SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 2345

Proceduralno generiranje trave i niskog raslinja

Mihael Međan

Zagreb, siječanj 2021.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 9. listopada 2020.

DIPLOMSKI ZADATAK br. 2345

Pristupnik:	**Mihael Međan (0036487393)**
Studij:	Računarstvo
Profil:	Računarska znanost
Mentor:	prof. dr. sc. Željka Mihajlović

Zadatak: **Proceduralno generiranje trave i niskog raslinja**

Opis zadatka:

Proučiti tehnike generiranja trave i niskog raslinja uz različite razine detaljnosti u prikazu (LOD). Proučiti utjecaj fizikalnih veličina kao što je vjetar na izradu simuliranih i animiranih prikaza. Razraditi i ostvariti fizikalno temeljen simulacijski model prikaza trave i niskog raslinja iz različite razine detaljnosti prikaza. Diskutirati utjecaj raznih parametara. Načiniti ocjenu rezultata i implementiranih algoritama. Izraditi odgovarajući programski proizvod. Rezultate rada načiniti dostupne putem Interneta. Radu priložiti algoritme, izvorne kodove i rezultate uz potrebna objašnjenja i dokumentaciju. Citirati korištenu literaturu i navesti dobivenu pomoć.

Rok za predaju rada: 5. veljače 2021.

SADRŽAJ

1. Uvod

Uvod rada. Nakon uvoda dolaze poglavlja u kojima se obrađuje tema.

2. Generiranje modela trave i niskog raslinja

2.1. Osnovni pristup generiranju modela

2.2. Generiranje točaka i poligona modela biljaka

2.2.1. Generiranje modela uz različitu razinu detalja

2.3. Povezivanje generiranih podataka i njihov prikaz

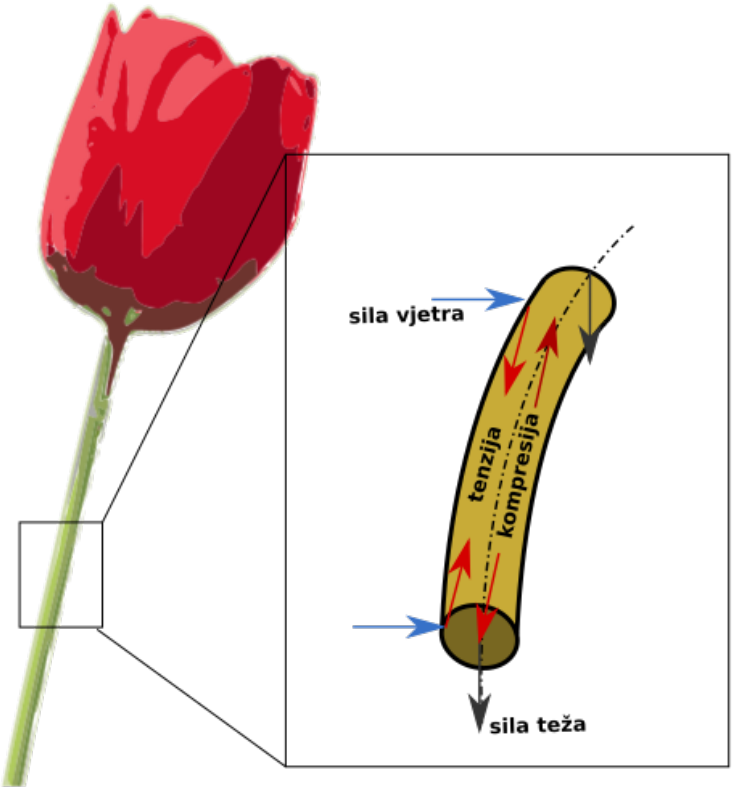
2.4. Programska implementacija generiranja i prikaza

2.5. Primjer korištenja razvijenog programa za generiranje jednostavne proizvoljne biljke

3. Simulacijski model

3.1. Fizički model ponašanja biljke

Osnovne sile koje utječu na ponašanje biljke su sila teža, unutrašnji otpor biljke i vanjski utjecaji. Sila teža konstantnim intenzitetom gura sve dijelove biljke prema dolje. Vanjski utjecaji poput vjetra, djeluju u svim smjerovima različitim intenzitetima na različite dijelove biljke. Te unutrašnji otpor biljke, koji uvijek djeluje u suprotnom smjeru od preostale dvije sile. Na slici 3.1 su slikovito prikazane sile koje djeluju na svaki mikroskopski malen dio biljke. Važno je napomenuti da su za tenziju i kompresiju prikazane sile koje one uzrokuju, a ne smjer kompresije odnosno tenzije.



