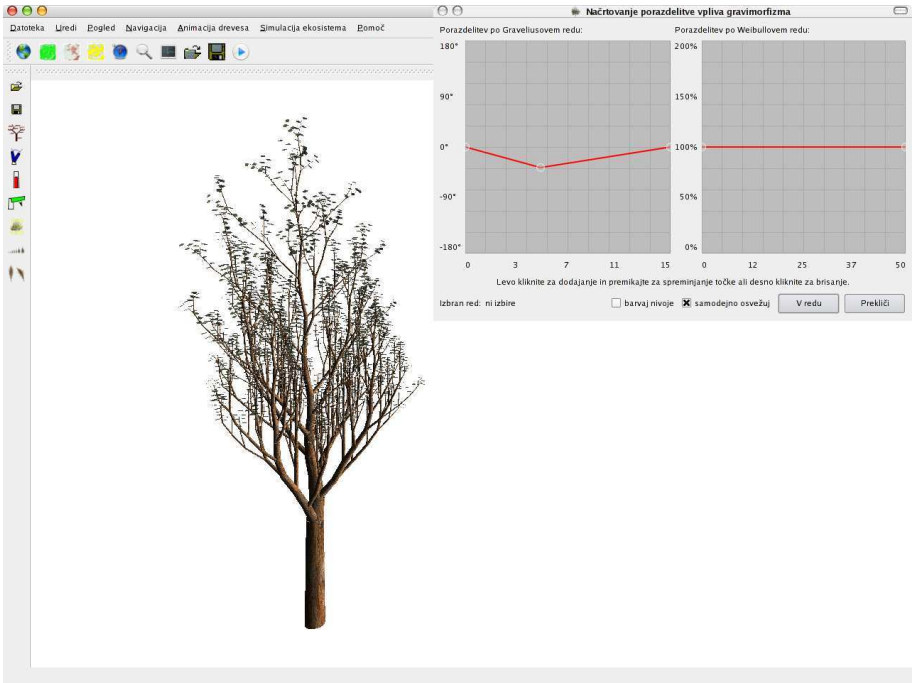
Slika 3.6: Interaktivno obarvanje delov veje, na katere bo vplivalo spreminjanje lokalnega parametra $\alpha^{g,w}$. Na sliki so z zeleno barvo obarvane vse veje z Graveliusovim redom $g = 1$.



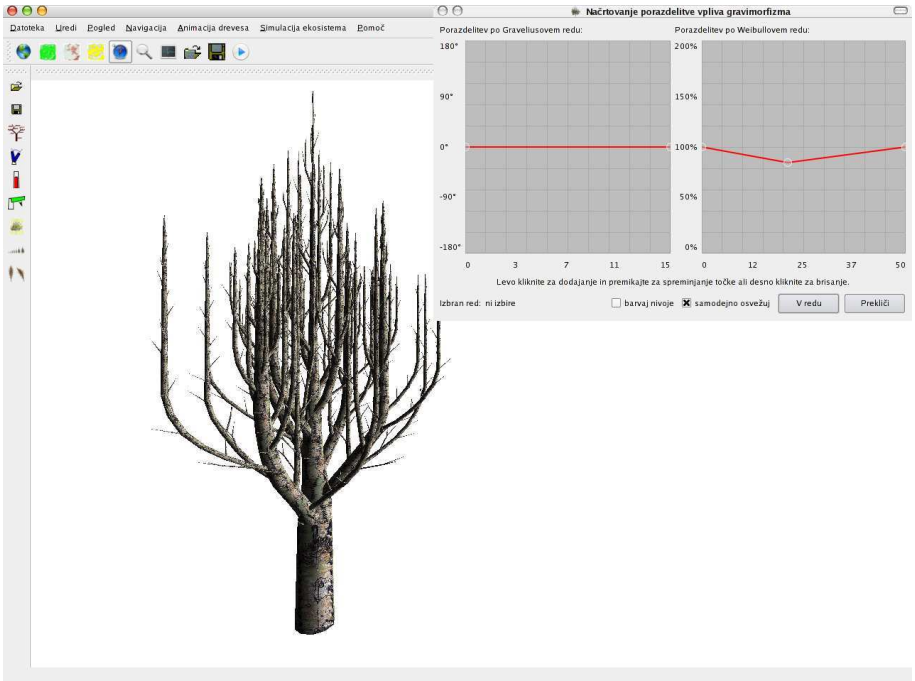
Slika 3.7: Interaktivno oblikovanje modela drevesa, osnovno drevo.



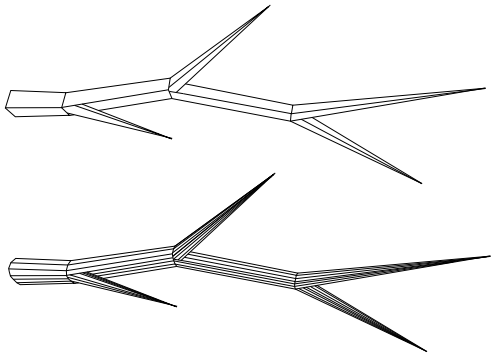
Slika 3.8: Interaktivno oblikovanje modela drevesa, dodan pozitivni gravimorfizem.



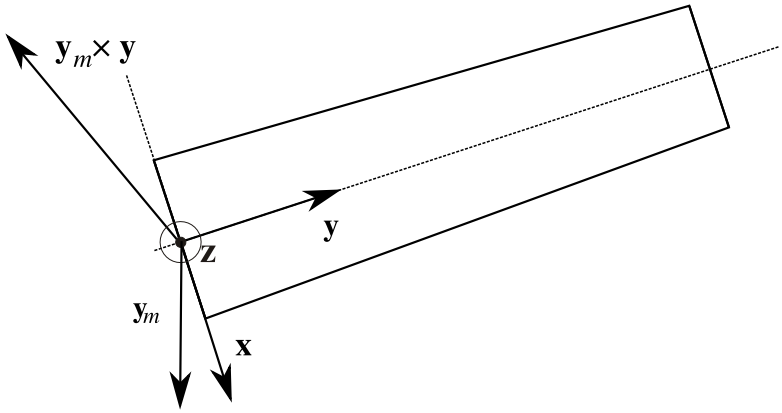
Slika 3.9: Interaktivno oblikovanje modela drevesa, dodan negativni gravimorfizem po redu g.



Slika 3.10: Interaktivno oblikovanje modela drevesa, dodan negativni gravimorfizem po redu w.



Slika 3.11: Geometrijska struktura vej z vrsto prisekanih piramid. Zgoraj je prikazana struktura s 4-strano piramido, spodaj s 15-strano piramido. Notranje veje ponazorimo s prisekanimi piramidami, končne veje (veje z eno žilo) pa s piramidami.



Slika 3.12: Določitev vektorja zasuka pri gravimorfizmu. Skozi razvoj drevesne strukture si vodimo vektor $\mathbf{y}_m$ v smeri navzdol, z vektorskim produktom tega vektorja in vektorja vzdolž veje v trenutnem koordinatnem sistemu, $\mathbf{y}$, pa dobimo nanju pravokoten vektor $\mathbf{y} \times \mathbf{y}_m$. Če sučemo okoli dobljenega vektorja, vejo upognemo navzdol glede na globalni koordinatni sistem.

ordinatni sistem vodimo inverzno matriko zasukov $\mathbf{M}_m^{-1}$, ki je za glavno podvejo zmnožek naslednjih inverznih transformacij (matrik zasukov $\mathbf{R}$):

$$\mathbf{M}_{m;1}^{-1} = \mathbf{R}_{\mathbf{y}\times\mathbf{y}_m}(-\alpha_m^{g,w})\mathbf{R}_y(-\alpha_p)\mathbf{R}_z(-\alpha_1)\mathbf{M}_{m;0}^{-1}, \tag{3.23}$$

kjer je $\mathbf{y}_m$ vektor iz globalnega koordinatnega sistema v smeri navzdol, izražen v koordinatnem sistemu trenutne veje in $\mathbf{y}$ vektor v smeri osi y v trenutnem koordinatnem sistemu. Z vektorskim produktom omenjenih vektorjev dobimo nanju pravokoten vektor $\mathbf{y} \times \mathbf{y}_m$ (glej sliko 3.12), ki določa os zasuka veje. $\mathbf{M}_{m;1}^{-1}$ je inverzna matrika zasukov iz osnovne veje (glej sliko 3.24). Inverzna matrika $\mathbf{M}_{m;0}^{-1}$ ima na začetku vrednost identitete.

Za stransko podvejo si analogno vodimo matriko $\mathbf{M}_{m;2}^{-1}$, pri čemer kot vejitve α_1 zamenjamo s kotom α_2.

Dolžino glavne podveje l_1 in stranske podveje l_2 do naslednje vejitve smo izračunali po enačbah:

$$r_1 = \max\left\{\min\left\{\sqrt{\frac{S_1}{S_0}}, M^{g,w}\right\}, m^{g,w}\right\}, \qquad r_2 = \max\left\{\min\left\{\sqrt{\frac{S_2}{S_0}}, M^{g,w}\right\}, m^{g,w}\right\}, \tag{3.24}$$

$$L_1 = r_1 L_0, \qquad L_2 = r_2 L_0, \tag{3.25}$$

$$l_1 = k_l^{g,w} L_1, \qquad l_2 = k_l^{g,w} L_2. \tag{3.26}$$

V enačbah nastopajo trije uporabniško nastavljivi parametri. Parametra $m^{g,w} \in [0, 20]$ in $M^{g,w} \in [0, 20]$ določata spodnjo in zgornjo mejo koeficientov r_1 ter r_2 in ju

Slika 3.13: Skupina dreves, ki smo jo upodobili s senčilnikom iz programskega paketa LightWave.

podamo s pomočjo kontrolnih lomljenk, podobnima tistima iz slike 3.5. r_1 določa relativno dolžino glavne podveje, L_1, glede na dolžino osnovne veje L_0, medtem ko r_2 določa relativno dolžino stranske podveje, L_2. Parameter $k_l^{g,w}$ je skalirni faktor, s pomočjo katerega izračunamo dejansko dolžino glavne podveje, l_1, in dolžino stranske podveje l_2.

Iz geometrijske strukture drevesa tvorimo 3D model tako, da veje oblečemo s prisekanimi stožci. Prisekane stožce aproksimiramo z n-stranimi prisekanimi piramidami ($n \in \{4, 15\}$, slika 3.11).

Geometrijsko strukturo drevesa lahko izrišemo takoj ali shranimo posamezna oglišča štirikotnikov. Točke dobimo tako, da njihove koordinate izrazimo v koordinatnem sistemu prvega odseka debla, kar dobimo iz matrike M_0. Če se odločimo za shranjevanje točk, lahko te uvozimo v drugih programih, na primer LightWave (slika 3.13).

3.3.2 Modeliranje listov

Listi so razporejeni enakomerno vzdolž končnih vej in rastejo pravokotno navzven. List je upodobljen kot kvadrat, na katerega prilepimo teksturo. Slika za teksturo vsebuje informacije o transparentnosti, ki določi videno obliko lista. Primer dveh tekstur je viden na sliki 3.14.

Število in velikost listov na posamezni veji določata posebna parametra, še en parameter pa določa tip porazdelitve listov vzdolž veje. Tipi porazdelitev listov so določeni po zgledu programskega paketa Corel Bryce 5 [Vera, 2001]. V kolikor

Slika 3.14: Primer dveh tekstur za liste. Ozadje teksture ima transparentnost, postavljeno na 1, zato je tisti del prozoren.

je listov neparno število, je neparni list postavljen na konico veje. Ostali listi so razporejeni glede na enega od petih tipov:

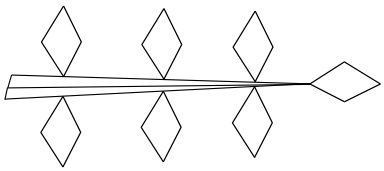
1. **nakopičeno** (angl. *stacked*) razporejeni listi so v paru po dva na vsaki strani veje, postavljeni pravokotno navzven od veje (slika 3.15),

2. **nestalno** (angl. *staggered*) razporejeni listi se izmenjujejo po eden na vsaki strani veje, postavljeni pravokotno na vejo (slika 3.16),

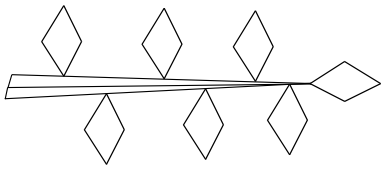
3. **spiralno** (angl. *spiral*) razporejeni listi se ovijajo okoli veje za nek določen kot pravokotno na vzdolžno os veje (slika 3.17),

4. **šopasto** (angl. *bunched*) razporejeni listi se zvrstijo vsi na koncu veje (slika 3.18) ali

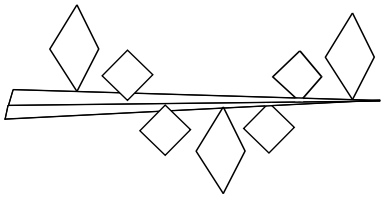
5. **igličasto** (angl. *coniferous*) razporejeni listi so razporejeni kot nestalni in oklepajo z vejo ostri kot (slika 3.19).



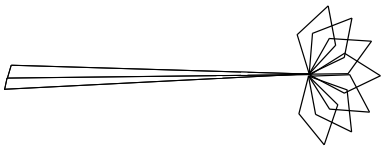
Slika 3.15: Nakopičeno razporejeni listi (tip 1).



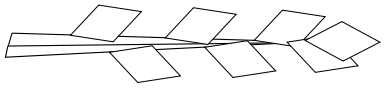
Slika 3.16: Nestalno razporejeni listi (tip 2).



Slika 3.17: Spiralno razporejeni listi (tip 3).



Slika 3.18: Šopasto razporejeni listi (tip 4).



Slika 3.19: Igličasto razporejeni listi (tip 5).

3.3.3 Poenostavljanje geometrijske strukture drevesa

Stopnjo poenostavitve drevesa z indeksom (s, p), l_{LOD} (*level-of-detail*, LOD), določimo iz njegove oddaljenosti od gledišča $\mathbf{p}_v$ po sledeči enačbi:

$$l_{LOD} = \left\lfloor \sqrt{\frac{\|\mathbf{p}_v - \mathbf{p}_{s,p}\|}{5}} \right\rfloor, \tag{3.27}$$

kjer je $\mathbf{p}_{s,p}$ lokacija drevesa z indeksom (s, p) in število 5 izkustveno določeno s testi.

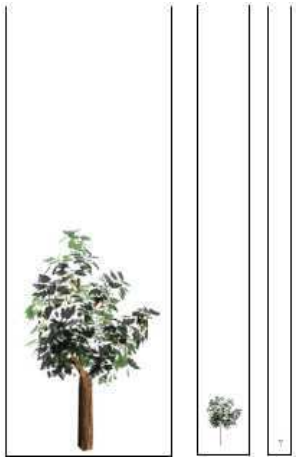
Če je $l_{LOD} = 0$, poenostavljanja ne izvršimo. Sicer pa geometrijsko strukturo drevesa razvijamo do vej z Weibullovim redom:

$$w_{geom} = (10 - l_{LOD})^2. \tag{3.28}$$

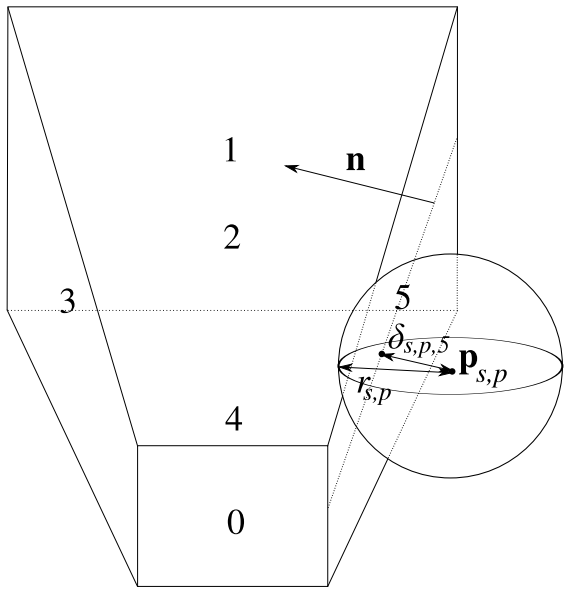
Nivoje zaključimo z listi, vendar pa nekaterih vej pri tem več ne rišemo. Veje rišemo do vej z Weibullovim redom:

$$w_{branch} = 10 - l_{LOD}. \tag{3.29}$$

Geometrijsko strukturo dreves poenostavimo glede oddaljenosti od opazovalca, tako da z izrisom prisekanih piramid pri bolj oddaljenih drevesih končamo pri manjšem redu w. Strukturo še gradimo, le piramid ne rišemo. Na sliki 3.20 vidimo isto drevo na različnih razdaljah od gledišča z izračunano stopnjo poenostavitve l_{LOD}.



Slika 3.20: Poenostavljanje geometrijske strukture drevesa z oddaljenostjo od opazovalca. Prvo drevo nastane, ko je opazovalec 50 m stran ($l_{LOD} = 3$, veje do globine $w_{branch} = 7$, geometrijska struktura do največ $w_{geom} = 49$), drugo pri 200 m ($l_{LOD} = 6$, veje do globine $w_{branch} = 4$, geometrijska struktura do največ $w_{geom} = 16$), tretje pri 700 m ($l_{LOD} = 10$, samo deblo $w_{branch} = 0$, geometrijska struktura do največ $w_{geom} = 0$).



Slika 3.21: Obrezovanje na vidni volumen. S primerjavo radija $r_{s,p}$ obkrožajoče krogle drevesa s centrom v $\mathbf{p}_{s,p}$ in oddaljenostjo od ravnine $\delta_{s,p,5}$ določimo, na kateri strani ravnine 5 se drevo nahaja.

Če je drevo v celoti izven vidnega volumna, drevesa ne narišemo. Tako na sceni z 10.000 drevesi, kjer imamo v vidnem volumnu le 100 dreves, izrišemo le-te, kar drastično poveča zmogljivost aplikacije. Da ugotovimo, ali bo drevo prikazano, moramo testirati, ali je kak del geometrijske očrtajoče krogle drevesa znotraj vidnega volumna. Hitra rešitev je testiranje, na kateri strani ravnin vidnega volumna se objekt nahaja [Picco, 2003].

Naša naloga je ugotoviti, ali je obkrožajoča krogla znotraj vidnega volumna, zunaj ali pa če ga obkroža. Rešitev dobimo na naslednji način: za vseh 6 ravnin vidnega volumna testiramo, ali je krogla na notranji strani ravnine, zunanji strani ravnine ali pa ravnino seka (slika 3.21).

Če ravnino predstavimo z eksplicitno enačbo:

$$ax + by + cz + d = 0 \tag{3.30}$$

in jo delimo z $\sqrt{a^2 + b^2 + c^2}$, da dobimo:

$$Ax + By + Cz + D = 0, \tag{3.31}$$

tedaj je vektor $\overline{\mathbf{n}} = [A, B, C]^T$ normaliziran vektor normale, D pa razdalja vzdolž $\overline{\mathbf{n}}$ od ravnine do koordinatnega izhodišča. Normalo postavimo tako, da kaže notri proti vidnemu volumnu. Razdaljo od poljubne točke $\mathbf{p}_{s,p} = [x_{s,p} \;\; y_{s,p} \;\; z_{s,p}]^T$ do ravnine,

δ, izračunamo po enačbi:

$$\delta = \mathbf{p}_{s,p}\overline{\mathbf{n}} + D. \tag{3.32}$$

Razdalja je lahko pozitivna ali negativna in to določa, na kateri strani ravnine točka $\mathbf{p}_{s,p}$ leži. Če je razdalja manjša od radija, je drevo zunaj vidnega volumna. Če je absolutna vrednost manjša od radija, se drevo seka z ravnino, sicer nadaljujemo s testiranjem še za preostale ravnine. Opisan postopek podaja algoritem 1.

Algoritem 1 Izračun vsebnosti drevesa v vidnem volumnu.

Vhod: $\mathbf{p}_{s,p}$ - središče obkrožajoče krogle drevesa
Vhod: $r_{s,p}$ - radij obkrožajoče krogle
Vhod: D_i - razdalja od izhodišča do ravnine vzdolž normale ravnine
Izhod: ali je drevo vsebovano v vidnem volumnu

```
for i = 0 to 5 do
  δ_{s,p,i} = p_{s,p} n̄_i + D_i; {skalarni produkt z lokacijo drevesa in razdalja do ravnine}
  if δ_{s,p,i} < −r_{s,p} then
    return OUT;{δ_{s,p,i} + r_{s,p} < 0, drevesa ne narišemo}
  end if
  if |δ_{s,p,i}| < r_{s,p} then
    return INTERSECT;{δ_{s,p,i} − r_{s,p} < 0 ali δ_{s,p,i} + r_{s,p} < 0, drevo narišemo}
  end if
end for
return IN; {drevo obkroža vidni volumen, ga narišemo}
```

Za ta postopek potrebujemo enačbe ravnin, ki si odvisne od pogleda v OpenGL. Enačbe ravnin hitro in direktno dobimo iz matrike pogleda OpenGL z uporabo ideje iz članka [Gribb in Hartmann, 2001].

3.3.4 Animacija rasti drevesa

Podprli smo dve obliki animacije dreves, in sicer simulacijo rasti in zibanje drevesa v vetru. Simulacijo rasti dosežemo z večanjem začetnega števila žil v drevesu. Če želimo doseči rast drevesa, linearno odvisno od časa, moramo število žil S večati kvadratično s časom t (to pomeni, da se debelina drevesa linearno veča). Hkrati povečujemo dejansko dolžino prvega odseka debla $l_0^{0,0}$, ki določa višino, na kateri se pojavijo prve veje (s tem linearno povečujemo dolžine vseh vej). Če čas rasti rastline t normaliziramo, število žil pri animaciji S' in dejansko dolžino prvega odseka debla ($l_0^{0,0'}$) določimo po enačbah:

$$S' = t^2 S, \tag{3.33}$$

$$l_0^{0,0'} = t^2 l_0^{0,0}. \tag{3.34}$$

Primer rasti drevesa kaže slika 3.22.

Slika 3.22: Simulacija rasti drevesa. Prvo drevo ima 100 žil, zadnje 1000. Normalizirana starost zadnjega drevesa je $t = 1$, prvega pa $t = 0,01$.

3.3.5 Animacija gibanja v vetru

Za razliko od gradnje modelov, ki je povsem deterministična, smo pri animaciji gibanja v vetru dopustili določeno stopnjo naključnosti. Na tak način je animacija precej bolj realistična, saj lahko v nasprotnem primeru hitro opazimo ponavljajoči se vzorec v gibanju. Animirali smo dva tipa gibanja v vetru, prvi je pozibavanje v vetru spremenljive smeri, drugi pa je nagib vej v smeri stalnega (usmerjenega) vetra. Parametri drevesa, ki vplivajo na izračunani model gibanja, so prožnost vej k_f, dejanska dolžina veje l_0, moč vetra spremenljive smeri w_s, smer usmerjenega vetra $\mathbf{w}$ in moč usmerjenega vetra w_g.

Pri animaciji pihanja neusmerjenega vetra vejo zasučemo okoli osi vzdolž veje in osi pravokotne na os vzdolž veje (slika 3.23). Kota omenjenih zasukov označimo z α_x in α_z ter izračunamo glede na čas t po enačbah:

$$\alpha_x(t) = w_s(1 - k_f)l_0 \sin(t + R_x), \qquad \alpha_z(t) = w_s(1 - k_f)l_0 \sin(t + R_z), \tag{3.35}$$

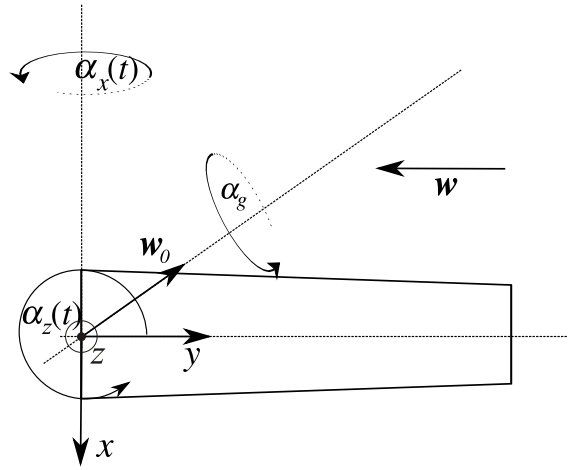
pri čemer sta R_x in R_z naključni racionalni števili za vsako vejo ($R_x, R_y \in [0, 2\pi]$).

Pri animaciji usmerjenega vetra normaliziran vektor $\mathbf{w}_0$ (smer vetra) določa smer zasuka veje. Jakost usmerjenega vetra w_g določa kot zasuka α_g okoli vektorja $\mathbf{w}$ (slika 3.23), ki je odvisen od števila žil S_0, po naslednji enačbi:

$$\alpha_w = \frac{S_0}{S} w_g. \tag{3.36}$$

Vektor $\mathbf{w}_0$ izračunamo kot:

$$\mathbf{w}_0 = \mathbf{y} \times \mathbf{M}_w^{-1}\mathbf{w}, \tag{3.37}$$



Slika 3.23: Zasuk veje zaradi usmerjenega vetra.

kjer $\mathbf{y}$ pomeni vektor v koordinatnem sistemu trenutne veje v smeri y in matrika $\mathbf{M}_w^{-1}$ pomeni inverzno transformacijo koordinatnega sistema trenutne veje v koordinatni sistem prvega odseka debla oz. globalni koordinatni sistem, v katerem je določen vektor $\mathbf{w}$. Inverzno matriko $\mathbf{M}_w^{-1}$ si zaradi hitrejšega izračunavanja vodimo sproti za vsako opravljeno transformacijo, tako da nad obstoječo matriko izvedemo inverzno transformacijo. Matriko $\mathbf{M}_{w;1}^{-1}$ za glavno podvejo dobimo iz matrike $\mathbf{M}_{w;0}^{-1}$ v osnovni podveji po naslednji enačbi:

$$\mathbf{M}_{w;1}^{-1} = \mathbf{R}_{\mathbf{y}\times\mathbf{y}_m}(-\alpha_m^{g,w})\mathbf{R}_y(-\alpha_p)\mathbf{R}_x(-\alpha_x(t))\mathbf{R}_z(-\alpha_1 - \alpha_z(t))\mathbf{R}_{\mathbf{w}_0}(-\alpha_w)\mathbf{M}_{w;0}^{-1}, \tag{3.38}$$

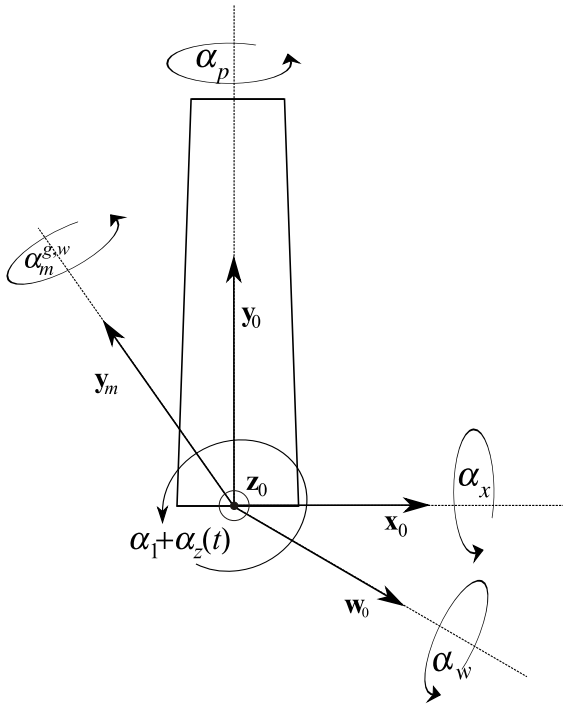
kjer vektor $\mathbf{w}_0$ pomeni vektor v smeri vetra iz koordinatenega sistema prvega odseka debla, izražen v trenutnem koordinatnem sistemu. Navedeni zasuki v algoritmu so vidni na sliki 3.24. $\mathbf{R}$ pomeni matriko zasuka okoli ustrezne smeri (osi, vektorja) za podan kot. Učinek vetra na drevo vidimo na sliki 3.25.

Celotni algoritem za izračun geometrijske strukture drevesa podaja algoritem 2.

3.4 Primeri modeliranja

Drevo lahko modeliramo interaktivno ali parametre vpišemo v vhodno datoteko. Globalni parametri so vpisani v datoteko en za drugim v nespremenjeni obliki. Grafi vektorskih parametrov so v datoteko zapisani uniformno, z normaliziranimi vrednostmi koordinat točk kontrolne lomljenke.

Interaktivno globalne parametre nastavimo z drsniki ali vpišemo numerične vrednosti, tip listov pa izberemo iz spustnega menija (glej sliko 3.4). Sliki za teksture izberemo z dialogom za izbiro datotek. Matrične parametre z indeksi (g, w) nastavljamo z dvema lomljenkama, kot je opisano na strani 19, v razdelku 3.3.

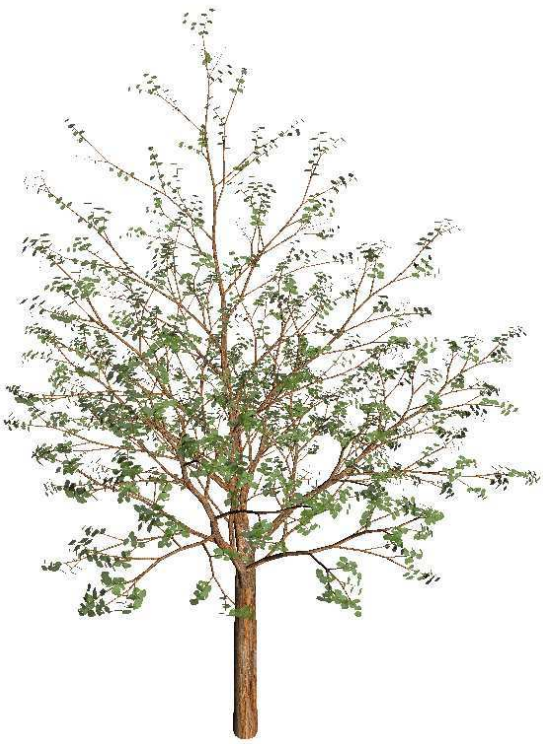


Slika 3.24: Zasuki veje pri razvijanju geometrijske strukture drevesa. Po opravljenih zasukih se pomaknemo vzdolž veje in rekurzivno nadaljujemo gradnjo drevesa.

Slika 3.25: Učinek vetra na geometrijo drevesa. Na levi strani je drevo brez učinka vetra, na desni smo uporabili učinek vetra $w_s = 1,5$ in $w_g = 2,5$.

Algoritem 2 Izračun geometrijske strukture drevesa. Proceduro pokličemo z **drevca**(0, 0, S, 1, $l_0^{0,0}$, **I**, **I**, **I**)

procedure drevca(g, w, S_0, L_0, l_0, $\mathbf{M}_0$, $\mathbf{M}_{m;0}^{-1}$, $\mathbf{M}_{w;0}^{-1}$)
Vhod: g, w - Graveliusov in Weibullov indeks osnovne veje
Vhod: S_0 - število žil v osnovni veji
Vhod: L_0 - relativna dolžina osnovne veje
Vhod: l_0 - dejanska dolžina osnovne veje
Vhod: $\mathbf{M}_0$ - koordinatni sistem trenutne veje
Vhod: $\mathbf{M}_{m;0}^{-1}$ - inverzna matrika zasukov za gravimorfizem v koordinatnem sistemu osnovne veje
Vhod: $\mathbf{M}_{w;0}^{-1}$ - inverzna matrika zasukov za usmerjen veter v koordinatnem sistemu osnovne veje
Vhod: globalni k_d, k_c, l_{type}, $k_s^{g,w}$, $M^{g,w}$, $m^{g,w}$, $k_l^{g,w}$, $\alpha_m^{g,w}$, $\alpha^{g,w}$,t, k_f, w_s, w_g (str. 17)
Izhod: narisano drevo
$d := k_d\sqrt{S_0}$; {izračunamo debelino po Mandelbrotu}
nariši osnovno vejo ($\mathbf{M}_0$, l_0, d);
if $S_0 = 1$ **then**
riši liste(l_{type});
return;
end if
$S_1 := \lceil 1 + k_s^{g,w}(S_0 - 2)\rceil$; {število žil v glavni podveji}
$S_2 = S_0 - S_1$; {število žil v stranski podveji}
$r_1 := \max\left\{\min\left\{\sqrt{\frac{S_1}{S_0}}, M^{g,w}\right\}, m^{g,w}\right\}$; {določimo razmerje dolžin v odvisnosti od razmerja debelin}
$r_2 := \max\left\{\min\left\{\sqrt{\frac{S_2}{S_0}}, M^{g,w}\right\}, m^{g,w}\right\}$;
$L_1 := r_1 L_0$; {relativna dolžina podvej}
$L_2 := r_2 L_0$;
$l_1 := k_l^{g,w} L_1$; {določimo dejansko dolžina podvej}
$l_2 := k_l^{g,w} L_2$;
$\alpha_1 := k_c\sqrt{\frac{S_2}{S_0}}\alpha^{g,w}$; {izračunamo kote vejitev}
$\alpha_2 := \alpha^{g,w} - \alpha_1$;
$\alpha_x(t) := \sin(t + R_x)w_s(1 - k_f)l_0$; {animacija neusmerjenega vetra}
$\alpha_z(t) := \sin(t + R_z)w_s(1 - k_f)l_0$;
$\alpha_w := \frac{S_0}{S}w_g$; {animacija usmerjenega vetra}
$\mathbf{M}_1 := \mathbf{R}_{\mathbf{w}_0}(\alpha_w,)\mathbf{R}_z(\alpha_1 + \alpha_z(t))\mathbf{R}_x(\alpha_x(t))\mathbf{R}_y(\alpha_p)\mathbf{R}_{\mathbf{y}\times\mathbf{y}_m}(\alpha_m^{g,w})\mathbf{T}_y(l_0)\mathbf{M}_0$;
$\mathbf{M}_2 := \mathbf{R}_{\mathbf{w}_0}(\alpha_w,)\mathbf{R}_z(\alpha_2 + \alpha_z(t))\mathbf{R}_x(\alpha_x(t))\mathbf{R}_y(\alpha_p)\mathbf{R}_{\mathbf{y}\times\mathbf{y}_m}(\alpha_m^{g,w})\mathbf{T}_y(l_0)\mathbf{M}_0$;
$\mathbf{M}_{m;1}^{-1} := \mathbf{R}_{\mathbf{y}\times\mathbf{y}_m}(-\alpha_m^{g,w})\mathbf{R}_y(-\alpha_p)\mathbf{R}_x(-\alpha_x(t))\mathbf{R}_z(-\alpha_1 - \alpha_z(t))\mathbf{M}_{m;0}^{-1}$; {osvežimo inverzno matriko za konstrukcijo vektorja gravimorfizma, veter se ne upošteva}
$\mathbf{M}_{m;2}^{-1} := \mathbf{R}_{\mathbf{y}\times\mathbf{y}_m}(-\alpha_m^{g,w})\mathbf{R}_y(-\alpha_p)\mathbf{R}_x(-\alpha_x(t))\mathbf{R}_z(-\alpha_2 - \alpha_z(t))\mathbf{M}_{m;0}^{-1}$;
$\mathbf{M}_{w;1}^{-1} := \mathbf{R}_{\mathbf{y}\times\mathbf{y}_m}(-\alpha_m^{g,w})\mathbf{R}_y(-\alpha_p)\mathbf{R}_x(-\alpha_x(t))\mathbf{R}_z(-\alpha_1 - \alpha_z(t))\mathbf{R}_{\mathbf{w}_0}(-\alpha_w)\mathbf{M}_{w;0}^{-1}$; {osvežimo inverzno matriko za konstrukcijo vektorja usmerjenega vetra}
$\mathbf{M}_{w;2}^{-1} := \mathbf{R}_{\mathbf{y}\times\mathbf{y}_m}(-\alpha_m^{g,w})\mathbf{R}_y(-\alpha_p)\mathbf{R}_x(-\alpha_x(t))\mathbf{R}_z(-\alpha_2 - \alpha_z(t))\mathbf{R}_{\mathbf{w}_0}(-\alpha_w)\mathbf{M}_{w;0}^{-1}$;
drevca ($g + 1$, $w + 1$, S_2, L_2, l_2, $\mathbf{M}_2$, $\mathbf{M}_{m;2}^{-1}$, $\mathbf{M}_{w;2}^{-1}$); {razvijemo stransko vejo}
drevca (g, $w + 1$, S_1, L_1, l_1, $\mathbf{M}_1$, $\mathbf{M}_{m;1}^{-1}$, $\mathbf{M}_{w;1}^{-1}$); {razvijemo glavno vejo}



Slika 3.26: Upodobitev proceduralnega modela za bukev.

3.4.1 Bukev

Za modeliranje bukve iz slike 3.26 smo izbrali naslednje globalne parametre: $S = 1000$, $l_0^{0,0} = 5$, $k_d = 0,01$, $\alpha_p = 85$, $k_c = 0,3$, $l_l = 0,1$, $\rho_l = 5$, $l_{type} = Spiral$ in $l_{LOD} = 0$.

Razmerje porazdelitve žil $k_s^{g,w}$ določimo za vse rede (g, w), kjer je $g \in [0, 15]$, $w \in [0, 50]$ (g, w so nenegativna cela števila), s pomočjo enačbe 3.17. Vrednosti k_s^g in k_s^w dobimo iz obeh grafov, ki sta podana kot lomljenki $k_s^g = \{ (0, \ \ 0,75), (15, \ \ 0,9) \}$ in $k_s^w = \{ (0, \ \ 1), (50, \ \ 1) \}$.

Kot med izhajajočima podvejama pri delitvi, $\alpha^{g,w}$, določimo za vse rede (g, w) s pomočjo enačbe 3.18. Potrebne vrednosti v omenjeni enačbi α^g in α^w definirata dva grafa, ki sta podana kot lomljenki $\alpha^g = \{ (0, \ \ 45°), (15, \ \ 45°) \}$ in $\alpha^w = \{ (0, \ \ 1), (50, \ \ 1) \}$.

Spodnjo mejo relativne dolžine veje, $m^{g,w}$, določimo za vse rede (g, w) po enačbi 3.19, v katero vstavimo $m^g = \{ (0, \ \ 1), (7,5, \ \ 0,45), (15, \ \ 1) \}$ in $m^w = \{ (0, \ \ 1), (50, \ \ 1) \}$.

Zgornjo mejo relativne dolžine veje, $M^{g,w}$, določimo za vse rede (g, w) po enačbi 3.20, v katero vstavimo $M^g = \{ (0, \ \ 1), (15, \ \ 1) \}$ in $M^w = \{ (0, \ \ 1), (50, \ \ 1) \}$.

Skalirni faktor dolžine veje, $k_l^{g,w}$, določimo za vse rede (g, w) po enačbi 3.22, v katero vstavimo $k_l^g = \{ (0, \ \ 1), (15, \ \ 1) \}$ in $k_l^w = \{ (0, \ \ 1), (50, \ \ 1) \}$.

Slika 3.27: Upodobitev proceduralnega modela za smreko.

Kot zasuka za gravimorfizem, $\alpha_m^{g,w}$, določimo za vse rede (g, w) po enačbi 3.21, v katero vstavimo $\alpha_m^g = \{ (0, \ 0°), (15, \ 0°) \}$ in $\alpha_m^w = \{ (0, \ 1), (50, \ 1) \}$.

3.4.2 Smreka

Za modeliranje smreke na sliki 3.27 smo izbrali naslednje globalne parametre: $S = 6000$, $l_0^{0,0} = 5,47$, $k_d = 0,005$, $\alpha_p = 85$, $k_c = 0,98$, $l_l = 0,1815$, $\rho_l = 30$, $l_{type} =$ *Stacked* in $l_{LOD} = 0$.

Razmerje porazdelitve žil $k_s^{g,w}$ določimo za vse rede (g, w), kjer je $g \in [0, 15]$, $w \in [0, 50]$ (g, w so nenegativna cela števila), s pomočjo enačbe 3.17. Vrednosti k_s^g in k_s^w dobimo iz obeh grafov, ki sta podana kot lomljenki $k_s^g = \{ (0, \ 0,955), (0,7749, \ 0,9615), (1,827, \ 0,8288), (4,207, \ 0,8269), (15, \ 0,8294) \}$ in $k_s^w = \{ (0, \ 1), (42,89, \ 1,005), (42,89, \ 0,654), (50, \ 0,6256) \}$.

Kot med izhajajočima podvejama pri delitvi, $\alpha^{g,w}$, določimo za vse rede (g, w) s pomočjo enačbe 3.18. Potrebne vrednosti v omenjeni enačbi α^g in α^w definirata dva grafa, ki sta podana kot lomljenki $\alpha^g = \{ (0, \ 128,8°), (1,164, \ 17,91°), (3,621, \ 45,21°), (15, \ 35,83°) \}$ in $\alpha^w = \{ (0, \ 1), (50, \ 1) \}$.

Slika 3.28: Upodobitev proceduralnega modela za vrbo.

Spodnjo mejo relativne dolžine veje, $m^{g,w}$, določimo za vse rede (g, w) po enačbi 3.19, v katero vstavimo $m^g = \{ (0, \quad 0,7109), (1,293, \quad 0,8531), (15, \quad 1) \}$ in $m^w = \{ (0, \ 1), (50, \ 1) \}$.

Zgornjo mejo relativne dolžine veje, $M^{g,w}$, določimo za vse rede (g, w) po enačbi 3.20, v katero vstavimo $M^g = \{ (0, \quad 0,9953), (15, \quad 1) \}$ in $M^w = \{ (0, \quad 1), (50, \ 1) \}$.

Skalirni faktor dolžine veje, $k_l^{g,w}$, določimo za vse rede (g, w) po enačbi 3.22, v katero vstavimo $k_l^g = \{ (0, \ 0,7109), (0,7749, \ 0,5385), (2,522, \ 1,327), (15, \ 1) \}$ in $k_l^w = \{ (0, \ 1,014), (38,58, \ 1,014), (45,26, \ 0,5877), (50, \ 0,01896) \}$.

Kot zasuka za gravimorfizem, $\alpha_m^{g,w}$, določimo za vse rede (g, w) po enačbi 3.21, v katero vstavimo $\alpha_m^g = \{ (0, \quad 0°), (0,3875, \quad 34,62°), (1,034, \quad -4,265°), (1{,}55, -4,154°), (2,974, \ -5,972°), (15, \ -0,8532°) \}$ in $\alpha_m^w = \{ (0, \ 1), (50, \ 1) \}$.

3.4.3 Vrba žalujka

Za modeliranje vrbe iz slike 3.28 smo izbrali naslednje globalne parametre: $S = 1500$, $l_0^{0,0} = 1$, $k_d = 0,01$, $\alpha_p = 85$, $k_c = 0,2$, $l_l = 0,3452$, $\rho_l = 5$, $l_{type} = \textit{Coniferous}$ in $l_{LOD} = 0$.

Razmerje porazdelitve žil $k_s^{g,w}$ določimo za vse rede (g, w), kjer je $g \in [0, 15]$, $w \in [0, 50]$ (g, w so nenegativna cela števila), s pomočjo enačbe 3.17. Vrednosti k_s^g in k_s^w dobimo iz obeh grafov, ki sta podana kot lomljenki $k_s^g = \{ (0, \quad 0,8507),$

$(3,75,\ \ 0,7749)$, $(15,\ \ 0,9)$ } in $k_s^w = \{$ $(0,\ \ 1)$, $(50,\ \ 1)$ }.

Kot med izhajajočima podvejama pri delitvi, $\alpha^{g,w}$, določimo za vse rede (g,w) s pomočjo enačbe 3.18. Potrebne vrednosti v omenjeni enačbi α^g in α^w definirata dva grafa, ki sta podana kot lomljenki $\alpha^g = \{$ $(0,\ \ 17,91°)$, $(1,875,\ \ 57,16°)$, $(15,\ \ 45°)$ } in $\alpha^w = \{$ $(0,\ \ 1)$, $(50,\ \ 1)$ }.

Spodnjo mejo relativne dolžine veje, $m^{g,w}$, določimo za vse rede (g,w) po enačbi 3.19, v katero vstavimo $m^g = \{$ $(0,\ \ 1)$, $(1,81,\ \ 0,9953)$, $(15,\ \ 1)$ } in $m^w = \{$ $(0,\ \ 0,9668)$, $(50,\ \ 1)$ }.

Zgornjo mejo relativne dolžine veje, $M^{g,w}$, določimo za vse rede (g,w) po enačbi 3.20, v katero vstavimo $M^g = \{$ $(0,\ \ 0,9953)$, $(1,422,\ \ 1,848)$, $(15,\ \ 1)$ } in $M^w = \{$ $(0,\ \ 1)$, $(50,\ \ 1)$ }.

Skalirni faktor dolžine veje, $k_l^{g,w}$, določimo za vse rede (g,w) po enačbi 3.22, v katero vstavimo $k_l^g = \{$ $(0,\ \ 1)$, $(15,\ \ 1)$ } in $k_l^w = \{$ $(0,\ \ 1)$, $(22,63,\ \ 0,9953)$, $(37,07,\ \ 0,7678)$, $(50,\ \ 1)$ }.

Kot zasuka za gravimorfizem, $\alpha_m^{g,w}$, določimo za vse rede (g,w) po enačbi 3.21, v katero vstavimo $\alpha_m^g = \{$ $(0,\ \ 84,45°)$, $(1,746,\ \ 94,69°)$, $(15,\ \ 110°)$ } in $\alpha_m^w = \{$ $(0,\ \ 0,6446)$, $(50,\ \ 1)$ }.

Poglavje 4

Modeliranje in simulacija ekoloških parametrov

V tem poglavju opišemo ekološki model dreves. Drevesa razporejamo po terenu v odvisnosti od parametrov, ki vplivajo na njihovo rast. Na ta način simuliramo dogajanje v naravnem okolju. Rast dreves je odvisna predvsem od naslednjih parametrov:

- **nadmorske višine** (*altitude*) [Martinčič s sodelavci, 1981],
- **naklona** (*slope*),
- **vlage** (*humidity*),
- **osončenosti** (*sunniness*),
- **vetrovnosti** (*windiness*),
- **medsebojnega tekmovanja za prostor med rastlinami** (*competition for space*) [Lane in Prusinkiewicz, 2002],
- **rasti blizu starševskih rastlin** (*clustering*, *clumping*, *underdispersion*) [Lane in Prusinkiewicz, 2002; Dale, 1999] in
- **umiranja rastlin zaradi staranja** (*dying with senescence*).

Nekatere parametre dobimo iz točk terena. Pri naši študiji smo koristili prosto dostopni vizualizator, ki pa smo ga kasneje zamenjali z lastnim vizualizatorjem.

Primer vizualizacije terena je prikazan na sliki 4.1.

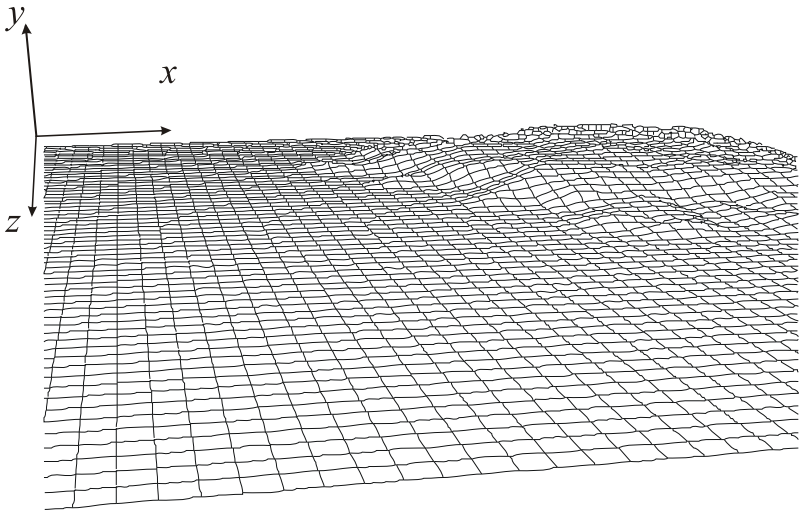
Površje je razdeljeno na 10.000 krp - četverokotnikov (100×100). Mreža, na kateri temelji predstavitev, je dodana terenu na sliki 4.2 in je prikazana v perspektivni

Slika 4.1: Vizualizacija terena z vizualizatorjem za prikaz pokrajine.

projekciji. Mreža četverokotnikov v ravnini xz je uniformna (slika 4.3). Oglišča četverokotnikov v prostoru xyz določajo naslednje koordinate:

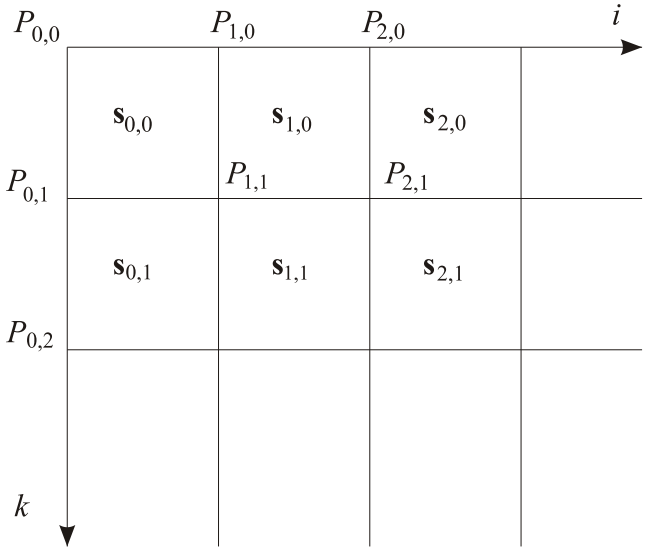
$$\mathbf{p}_{i,k} = [\, x_i \quad y_{i,k} \quad z_k \,]^T, \qquad i, k \in [0, 99]. \tag{4.1}$$

Četverokotniki so v prostoru zaradi različne nadmorske višine različnih velikosti. Označimo jih kot $\mathbf{s}_{i,k}$ $(i, k \in [0, 99])$.



Slika 4.2: Mreža terena, sestavljena iz četverokotnikov.