Slika 3.1: Prikaz sila koje djeluju na biljku

Usred djelovanja te tri sile, biljka u svakom trenutku pokušava doći u stanje gdje se te tri sile poništavaju. Svako gibanje biljke prouzrokovano je promjenom vanjskih utjecaja kojima se biljka pokuša prilagoditi kako bi ukupna sila koja djeluje na biljku bila nula.

Čak i naizgled jednostavan problem kao što je simuliranje ponašanja biljke je u svojoj naravi izrazito kompleksan. Sila teža djeluje na svaki mikroskopski dio biljke. Na isti način djeluju i vanjske sile. Vanjske sile i sila teže u biljci uzrokuju napetosti i kompresije koje uzrokuju silu otpora biljke. To sve se događa na mikroskopskoj razini na svakom dijelu površine biljke. Savršena simulacija ovakvog sustava bila bi izrazito računalno zahtjevna.

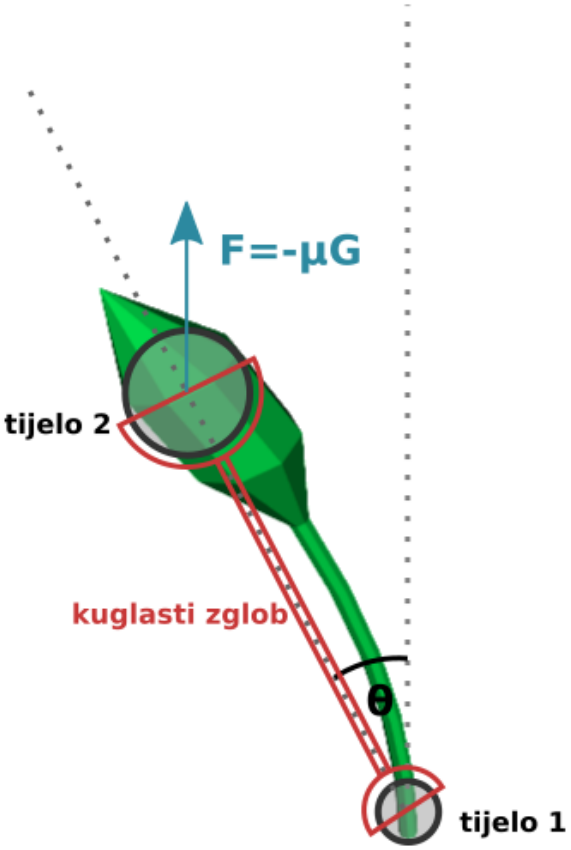
3.2. Programski model fizičkog ponašanja

Zbog računalne zahtjevnosti i problema preciznog definiranja svih sila, potrebno je pronaći matematički model koji bi rezultirao istim (ili barem vrlo sličnim) ponašanjem, a imao bi mnogo manju računalnu složenost i bio bi jednostavniji za definirati.

U računalnom programiranju se fizičke interakcije obično modeliraju preko tri osnovna građevna bloka. Tijela, ograničenja i sile. Tijela su geometrijski oblici u trodimenzijskom prostoru koja zauzimaju volumen i podložna su djelovanju sila. Ograničenja su skup pravila koja određuju kako se neka dva tijela mogu ponašati relativno jedno prema drugom. Sile djeluju na tijela i uzrokuju promjene koje se tada na temelju ograničenja rješavaju.

Primjer tijela je kocka u trodimenzijskom prostoru. Primjer sile je sila teža koja djeluje konstantnim intenzitetom prema negativnoj y osi. Primjer ograničenja je ako se volumen jednog tijela nađe unutar volumena drugog tijela, na oba tijela se primjenjuje sila u smjeru suprotnom od središta mase drugog tijela. Ovo ograničenje je naivni pristup rješavanju kolizija između tijela.