Točke terena je možno določiti z bitno sliko ali z nalaganjem DMR (digitalnega modela reliefa) [Li s sodelavci, 2005]. V datoteki formata DMR so višinski podatki



Slika 4.3: Mreža četverokotnikov v ravnini xz.

resničnega terena iz satelitov vzorčeni na 25 metrov. Koordinate oglišč terena so zapisane s po tremi vrednostmi, vsaka v svoji vrstici. Prva vrednost pomeni v našem simulatorju koordinato x, druga z in tretja y.

Parametri, ki izhajajo neposredno iz razgibanosti terena, so:

- nadmorska višina,
- naklon,
- vlaga,
- osončenost in
- vetrovnost.

Omenjene parametre dobimo z različnimi algoritmi pred pričetkom simulacije ekosistema in jih ne spreminjamo med tekom simulacije. Te parametre računamo za vsako krpo pokrajine posebej. V nadaljevanju bomo opisali algoritme, ki omogočajo izračun parametrov.

4.1 Nadmorska višina

Nadmorsko višino terena y normaliziramo po formuli:

$$\overline{y} = \frac{y - y_{min}}{y_{max} - y_{min}}, \tag{4.2}$$

pri čemer je y_{min} minimalna nadmorska višina na terenu in y_{max} maksimalna nadmorska višina na terenu. Parameter višine terena $\overline{y}$ bomo uporabili pri odtekanju padavin in določanju vlažnosti zemlje.

4.2 Naklon

Naklon $s_{i,k}$ izračunamo iz komponente y enotskega normalnega vektorja:

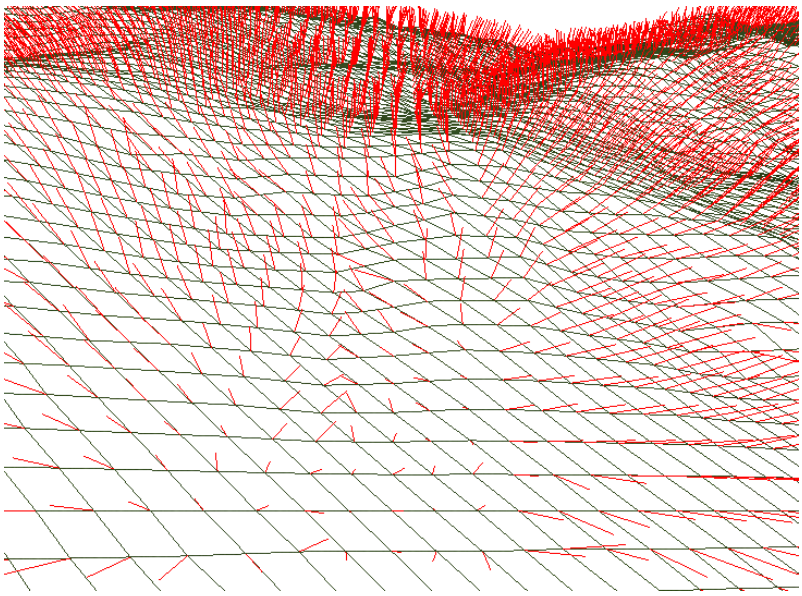
$$s_{i,k} = 1 - \overline{n}_{y;i,k}. \tag{4.3}$$

Popolnoma ravni teren ima zato vrednost naklona 0, popolnoma navpični teren pa vrednost 1.

Pri izračunu enačbe 4.3 potrebujemo normalo $\mathbf{n}_{i,k}$ četverokotnika (krpe) $\mathbf{s}_{ik}$. Leto določimo z vektorskim produktom dveh vektorjev: prvi vektor povezuje obravnavano točko s sosednjo po dolžini terena, drugi vektor pa s sosednjo točko po širini terena:

$$\mathbf{n}_{i,k} = (\mathbf{p}_{i+1,k} - \mathbf{p}_{i,k}) \times (\mathbf{p}_{i,k+1} - \mathbf{p}_{i,k}) = \begin{bmatrix} n_{x;i,k} & n_{y;i,k} & n_{z;i,k} \end{bmatrix}^T. \tag{4.4}$$

Normale $\mathbf{n}_{i,k}$ normiramo in jih označimo kot $\overline{\mathbf{n}}_{i,k}$. Enotski vektorji normal terena so prikazani na sliki 4.4.



Slika 4.4: Enotski vektorji normal po terenu ($\overline{\mathbf{n}}_{i,k}$).

4.3 Vlaga

Na vsako krpo pade enaka količina padavin, te padavine pa ne ostanejo v celoti na določeni krpi, ampak se stekajo k sosednjim krpam (ali pritekajo od sosednjih krp) v odvisnosti od naklona krpe in vpojnosti tal. To pomeni, da vlago v krpi sestavljajo

padavine, ki jih krpa neposredno vpije, in padavine, ki pritečejo iz višjih krp. V bližini tekoče ali mirujoče vode pa se vlagi v krpi doda tudi učinek kapilarnosti.

Krpe, ki sestavljajo teren, imajo v prostoru različno ploščino, vendar pa ima njihova projekcija na ravnino xz vedno enako ploščino. Privzamemo, da so padavine enakomerno porazdeljene po površini ravnine xz in da padajo pravokotno na ravnino xz. Po tej predpostavki zato prejme vsaka krpa (ne glede na velikost) enako količino padavin $m_{climate}$.

Vpojnost tal je funkcija strukture tal in vrste padavin (rahel dež, naliv). V naši simulaciji vpojnost tal trenutno opišemo z odstotkom padavin, ki jih tla vsrkajo.

Na začetku vsem krpam priredimo enako vrednost padavin $m_{climate}$ (na primer $m_{climate} = 1$). Za vsako od krp upoštevamo še širjenje vlage okoli vodotokov in stoječih voda, tako da krpam vodotokov in stoječih voda dodamo velike količine padavin m_w (na primer $m_w = 10$), kot kaže enačba 4.5. Krpe, ki predstavljajo vodotoke in stoječe vode, so tiste, ki imajo komponento $\overline{y}_{i,k}$ manjšo ali enako normalizirani višini nivoja vode $\overline{y}_w \in [0, 1]$, ta pa je nastavljivi parameter simulacije.

$$m_{i,k} := \begin{cases} m_{i,k} + m_w, & \overline{y}_{i,k} \leq \overline{y}_w \\ m_{i,k}, & \text{sicer} \end{cases} . \tag{4.5}$$

Krpe nato izbiramo po višini od najvišje proti nižjim. Denimo, da je izbrana krpa $\mathbf{s}_{i,k}$. Najprej določimo število strani odtekanj te krpe $n_{neigh;i,k}$, ki ga dobimo tako, da preštejemo sosednje krpe (0 do 4), ki imajo nižjo središčno točko.

Pri odtekanju padavin upoštevamo še moč odtekanja k_m, ki je uporabniško nastavljiv parameter. Delež odlitja padavin na eno sosednjo krpo je za trenutno krpo določen s formulo:

$$\Delta m_{i,k} = \begin{cases} 0, & n_{neigh;i,k} = 0 \\ \dfrac{s_{i,k} k_m}{n_{neigh;i,k}}, & n_{neigh;i,k} = 1, 2, 3, 4 \end{cases} . \tag{4.6}$$

To pomeni, da iz bolj strme krpe padavine odtekajo močneje, iz ravne krpe pa padavine ne odtekajo.

Ta delež prištejemo sosednjim krpam, ki imajo nižjo središčno točko:

$$m_{i\pm1,k\pm1} := m_{i\pm1,k\pm1} + \Delta m_{i,k}. \tag{4.7}$$

Vse odlite deleže pa trenutni krpi vselej tudi odštejemo:

$$m_{i,k} := m_{i,k} - n_{neigh;i,k} \Delta m_{i,k}. \tag{4.8}$$

Celotni izračun vlage na koncu N_{mblur}-krat povprečimo (npr. $N_{mblur} = 10$),

s čimer zmehčamo izstopajoče vrednosti in dobimo zvezno širjenje vlage po prsti. Povprečenje za neko krpo v enem koraku opravimo tako, da zanjo shranimo povprečno količino vlage na njej in štirih ostalih obkrožajočih krpah. Splošneje rečeno, nad podatki izvedemo digitalni filter nizkega sita (tekoče povprečenje). Postopek za izračun vlage je zapisan z algoritmom 3.

Algoritem 3 Izračun padavin za ves teren.

Vhod: i_{max}, k_{max} - dimenzije terena
Vhod: $\mathbf{c}_{i,k}$ - središčne točke krp terena
Vhod: N - število krp terena
Izhod: matrika elementov $m_{i,k}$ - vlaga za vsako krpo

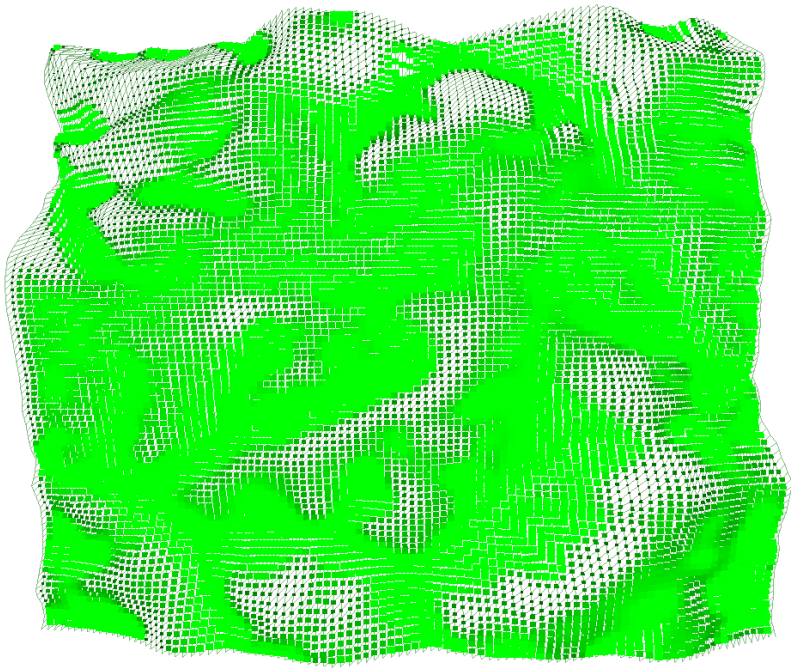
```
for i = 0 to i_max − 1 do
  for k = 0 to k_max − 1 do
    m_{i,k} := m_climate;
    if ȳ_{i,k} ≤ ȳ_w then
      m_{i,k} := m_{i,k} + m_w; {dodamo vlago krpam, ki ustrezajo vodotokom in
      stoječim vodam}
    end if
  end for
end for
krpe uredi nenaraščajoče po višini (c_{i,k}, N); {dobimo na primer seznam (34, 13,
17, ...)}
for j = 0 to N − 1 do
  i, k := dobi indeks i, k iz indeksa j; {dobimo 0 → 34, 1 → 13, 2 → 17, ...}
  n_{neigh;i,k} := določi število strani odtekanj iz krpe s_{i,k};
  Δm_{i,k} := m_{i,k} s_{i,k} k_m / n_{neigh;i,k}; {izračunamo delež odlitja za eno krpo}
  m_{i±1,k±1} := m_{i±1,k±1} + Δm_{i,k}; {sosednjim krpam prištejemo delež padavin}
  m_{i,k} := m_{i,k} − n_{neigh;i,k} Δm_{i,k}; {trenutni krpi odštejemo deleže odteklih padavin}
end for
for j = 0 to N_mblur − 1 do
  m_{i,k} := (m_{i,k} + m_{i−1,k} + m_{i,k+1} + m_{i+1,k} + m_{i,k+1}) / 5; {povečamo vlago krpam ob vodotokih in
  stoječih vodah}
end for
```

Izračunana vlažnost v krpah, ki jo dobimo v programu, je prikazana z velikostjo kvadratov na sliki 4.5.

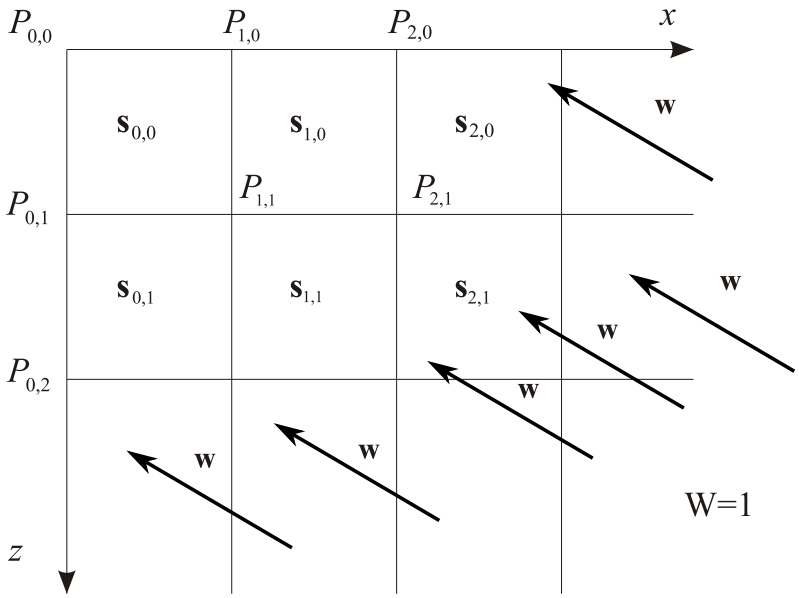
4.4 Vetrovnost

Vetrovnost $w_{i,k}$ v krpi $\mathbf{s}_{i,k}$ določa izpostavljenost te krpe vetru. Ta piha konstantno z jakostjo $W = 1$ s smerjo vektorja $\mathbf{w}$ v ravnini xz, kot prikazuje slika 4.6.

Če so sosednje krpe v smeri $-\mathbf{w}$ višje ležeče od obravnavane, je krpa v zavetrju. Količino vetrovnosti $w_{i,k}$ določimo s pomočjo kota zavetrja $\varphi_{i,k,w} = \measuredangle(\mathbf{c}_{i,k,w}, \mathbf{c}_{i,k},$



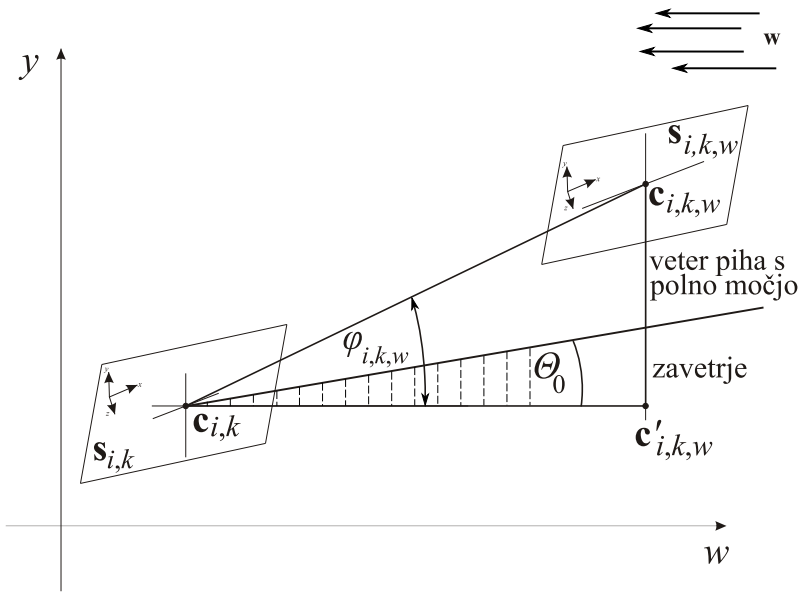
Slika 4.5: Izračunana vlažnost terena s simulacijo odtekanja padavin. Območja z več zelene barve so bolj vlažna.



Slika 4.6: Pihanje vetra v ravnini xz čez cel teren v isti smeri. Jakost vetra W je konstantna za ves teren, prav tako pa je konstantna tudi smer vetra.

$\mathbf{c}'_{i,k,w}$), pri čemer je točka $\mathbf{c}_{i,k,w}$ središčna točka krpe v smeri proti vetru, $\mathbf{c}_{i,k} = \begin{bmatrix} c_{x;i,k} & c_{y;i,k} & c_{z;i,k} \end{bmatrix}^T$ središčna točka obravnavane krpe in $\mathbf{c}'_{i,k,w}$ projekcija točke $\mathbf{c}_{i,k,w}$ na ravnino $y = c_{y;i,k}$ (slika 4.7). Izračun kota zavetrja podaja enačba 4.9:

$$\varphi_{i,k,w} = \arctan \frac{c_{y;i,k,w} - c_{y;i,k}}{\|\mathbf{c}'_{i,k,w} - \mathbf{c}_{i,k}\|}. \tag{4.9}$$



Slika 4.7: Kot zavetrja $\varphi_{i,k,w}$ med središčnima točkama $\mathbf{c}_{i,k}$ in $\mathbf{c}_{i,k,w}$ krp $\mathbf{s}_{i,k}$ in $\mathbf{s}_{i,k,w}$. Krpa $\mathbf{s}_{i,k,w}$ ustreza w-temu vzorcu — ti so na sliki 4.8 označeni s črticami.

Zavetrje v krpi določimo iz najmanjšega kota zavetrja ($\varphi_{i,k} = \min_w \{\varphi_{i,k,w}\}$). Postopek sledenja vektorju proti vetru je prikazan na sliki 4.8.

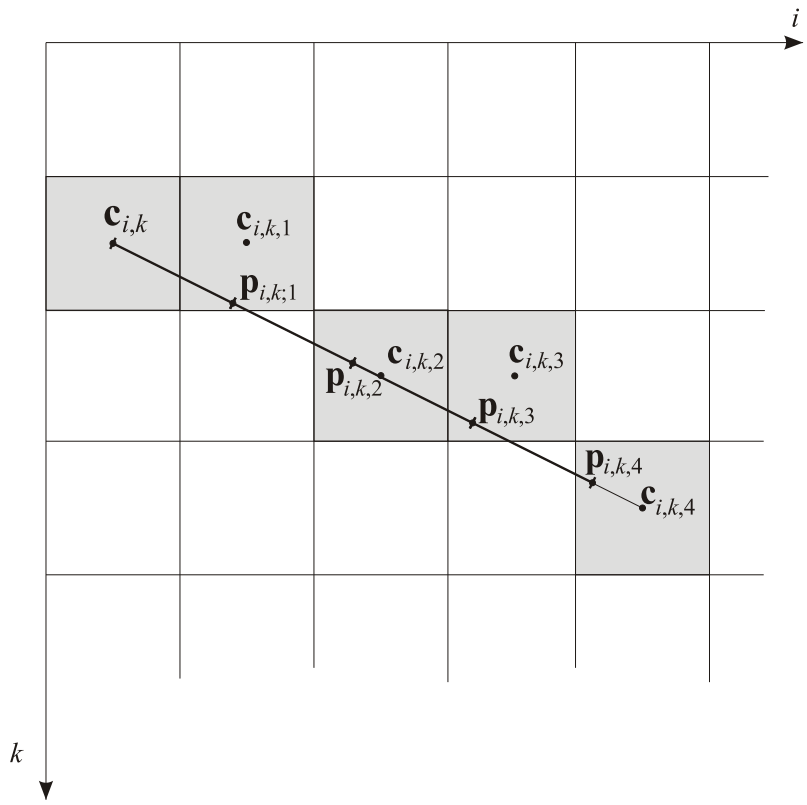
Da dokončno določimo vetrovnost $w_{i,k}$, upoštevamo le še spuščanje vetra, ki ga postopoma modeliramo na dva načina, kot to prikazuje slika 4.9.

Prvi način je z največjim radijem N_w, ki omejuje dolžino potovanja v nasprotni smeri vetra in je določen s številom krp (običajno 10 [van Lieshout, 2004]). Drugi način je s pomočjo praga $\Theta_0 \in [0, \frac{\pi}{2}]$ (npr. $\frac{\pi}{4}$) za kot $\varphi_{i,k}$, ko zavetrje popolnoma izgine, ker se veter usmerja navzdol. V preseku kot Θ_0 prikazuje slika 4.7. Dokler je $\varphi_{i,k}$ manjši od Θ_0, vetrovnost $w_{i,k}$ narašča linearno, kasneje pa je ta vrednost enaka 1, kot prikazuje enačba:

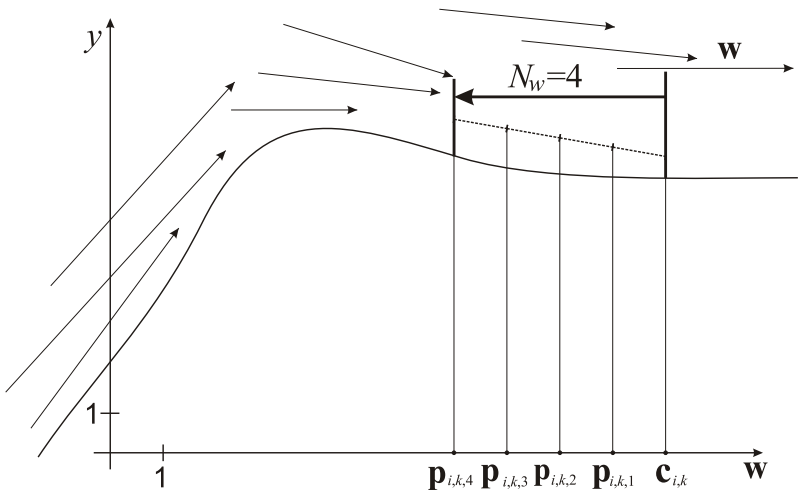
$$w_{i,k} = \begin{cases} \varphi_{i,k}, & \varphi_{i,k} \leq \Theta_0 \\ 1, & \text{sicer} \end{cases}. \tag{4.10}$$

Opisan postopek je prikazan kot algoritem 4.

Izračunana vetrovnost, ki jo izračuna program, je prikazana na sliki 4.10.



Slika 4.8: Sledenje proti vetru iz točke $c_{i,k}$ za $N_w = 4$ vzorcev. Smer potovanja je določena z $-\mathbf{w}$, dolžina koraka je enaka dimenziji krpe K. Iz vzorcev (črtice) dobimo pripadajoče krpe in njihove središčne točke.

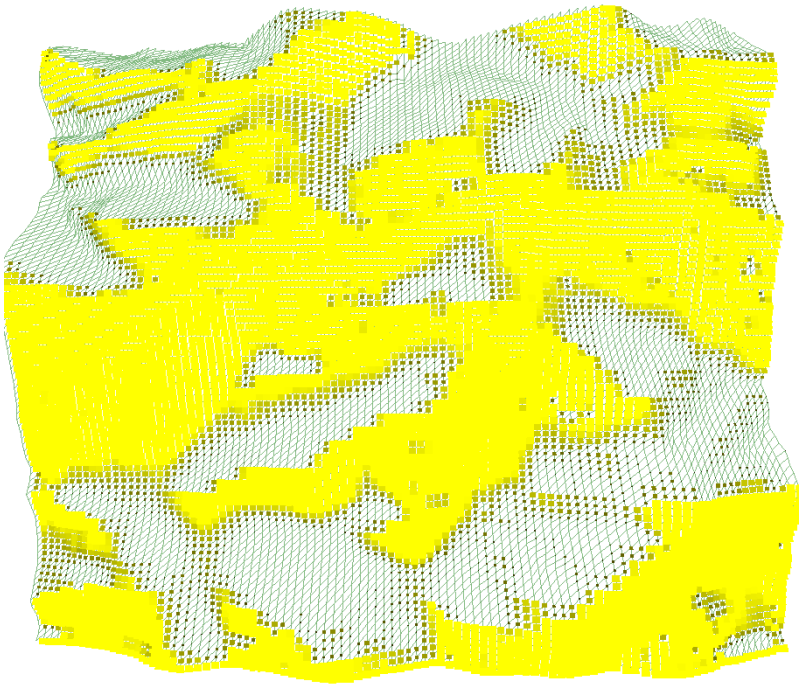


Slika 4.9: Spuščanje vetra. Veter se lahko po nekaj krpah toliko spusti, da že piha s polno močjo.

Algoritem 4 Izračun vetrovnosti za ves teren.

Vhod: i_{max} - število krp terena po osi x
Vhod: k_{max} - število krp terena po osi z
Vhod: N_w - število krp v smeri proti vetru
Izhod: matrika elementov $w_{i,k}$ - vetrovnost za vsako krpo

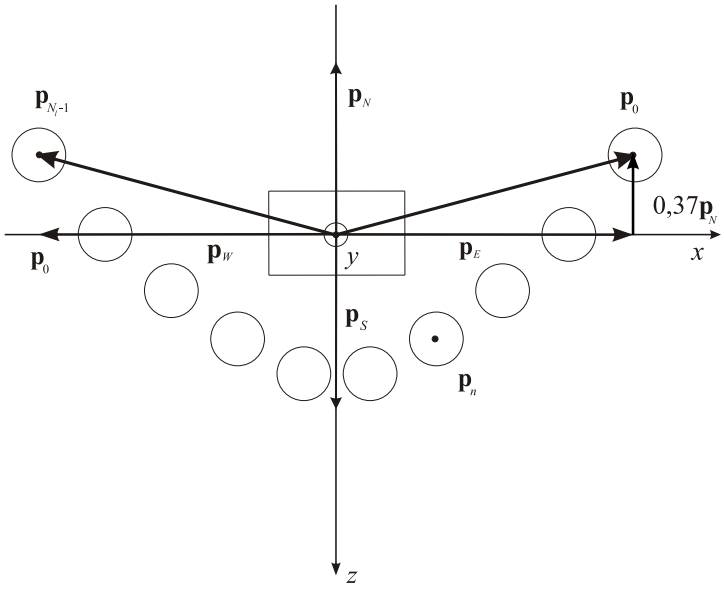
```
for i = 0 to i_max − 1 do
  for k = 0 to k_max − 1 do
    φ_{i,k} := 0;
    for w = 0 to N_w − 1 do
      φ_{i,k,w} = arctan (c_{y;i,k,w} − c_{y;i,k}) / ‖c'_{i,k,w} − c_{i,k}‖;
      φ_{i,k} := min(φ_{i,k,w}, φ_{i,k}); {manjši ko je kot zavetrja, bolj piha veter}
    end for
    if φ_{i,k} ≤ S_0 then
      w_{i,k} := φ_{i,k};
    else
      w_{i,k} := 1;
    end if
  end for
end for
```



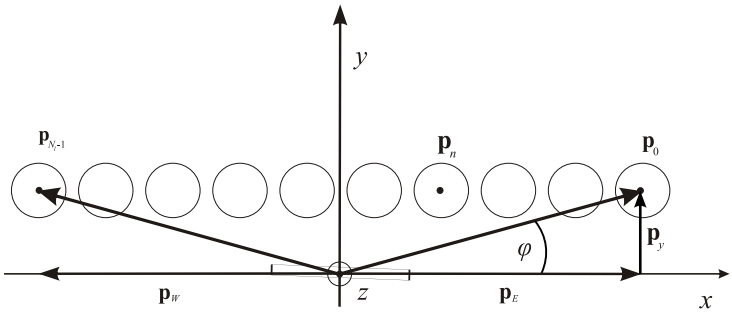
Slika 4.10: Vpliv vetra na terenu. Območja z več rumene barve so bolj vetrovna.

4.5 Osončenost

Na terenu določimo normirane vektorje v smeri strani neba $\mathbf{p}_N$, $\mathbf{p}_S$, $\mathbf{p}_E$, $\mathbf{p}_W$ in simuliramo potovanje sonca od vzhoda proti zahodu [Zakšek s sodelavci, 2004]. Ker je simulator prirejen za severno zemeljsko poloblo, sonce nekoliko nagnemo proti jugu, tako da so južni deli terena bolj osvetljeni od severnih. Skrajne položaje sonca $\mathbf{p}_0$ in $\mathbf{p}_{N_l-1}$ ter gibanje sonca $\mathbf{p}_n$ med njima vidimo na slikah 4.11 in 4.12.



Slika 4.11: Vzorci položajev sonca pri izračunu osvetljenosti za neko krpo $\mathbf{s}_{i,k}$.



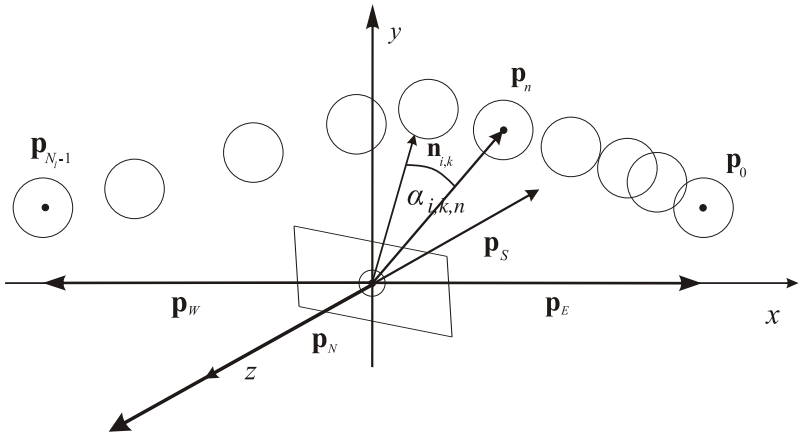
Slika 4.12: Vpadni kot sončne svetlobe, ki odreja osončenost krpe. S spreminjanjem točk vzorčenja se spreminja tudi ta kot.

Vektorja $\mathbf{p}_0$ in $\mathbf{p}_{N_l-1}$ določata naslednji enačbi:

$$\mathbf{p}_0 = \mathbf{p}_E + 0,37\mathbf{p}_N, \tag{4.11}$$

$$\mathbf{p}_{N_l-1} = \mathbf{p}_W + 0,37\mathbf{p}_N, \tag{4.12}$$

kjer vektor $0,37\mathbf{p}_N$ pomeni, da kot vzhoda in zahoda pomaknemo proti severu (za



Slika 4.13: Vzorci položajev sonca pri izračunu osvetljenosti za neko krpo $\mathbf{s}_{i,k}$, v prostoru.

izpeljavo konstant glej dodatek B).

Osončenost računamo v N_l položajih sonca, tako da upoštevamo še spremembo deklinacijskega kota srednjega vzorca $\frac{N_l-1}{2}$. Vektor sonca v najvišji legi je na 46. vzporedniku (kot deklinacije je 46°) zaradi vpadnega kota $\alpha_{i,k,\frac{N_l-1}{2}} = 90° - 46° = 44°$ enak $\mathbf{p}_{\frac{N_l-1}{2}} = (0,\ \sin 44°,\ \cos 44°)$, sonce pa je po sončni uri ob 12:00 natanko med vzhodom in zahodom. Opisan izračun podaja naslednja enačba:

$$\mathbf{p}_n = \begin{cases} \mathbf{p}_0 + \frac{2n}{N_l-1}(\mathbf{p}_{\frac{N_l-1}{2}} - \mathbf{p}_0), & n = 0, 1, ..., \frac{N_l-1}{2} \\ \mathbf{p}_0 + \frac{2(n-\frac{1}{2})}{N_l-1}(\mathbf{p}_{N_l-1} - \mathbf{p}_{\frac{N_l-1}{2}}), & n = \frac{N_l-1}{2} + 1, ..., N_l - 1 \end{cases}. \tag{4.13}$$

Dobljeni vektor $\mathbf{p}_n$ normaliziramo in izračunamo kosinus vpadnega kota, t. j. kota $\alpha_{i,k,n}$ med enotsko normalo $\overline{\mathbf{n}}_{i,k}$ terena in vektorjem proti soncu $\mathbf{p}_n$ (sliki 4.14 in 4.13):

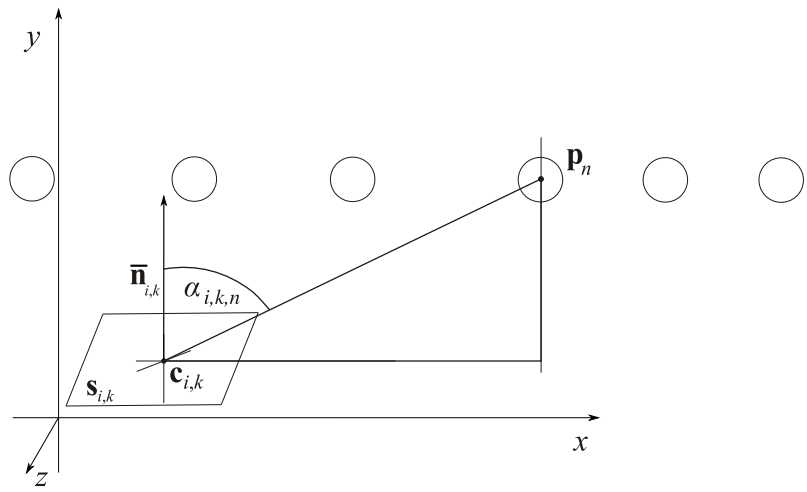
$$\cos \alpha_{i,k,n} = \overline{\mathbf{n}}_{i,k} \cdot \frac{\mathbf{p}_n}{\parallel \mathbf{p}_n \parallel}. \tag{4.14}$$

Osončenost krpe $l_{basic;i,k,n}$, če je sonce v položaju $\mathbf{p}_n$, določa enačba 4.15:

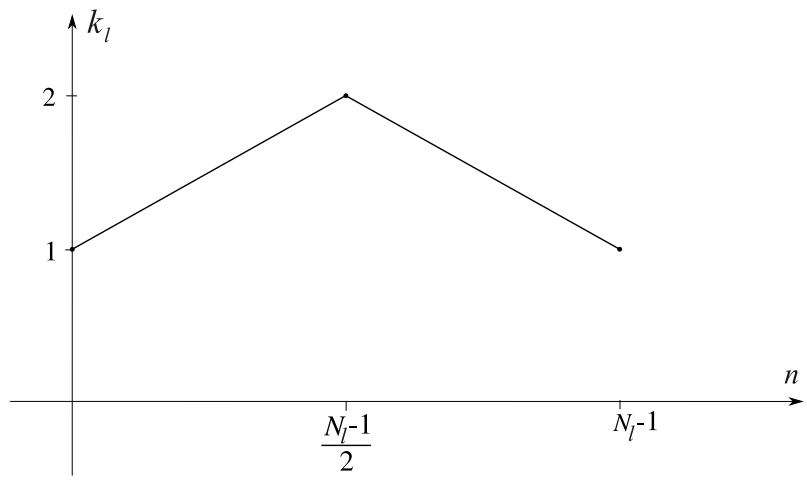
$$l_{basic;i,k,n} = \begin{cases} \cos \alpha_{i,k,n}, & 0 \leq \alpha_{i,k,n} \leq \frac{\pi}{2} \\ l_{ambient}, & \text{sicer} \end{cases}. \tag{4.15}$$

Vrednost osončenosti $l_{basic;i,k,n}$ smo utežili s faktorjem k_l, ki ga prikazuje slika 4.15, kjer predpostavljamo, da je N_l liho število. S tem damo večjo težo položajem, ki so sredi dneva, saj je ozračje bolj segreto in ima sonce večji vpliv. Nove vrednosti osončenosti so tako:

$$l_{i,k,n} = \begin{cases} l_{basic;i,k,n}\left(1 + \frac{2n}{N_l-1}\right), & 0 \leq n \leq \frac{N_l-1}{2} \\ l_{basic;i,k,n}\left(3 - \frac{2n}{N_l-1}\right), & \frac{N_l-1}{2} < n \leq N_l - 1 \end{cases}. \tag{4.16}$$



Slika 4.14: Vpadni kot sončne svetlobe, ki odreja osončenost krpe. S spreminjanjem točk vzorčenja se spreminja tudi ta kot.



Slika 4.15: Korekcijski faktor k_l osončenosti sredi dneva, saj je ozračje bolj segreto in ima sonce večji vpliv.

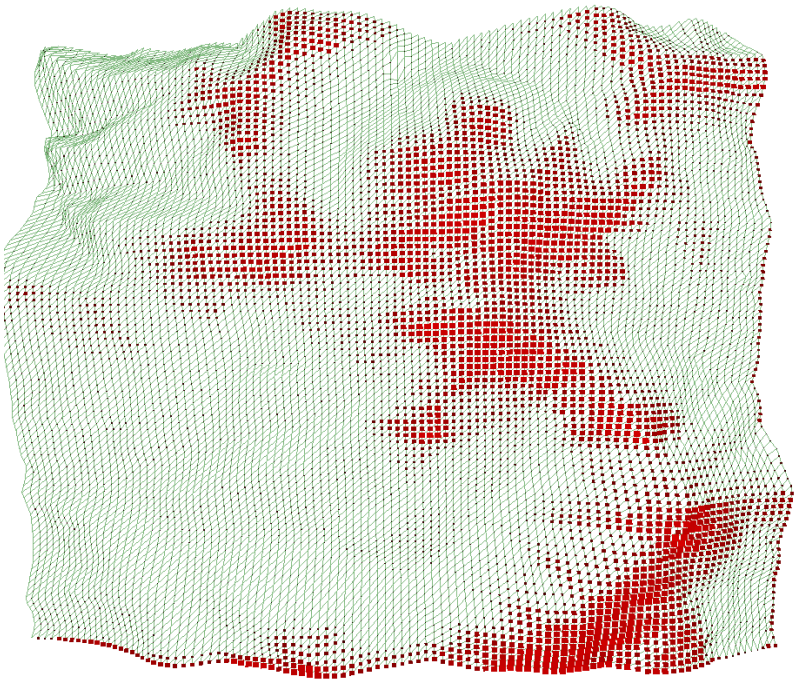
Pri izračunu osončenosti upoštevamo še senčnost krp, ki jih izračunamo podobno kot pri algoritmu za vetrovnost (algoritem 4). Če je krpa v senci, osončenost iz trenutnega vzorca zmanjšamo glede na nastavljivi koeficient $l_{ambient} \in (0, 1)$ za prisotnost razpršene svetlobe (npr. $l_{ambient} = 0,2$), sicer ga pustimo nespremenjenega.

Pri sledenju v smeri proti vzorcu položaja sonca $\mathbf{p}_n$ tako preverjamo, ali obstaja kakšna krpa, ki zastira sončevo svetlobo. To ugotovimo s primerjavo kotov od srednje točke trenutne krpe $\mathbf{c}_{i,k}$ do vzorca sonca in od iste točke do n-te srednje točke $\mathbf{c}_{i,k;n}$ krpe na sledi proti vzorcu sonca. Omenjen kot izračunamo enako kot pri algoritmu za veter, po enačbi 4.9. Če je izračunan kot večji od kota med trenutno krpo $\mathbf{c}_{i,k}$ in vzorcem položaja sonca, je trenutna krpa v senci in sledenje končamo. Končamo tudi takrat, če pridemo na rob terena in tedaj krpa ni v senci.

Osončenost krpe $\mathbf{s}_{i,k}$ predstavlja vsota osončenosti v vseh položajih sonca:

$$l_{i,k} = \frac{\sum_{n=0}^{N_l-1} l_{i,k,n}}{N_l}. \tag{4.17}$$

Algoritem za izračun osončenosti kaže algoritem 5. Slika 4.16 kaže vrednost osončenosti s pomočjo velikosti kvadratov na krpah.



Slika 4.16: Vpliv sonca izračunan na realnih podatkih terena. Območja z več rdeče barve so bolj osončena.

4.6 Način razširjanja dreves

Dodajanje rastlin rastlinskim vrstam lahko izvajamo po vrsti, tako da za vsako vrsto dodamo določeno število $\Delta N_{g,s}$ dreves na teren. Zaenkrat povejmo le, da je to število določeno za posamezno vrsto posebej (iz zbiranja semen na terenu), saj je natančneje definirano kasneje pri postopku rasti rastlin (razdelek 4.8). Drevesa nameščamo z upoštevanjem koncepta o rasti v bližini starševskih rastlin. Opisan postopek je predstavljen z algoritmom 6.

4.6.1 Kopičenje rastlin v bližini starševskih rastlin

Algoritem 6 se ne obnese dobro pri velikem številu rastlin (npr. nad 5.000 rastlin). Struktura za hranjenje rastlin v rastlinski vrsti je linearno povezan seznam $P_{s,p}$ in

Algoritem 5 Izračun osvetljenosti za ves teren.

Vhod: i_{max} - število krp terena po osi x
Vhod: k_{max} - število krp terena po osi z
Vhod: N_l - število vzorcev za položaje sonca
Vhod: $\mathbf{p}_N$, $\mathbf{p}_S$, $\mathbf{p}_E$, $\mathbf{p}_W$ - vektorji iz terena v smeri strani neba
Izhod: matrika elementov $l_{i,k}$ - osvetljenost za vsako krpo

$\mathbf{p}_0 := \mathbf{p}_E + 0,37\mathbf{p}_N$; {določimo skrajne lege sonca}
$\mathbf{p}_{N_l-1} := \mathbf{p}_W + 0,37\mathbf{p}_N$;
$\mathbf{p}_{\frac{N_l-1}{2}} = (0,\ \sin 44°,\ \cos 44°)$;
for $i = 0$ **to** $i_{max} - 1$ **do**
  for $k = 0$ **to** $k_{max} - 1$ **do**
    $l_{i,k} := 0$; {inicializacija celotne osvetljenosti za krpo}
    for $n = 0$ **to** $N_l - 1$ **do**
      if $n \leq N_l - 1$ **then**
        $\mathbf{p}_n = \mathbf{p}_0 + \frac{2n}{N_l-1}(\mathbf{p}_{\frac{N_l-1}{2}} - \mathbf{p}_0)$; {položaj sonca pri dopoldanskem vzorcu}
      else
        $\mathbf{p}_n = \mathbf{p}_0 + \frac{2(n-\frac{1}{2})}{N_l-1}(\mathbf{p}_{N_l-1} - \mathbf{p}_{\frac{N_l-1}{2}})$; {položaj sonca pri popoldanskem vzorcu}
      end if
      $\cos\alpha_{i,k,n} = \overline{\mathbf{n}}_{i,k} \cdot \frac{\mathbf{p}_n}{\| \mathbf{p}_n \|}$; {kot osvetljenosti vzorca}
      if $\cos\alpha_{i,k,n} \geq 0$ **then**
        $l_{basic;i,k,n} = \cos\alpha_{i,k,n}$; {osvetljenost direktno osončenega vzorca}
        if $0 < l < \frac{N_l-1}{2}$ **then**
          $l_{i,k,n} = l_{basic;i,k,n}\left(1 + \frac{2n}{N_l-1}\right)$; {jakost osvetljenosti vzorca čez dan}
        else
          $l_{i,k,n} = l_{basic;i,k,n}\left(3 - \frac{2n}{N_l-1}\right)$; {jakost osvetljenosti vzorca na robovih dneva}
        end if
        if ni v senci($\mathbf{p}_n$) **then**
          $l_{i,k} := l_{i,k} + \frac{l_{i,k,n}}{N_l}$; {akumulacija celotne osvetljenosti za krpo}
        else
          $l_{i,k} := l_{i,k} + \frac{l_{i,k,n} l_{ambient}}{N_l}$; {upoštevamo razpršeno svetlobo v senci}
        end if
      end if
    end for
  end for
end for

Algoritem 6 Osnovni algoritem za sajenje novih rastlin.

Vhod: N_s - število rastlinskih vrst
Vhod: $\Delta N_{g,s}$ - število novih rastlin v tej generaciji pri s-ti rastlinski vrsti ($s = 1, ...N_s$)
Vhod: p_s - verjetnost rasti nove rastline blizu svoje vrste za vsako vrsto posebej
Izhod: nove rastline $P_{s,n}$ so dodane rastlinskim vrstam in na teren

```
for s = 0 to N_s − 1 do
  for n = 0 to ΔN_{g,s} − 1 do
    if izpolni(p_s) then
      P_{parent;s,n} := poišči bližnjo rastlino(s); {izberemo obstoječo rastlino iz rastlinske vrste s}
      s_{s,n} := vrni naključno sosednjo krpo(P_{parent;s,n}); {naključje lahko sicer vrne tudi isto krpo}
      p_{s,n} := naključna točka na krpi(s_{s,n}); {dokončno določimo položaj nove rastline}
    else
      s_{s,n} := vrni naključno krpo();
      p_{s,n} := naključna točka na krpi(s_{s,n});
    end if
    if položaj p_{s,n} ni v vodi then
      P_{s,n} := tvori novo rastlino(p_{s,n}); {v podatkovno strukturo rastline shranimo položaj p_{s,n} in ostale pogoje terena na tem položaju}
    end if
  end for
end for
```

algoritem bi pri iskanju naključne bližnje rastline vsakokrat znova preiskal obstoječe rastline za vrsto. Zahtevnost algoritma je tako odvisna od števila rastlin in je $O(n^2)$. Zato raje uporabimo prilagojen algoritem 7, ki z uporabo **referenčnega seznama porazdelitve rastlin po krpah** preišče sezname bližnjih rastlin posameznih vrst rastlin le enkrat na simulacijski korak in zmanjša zahtevnost algoritma dodajanja rastlin na $O(n)$.

Iskanje bližnjih rastlin

Indekse omenjenega referenčnega seznama označimo z $I_{s,n}$ in jih tvorimo za vsako rastlinsko vrsto posebej. Seznam osvežujemo tako, da se za novo nastale rastline na simulacijskem koraku naključno odločimo, ali rastlina raste blizu druge rastline ali kjerkoli drugje na terenu. Če se odločimo za slednjo možnost, povečamo števec rastlin, ki jih dodamo kjerkoli (označimo ga z A_s) in nadaljujemo. Pri prvi možnosti pa naključno izberemo indeks rastline iste rastlinske vrste kot je trenutno dodajana rastlina, blizu katere naj zraste nova rastlina in ga shranimo v seznam elementov $I_{s,n}$. Po končani gradnji seznama elementov $I_{s,n}$ tega nepadajoče uredimo glede na hranjene indekse $I_{s,n}$. Ker je tedaj seznam urejen, se lahko skozi seznam rastlin v njeni vrsti sprehodimo linearno od 0 do $dolzinaseznama(I_{s,n})$ in zgolj primerjamo indekse iz $I_{s,n}$. Če je tekoči indeks enak indeksu $I_{s,n}$, je trenutna rastlina starš novi rastlini, ki jo kreiramo blizu. Novo rastlino tako posadimo v majhnem naključnem odmiku od obstoječe rastline. Opisana izboljšava je zapisana v algoritmu 7.

Primer izrazite rasti blizu svoje vrste in prevlado na tem območju zaradi ugodnih pogojev prikazuje slika 4.17.

Odmik izberemo kot poljubno sosednjo točko mreže terena glede na pozicijo stare rastline na terenu in na pripadajočem polju kjerkoli posadimo novo rastlino. Sosednje štiri točke za možne nove položaje rasti v bližini staršev so prikazane na sliki 4.18.

Določanje premika v sosednjo krpo

V sosednjo krpo se iz krpe $\mathbf{s}_{s,n}$ premaknemo tako, da naključno izberemo eno izmed štirih sosednjih krp, glede na indeks (i, k), ki ga $\mathbf{s}_{s,n}$ hrani. Tako naključno izberemo enega izmed indeksov $(i-1, k)$, $(i+1, k)$, $(i, k-1)$ in $(i, k+1)$, pri čemer pri robovih terena pazimo, da krpa s takšnim indeksom dejansko obstaja.

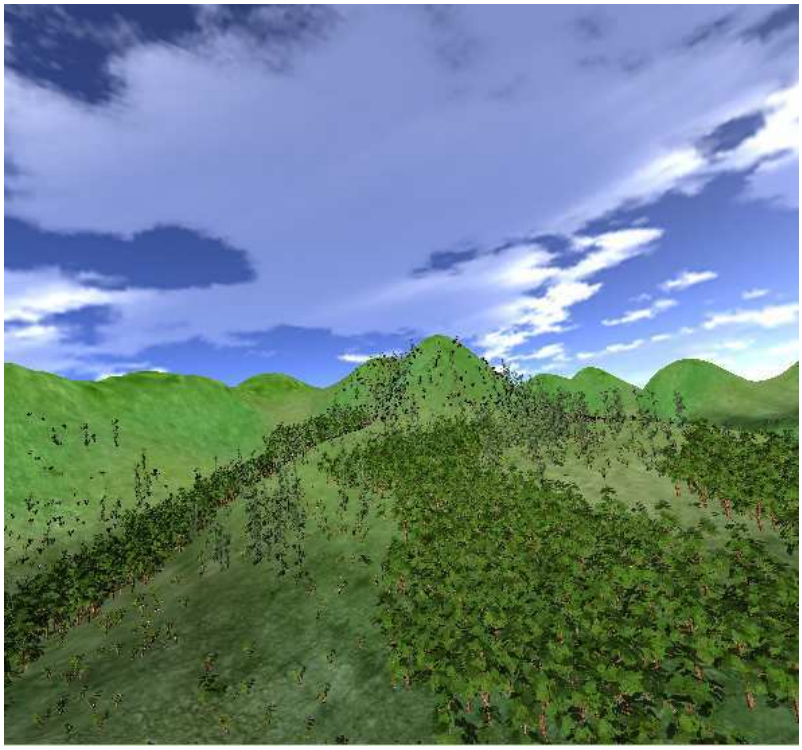
Ustvarjanje rastline na podani krpi

Za n-to krpo $\mathbf{s}_{s,n}$ iz njenega indeksa krpe (i, k) določimo lokacijo nove rastline z naključno izbiro točke na ploskvi krpe $\mathbf{s}_{i,k}$. To storimo tako, da določimo vektorja

Algoritem 7 Izboljšan algoritem za dodajanje novih rastlin.

Vhod: N_s - število rastlinskih vrst
Vhod: $\Delta N_{g,s}$ - število novih rastlin pri tej generaciji pri rastlinski vrsti s
Vhod: p_s - verjetnost rasti nove rastline blizu svoje vrste za vsako vrsto posebej
Izhod: nove rastline $P_{s,n}$ so dodane rastlinskim vrstam in na teren

```
for s = 0 to N_s − 1 do
  A_s := 0;
  for n = 0 to ΔN_{g,s} − 1 do
    if izpolni(p_s) and dolžina seznama(P_{s,p})-1≥ 0 then
      I_{s,n} := naključni indeks med vključno (0, dolžina seznama(P_{s,p})-1);
    else
      A_s := A_s + 1;
    end if
  end for
  uredi seznam I_{s,n} po vrednostih shranjenih indeksov;
  for n = 0 to dolžina seznama(I_{s,n}) −1 do
    P_{parent;s,n} := I_{s,n};
    s_{s,n} := vrni naključno sosednjo krpo(P_{parent;s,n});
    p_{s,n} := naključna točka na krpi (s_{s,n});
    if položaj ni v vodi then
      P_{s,n} := tvori novo rastlino(p_{s,n}); {seme pade na plodna tla}
    end if
  end for
  for n = 0 to A_s − 1 do
    s_{s,n} := vrni naključno krpo(); {dodamo še rastline, ki lahko rastejo daleč
    stran od staršev}
    p_{s,n} :=naključna točka na krpi (s_{s,n});
    if položaj ni v vodi then
      P_{s,n} := tvori novo rastlino (p_{s,n});
    end if
  end for
end for
```



Slika 4.17: Rastline rastejo blizu starševskih rastlin.