Stvarnu simulaciju jednostavne biljke možemo vrlo dobro aproksimirati korištenjem svega dva tijela, jednog ograničenja i jedne sile koja djeluje na jedno od tijela. Ovaj jednostavni model prikazan je slikom 3.2.



Slika 3.2: Ilustracija prikaza fizickog modela stvarnog gibanja

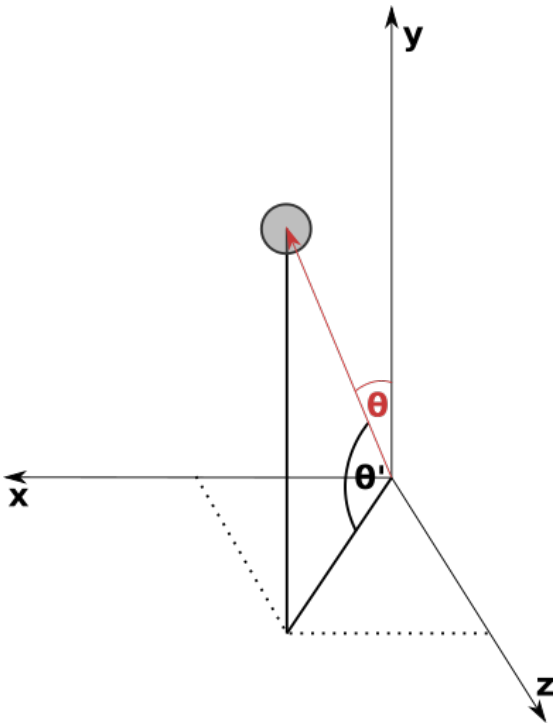
Tijelo 1 u fizickom modelu pretstavlja korijen biljke i u simulaciji je u cijelosti staticno. Tijelo 2 pretstavlja vrh biljke i dinamičko je, odnosno može reagirati na utjecaj sila. Ta dva tijela spojena su kuglastim zglobom. Kuglasti zglob je ograničenje na tijelo 2 koje dozvoljava tijelu da se slobodno giba i rotira u prostoru oko bilo koje osi pod uvjetom da je uvijek jednako udaljeno od tijela 1.

Na tijelo 2 također djeluje sila F koja je svojim smjerom obrnuta od smjera sile teže, a njezin iznos je određen kutom otklona od neutralne pozicije. Neutralna pozicija ima tijelo 2 direktno iznad tijela 1 i u takvom slučaju je faktor μ jednak 0.

Računanje otklona θ u tro dimenzijskom prostoru radimo kao omjer duljine stranica trokuta. Jedna od stranica je projekcija pozicije tijela 2 na ravninu xz, a druga stranica je visina tijela 2 iznad te ravnine. Zbog kuglastog zgloba očekujemo da je duljina vektora pozicije tijela 2 od ishodišta uvijek jednaka, normalizacijom pozicije tijela 2, duljina projekcije na xz ravninu ima vrijednost cos kuta θ'. Zanimljiva informacija nam je otklon od y osi, koristimo $sin(\theta')$ te vrijednost kako bi dobili suprotan kut $(\pi/2-\theta')$.

U takvom slučaju nam je dovoljna informacija uzeti y vrijednost normalizirane pozicije i možemo precizno izračunati otklon. Računaje kuta otklona prikazano je formulom 3.1.

$$\theta = asin(normalized(position).y) \tag{3.1}$$



Slika 3.3: Prikaz izračuna kuta otklona

Nakon izračuna kuta otklona, računanje sile koja djeluje na tijelo 2 je sada jednostavno i prikazano formulom 3.2. Računanje sile je također proporcionalno sa sinusom kuta na intervalu između 0 i 90 stupnjeva. Sila koja djeluje na tijelo kod otklona većeg od 90 stupnjeva veća je od sile koja djeluje na tijelo na otklonu manjem od 90 stupnjeva. Kod jednostavnog modela s jednim zglobom ovo je nerealan slučaj jer se tijelo 2 ne može naći u takvoj poziciji (tijelo 2 bi tada bilo ispod tijela 1, i prošlo bi kroz površinu poda), ali ga je bitno spomenuti zbog skaliranja na model sa većim brojem zglobova.

$$\vec{F} = sin(\theta) * (1 + elasticity) * \vec{G} * -1 \tag{3.2}$$

Rezultirajuća sila $\vec{F}$ djeluje u negativnom y smjeru (suprotno od smjera sile teže). Takav smjer je rezultat faktora $\vec{G} * -1$. U formuli se javlja i faktor $(1 + elasticity)$.