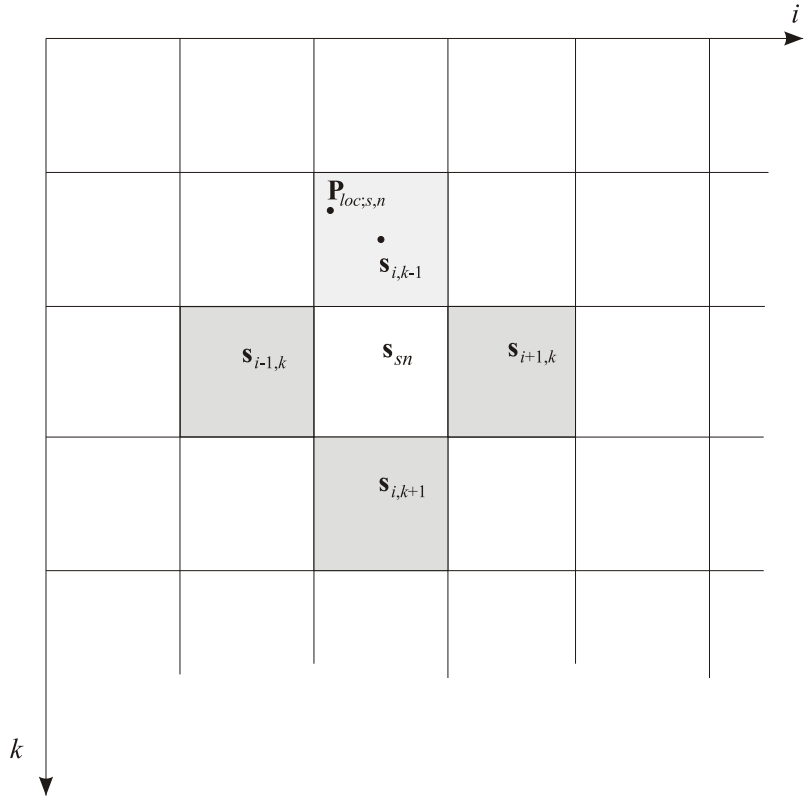
$\mathbf{k}_1$ in $\mathbf{k}_2$ od izhodiščne točke krpe do skrajnih točk krpe, to je začetnih točk naslednjih krp $(i+1, k)$ in $(i, k+1)$. Z naključnim skaliranjem vektorjev $\mathbf{k}_1$ in $\mathbf{k}_2$ ter seštevanjem dobljenih skaliranih vektorjev dobimo vektor, ki predstavlja iskano lokacijo nove rastline $\mathbf{P}_{loc;s,n}$. Opisan postopek je prikazan na sliki 4.19. Pri tem poskrbimo še, da se y-koordinata rastline pravilno prilega površini krpe, kar dosežemo s projiciranjem izbrane točke na to krpo.

Pri tvorbi novih rastlin poleg rastline shranimo njene faktorje, tako da so kasneje lažje dostopni. Inicializiramo starost, velikost, radij, živost in moč rastline na 0. Lokacijo rastline nastavimo iz dobljene lokacije na krpi, prav tako pa shranimo indeks krpe, ki ji rastlina pripada. Iz podatkov o terenu odčitamo strmino, višino, vlažnost, osončenost in vetrovnost ter jih shranimo. Za rastlino prav tako določimo največjo možno starost v letih, ki jo lahko doseže.

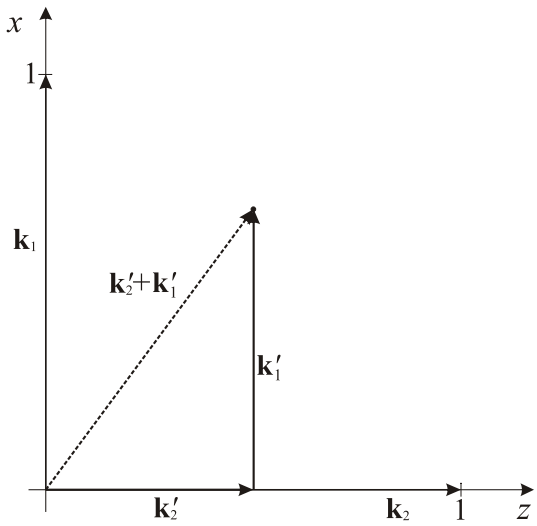
4.7 Tekmovalnost med rastlinami

Ob zgostitvi rastlin na nekem območju upoštevamo, da dominirane rastline lahko odmrejo, kar je posledica samo-redčenja. Ta pravi, da funkcija povprečne teže (debeline rastlin) v odvisnosti od gostote rastlin (števila primerkov) na logaritemski skali pada s strmino $-\frac{3}{2}$ (slika 4.20).

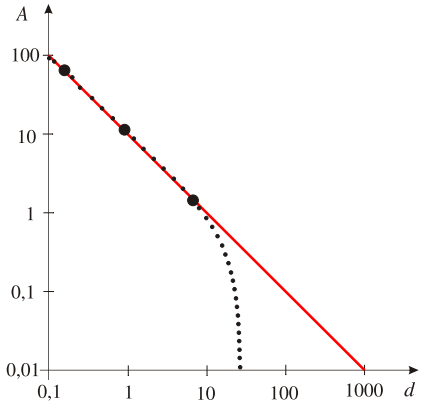
Da določimo, katera rastlina dominira kateri, vsaki rastlini priredimo moč $p_{s,p}$.



Slika 4.18: Primer izbire naključne rasti rastline blizu svojega starša. Krpi $\mathbf{s}_{s,n}$ smo izmed obarvanih možnih krp naključno izbrali sosednjo krpo z manjšim indeksom k. Na tej krpi smo izbrali naključno točko $\mathbf{p}_{s,n}$, ki predstavlja lokacijo nove rastline.



Slika 4.19: Določitev naključne točke na krpi.



Slika 4.20: Ricklefsova krivulja samo-redčenja (povprečna površina rastlin v odvisnosti od njihove gostote). Proti nasičenju začnejo gostote manj naraščati, kar pomeni, da nekatere rastline odmirajo.

Moč določimo iz produkta živosti rastline $v_{s,p}$, trenutne velikosti rastline $h_{s,p}$ in **največje možne relativne doživete moči** $p_{r;p}$, kot prikazuje enačba 4.18.

$$p_{s,p} = v_{s,p} h_{s,p} p_{r;p}. \tag{4.18}$$

Največja možna relativna doživeta moč $p_{r;p}$ (*relative power*) predstavlja razmerje med časom največje starosti rastline $t_{f;s}$ za neko rastlinsko vrsto napram času $\max\{t_{f;s}\}$, kar je največja potencialno možna življenjska doba posamezne rastline (npr. $\max\{t_{f;s}\} = 300$ let):

$$p_{r;p} = \frac{t_{f;s}}{\max\limits_{s}\{t_{f;s}\}}. \tag{4.19}$$

Živost rastline $v_{s,p} \in [0,1]$ določimo s seštevanjem ugodnosti pogojev terena na krpi $\mathbf{s}_{i,k}$. Živost je aritmetična sredina vseh pogojev in je prikazana z naslednjo enačbo:

$$v_{s,p} = \frac{k_{\overline{y};s,p} + m_{s,p} + l_{s,p} + w_{s,p} + s_{s,p}}{5}. \tag{4.20}$$

Omenjeni pogoji so nadmorska višina $k_{\overline{y};s,p}$, vlaga $m_{s,p}$, osončenost $l_{s,p}$, vetrovnost $w_{s,p}$ in strmina $s_{s,p}$. Vpliv slednjega izračunamo z upoštevanjem škodljivosti naklona $f_{s;s}$, ki jo podamo za vsako rastlinsko vrsto posebej:

$$s_{s,p} = 1 - (f_{s;s} s_{i,k})^2. \tag{4.21}$$

Indeksi (i,k) označujejo lastnost terena na mestu obravnavane rastline.

Ostale štiri pogoje izračunamo glede na odmike od **optimalnih vrednosti la-**

stnosti terena $\overline{y}_{o;s}$, $m_{o;s}$, $l_{o;s}$ in $w_{o;s}$ po naslednjih enačbah:

$$k_{\overline{y};s,p} = \begin{cases} 1, & |\overline{y}_{i,k} - \overline{y}_{o;s}| \leq \Delta\overline{y}_{o;s} \\ 1 - \max\left\{1, |\overline{y}_{i,k} - \overline{y}_{o;s}| f_{\overline{y};s}\right\}, & \text{sicer} \end{cases}, \tag{4.22}$$

$$m_{s,p} = \begin{cases} 1, & |m_{i,k} - m_{o;s}| \leq \Delta m_{o;s} \\ 1 - \max\left\{1, |m_{i,k} - m_{o;s}| f_{m;s}\right\}, & \text{sicer} \end{cases}, \tag{4.23}$$

$$l_{s,p} = \begin{cases} 1, & |l_{i,k} - l_{o;s}| \leq \Delta l_{o;s} \\ 1 - \max\left\{1, |l_{i,k} - l_{o;s}| f_{l;s}\right\}, & \text{sicer} \end{cases}, \tag{4.24}$$

$$w_{s,p} = \begin{cases} 1, & |w_{i,k} - w_{o;s} \leq \Delta w_{o;s} \\ 1 - \max\left\{1, |w_{i,k} - w_{o;s}| f_{w;s}\right\}, & \text{sicer} \end{cases}. \tag{4.25}$$

Dopustne odmike določimo s parametri $\Delta\overline{y}_{o;s}$, $\Delta m_{o;s}$, $\Delta l_{o;s}$ in $\Delta w_{o;s}$. V kolikor je lastnost izven želenih meja, pogoj iz ugodne vrednosti zmanjšamo za produkt tega odmika in **faktorje slabega vpliva** $f_{\overline{y};s}$, $f_{m;s}$, $f_{l;s}$ in $f_{w;s}$.

4.7.1 Odmiranje rastlin

Rastlina lahko odmre, če:

- doseže običajno starost odmrtja (življenjska doba) ali
- je predolgo dominirana od drugih rastlin.

Postaranje

Čas naravnega odmrtja rastline $t_{max;s,p}$ naključno določimo pri prvem dodajanju rastline in je med časom starosti $t_{m;s}$ in časom največje starosti $t_{f;s}$:

$$t_{max;s,p} = t_{m;s} + (t_{f;s} - t_{m;s}) f_{random}, \qquad f_{random} \in [0, 1]. \tag{4.26}$$

Dominiranost

Če rastlina A z močjo p_A raste poleg močnejše rastline B, ki ima večjo moč p_B, postane dominirana. V tem primeru raste počasneje ali sploh ne in lahko odmre z verjetnostjo $p_{m;s}$ (nastavljiv parameter simulacije), ki je določena za vsako rastlinsko vrsto posebej in modelira odpornost na rast v sobivanju.

Kot pomoč za nadaljnjo razlago določimo število krp N_{opt}. To določa, za koliko krp sta največ lahko oddaljeni poljubni rastlini, da se njuna ekološka kroga še lahko prekrivata. Ugotovimo lahko, da je to natanko toliko, čez koliko krp se lahko razteza premer ekološke sosednosti največje rastline. Ta pa je enak največjemu možnemu

premeru R_s ekološkega kroga za rastlinsko vrsto. Če s K označimo dimenzije krpe, podaja izračun N_{opt} naslednja enačba:

$$N_{opt} = \left\lceil \frac{2 \max_s \{R_s\}}{K} \right\rceil . \tag{4.27}$$

Potencialno močnejše rastline glede na trenutno rastlino na krpi (i, k) poiščemo tako, da preverimo moč rastlin v okolici te krpe. Če bi preiskovali celoten teren, bi simulacija potekala prepočasi, zato pregledamo le potencialno število sosednjih krp N_{opt}. Da lahko obiščemo le N_{opt} krp, si ponovno pomagamo z referenčnim seznamom lokacij rastlin po krpah, ki smo ga predstavili že pri dodajanju rastlin v bližini starševskih rastlin. V seznamu imamo za vsako krpo shranjene rastline, ki na njej rastejo. Vse rastline na neki krpi primerjamo z obravnavano rastlino in preverimo, ali se ekološki krogi prekrivajo ter ali ima obravnavana rastlina manjšo moč. Če sta pogoja izpolnjena, prenehamo z iskanjem in obravnavano rastlino določimo kot dominirano.

Geometrijsko razdaljo $d(A, B)$ med rastlinama A in B na lokacijah $\mathbf{p}_A$ in $\mathbf{p}_B$ določimo z:

$$d(A, B) = \|\mathbf{p}_A - \mathbf{p}_B\|. \tag{4.28}$$

Prekrivanje ekoloških krogov dveh rastlin $F_{A,B}$ (glej sliko 4.21) določimo iz geometrijske razdalje med rastlinama. Če je razdalja med rastlinama manjša od vsote radijev r_A in r_B, se ekološka kroga sekata. Prekrivanje izračunamo po enačbi 4.29, dokler ne preverimo vseh bližnjih rastlin ali pa katera bližnja rastlina dominira rastlino A.

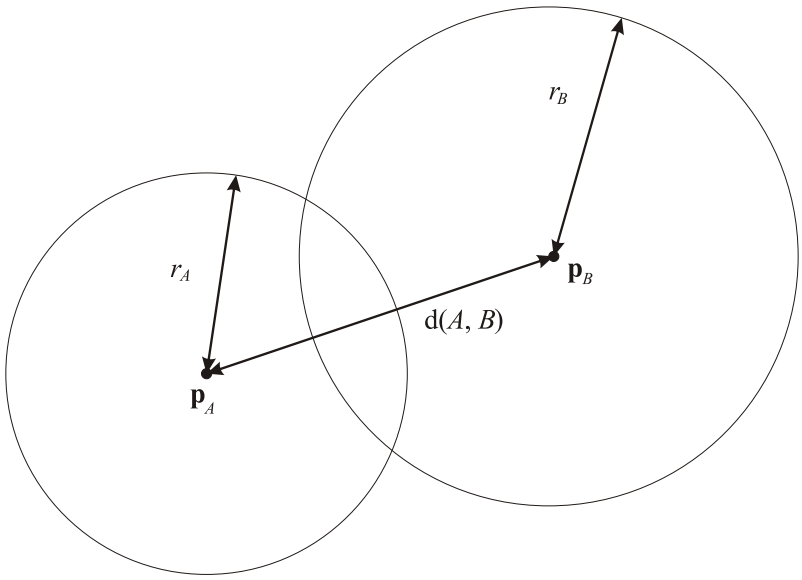
$$F_{A,B} = \begin{cases} true, & d(A, B) < r_A + r_B \\ false, & \text{sicer} \end{cases} . \tag{4.29}$$

Rastline za dominiranost iščemo tako na polju krp vključno med krpama $(i - N_{opt}, k - N_{opt})$ in $(i + N_{opt}, k + N_{opt})$ (slika 4.22). Postopek je opisan z algoritmom 8.

4.8 Reprodukcija rastlin

Rastline vrste s se po terenu reproducirajo glede na število svojih semen na terenu s_s in faktor razploda te vrste g_s (nastavljiv parameter simulacije, ponavadi celo število, npr. od 1 do 5). Število novih rastlin ΔN_s vrste s, ki vzkalijo v trenutnem koraku simulacije, izračunamo po naslednji enačbi:

$$\Delta N_s = s_s g_s, \qquad g_s \in [0, 100]. \tag{4.30}$$

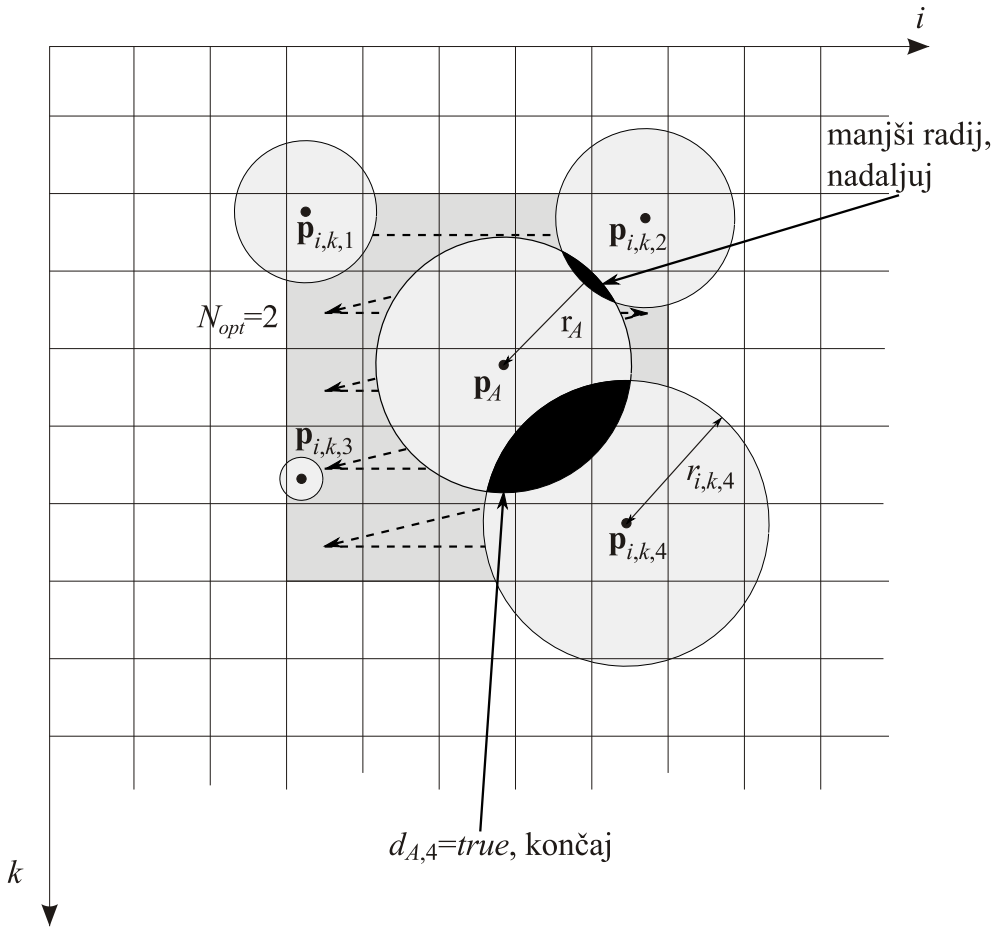


Slika 4.21: Dominiranost izračunamo iz prekrivanja ekoloških krogov.

Algoritem 8 Iskanje dominiranosti.

Vhod: $\mathbf{p}_A$ - položaj rastline A
Vhod: p_A - moč rastline A
Vhod: r_A - polmer rastline A
Vhod: i, k - indeksa krpe obravnavane rastline A
Vhod: N_{opt} - velikost okolice za iskanje po sosednjih krpah
Vhod: seznam $\mathbf{p}_{i,k,n}$ - lokacije rastlin n, ki so blizu krpe $\mathbf{s}_{i,k}$
Vhod: seznam $p_{i,k,n}$ - moči rastlin n, ki so blizu krpe $\mathbf{s}_{i,k}$
Vhod: seznam $r_{i,k,n}$ - radiji rastlin n, ki so blizu krpe $\mathbf{s}_{i,k}$
Izhod: za obravnavano rastlino določimo zastavico $D_{A,B}$ - ali je dominirana od katerekoli bližnje rastline

```
D_{A,B} := false;
for z = k − N_opt to k + N_opt do
  for x = i − N_opt to i + N_opt do
    for n = 0 to dolžina seznama(p_{i,k})-1 do
      if |p_A − p_{i,k,n}| < r_A + r_{i,k,n} and p_A < p_{i,k,n} then
        D_{A,B} := true;
        return;
      end if
    end for
  end for
end for
```

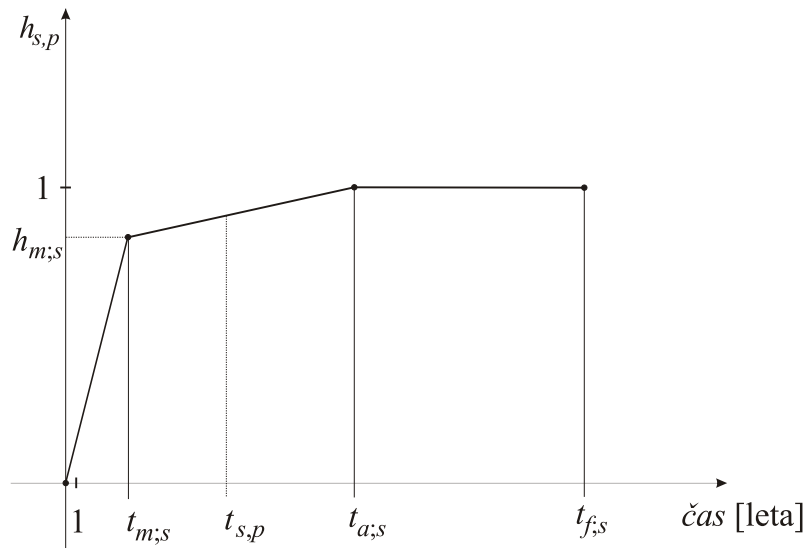


Slika 4.22: Iskanje dominiranosti rastlin v omejenem območju okoli rastline A, $N_{opt} = 2$. V primeru, da se ekološka kroga rastlin prekrijeta in je sosednja rastlina močnejša, končamo s preiskovanjem in vrnemo *da*, sicer *ne*. Primerjati moramo moč omenjenih rastlinih, ne zgolj velikosti radijev, saj na radije poleg velikosti rastlin $h_{s,p}$ vpliva R_s (enačba 4.33), na moč rastline pa poleg velikosti rastlin $h_{s,p}$ vpliva še živost rastline $v_{s,p}$ in največja možna doživeta moč rastline $p_{r;p}$ (enačba 4.18). Zaporedje pregledovanja krp označuje črtkana črta. Slika prikazuje primer s štirimi rastlinami, ki jih lahko označimo po vrsti z indeksi 1, 2, 3 in 4. Položaj rastline $\mathbf{p}_A$ po vrsti primerjamo s položaji $\mathbf{p}_{i,k,1}$, $\mathbf{p}_{i,k,2}$, $\mathbf{p}_{i,k,3}$ in $\mathbf{p}_{i,k,4}$. Ker se prvi krog ekoloških sosednosti rastlin ne seka s krogom ekološke sosednosti osnovne rastline, nadaljujemo. Čeprav se v drugem primeru zgodi presek, pa je tokrat primerjana rastlina prešibka, zato ponovno nadaljujemo. V primeru tretje sosednje rastline prav tako nadaljujemo iskanje. Ustavimo se pri četrti rastlini, ki dominira osnovno rastlino, saj za njuni moči velja $p_A < p_{i,k,4}$ in njuna ekološka kroga se prekrivata ($|\mathbf{p}_A - \mathbf{p}_{i,k,4}| < r_A + r_{i,k,4}$).

Zbiranje semen določimo z vsoto moči rastlin, ki so že zrele in tvorijo semena. Tako večja drevesa dajejo več semen kot manjša. V prvem koraku postavimo $\Delta N_s = 1$. Iz enačbe 4.30 sledi, da so kumulacije semen za vse rastlinske vrste enake $\frac{1}{g_s}$ in zato nastane po ena rastlina na rastlinsko vrsto, kasneje pa semena ustrezno dodajamo. Povezavo med močmi rastlin $p_{s,p}$ (enačba 4.18, stran 59) in kumulacijo semen vseh $N_{g,s}$ rastlin iste vrste podaja enačba 4.31. Ker je prava moč rastline enaka $p_{s,p}$, jo v izračunu označimo s $p'_{s,p}$, kar pomeni, da postavimo moč na 0, dokler rastlina še ni zrela.

$$s_s = \sum_{p=0}^{N_{g,s}} p'_{s,p}, \qquad p'_{s,p} = \begin{cases} p_{s,p}, & t_{s,p} \geq t_{m;s} \\ 0, & \text{sicer} \end{cases} . \tag{4.31}$$

Simulacijski koraki predstavljajo leta. Starost p-te rastline iz rastlinske vrste s označimo s $t_{s,p}$. Mlade rastline rastejo hitreje, v zrelosti rastejo počasneje, v starosti pa več ne rastejo. Če so rastline dominirane (opisano kasneje), rastejo zelo počasi ali sploh ne (v modelu smo upoštevali le slednje). Hitrost rasti rastline spreminjamo v odvisnosti od prej omenjenih treh obdobij rastline in koeficienta ugodnosti pogojev v_p (opisan kasneje). V mladosti rastlina zgolj hitro raste. Zrelost pomeni, da rastlina prvič odvrže svoja semena in iz teh lahko nastanejo nove rastline. Starost poimenujemo obdobje, ko rastlina več ne raste. Starost traja največ do leta $t_{f;s}$, ko vsaka rastlina te vrste zagotovo odmre. Meji med opisanimi obdobji predstavljata leto zrelosti (*matured*) $t_{m;s}$ in leto starosti (*aged*) $t_{a;s}$ posamezne rastlinske vrste. Odvisnost med značilnimi starostmi rastline in hitrostjo rasti je prikazana na sliki 4.23. Za vsako vrsto prav tako podamo višino $h_{m;s}$, ki jo rastlina doseže v letu



Slika 4.23: Hitrost rasti skozi staranje. Do zrelosti rastlina raste hitreje, v času zrelosti raste počasneje, v starosti pa rastlina več ne raste. S $h_{m;s}$ podamo velikost rastline ob času zrelosti.