Njegova zadaća je osigurati da biljke koje imaju veći faktor elastičnosti stoje bliže centru ravnoteže čak i kod malih otklona od središta. Primjer biljke koja ima velik faktor elastičnosti je rogoz, koji stoji potpuno uspravno u neutralnoj poziciji. Primjer male elastičnosti biljke je zvončić, koji nema dovoljno sile kako bi se vratio u neutralni (uspravni) položaj. Ime faktora je elastičnost jer odgovara na pitanje kolikom silom će se biljka pokušati vratiti u neutralni položaj nakon što je otklonjena od centra ravnteže.

3.3. Programska implementacija fizičkog modela

Fizički pokretač korišten u prezentiranom rješenju je OimoPhysics. Taj pokretač je napisan u programskom jeziku Haxe i preveden u programski jezik Javascript. Izbor ovog fizičkog pokretača prvenstveno je potaknut njegovom relativnom jednostavnošću i njegovom mogućnosti izvršavanja direktnog izvođenja unutar preglednika. Zbog jednostavnastavnog modela kojeg simuliramo, nije predana velika težina performansama koje zbog izvođenja u interpretiranom jeziku mogu negativno utjecati na brzinu izvođenja programskog rješenja.

Osnova svakog fizičkog pokretača je svijet simulacije (engl. *world*) koji sadržava sve osnovne gradivne blokove fizičke simulacije objašnjene u sekciji 3.2. Svijet implicitno definira ograničenja za rješavanje kolizija između objekata. Inicijalno postavljanje vrijednosti svijeta u OimoPhysics pokretaču je iznimno jednostavno i prikazano je isječkom koda 3.1.

```
1 const world = new OIMO.World(1, new Vec3(0, -9.81, 0))
```

Isječak koda 3.1: Postavljanje simulacijskog svijeta

Prvi parametar kod inicijalizacije je izbor algoritma rješavača kolizija (engl. *collision-solver*. Različiti algoritmi pridaju različite važnosti pojedinim elementima simulacije. Za znanstvene radove izabrali bi algoritam koji manju pažnju pridaje brzini izvođenja simulacije, a veću pažnju točnosti same simulacije. Za igre i ostale sustave koji se izvršavaju u stvarnom vremenu, manje nam je bitna točnost, a bitnija je brzina izvođenja. Algoritmi unutar OimoPhysics pokretača su označeni od 0 do 2, gdje vrijednost implicira kolika se težina pridaje točnosti simulacije. U prezentiranom radu korištena je postavka 1, koja postiže balansirani odnos između točnosti i brzine izvođenja.

Nakon kreiranja simulacijskog svijeta dodajemo fizički kostur svakoj biljci. Fizički kostur biljke sastoji se od 3 dijela. tijela 1 koje je nepomično i opisuje korijen biljke, tijela 2 koje je vrh biljke i podložno je utjecaju sila te kuglastog zgloba koji povezuje tijelo 1 i tijelo 2.

Tijelo u simulacijskom svijetu opisano je s dva konfiguracijska objekta. Svaki konfiguracijski objekt objašnjava zaseban dio simulacije. Prvi konfiguracijski objekt su općenita svojstva tijela poput njegove inijcijalne pozicije u simulacijskom svijetu i vrsta ponašanja tijela. Konfiguracijski razred za općenita svojstva tijela u OimoPhysics pokretaču je `RigidBodyConfig`. U korištenom fizičkom pokretaču postoje tri vrste ponašanja tijela - statično, dinamički i kinematičko. Na statična tijela ne utječe sila i ne mijenjaju svoju poziciju i rotaciju čak ni prilikom kolizije s drugim tijelima. Na dinamička tijela utječe sila i ona se ponašaju kao svi objekti u stvarnom svijetu - prilikom kolizije se odbijaju i rotiraju, a prilikom djelovanja sile ubrzavaju ili usporavaju u smjeru sile. Dinamička tijela reagiraju na koliziju sa statičnim i kinematičkim tijelima. Kinematička tijela ne reagiraju na kolizije, ali reagiraju na sile i njihovim utjecajem mogu mijenjati svoju poziciju i rotaciju.

```
1  const origin = new OIMO.Vec3(
2    plant.pos[0], plant.pos[1], plant.pos[2]
3  );
4
5  const baseConfig = new OIMO.RigidBodyConfig();
6  baseConfig.position = origin;
7  baseConfig.type = OIMO.RigidBodyType.STATIC;
8
9  let shapeConfig = new OIMO.ShapeConfig();
10 shapeConfig.geometry = new OIMO.BoxGeometry(
11   new OIMO.Vec3(0.1, 0.1, 0.1)
12 );
13
14 const base = new OIMO.RigidBody(baseConfig);
15 base.addShape(new OIMO.Shape(shapeConfig));
16
17 this.world.addRigidBody(base);
```

Isječak koda 3.2: Postavljanje tijela korijena biljke i njegovih svojstava

Za tijelo koje simulira korijen biljke pozicija tijela odgovara poziciji tijela u 3D prikazu. Vrsta tog tijela je statična. Ne mijenja svoju poziciju i na rotira se. Ovakvo ponašanje prati ponašanje stvarnih biljaka u normalnim atmosferskim uvjetima.

Drugi konfiguracijski razred je `ShapeConfig`. `ShapeConfig` opsije geometriju modela u 3D prostoru. U fizičkim pokretačima uobičajeno je pojednostaviti geometriju modela što je više moguće kako bi brzina izvođenja bila što veća. Iz tog razloga su geometrije tijela obično jednostavni geometrijski oblici poput kvadra, kugle, piramide i stošca. Složeniji oblici mogu se dobiti dodavanjem dodatnih jednostavnih oblika na tijelo.

Tijelo koje simulira korijen biljke je statično i nema interakciju s ostalim tijelima, pa je u prezenitranom rješenju opisan samo kao mala kocka na dnu biljke. Postavljanje svojstava tijela korijena biljke i njegove geometrije prikazano je u isječku koda 3.2.

Tijelo koje simulira vrh biljke iako složenije, također ima vrlo jednostavan proces postavljanja početnih vrijednosti. Pozicija tijela je točno iznad tijela korijena biljke na y vrijednosti koja odgovara visini biljke. Vrsta ovog tijela je dinamičko, jer osim reagiranja na sile vjetra reagira i kod potencijalnih interakcija s ostalim biljkama. Promjene koje sadrži postavljanje početnih vrijednosti tijela vrha biljke u odnosu na tijelo korijena prikazan je isječkom koda 3.3.

```
1 ...
2 tipConfig.position = origin.add(new OIMO.Vec3(0, plant.height, 0));
3 tipConfig.type = OIMO.RigidBodyType.DYNAMIC;
4
5 ...
6 shapeConfig.geometry = new OIMO.SphereGeometry(
7   plant.collisionRadius
8 );
```

Isječak koda 3.3: Postavljanje tijela vrha biljke

Vrh biljke umjesto kocke ima kuglu. Iako neprecizno, kugla dobro opisuje ponašanje interakcije vrha biljaka i pogodno je za brzinu izvođenja.

Definicija kuglastog zgloba u OimoPhysics pokretaču također je jednostavna. Sastoji se od jednog konfiguracijskog objekta koji opisuje sva svojstva ovog ograničenja.