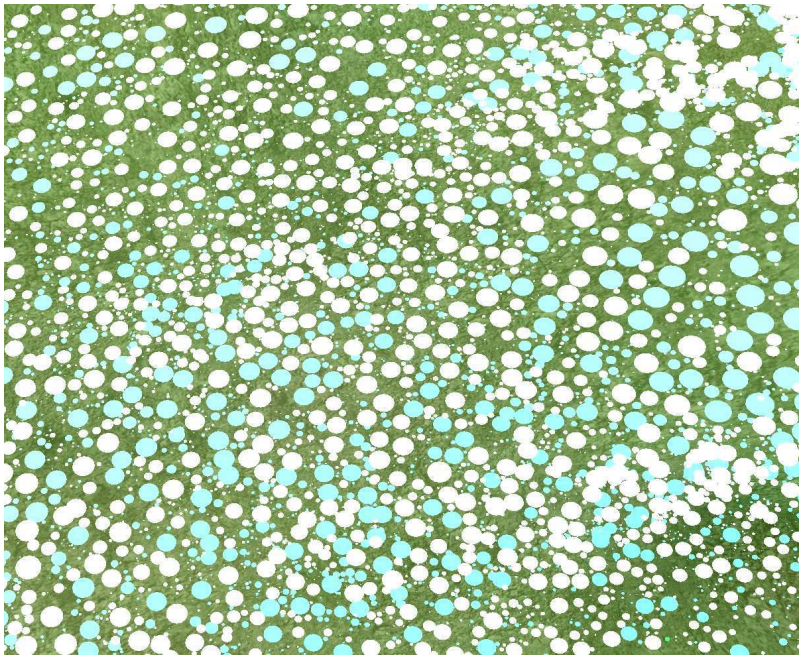
zrelosti. S podanimi parametri lahko izračunamo prirast k velikosti rastline $\Delta h_{s,p}$ v enem simulacijskem ciklu po enačbi:

$$\Delta h_{s,p}[\text{m/leto}] = \begin{cases} v_p \frac{h_{m;s}}{t_{m;s}}, & 0 \leq t_{s,p} \leq t_{m;s} \\ v_p \frac{1-h_{m;s}}{t_{a;s}-t_{m;s}}, & t_{m;s} \leq t_{s,p} \leq t_{a;s} \\ 0, & \text{sicer} \end{cases}. \tag{4.32}$$

Velikost ekološkega radija $r_{s,p}$ rastline izračunamo iz višine rastline $h_{s,p} \in [0, 1]$ (normaliziranost izhaja iz enačbe 4.22, stran 60) s pomočjo parametra R_s za razmerje med starostjo v letih in velikostjo radija v metrih (uporabniško podan za vsako rastlinsko vrsto posebej):

$$r_{s,p} = h_{s,p} R_s, \qquad R_s \in (0, 10). \tag{4.33}$$

Slika 4.24 prikazuje ekološke sosednosti rastlin, ki so odvisne od velikosti rastlin.



Slika 4.24: Krogi predstavljajo radije rastlin: večje rastline imajo večje radije. Kot vidimo, se rastline dokaj malo prekrivajo. Na začetku simulacije je bilo veliko manjših rastlin, nato pa se je število rastlin močno zmanjšalo, saj so začele prevladovati večje rastline.

4.9 Potek simulacije

Simulacijo začnemo z neporaslim terenom, na katerega vsako leto dodamo določeno število rastlin posamezne vrste. Pri odvijanju simulacije opazimo želeno redčenje, ki

je eden od osnovnih naravnih principov ekologije rastlin. Na danem območju se tako najprej razvije več manjših rastlin, kasneje pa prevladajo večje, ki jih je ustrezno manj.

Opisani pogoji pokrajine vplivajo na rastline tako, da rastline, ki ne tolerirajo določenega življenjskega pogoja na svojem območju rasti, počasneje rastejo in kasneje skozi simulacijske korake odmrejo s podano verjetnostjo na simulacijski korak. Med rastjo rastline pridobivajo moč.

V vsakem koraku simulacije izvedemo dodajanje, rast in odstranjevanje rastlin, kot je predstavljeno z algoritmom 9. Naštete rutine se obnašajo stohastično; pri tem uporabljajo naključni generator števil *SUPER-DUPER* z obliko $LCG(2^{32}, 69069, 0, 1)$ [Marsaglia, 1972]. Uveljavitev verjetnostne trditve smo izvedli, če se je normalizirano generirano naključno število uvrstilo pod določen prag a:

$$izpolni(a) = \begin{cases} true, & F(RNi) < a \\ false, & \text{sicer} \end{cases} . \tag{4.34}$$

Algoritem 9 Nadzorna rutina simulacije.

Vhod: v - seznam rastlinskih vrst
Vhod: r - seznam rastlin za rastlinske vrste
Vhod: f - seznam matrik za vrednosti faktorjev na površju
Izhod: izvaja koračno simulacijo razvoja ekosistema

```
loop
    dodaj nove rastline rastlinskim vrstam(v, r);
    zrasti vse rastline (r, f);
    odstrani odmrle rastline(r);
end loop
```

Poglavje 5

Vizualizator pokrajine

Pri vizualizatorju uporabljamo rezultate tekočega koraka simulacije in animiramo rast dreves. Za prikaz uporabimo tehnologijo OpenGL. Pred upodabljanjem dodamo:

- obrezovanje scenskih objektov na vidni volumen in
- poenostavljanje drevesne geometrije.

Tako v sceni prikažemo le trenutno vidne rastline [Picco, 2003; Gribb in Hartmann, 2001], ki se poenostavijo glede na oddaljenost od gledišča. Poenostavitev dreves izvedemo tako, da izrišemo manj vej in listov za poenostavljeno drevo. Zaenkrat poenostavljamo samo drevesa, saj izrisovanje terena ni časovno potratno napram času izrisovanja dreves.

5.1 Upodobitev površja pokrajine

Površje pokrajine upodobimo s trikotniki, ki jih dobimo po triangulaciji množice točk terena. Ker so te točke v ravnini xz razporejene ekvidistančno, jih zapišemo v dvodimenzionalno polje točk, ki ga lahko naslavljamo s celoštevilskimi indeksi. Z naslavljanjem točk je možno hitro sestaviti štirikotnike, ki pokrijejo celoten teren, štirikotnike pa delimo na polovico in dobimo potrebne trikotnike.

5.2 Prevedena polja oglišč in multiteksturiranje ploskev v OpenGL

Upodobitev terena smo opravili po specifikaciji OpenGL verzije 1.2 [Wikipedia, 2006a; Woo s sodelavci, 1999], ki omogoča posredovanje **prevedenih polj oglišč**

Slika 5.1: Tekstura terena za čez celi teren. Ker je slika manjše ločljivosti, preko nje nalepimo teksturo za dodajanje podrobnosti.

(*compiled vertex arrays*). To pomeni, da na odjemalcu sestavimo vse potrebne točke in povezave med njimi za gradnjo trikotnikov v obliki polja. Polje pošljemo na strežnik OpenGL in ga zaklenemo, tako da ga strežnik shrani v pomnilnik grafične kartice. Uporaba te tehnike pohitri izris mreže terena za več kot 25-krat napram implicitnemu podajanju vsake točke posebej v vsakem novem okvirju animacije [Dobry, 2000].

Za uporabo ukazov iz specifikacije OpenGL 1.2 moramo v programsko kodo vključiti ustrezne glave za delo z razširitvami OpenGL in prevedeno objektno kodo povezati s knjižnico za OpenGL. Razširitev, ki nam omogoča delo s prevedenimi polji oglišč, se imenuje `GL_EXT_compiled_vertex_array`. Ta nam omogoča, da s funkcijo grafične knjižnice **DrawElements** izrišemo seznam gradnikov, kot so na primer trikotniški trakovi.

Teksturiranje objektov je opravljeno z nalaganjem rastrskih slik, ki jih posredujemo stroju OpenGL. Ta bitne podatke shrani in vrne ime teksture, s katero nadalje operiramo. Z razširitvijo OpenGL za več tekstur na eni površini dosežemo čistejše slike. Razširitev se imenuje `GL_ARB_multitexture` in omogoča izbiro aktivne med več teksturami s funkcijo **glClientActiveTextureARB**. Tej podamo parameter `GL_TEXTURE<texid>_ARB`, pri čemer `texid` predstavlja indeks izbirane teksture: 0, 1 ali več. Teksture se med seboj kombinirajo. Primer teksture čez celi teren prikazuje slika 5.1, prelepljena tekstura za podrobnosti na terenu pa je vidna na sliki 5.2.

Slika 5.2: Slika za dodajanje podrobnosti (visokih frekvenc), da teksturo naredimo bolj informacijsko polno in s tem bolj grobo.

5.3 Navigacija

Aplikacija omogoča dva načina pogleda. Prvi prikazuje eno samo drevo v polni podrobnosti izrisa in je posebej primeren za modeliranje drevesa. Med modeli dreves lahko med modeliranjem preklapljamo in tako nastavimo parametre proceduralnih modelov za izračun geometrije vsem vrstam rastlin v ekosistemu. V drugem načinu pogleda na ekosistem je prikazanih veliko dreves na terenu, s prilagojeno podrobnostjo izrisa. Namenjen je opazovanju poteka simulacije.

V načinu pogleda na eno samo drevo gledišče ostaja fiksno, premikamo pa drevo. Viden del drevesa določimo s sukanjem drevesa okoli glavne osi debla in okoli osi, pravokotne na glavno os drevesa. Drevo lahko pomikamo v treh osnih smereh.

V načinu pogleda na ekosistem premikamo gledišče. Parametre proceduralnih modelov dreves lahko sicer nastavljamo tudi v tem načinu pogleda, vendar ga pogosteje uporabimo za oddaljen pogled na celoten ekosistem. Pogled nastavimo s šestimi prostostnimi stopnjami, s katerimi lahko nastavimo poljuben položaj gledišča in smer gledanja. Postavljanje pogleda v načinu ekosistema poteka s klicem funkcije **gluLookAt** iz pomožne grafične knjižnice, ki mu podamo točko gledanja, referenčno točko in vektor v smeri navzgor.

Poglavje 6

Rezultati simulacije drevesnih ekosistemov

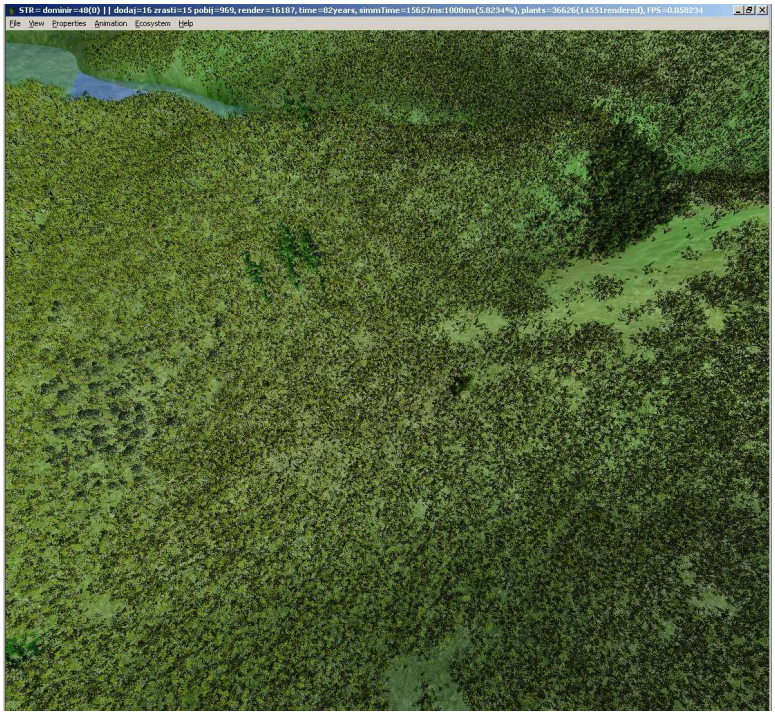
S simulacijo tvorimo porazdelitev rastlin, ki jih lahko upodobimo kot žične modele ali s polnimi teksturami. Namesto rastlin lahko upodobimo tudi samo kroge sosednosti, kar je primerno za shematski prikaz. V nadaljevanju so predstavljene upodobljene slike, ki so dobljene z enim od tekov simulacije.

V simulacijo so vključene štiri vrste dreves: bukve, javorji, smreke in grmovnice. Simulacijo pričnemo s praznim terenom brez dreves, na katerega se ta naselijo, kot bi človek pokrajino pustil neobdelano (slika 6.1). Ker se grmovnice po terenu najhitreje razrastejo, je teh po prvih nekaj letih največ (slika 6.2). Te so dokaj šibke v boju za prostor proti večjim drevesnim vrstam, zato se začno umikati in teren začno poseljevati večji listavci (slika 6.3). Kasneje se iz desne strani vse bolj začno širiti tudi smreke (slika 6.4).

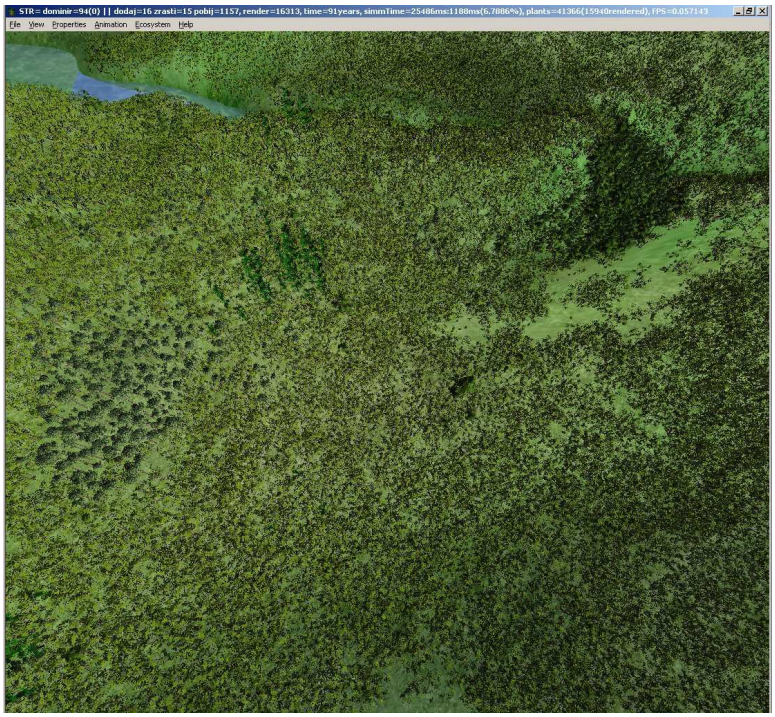
Po približno 180 letih simulacije na terenu rastejo vsa uporabljena drevesa v simulaciji. Vsaka vrsta od njih si je za življenjski prostor izborila tisti del, ki je zanjo najugodnejši in kjer lahko izpodrine ostale vrste. Na sliki 6.5 lahko vidimo tudi želeno rast dreves v gručah, še vedno pa je med drevesi ene vrste nekaj dreves iz kakšne druge drevesne vrste, kar sliko naredi bolj verodostojno.

Slika 6.1: Začetek simulacije: človek preneha s košnjo trave, pričnejo se pojavljati grmovnice.

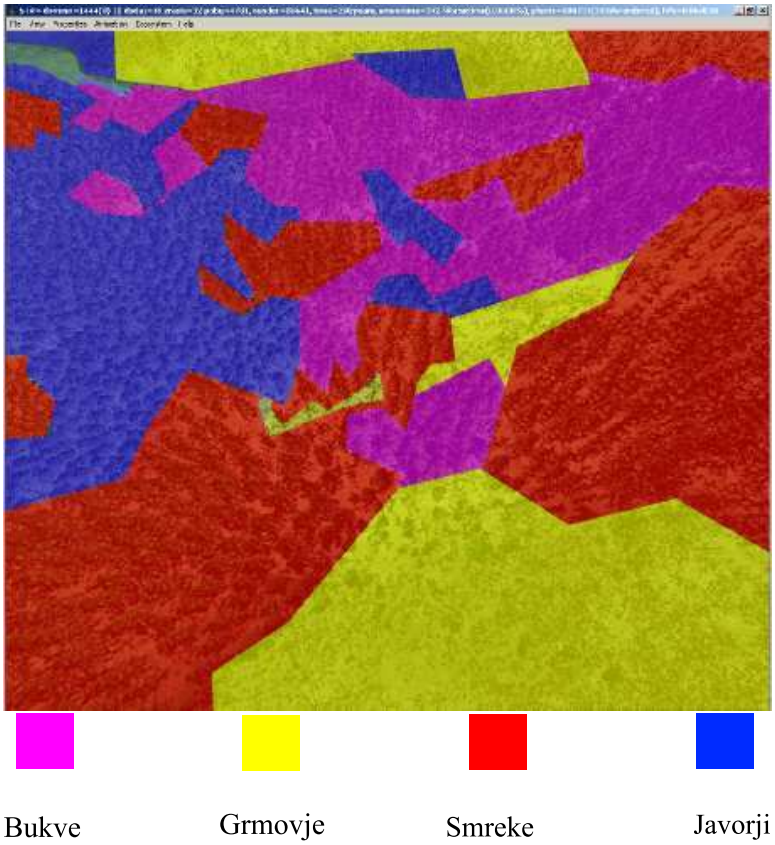
Slika 6.2: Čez 20 let: grmovnice se hitro razširijo po terenu.



Slika 6.3: Simulacija po 60 letih: rasti začnejo tudi druge drevesne vrste, ki izpodrivajo grmovnice, teren je zaraščen.



Slika 6.4: Simulacija po 80 letih: vse bolj prodirajo tudi smreke.



Slika 6.5: Simulacija po 180 letih: bukve, javorji, smreke in grmovnice se porazdelijo glede na lastnosti terena in medsebojne vplive.

Poglavje 7

Zaključek

V diplomskem delu smo predstavili problematiko ekološkega modeliranja in vlogo računalniške grafike pri realističnem prikazu naravnega procesa. Pri tem smo morali pretvoriti znane zakonitosti iz ekologije in biologije v računalniške algoritme, ki omogočajo njihovo simulacijo v računalniku.

V ekosistemu smo na pokrajini vizualizirali drevesa, ki smo jih tvorili z modelirnikom dreves. Slednji je omogočal tudi animacijo in poenostavljanje geometrijske strukture, ki smo jo povezali z vizualizacijo pokrajine. Animirali smo rast dreves in zibanje v vetru, geometrijsko strukturo dreves pa poenostavljali glede na oddaljenost od gledišča.

Simulacija ekosistema je za vernejšo porazdelitev rastlin na terenu poleg tekmovanja med rastlinami za prostor in rast v bližini starševske rastline upoštevala še lastnosti terena, kot so naklon, vlaga, osončenost in vetrovnost. S tem smo povečali vernost porazdelitve rastlin in dosegli rast rastlin v skupinah glede na ugodnosti življenjskih pogojev na določenem mestu.

Nadaljnje raziskave bi bilo možno opraviti pri modeliranju, simulaciji kot tudi pri upodabljanju. Drevesa bi lahko generirali s kakšnim drugim modelom. Modeliranje ekosistema bi lahko opravili z drugačnimi modeli za izračun lastnosti terena. Model bi prav tako lahko dodatno preverili z realnimi podatki iz kakšne slovenske pokrajine. Z razširitvijo modela bi lahko prikazali vpliv onesnaževanja okolja na rast dreves. Simulacijo bi lahko za še hitrejše izvajanje porazdelili z vmesnikom MPI. Upodabljanje bi lahko med več OpenGL strežnikov porazdelili tako, da bi uporabili sistem Chromium, vsak strežnik pa bi izračunal geometrijo proceduralnih modelov dreves in izrisal samo tista, ki so vidna v pripadajočem delu vidnega polja.

Dodatek A

Seznam uporabljenih simbolov

Pri izdelanem modelu ekosistema so uporabljeni sledeči simboli. Pri tem indeksi p pomenijo rastlino, s vrsto rastline, i in k pa indeksa krpe. Imena indeksov so ločena z vejico. Opisni indeksi (ločeni s podpičjem od ostalih, npr. $n_{neigh;i,k}$), dodatno pojasnjujejo glavni simbol.

A_s	skupno število rastlin, ki jih dodamo v poljubni točki terena, za ves teren
$\mathbf{c}_{i,k}$	središčna točka krpe $\mathbf{s}_{i,k}$
$\mathbf{c}'_{i,k,w}$	projekcija točke w-tega vzorca vetra v ravnino izhodiščne krpe
d	premer veje
$d(A, B)$	razdalja med dvema rastlinama A in B, ki se nahajata v točkah $\mathbf{p}_A$ in $\mathbf{p}_B$.
$D_{s,p}$	dominiranost p-te rastline iz s-te rastlinske vrste; rastlina je lahko dominirana od poljubne močnejše rastline, $D \in \{0, 1\}$
$F_{A,B}$	(zastavica da/ne) prekrivanje območij ekoloških sosednosti dveh rastlin A in B, ki se nahajata v točkah $\mathbf{p}_A$ in $\mathbf{p}_B$.
$f_{m;s}$	faktor škodljivosti zaradi odmika od dopustne vlažnosti za s-to rastlinsko vrsto
$f_{l;s}$	faktor škodljivosti zaradi odmika od dopustne osončenosti za s-to rastlinsko vrsto
$f_{\overline{y};s}$	faktor škodljivosti zaradi odmika od dopustne nadmorske višine za s-to rastlinsko vrsto
$f_{w;s}$	faktor škodljivosti zaradi odmika od dopustne vetrovnosti za s-to rastlinsko vrsto
$f_{s;s}$	faktor škodljivost naklona za s-to rastlinsko vrsto
g	Graveliusov red (indeks) veje
g_s	faktor razploda rastlinske vrste s

$h_{s,p}$	višina p-te rastline v metrih, ki pripada rastlinski vrsti s
$h_{m;s}$	povprečna višina, ki naj bi jo rastlina iz vrste s dosegla v letu zrelosti
i_{max}	dimenzija terena v številu krp v smeri osi x
$\mathbf{k}_1$, $\mathbf{k}_2$	vektorja na stranicah krpe $\mathbf{s}_{i,k}$
k_l	faktor utežitve osončenosti
k_{max}	dimenzija terena v številu krp v smeri osi z
$k_l^{g,w}$	skalirni faktor dolžine veje
$k_c^{g,w}$	koeficient gravicentralizma za veje na indeksu (g, w)
k_p	fototropizem (tendenca rasti v smeri proti svetlobi)
k_f	prožnost vej drevesa pri vetru
k_m	koeficient razširjanja tekočin pri izračunu vlažnosti
$k_o^{g,w}$	ortotropizem (t. j. tendenca po rasti vej navpično navzgor)
$k_p^{g,w}$	plagiotropizem (t. j. tendenca po rasti vej horizontalno navzven od debla)
$k_s^{g,w}$	delež žil osnovne veje, ki se cepijo v glavno podvejo
k_d	koeficient debeline veje
k_w	Writhov koeficient za naključnost rasti
$k_{\overline{y};s,p}$	vpliv nadmorske višine na mestu p-te rastline, ki pripada rastlinski vrsti s
K	dimenzija stranice krpe v ravnini xz (v metrih)
L_0	relativna dolžina osnovne veje
L_1 (L_2)	relativna dolžina glavne (stranske) podveje
$l_0^{0,0}$	dolžina prvega odseka debla
l_0	dejanska dolžina osnovne veje
l_1 (l_2)	dejanska dolžina glavne (stranske) podveje
$l_{ambient}$	razpršena svetloba na terenu, ki jo upoštevamo pri krpah v senci
$l_{basic;i,k,n}$	osončenost krpe $\mathbf{s}_{i,k}$ iz n-tega vzorca položaja sonca brez upoštevanja razlik v jakosti sevanja sonca
$l_{i,k}$	osončenost krpe $\mathbf{s}_{i,k}$
$l_{i,k,n}$	osončenost krpe $\mathbf{s}_{i,k}$ iz n-tega vzorca položaja sonca
$l_{o;s}$	optimalna osončenost za s-to rastlinsko vrsto
$\Delta l_{o;s}$	dopusten, še neškodljiv odmik od osončenosti za s-to rastlinsko vrsto
$l_{s,p}$	ugodnost pogoja osončenosti p-te rastline
l_{type}	način porazdelitve listov
l_l	velikost listov
l_{LOD}	stopnja poenostavitve geometrije