Taj konfiguracijski razred je `SphericalJointConfig`. `init` metodi ovog konfiguracijskog objekta predajemo tijela na koja će se ovo ograničenje primjenjivati. U prezentiranom rješenju to su tijela `base` i `tip`. Osim tijela na koja se ograničenje primjenjuje potrebno je definirati i referentnu točku za to ograničenje. Kod simulacije se vrh biljke rotira oko korijena biljke, pa poziciju korijena biljke uzimamo kao referentnu točku.

```
1 const jointConfig = new OIMO.SphericalJointConfig();
2 jointConfig.init(base, tip, baseConfig.position);
3 jointConfig.springDamper = new OIMO.SpringDamper().setSpring(0, 0);
4 jointConfig.breakForce = 0;
5 jointConfig.breakTorque = 0;
6
7 const joint = new OIMO.SphericalJoint(jointConfig);
8 this.world.addJoint(joint);
```

Isječak koda 3.4: Postavljanje kuglastog zgloba

U ostatku konfiguracije definiran je i prigušivač opruge. Prigušivač opruge simulira ograničenje kao oprugu umjesto čvrstu vezu između tijela. U prezentiranom rješenju postavke prigušivača su postavljene na nulu. Ovakve postavke postavljaju ograničenje kao čvrstu vezu između dva tijela. Biljke možemo zamisliti kao opruge pa ovakav izbor djeluje nelogično. Odabir baš takvih parametara rezultat je ručne implementacije vrlo sličnog ponašanja biljke. Korištenje oba sustava za simulaciju elastičnosti rezultira fizičkim nekonzistencijama u ponašanju biljke.

Parametri `breakForce` i `breakTorque` opisuju silu i obrtni moment pod kojim bi ograničenje popustilo, odnosno prestalo raditi. Vrijednost nula u konfiguraciji signalizira fizičkom pokretaču da je ograničenje konstantno, odnosno da neće prestati vrijediti bez obzira na sile koje djeluju na tijela. Cijelovito postavljanje vrijednosti ograničenja kuglastog zgloba prikazano je isječkom koda 3.4.

Nakon početnog postavljanja kostura biljke potrebno je definirati ponašanje biljke u simulaciji. Definicija ponašanja prikazana je u isječku koda 3.5. U simulaciji prolazimo kroz sve kuglaste zglobove u kosturu biljke i izvršavamo simulaciju za svaki od njih. U jednostavnim primjerima postoji samo jedan zglob za svaku od biljaka, ali na ovaj način je sačuvana mogućnost simuliranja mnogo složenijih konfiguracija.

Na početku jedne iteracije simulacije uzimamo tijela na koja je primjenjeno ograničenje kuglastog zgloba. Na drugo tijelo (vrh biljke) primjenjujemo silu vjetra. Sila vjetra je vektor neke duljine u 3D prostoru. Duljina vektora odgovara snazi koju vjetar ima u nekom trenutku. Nakon primjene sile vjetra, na tijelo se primjenjuje izrazito mala sila koja djeluje suprotno od vektora brzine tijela. Tijelo zbog te sile konstantno usporava i time riješava problem beskonačne simulacije čak i za vrlo male i kratkotrajne početne sile.

```
1  simulateMovement() {
2    for (const joint of this.skeleton.joints) {
3      const rigid1 = joint.getRigidBody1();
4      const rigid2 = joint.getRigidBody2();
5
6      this.applyWind(rigid2);
7      this.applyInnerFriction(rigid2);
8
9      // we need relative position of joint
10     const pos1 = rigid1.getPosition();
11     const pos2 = pos1.sub(rigid2.getPosition());
12
13     const force = Math.abs(this.forceFactor(pos2));
14     rigid2.applyForceToCenter(new OIMO.Vec3(0, force, 0));
15   }
16 }
```

Isječak koda 3.5: Fizička simulacija ponašanja biljke.

Nakon završene simulacije vjetra računamo silu koja objašnjava ponašanje biljke na vjetru. Sila je detaljno objašnjena u poglavlju 3.2. Funkcija `forceFactor` računa iznos sile kojom djelujemo, a opisana je formulama 3.1 i 3.2.

4. Simulacija ponašanja generiranog modela

4.1. Povezivanje fizičkog modela sa generiranim prikazom modela

Do sada su definirana dva sustava.

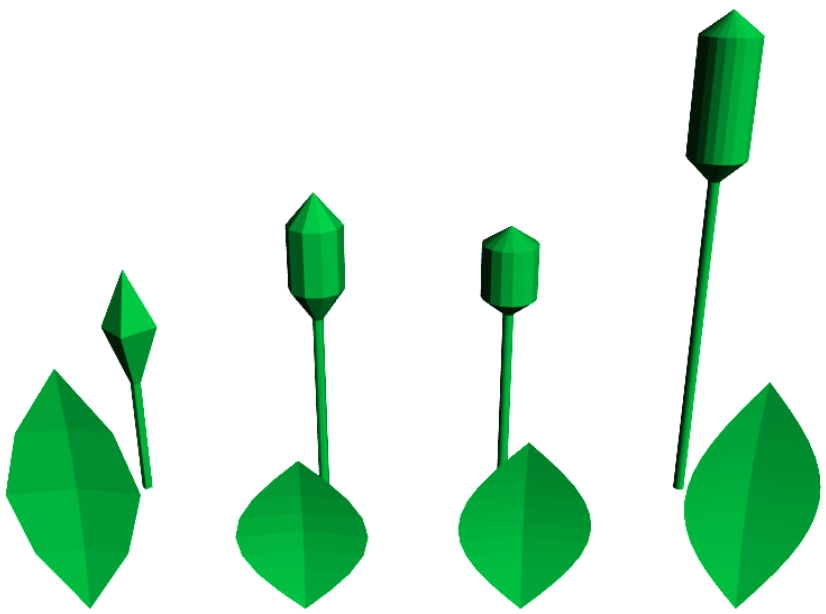
4.2. Programska implementacija modela

4.3. Utjecaj parametara na simulaciju

5. Ocjena rezultata

5.1. Realističnost prikaza

Realističnost prikaza nije bila primarna zadaća ovog rada. Generirane biljke ne izgledaju osobito prirodno bez obzira na razinu detalja kojom su generirane. Primjer generiranih biljaka vidljiv je na slici 5.1.



Slika 5.1: Prikaz modela u različitoj razini detalja

Prvi razlog je primjena konstantnog sjenčanja poligona. U ovakvom načinu sjenčanja, normale za pojedine vrhove računamo kao normalu površine (poligona) kojeg oni zatvaraju. Izračunata vrijednost normale se pripisuje svim vrhovima poligona. Korištenjem konstantnog sjenčanja poligona postižemo jasno vidljivu granicu između pojedinih površina. Takav način prikaza, osim što je vrlo brz u izvođenju, znatno je

olakšao razvoj prezentiranog programskog rješenja jer daje programeru jasan uvid u poziciju pojedinih vrhova na ekranu i njihovo ponašanje.

Bez obzira na prednosti u razvoju koje ovakav pristup implicira, prikazani modeli izgledaju umjetno i izrazito su pravilni. Pravilnost modela je najveći uzrok neprirodnog izgleda generiranih modela, jer na biljkama rijetko očekujemo savršenu simetriju oko geometrijskih osi.

Drugi razlog je nedostatak detalja na generiranim modelima biljaka. Ovaj razlog dodatno naglašava pravilnost biljaka i smanjuje osjećaj realnosti prikaza. Smanjenje broja detalja donijelo je iste prednosti kao i prethodni pristup - lakši razvoj i bolje performanse po cijenu realističnosti prikaza.

Treći razlog je izolacija prikaza. Biljke su prikazane kao centralni i jedini dio prikaza. Ovakav vakuum u prostoru dodatno naglašava oba prije spomenuta problema. Nedostatak simulacije atmosfere (prašina, distorzija zraka svjetlosti kao posljedica vlage u zraku itd.) uzrokuje dojam statičnosti biljaka bez obzira na animirani prikaz ponašanja vjetra. Izostavak navedenih značajki, kao i prethodna pojednostavljenja, olakšalo je razvoj programskog rješenja i oslobodilo vrijeme za ostvarenje kvalitetnije simulacije vjetra.

5.1.1. Mogućnosti poboljšanja

Problem sjenčanja može se riješiti primjenom nekog drugog načina sjenčanja modela. Naprimjer korištenje Gouraudovog sjenčanja. Korištenje ove metode sjenčanja eliminiralo bi vidljivost pojedinih vrhova na modelu i interpolacijom između vrhova osiguralo gladak prijelaz između osjenčanih i ne osjenčanih dijelova modela.