$\mathbf{M}_{m;1}^{-1}$	inverzna matrika za konstruiranje vektorja proti tlom v koordinatnem sistemu glavne podveje
$\mathbf{M}_{w;1}^{-1}$	inverzna matrika za konstrukcijo vektorja pihanja vetra v koordinatnem sistemu glavne podveje
M_0	koordinatni sistem prvega odseka debla (globalni koordinatni sistem)
$m_{climate}$	povprečna količina padavin na terenu za eno krpo glede na dano klimo
$m_{i,k}$	vlažnost na krpi $\mathbf{s}_{i,k}$ (količina padavin po izvedbi nizkega sita)
$m_{o;s}$	optimalna vlažnost za s-to rastlinsko vrsto
$\Delta m_{o;s}$	dopusten, še neškodljiv odmik od vlažnosti za s-to rastlinsko vrsto
$m_{s,p}$	ugodnost pogoja vlažnosti za p-to rastlino, ki pripada rastlinski vrsti s
m_w	količina padavin, ki jo dodamo za reke; vpliva na kasnejše širjenje vlage z interpolacijo
$\Delta m_{i,k}$	delež odtečenih padavin iz krpe $\mathbf{s}_{i,k}$
$m_t^{g,w}$ $(M_t^{g,w})$	spodnja (zgornja) meja relativne dolžine podvej glede na dolžino osnovne veje, ločeno za tip podveje ($t \in \{major, minor\}$)
$n_t^{g,w}$ $(N_t^{g,w})$	spodnja (zgornja) meja absolutne dolžine podvej glede na dolžino osnovne veje, ločeno za tip podveje ($t \in \{major, minor\}$)
$m^{g,w}$ $(M^{g,w})$	spodnja (zgornja) meja relativne dolžine podvej glede na dolžino osnovne veje združeno za tip podveje
$\mathbf{n}_{i,k}$	normalni vektor v točki $\mathbf{p}_{i,k}$ in tudi na krpi $\mathbf{s}_{i,k}$
$\overline{\mathbf{n}}_{i,k}$	enotski normalni vektor v točki $\mathbf{p}_{i,k}$
N_l	število vzorcev položaja sonca
N_{mblur}	število povprečenj vzorcev vlage
N_{opt}	največje možno število krp za katerega sta lahko oddaljeni dve sosednji rastlini, glede na njuna največja možna ekološka radija
$\Delta N_{g,s}$	število novih rastlin vrste s v novi generaciji
N_s	število rastlinskih vrst
N_w	število krp velikosti K v zavetrju
$n_{neigh;i,k}$	število strani odtekanja padavin iz krpe $\mathbf{s}_{i,k}$
$\mathbf{p}_v$	točka gledišča, kjer stoji opazovalec
$p_{s,p}$	moč p-te rastline, ki pripada s-ti rastlinski vrsti
$p'_{s,p}$	moč zrele rastline, sicer 0
$p_{r;p}$	največja možna relativna doživeta moč p-te rastline

$p_{m;s}$	verjetnost odmrtja dominirane rastline s-te rastlinske vrste v tekočem letu simulacije
$\mathbf{p}_N$, $\mathbf{p}_S$, $\mathbf{p}_E$, $\mathbf{p}_W$	vektorji v smeri strani neba (sever, jug, vzhod, zahod)
p_s	verjetnost rasti nove rastline blizu svoje rastlinske vrste s
$P_{s,p}$	rastlina p rastlinske vrste s v linearno povezanem seznamu, kjer je dostop do elementov lahko le iterativen in ne poljuben
$P_{s,n}$	n-ti nov primerek iz rastlinske vrste s v trenutnem simulacijskem koraku
$P_{parent;s,n}$	naključna obstoječa rastlina iz vrste s za n-to dodajano rastlino
$\mathbf{p}_{s,p}$	položaj p-te rastline iz rastlinske vrste s
r_1 (r_2)	relativna dolžina glavne (stranske) podveje do naslednje vejitve
r_A	radij rastline A
$I_{s,n}$	indeks iz referenčnega seznama dodajanih rastlin za rastlinsko vrsto s
$r_{s,p}$	velikost ekološkega radija p-te rastline iz s-te rastlinske vrste (funkcija $t_{s,p}$)
R_s	koeficient razmerja med velikostjo radija v metrih in starostjo v letih
S	število žil v deblu drevesa (začetni odsek do prve vejitve)
S_0	število žil v tekoči veji, ki je lahko tudi deblo
S_1	število žil, ki se odcepi od veje s S_0 v glavno podvejo
S_2	število žil, ki se odcepi od veje s S_0 v stransko podvejo
$s_{i,k}$	naklon krpe terena
$\mathbf{s}_{i,k}$	(i,k)-ta krpa-štirikotnik terena
$\mathbf{s}_{s,n}$	krpa za n-to dodano rastlino, ki pripada rastlinski vrsti s
$s_{s,p}$	strmina na mestu p-te rastline, ki pripada rastlinski vrsti s
s_s	število semen na terenu za rastlinsko vrsto s
$t_{s,p}$	starost p-te rastline iz s-te rastlinske vrste
$t_{max;s,p}$	pričakovana najvišja starost rastline p iz vrste s
$t_{f;s}$	največja možna starost rastlin s-te rastlinske vrste
$t_{m;s}$	leto zrelosti rastline s-te rastlinske vrste
$t_{a;s}$	starost p-te rastline, ko doseže maksimalno višino
$v_{s,p}$	živost p-te rastline, ki pripada s-ti rastlinski vrsti
w	Weibullov red (indeks) veje
$\mathbf{w}$	smer pihanja vetra, v globalnem koordinatnem sistemu
$\mathbf{w}_0$	smer pihanja vetra, v koordinatnem sistemu trenutne veje
W	jakost vetra
$w_{i,k}$	vetrovnost krpe $\mathbf{s}_{i,k}$

$w_{o;s}$	optimalna vetrovnost za s-to rastlinsko vrsto
$\Delta w_{o;s}$	dopusten, še neškodljiv odmik od vetrovnosti za s-to rastlinsko vrsto
$w_{s,p}$	ugodnost vetrovnosti na mestu p-te rastline, ki pripada rastlinski vrsti s
$w_s(t)$	hitrost (jakost) vetra spremenljive smeri na drevo
$w_g(t)$	hitrost (jakost) pihanja sunka vetra nespremenljive smeri na drevo
w_x (w_y)	kot zasuka trenutne veje vzdolž osi veje (pravokotno na vejo) zaradi neusmerjenega pihanja vetra
w_{geom}	Weibullov indeks, do katerega razvijamo geometrijsko strukturo drevesa
w_{branch}	Weibullov indeks, do katerega rišemo veje drevesa
y	nadmorska višina
$\overline{y}$	normalizirana nadmorska višina
$\overline{y}_{o;s}$	optimalna nadmorska višina za s-to rastlinsko vrsto
$\Delta\overline{y}_{o;s}$	dopusten, še neškodljiv odmik od nadmorske rastline iz s-te rastlinske vrste
$\overline{y}_w$	normalizirana višina nivoja vode
$\alpha_{i,k,n}$	kot med normalo krpe in vektorjem proti soncu na krpi $\mathbf{s}_{i,k}$
$\alpha^{g,w}$	kot med izhajajočima podvejama za veje na indeksu (g, w)
$\alpha_m^{g,w}$	kot zasuka za gravimorfizem za veje na indeksu (g, w)
α_1	kot odklona glavne podveje
α_2	kot odklona stranske podveje
α_p	kot zasuka okoli osi vzdolž veje
$\alpha_x(t)$, $\alpha_z(t)$	kota zasuka pri vetru spremenljive smeri
α_w	kot zasuka pri usmerjenem vetru
$\varphi_{i,k}$	kot zavetrja krpe $\mathbf{s}_{i,k}$
Θ_0	prag kota brez zavetrja
ρ_l	gostota listov
τ	korak simulacije (čas)

Dodatek B

Izpeljava povprečnega kota osončenja v poletnem času med enakonočjema

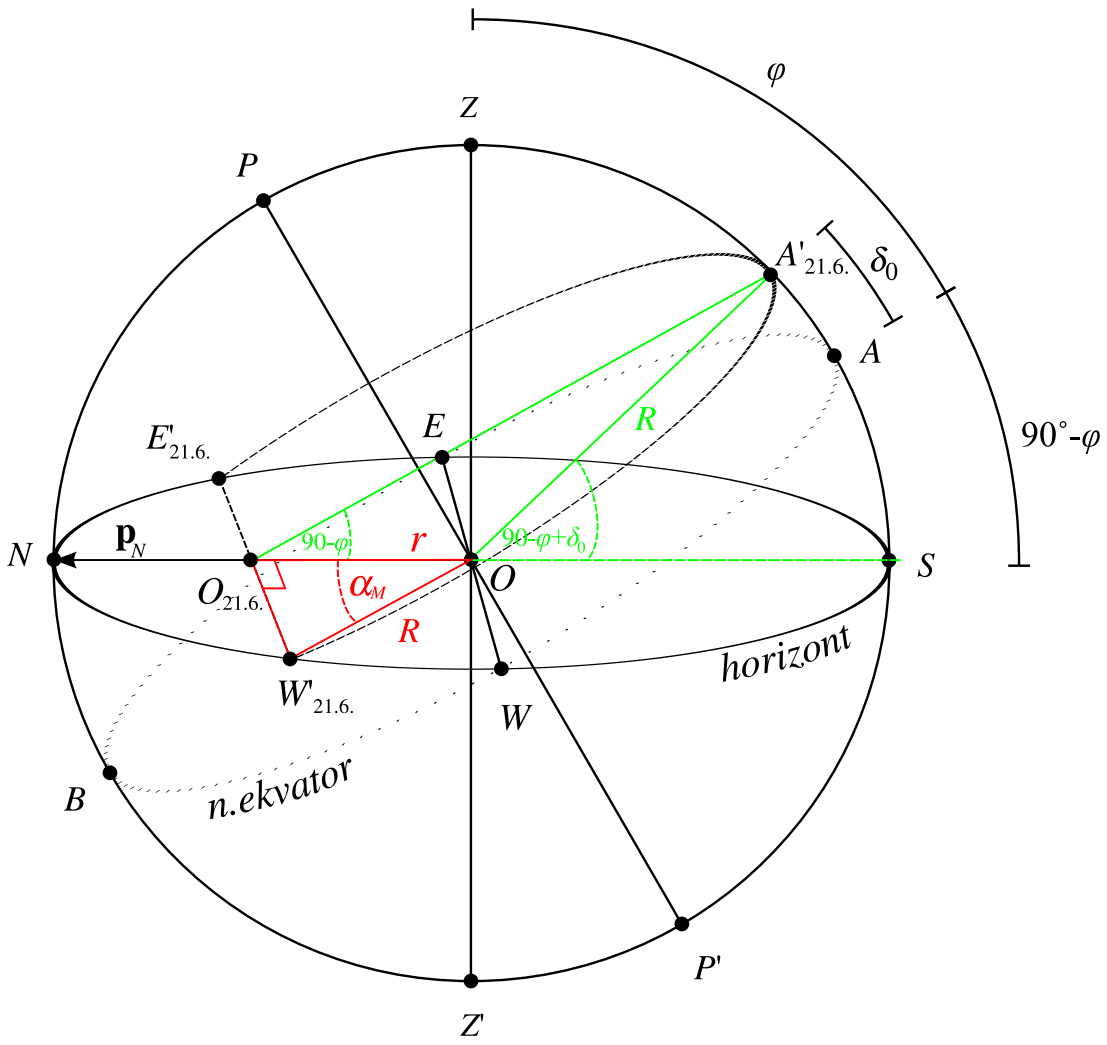
V modelu osončenja vzamemo povprečni kot vzhoda in zahoda sonca skozi pomladansko in poletno obdobje (od pomladanskega enakonočja 21. marca do jesenskega enakonočja 23. septembra), da pri osončenju terena izračunamo vektor $\mathbf{p}_0$. Ta kot znaša približno $\alpha = 21,98°$ proti severu, zato vektorju položaja sonca $\mathbf{p}_0$ prištejemo približno $0,37\mathbf{p}_N$. Izpeljava sledi.

Znano je, da 21. marca in 23. septembra sonce vzide na vzhodu in zaide na zahodu. Takrat namreč sonce v ekvatorju sije pravokotno na Zemljo in nastopi enakonočje, saj sta dan in noč enako dolga. Od enega do drugega enakonočja v poletnem času na severni polobli sonce vzhaja nekoliko proti severu in zahaja prav tako nekoliko proti severu. To pomeni, da sonce čez dan po svoji poti na nebu opravi več kot 180°. Da bi dobili povprečen kot, koliko proti severu sonce vzhaja, moramo najprej določiti njegov ekstrem α_M, to je 21. junija, na dan poletnega solsticija.

Slovenija leži na 46. vzporedniku ($\varphi = 46°$), kar pomeni, da je v času poletnega solsticija zaradi nagiba $\delta_0 = 23,5°$ kot sonca enak $90° - 46° + 23,5° = 67,5°$ (kot $\measuredangle A_{21.6.}OS$ na sliki B.1). Iz kota $\measuredangle A_{21.6.}O_{21.6.}S = 44°$ pa lahko po sinusnem izreku izračunamo razmerje med dolžinama r in R. Razmerje med polmerom Zemlje R in delčkom polmera, ki se seka s horizontom (ESW: vzhod-jug-zahod), znaša:

$$\frac{r}{\sin \measuredangle O_{21.6.}A_{21.6.}O} = \frac{R}{\sin \measuredangle OO_{21.6.}A_{21.6.}}, \tag{B.1}$$

$$\frac{r}{\sin\left(180° - (90° - \varphi) - (180° - (90° - \varphi + \delta_0))\right)} = \frac{R}{\sin\left(90° - \varphi\right)}, \tag{B.2}$$



Slika B.1: Točke in krogi nebesne krogle (sfere). O označuje središče nebesne polkrogle, kjer je opazovalec. Z označuje zenit (nadglavišče), Z' nadir (podnožišče), P severni nebesni pol (tečaj), P' južni nebesni pol, E, W, S in N pa vzhod, zahod, sever, jug. Premico skozi ZO imenujemo navpičnica, premico skozi PP' nebesna os. Krivulja skozi EAW določa nebesni ekvator, skozi ESW pa horizont (obzorje). Krivulja NPZ določa nebesni meridian (poldnevnik). φ označuje geografsko širino opazovalca. δ_0 označuje odmik v stopinjah od enakonočja in je v času poletnega solsticija enak $23,5°$. Oznaki $E'_{21.6.}$ in $W'_{21.6.}$ označujeta točki vzhoda in zahoda sonca na dan poletnega solsticija na točki z geografsko širino $\varphi - \delta_0$, krivulja skozi njiju in $A'_{21.6.}$ pa leži v ravnini, vzporedni EAW. Z rdečo barvo poudarjen trikotnik nam da iskani kot α_M, iz katerega kasneje izračunamo povprečni α. Kot α_M je napet med vektorjem dolžine polmera zemlje R, v smeri do zahoda sonca na dan poletnega solsticija in vektorjem v smeri proti severu, dolžine r.

$$\frac{r}{\sin 23,5^\circ} = \frac{R}{\sin 44^\circ}, \tag{B.3}$$

$$\frac{r}{R} = \frac{\sin 23,5^\circ}{\sin 44^\circ}. \tag{B.4}$$

Kot α_M v spodnjem pravokotnem trikotniku med stranico r in hipotenuzo R izračunamo:

$$\sin \alpha_M = \frac{r}{R}, \tag{B.5}$$

$$\alpha_M = \sin^{-1} \frac{r}{R}, \tag{B.6}$$

$$\alpha_M = \sin^{-1} \frac{sin 23,5^\circ}{sin 44^\circ}, \tag{B.7}$$

$$\alpha_M = \sin^{-1} 0,574, \tag{B.8}$$

$$\alpha_M = 35,031. \tag{B.9}$$

Ker se Zemlja okoli sonca vrti po elipsi, ki jo je za naš izračun možno aproksimirati s krožnico, se α_M s časom spreminja po sinusnem zakonu, če si za $t = 0$ izberemo pomladno enakonočje.

Za izračun povprečnega kota α čez poletje moramo izračunati polovico celoletne sinusne periode s $t_0 = 365,25$ dni. Kot $\alpha = 0$ je v času spomladanskega enakonočja ($t = 0$), v času poletnega solsticija $t = \frac{t_0}{4}$ je $\alpha = \alpha_M$, v času jesenskega enakonočja pri $t = \frac{t_0}{2}$ pa spet $\alpha = 0$. Ker je sinusoida simetrična, moramo izračunati samo povprečje na prvi četrtini. Najprej pa izračunamo integral, da bi kasneje izračunali še povprečje:

$$\omega = \frac{2\pi}{t_0}, \tag{B.10}$$

$$\alpha_{sum} = \int_0^{\frac{t_0}{2}} \alpha_M \sin(\omega t) dt, \tag{B.11}$$

$$\alpha_{sum} = 2\alpha_M \int_0^{\frac{t_0}{4}} \sin(\omega t) dt, \tag{B.12}$$

$$\alpha_{sum} = 2\alpha_M(-\cos(\omega t)|_{\frac{t_0}{4}} - (-\cos(\omega t)|_0)), \tag{B.13}$$

$$\alpha_{sum} = 2\alpha_M(0 + 1), \tag{B.14}$$

$$\alpha_{sum} = 70^\circ. \tag{B.15}$$

Da bi dobili povprečje, rezultat integrala delimo s časom, skozi katerega se vred-

nosti kumulirajo, t. j. pol leta ($\Delta t = \frac{t_0}{2}$):

$$\alpha = \frac{\alpha_{sum}}{\frac{t_0}{2}}, \tag{B.16}$$

$$\alpha = \frac{70^\circ}{\frac{365,25}{2}}, \tag{B.17}$$

$$\alpha = 0,383 = 21,980^\circ. \tag{B.18}$$

Za izračun položaja sonca $\mathbf{p}_0$ moramo prišteti vektor proti severu, ki je dolžine sinusa kota α:

$$\mathbf{p}_0 = \mathbf{p}_E + \sin(\alpha)\mathbf{p}_N, \tag{B.19}$$

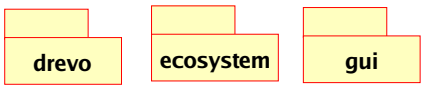
$$\mathbf{p}_0 = \mathbf{p}_E + 0,374\mathbf{p}_N. \tag{B.20}$$

Za pomoč pri razlagi astronomskih koordinatnih sistemov in izpeljavi kota α_M se zahvaljujemo asist. mag. Vladimirju Grubelniku.

Dodatek C

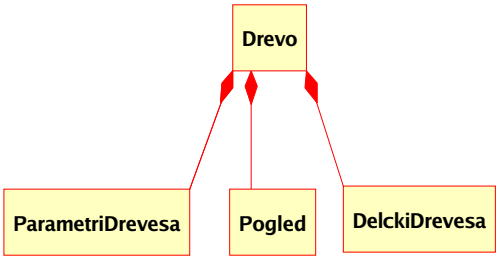
Programski paket EcoMod

Programski paket **EcoMod** je razdeljen na knjižnici **drevo** in **ecosystem** ter glavni program **gui** za interaktivno delo (slika C.1). Vsi paketi skupno obsegajo več kot 50 razredov, ki se ob povezovanju aplikacije sestavijo v eno izvršilno datoteko.



Slika C.1: Diagram paketov v aplikaciji.

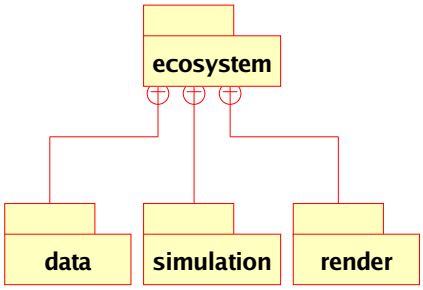
Knjižnica **drevo** služi upodabljanju dreves. Glavni razred je **Drevo** (slika C.2), ki vsebuje razred **ParametriDrevesa** za hranjenje parametrov za proceduralni model geometrije drevesa. Vsebuje tudi drugi razred **Pogled**, ki definira izgled geometrije in nastavitve animacije drevesa.



Slika C.2: Diagram pomembnejših razredov v knjižnici **drevo**.

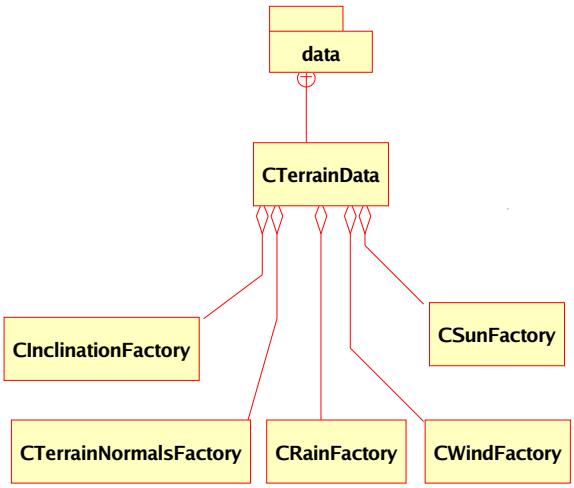
Knjižnica **ecosystem** služi simulaciji in upodabljanju ekosistema. Razdeljena je na pakete **data**, **simulation** in **render** (slika C.3).

Terenski podatki so shranjeni v paketu **data** (slika C.4). Razred **CTerrainData** je krovni razred, ki poleg splošnih podatkov terena, kot njegova velikost in ločljivost krp, vsebuje še podatke na posameznih krpah. Te podatke vračajo metode v vsebovanih objektih razredov za podatke terena. Razred **CTerrainNormals** iz podanih točk terena izračunava normale krp terena. Razred **CInclinationFactory**



Slika C.3: Diagram paketov v knjižnici **ecosystem**.

iz normal določa strmine po krpah, razred **CRainFactory** vlažnost krp, razred **CSunFactory** pa osončenost krp. Razred **CWindFactory** definira vetrovnost na krpah. Objekti naštetih razredov po zagonu aplikacije ob tvorbi objekta tipa **CTerrainData** izračunajo lastnosti terena, ki skozi simulacijo ostanejo enake.

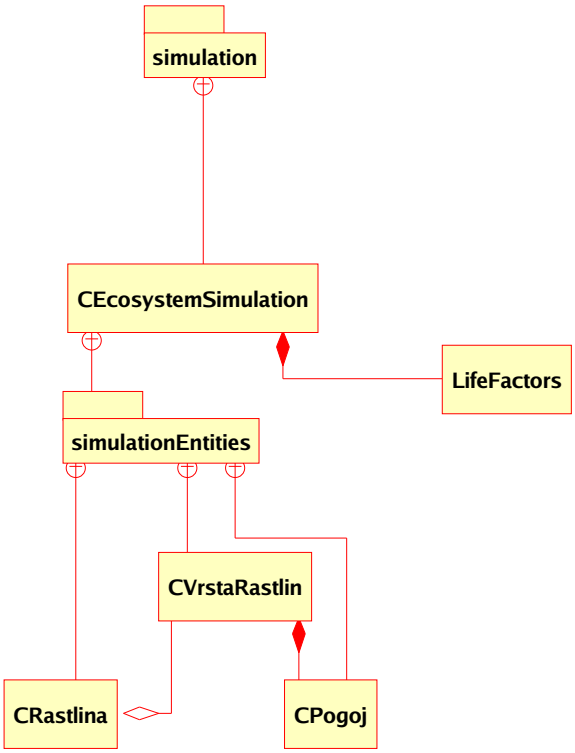


Slika C.4: Diagram pomembnejših razredov v paketu **data** knjižnice **ecosystem**.

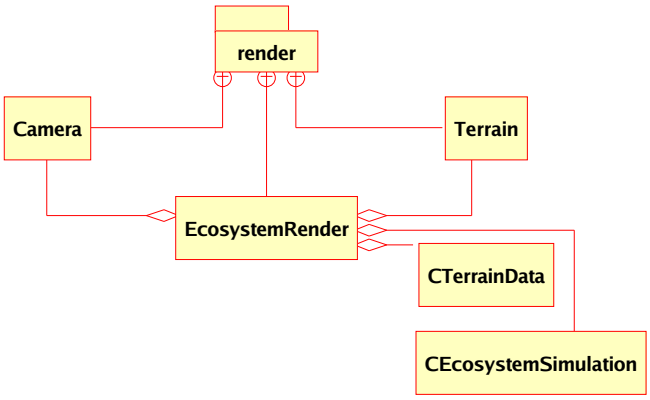
Simulacijo vodi paket **simulation** (slika C.5) z glavnim razredom **CEcosystemSimulation**. Ta razred hrani razpoložljive rastlinske vrste v polju objektov razreda **CVrstaRastlin**, vse rastline v ekosistemu v polju objektov razreda **CRastlina** in objekt za izračunavanje življenjskih pogojev rastlinam tipa **LifeFactors**. Prav tako pa uporablja podatke o terenu iz objekta razreda **CTerrainData**.

V paketu **render** (slika C.6) so vsebovani razredi za upodobitev elementov ekosistema. Razred **EcosystemRender** je glavni razred paketa, uporablja ostale razrede v paketu in sodeluje z nekaterimi razredi iz drugih paketov. Vsebuje objekt razreda **CTerrainData**, ki ga deli z objekti razredov **Camera**, **Terrain** in **CEcosystemSimulation**. Razred **Camera** služi postavljanju parametrov pogleda na sceno, kot sta položaj in smer gledanja, prav tako pa omogoča hojo po površini terena. Razred **Terrain** služi izrisovanju trikotnikov za mrežo terena.

Paket **gui** služi interaktivnemu upravljanju z aplikacijo. Uporablja paketa **drevo**

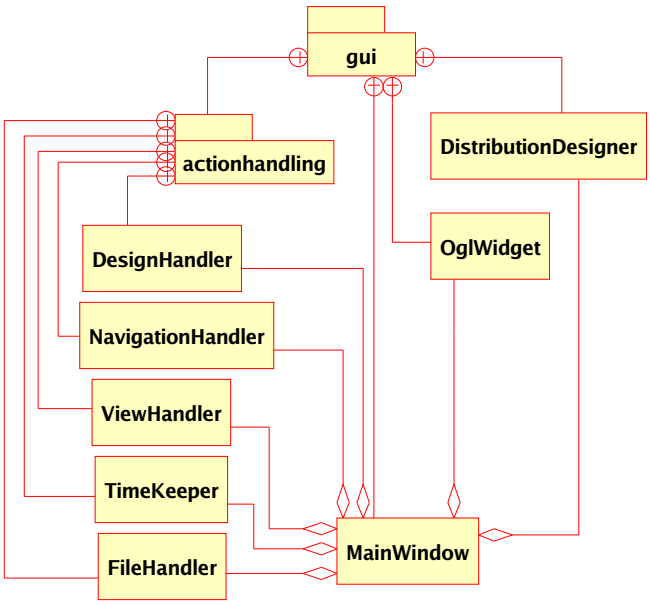


Slika C.5: Diagram pomembnejših razredov v paketu **simulation** knjižnice **ecosystem**.



Slika C.6: Diagram pomembnejših razredov v paketu **render** knjižnice **ecosystem**.

in **ecosystem**. Za grafično upodobitev scene na okno skrbi razred **OglWidget** (slika C.7), ki služi zgolj kot vmesnik za vzpostavitev konteksta OpenGL med okenskim sistemom in razredom **EcosystemRender**. Glavni razred aplikacije je **MainWindow**, ki sprejema uporabnikove interaktivne ukaze ter delegira ustrezne akcije. Povezan je tudi z uporabniškimi dialogi v paketu, med katere spada **DistributionDesigner**. S pomočjo slednjega lahko interaktivno načrtujemo graf porazdelitve količin v odvisnosti od indeksov g in w, pri tem pa se geometrija sproti osvežuje zaradi proženja signalov, ki jih sprejme **OglWidget** in sliko osveži. Ostale uporabniške akcije urejanja parametrov drevesa so zajete v razredu **DesignHandler**. Razred **NavigationHandler** služi spreminjanju nastavitev kamere, **ViewHandler** za spreminjanje nastavitev izgleda objektov na sceni, razred **TimeKeeper** skrbi za tek morebitne animacije, razred **FileHandler** pa za operacije z datotekami, kot so shranjevanje in nalaganje parametrov drevesa iz datoteke ali nalaganje tekstur drevesa.



Slika C.7: Diagram pomembnejših razredov v paketu **gui** knjižnice **gui**.

C.1 Ključni deli programske kode

C.2 Navodila za uporabo programa

Jedro sistema je zasnovano tako, da je prenosljivo med Win32 in sistemi X11 v jeziku C++ [Eckel, 2000a; Eckel, 2000b]. Koda je bila v prvih različicah vezana na knjižnico Win32 API [Wikipedia, 2006b], kasneje pa na knjižnico Qt [Blanchette in

```
304 void DelckiDrevesa::risiList( float velikost ) {
305     glColor4f(0.5f, 0.5f, 0.5f, 0.8f);
306     if ( lowQuality < LEAF_WHOLE ) {
307         glRotatef( -90, 1.0, 0.0, 0.0 ); //z postane y->disk na y=0
308         gluDisk( pogled.kvadraticnePloskve, 0, velikost, 4, 1 );
309         glRotatef( 90, 1.0, 0.0, 0.0 );
310     } else {
311         glPointSize( 1.4f );
312         glBegin( GL_POINTS ); {
313             glVertex3f( 0, 0, 0 );
314         }
315         glEnd();
316     }
317 } //DelckiDrevesa::risiList
```

Slika C.8: Izvorna koda za risanje lista iz datoteke **delckidrevesa.cpp**.

```
49         //steblo renderiramo kar s cilindrom
50         if ( lowQuality < BRANCH_WHOLE ) {
51             glRotatef( -90, 1.0, 0.0, 0.0 );
52             glBindTexture( GL_TEXTURE_2D, pogled.textureNames[ 0 ] );
53             if ( pogled.highQualityRendering && lowQuality == 0 )
54                 gluCylinder( pogled.kvadraticnePloskve, sirinaSpodaj,
   sirinaZgoraj, visina, 10, 5 );
55             else
56                 gluCylinder( pogled.kvadraticnePloskve, sirinaSpodaj,
   sirinaZgoraj, visina, 5, 1 );
57         } else {
58             glBindTexture( GL_TEXTURE_2D, pogled.textureNames[ 0 ] );
59             glBegin( GL_LINES );
60             {
61                 glTexCoord1d( 0 );
62                 glVertex3f( 0, 0, 0 );
63                 glTexCoord1d( 1 );
64                 glVertex3f( 0, visina, 0 );
65             }
66             glEnd();
67         }
```

Slika C.9: Del izvorne kode za risanje enega odseka stebla, iz razreda **DelckiDrevesa** v datoteki **delckidrevesa.cpp**.

```
122         //pomik gor po steblu, zamik na stran novega stebla (raste direktno iz
    konca skorje, ne iz srede debla)
123         glTranslatef( 0, dejanskaVisinaTrenutneVeje, 0 );
124
125         //gravicentralizem!!
126         double gravicentralizem = kotVejitveGlavno * pd.graviCentralizem;
127         if ( G == 0 )
128             glRotatef( gravicentralizem, 0.0, 0.0, 1.0 );
129         else
130             gravicentralizem = 0;
131
132         //stransko steblo?
133         if ( stZilStransko >= 1 ) {
134             Matrika nova = modelMatrixInverse;
135             priraviInverznoMatriko( nova, gravimorfizemAngle,
    gravimorfizemVektor, gravicentralizem, kotVejitveStransko, 0, windVektor );
136
137             Matrika novaPostEffects = modelMatrixInversePostGrowthEffects;
138             priraviInverznoMatriko( novaPostEffects, gravimorfizemAngle,
    gravimorfizemVektor, gravicentralizem, kotVejitveStransko, windAngle, windVektor
    );
139
140             double sirinaVeje = sqrt( ( double ) stZilStransko ) *
    pd.sirinaSteblaNaZilo;
141             double pomikVejeVRastOdZnotraj = -( sirinaZgoraj - sirinaVeje ) / 2.0;
142
143             //trenutno matriko postavimo na sklad
144             glPushMatrix();
145             {
146                 glTranslatef( pomikVejeVRastOdZnotraj, 0, 0 );
147                 //zasuk v smer stranske veje
148                 glRotatef( kotVejitveStransko, 0.0, 0.0, 1.0 );
149                 //kliče rast stranske veje, presek za to je novi: NULL
150                 drevca( stZilStransko, G + 1, W + 1, L2, l2, nova,
    novaPostEffects, NULL, idVejaNext++ );
151             }
152             glPopMatrix();
153
154             //zasuk v smer za glavno vejo
155             glRotatef( -kotVejitveGlavno, 0.0, 0.0, 1.0 );
156         } else
157             kotVejitveGlavno = 0;
158
159         //glavno deblo
160         if ( stZilGlavno >= 1 ) {
161             //tu lahko povozimo trenutno inverzno matriko
162             priraviInverznoMatriko( modelMatrixInverse, gravimorfizemAngle,
    gravimorfizemVektor, gravicentralizem, -kotVejitveGlavno, 0, windVektor );
163             priraviInverznoMatriko( modelMatrixInversePostGrowthEffects,
    gravimorfizemAngle, gravimorfizemVektor, gravicentralizem, kotVejitveStransko,
    windAngle, windVektor );
164             //glavna veja, gravelius se ohranja
165             drevca( stZilGlavno, G, W + 1, L1, l1, modelMatrixInverse,
    modelMatrixInversePostGrowthEffects, &presekSpodajNovi, idPraveVeje );
166         }
```

Slika C.10: Del izvorne kode za izračun delitve veje drevesa iz datoteke **drevo.cpp**.

```
71 void CEcosystemSimulation::zrastiVseRastlineInDolociMoci() {
72     for (std::vector<simulationEntities::CVrstaRastlin>::iterator vr =
   rastlinskeVrste.begin(); vr != rastlinskeVrste.end(); vr++) {
73         simulationEntities::CVrstaRastlin &rastlinskaVrsta = *vr;
74         ///semena so tu že od nekdaj ali pa jih sedaj prinesejo živali
75         rastlinskaVrsta.komulacijaSemen =
   1.0f/rastlinskaVrsta.stNovihRastlinNaSimulacijskiKorak;
76         for (simulationEntities::CVrstaRastlin::ziviPrimerkiTeVrsteIter r =
   rastlinskaVrsta.ziviPrimerkiTeVrste.begin(); r !=
   rastlinskaVrsta.ziviPrimerkiTeVrste.end(); r++) {
77             simulationEntities::CRastlina &rastlina = *r;
78             if (!rastlina.jeDominirana)
79                 zrastiRastlino(rastlina, rastlinskaVrsta);
80             else
81                 rastlina.prodornost=rastlina.prodornost-1.0f>0?rastlina.prodornost
   -1.0f:0;
82             rastlina.trenutnaStarost++;
83         }
84     }
85 }//CEcosystemSimulation::zrastiVseRastlineInDolociMoci
```

Slika C.11: Izvorna koda za izračun rasti rastlin iz datoteke **ecosystemsimulation.cpp**.

```
75 bool CObjectCulling::containsSphere(const drevo::Tocka& centerObjekta, float
   radijObjekta) const {
76     float razdalja;/// izračunamo razdaljo do vsake od ravnin
77     for(int i = 0; i < 6; i++) {
78         drevo::Tocka n((float)p_planes[i].a, (float)p_planes[i].b,
   (float)p_planes[i].c);
79         razdalja = n.skalarniProdukt(centerObjekta)+(float)p_planes[i].d;
80
81         if(razdalja < -radijObjekta) /// če je zunaj za keterokoli, je zunaj
82             return false;   ///če je razdalja manjša od -radijKrogle, smo zunaj
83
84         if((float)fabs(razdalja) < radijObjekta)/// če se vsaj z eno seka, je
   notri!
85             return true;    ///če je razdalja med +-radij, se objekt seka z ravnino
86     }
87     return true;///sicer je cel objekt notri v pogledu
88 }
89
```

Slika C.12: Del izvorne kode za obrezovanje objektov na vidni volumen iz datoteke **objectculling.cpp**.

Summerfield, 2004; Trolltech, 2005], s katero smo dosegli njeno prenosljivost.

Za vizualizacijo je uporabljen Gouraudov model senčenja iz specifikacije OpenGL z razširitvami. Programski paket omogoča:

- prikaz dreves s spreminjajočo 3D geometrijo v realnem času,
- izvoz geometrije za upodabljanje v kakšnem drugem programskem paketu (preizkušeno s paketom LightWave),
- prikaz realističnega terena in obzorja okolja,
- simulacijo ekosistema z interakcijo med rastlinami in
- interakcijo z uporabnikom v slovenskem ali angleškem jeziku, odvisno od nastavitev sistema.

C.2.1 Modeliranje dreves

Najprej načrtujemo porazdelitev žil, nato načrtujemo kote vejitev, spremenimo dolžine vej, morda uporabimo nekaj gravimorfizma ter uredimo dodatne napredne lastnosti drevesa. Tam lahko določimo število žil v drevesu, velikost prvega odseka debla, relativno debelino glede na žilo, vpliv gravicentralizma, kot filotakse, tip, gostoto in velikost listov ter hitrost sunkov vetra.

Pogled na drevo spreminjamo preko miške in tipkovnice: *levi gumb+premik miške* premakne drevo ven/notri, *desni gumb+premik miške* suče drevo okoli osi x in y, tipke *(shift+)gor/dol/levo/desno* pa premikajo drevo gor/dol/levo/desno in tipka *ESC* vrne privzete nastavitve pogleda.

Meni **Datoteka** služi operacijam z datotekami. **Shrani parametre** shrani nastavljive parametre drevesa v datoteko. **Naloži parametre** naloži nastavljive parametre iz datoteke. **Izvozi sliko** izvozi sliko v rastrski format (BMP, JPEG, PNG). **Izvozi oglišča** izvozi geometrijsko strukturo drevesa v datoteko kot nabor kvadratov. **Izhod** zapre aplikacijo.

Meni **Pogled** spreminja način izrisovanja in pogleda na geometrijo. **Prikaži liste** spreminja prikaz listov. **Upodabljanje z visoko kvaliteto** spreminja visoko/nizko ločljivostno tvorbo geometrije za upodabljanje. **Obarvaj po G,W** obarva vsak G/W red veje v drugi barvi. **Žični model** spreminja prikaz geometrije ožičeno ali s teksturami. **Luči omogočene** omogoči upodabljanje z lučmi. **Ponastavi pogled** parametre kamere vrne na začetne vrednosti.

Meni **Uredi** spreminja parametre proceduralnega modela drevesa. Izbira **Načrtovanje porazdelitve žil** prikaže dialog za načrtovanje grafov debeline glavnih vej, **Načrtovanje kotov vejitev** kot med podvejama, **Načrtovanje dolžine vej**

karakteristične dolžine veje, **Načrtovanje skaliranja dolžine vej** skaliranje dolžin vej in **Načrtovanje vpliva gravimorfizma** vpliv gravimorfizma. **Povečaj velikost listov** poveča velikost listov za 10%, **Pomanjšaj velikost listov** pa pomanjša za 10%. **Naloži teksturo debla** naloži teksturo debla, **Naloži teksturo listov** pa teksturo listov. Dialog **Napredne lastnosti** služi nastavljanju skalarnih lastnosti drevesa.

Starost drevesa določa starost drevesa, kompleksnost in debelino osnovnega debla. **Višina debla** določa višino prvega odseka debla. **Relativna debelina vej na žilo** določa koeficient za debelino vej. **Gravicentralizem** določa pokončnost osrednjega debla. **Kot filotakse** določa zasuk veje okoli osnovne veje. **Tip listov** določa tip razporeditve listov: spiralno, nakopičeno, nestalno, šopasto ali igličasto. **Gostota listov** določa število listov na koncu vej. **Velikost listov** določa velikost listov. **Hitrost vetra** določa hitrost pihajočega vetra.

Meni **Animacija** služi vklopu in izklopu animacije dreves. **Rast** sproži simulacijo rasti drevesa. **Suči** sproži sukanje drevesa okoli samega sebe, da si ogledamo celotno drevo. **Zibanje v vetru** sproži simulacijo zibanja v vetru.

Pomoč prikaže dialoga s podatki o programu in navodila o programu. **O...** prikaže podatke o verziji, avtorstvu in licenčnih pogojih programa, **Uporaba** pa prikaže jedrnat vodič za uporabo programa.

C.2.2 Simulacija ekosistema

Meni **Simulacija ekosistema** skrbi za uporabo simulacije ekosistema. **Poženi simulacijo** vklopi ali izklopi tok simulacije. **Računaj več korakov na eno upodobljeno okno** in **Računaj manj korakov na eno upodobljeno okno** povečata ali zmanjšata število upodobljenih slik na en simulacijski korak. **Prikaži statistiko** prikaže statistiko s podatki o številu rastlin v ekosistemu po vrstah (slika C.13).

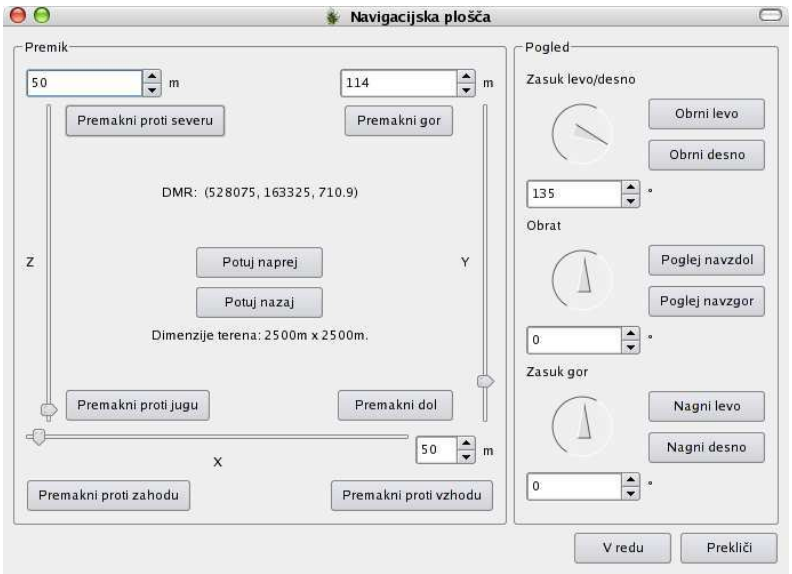
Gledišče v načinu pogleda na ekosistem spreminjamo na sledeč način: tipki *naprej/nazaj* pomakneta kamero naprej/nazaj, s tipkama *levo/desno* pogledamo levo/desno, tipki *home/end* služita pogledu gor/dol, s tipkama *delete/page up* se premikamo levo/desno. *Insert/PageUp* nas zavrtita levo/desno. *Pika* nas pomakne navzdol, tipka *vejica* pa navzgor.

Nastavljanje pogleda lahko opravimo tudi z **navigacijsko ploščo**, ki jo vidimo na sliki C.14.

S pogovornim oknom na sliki C.15 lahko nastavimo območje rasti, na katerem lahko rastline rastejo na terenu. Območje je podano z indeksi krp, ki območje omejujejo.

S pogovornim oknom C.16 lahko urejamo ali samo spremljamo semena genera-

Slika C.13: Statistika za ekosistem. Prikazano je število bukev skozi nekaj let simulacije. Z desnim klikom na točko v grafu se izpiše točna vrednost grafa, t.j. število rastlin v izbranem letu.



Slika C.14: Navigacijska plošča za spreminjanje pogleda. Po terenu se lahko premikamo z drsniki na plošči ali pa vrednost vpišemo ročno. Če držimo vklopljen gumb (npr. naprej), se premikamo v tisto smer. Pri spreminjanju pogleda se prav tako osvežujejo koordinate DMR (digitalni model reliefa).

Slika C.15: Nastavljanje območja rasti rastlin. Podamo indekse krp, med 0 in 100.

torjev naključnih števil.

Z menijsko izbiro **Shrani simulacijo** lahko simulacijo tudi shranimo. Pri tem shranimo vse lastnosti živih rastlin in semena generatorja naključnih števil. Z izbiro **Naloži simulacijo** shranjeno simulacijo lahko naložimo, prikažemo in tudi nadaljujemo.

Slika C.16: Pregled in urejanje vseh semen za generator naključnih števil, uporabljenih v simulaciji.

Literatura

Aono, M. in Kunii, T. (1984). Botanical tree image generation. *IEEE Computer Graphics and Applications*, 4(5):10–34.

Blanchette, J. in Summerfield, M. (2004). *C++ GUI Programming with Qt 3*. Prentice Hall v sodelovanju s Trolltech Press, Upper Saddle River (New Jersey).

Bloomenthal, J. (1985). Modeling the mighty maple. V Barsky, B. A., urednik, *SIGGRAPH '85 Conference Proceedings (San Francisco, CA, 22–26 julij 1985)*, strani 305–311.

Chiba, N., Muraoka, K., Doi, A. in Hosokawa, J. (1997). Rendering of forest scenery using 3D textures. *Journal of Visualization and Computer Animation*, 8(4):191–199.

Cook, R. L. (1986). Stochastic sampling in computer graphics. *ACM Transactions on Graphics*, 5(1):51–72.

Dale, M. R. T. (1999). *Spatial Pattern Analysis in Plant Ecology*. Cambridge Studies in Ecology. Cambridge University Press, Cambridge, UK.

Deussen, O., Colditz, C., Stamminger, M. in Drettakis, G. (2002). Interactive visualization of complex plant ecosystems. V *Proceedings of the conference on Visualization '02*, strani 219 – 226.

Deussen, O., Hanrahan, P., Lintermann, B., Mech, R., Pharr, M. in Prusinkiewicz, P. (1998). Realistic modeling and rendering of plant ecosystems. V *SIGGRAPH*, strani 275–286.

DigiBen (2005). Game tutorials. [Internet `http://www.gametutorials.com/gtstore/c-1-test-cat.aspx`; dostop 4. maj 2006].

Dobry, J. (2000). Opengl - getting high performance. [Internet `http://jdobry.webpark.cz/opengl/opengl_maximum_performance.html`; dostop 4. maj 2006].

Eckel, B. (2000a). *Thinking in C++, Volume 1.* Prentice Hall, druga izdaja.

Eckel, B. (2000b). *Thinking in C++ Volume 2: Standard Libraries and Advanced Topics.* Prentice Hall, druga izdaja.

Gribb, G. in Hartmann, K. (2001). Fast extraction of viewing frustum planes from the world-view-projection matrix. [Internet `http://www2.ravensoft.com/users/ggribb/planeextraction.pdf`; dostop 27. marec 2006].

Holton, M. (1994). Strands, gravity, and botanical tree imagery. *Comput. Graph. Forum*, 13(1):57–67.

Hopkins, B. (1954). A new method for determining the type of distribution of plant individuals. *Annals of Botany*, XVIII:213–226.

Lane, B. in Prusinkiewicz, P. (2002). Generating spatial distributions for multilevel models of plant communities. V *Proceedings of the Graphics Interface 2002 (GI-02)*, strani 69–80, Mississauga, Ontario, Canada. Canadian Information Processing Society.

Li, Z., Zhu, Q. in Gold, C. (2005). *Digital Terrain Modeling.* CRC Press, San Francisco.

Lindenmayer, A. (1968). Mathematical models for cellular interactions in development, I & II. *J. Theor. Biol.*, 18:280–315.

Lintermann, B. in Deussen, O. (1999). Interactive modeling of plants. *IEEE Computer Graphics and Applications*, 19(1):56–65.

Mandelbrot, B. (1982). *The fractal geometry of nature.* Freeman, San Francisco.

Marsaglia, G. (1972). The structure of linear congruential sequences. V Zaremba, S. K., urednik, *Appl. Number Theory numer. Analysis, Proc. Sympos. Univ. Montreal 1971, 249–285*, strani 249–285. Academic Press, New York, NY, USA.

Martinčič, A., Vrhovšek, D. in Batič, F. (1981). *Ekologija rastlin.* VTO za biologijo, Pleško, Ljubljana, Rožna dolina c. IV/36.

Oppenheimer, P. E. (1986). Real time design and animation of fractal plants and trees. *Computer Graphics*, 20(4):55–64.

Papert, S., Watt, D., diSessa, A. in Weir, S. (1979). Final report of the brookline LOGO project. Technical Report 545, MIT AI Lab.

Picco, D. (2003). Frustum culling. [Internet `http://www.flipcode.com/articles/article_frustumculling-pf.shtml`; dostop 27. marec 2006].

Pressman, O. E. (2001). Terrain engine. [Internet `http://ohad.visual-i.com/exper/exper.htm#terrain`; dostop 18. april 2005].

Prusinkiewicz, P., Hammel, M., Hanan, J. in Měch, R. (1997). L-systems: From the theory to visual models of plants. V Michalewicz, M. T., urednik, *Plants to Ecosystems*, volume 1 of *Advances in Computational Life Sciences*, poglavje 1, strani 1–27. CSIRO Publishing, P.O. Box 1139, Collingwood 3066, Australia.

Prusinkiewicz, P. in Lindenmayer, A. (1990). *The Algorithmic Beauty of Plants*. Springer-Verlag.

Reeves, W. (1985). Approximate and probabilistic algorithms for shading and rendering structured particle systems. *Proceedings of SIGGRAPH'85*, stran 313.

Rickefs, R. E. (1990). *Ecology*. W. H. Freeman, New York, tretja izdaja.

Strnad, D. in Guid, N. (2004). Modeling trees with hypertextures. *Comput. Graph. Forum*, 23(2):173–188.

Trolltech (2005). *Qt 4.0 Whitepaper*. Trolltech press.

van Lieshout, M. P. (2004). Time dependent enery calculations. V *Wind flow and Trees*. British Wind Energy Association.

Vera, E. (2001). *Bryce 5 User Guide*. MetaCreations Corporation.

Weber, J. in Penn, J. (1995). Creation and rendering of realistic trees. *Proceedings of SIGGRAPH '95*, strani 119–128.

Wikipedia (2006a). Opengl — wikipedia, the free encyclopedia. [Internet `http://en.wikipedia.org/w/index.php?title=OpenGL&oldid=53083766`; dostop 14. maj 2006].

Wikipedia (2006b). Windows api — wikipedia, the free encyclopedia. [Internet `http://en.wikipedia.org/w/index.php?title=Windows_API&oldid=53145546`; dostop 15. maj 2006].

Woo, M. s sodelavci (1999). *OpenGL programming guide: the official guide to learning OpenGL, version 1.2*. Addison-Wesley, tretja izdaja.

Zakšek, K., Oštir, K. in Podobnikar, T. (2004). Osončenost površja slovenije. *Geografski vestnik*, 76(1):79–90.