Biljke u stvarnosti rijetko imaju ikakve oštre bridove između svojih strana i zaglađivanje intenziteta osvjetljenja Gouraudovog sjenčanja između vrhova bi rezultiralo uvjerljivijim prikazom kod modela generiranih u većoj razini detalja. Kod modela generiranih manjom razinom detalja, rubovi samog modela bi ostali oštri iako je model zaglađen. Sraz između rubova i unutrašnjosti modela može uzrokovati smanjenje realističnosti prikaza.

Dodavanje malih varijacija kod generiranja modela poboljšalo bi realističnost prikaza. Mali istupi točaka od centra simetrije dali bi biljkama prirodniji izgled uvođenjem nesavršenosti i raznovrsnosti biljaka. Osim dodavanja istupa u poziciji točaka u modelu, istupi se mogu dodati i na boje pojedinih površina, gdje bi neke površine bile više ili manje intenzivne boje od drugih. Dodavanje boja u kombinaciji s Gouraudovim sjenčanjem dodatno bi pojačalo realističnost prikaza.

Dodavanje varijacija je programski izrazito jednostavno za implementaciju ali uzrokuje usporenje izvođenja, jer se svaki model zbog svoje unikatnosti mora preslikati u memoriju grafičke kartice umjesto korištenja istog modela za sve biljke iste vrste. Kompromis se može postići na sredini - imati limitiran broj različitih modela iste vrste biljke koji se nasumično odabere za svaku biljku prilikom njenog kreiranja.

Dodavanje atmosferskih efekata na simulaciju bi uvelike pomoglo realističnosti prikaza i dojmu živosti biljaka. Ovaj pristup riješavanju nerealističnosti prikaza je najkompleksniji od ponuđenih alternativa i zahtjeva razvoj nekolicine popratnih sustava, ali dao bi najveći doprinos realističnosti prikaza. Primjeri popratnih sustava koje je potrebno razviti uključuju: sustav ukrasnih čestica, sustav efekata nakon iscrtavanja (engl. *post-processing effects*) i druge.

5.2. Realističnost fizičke simulacije

Bez obzira na jednostavnost implementacije fizičke simulacije i snažne pretpostavke uključene u nju, fizička simulacija daje zadovoljavajuće rezultate. Ponašanje prati stvarno prirodno gibanje u tri dimenzije i ne potiče osjećaj umjetnosti ili ograničenosti prolaskom kroz prostor. Simulacija dobro modelira utjecaj visine biljke na njezinu simulaciju na vjetru.

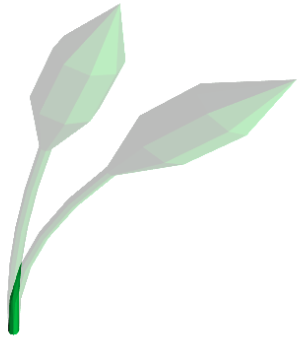
Na udarima vjetra visoke biljke rade velike i snažne zamahe i polagano gube energiju za nastavak daljnjeg osciliranja. Niske biljke rade kraće zamahe ali većom frekvencijom.

Kod konstantnog puhanja, biljke osciliraju prema točki konvergencije sile puhanja, sile teže i sila otpora unutar stabljke. Visoke biljke očekivano imaju veću amplitudu i manju frekvenciju oko te točke od niskih biljaka.

Puhanje s naletima vjetra također daje realistične rezultate. Niske biljke u naletu vjetra se brzo poravnavaju prema točki konvergencije i osciliraju oko nje, a po prestanku puhanja osciliraju oko centra ravnoteže velikom frekvencijom. Visoke biljke imaju veću tromost i rade veće oscilacije oko točke konvergencije vjetra i otpora biljke prilikom naleta vjetra. Kad vjetar prestane, naprave manje oscilacija oko centra ravnoteže prije nego vjetar ponovno počne.

Slika 5.2: Katično ponasanje niske biljke prilikom vjetra u valovima

Kad je simulacija vjetra realistična (takva da vjetar dolazi i prolazi u valovima - postepeno raste u snazi, i nakon toga postepeno opada u snazi) također imamo realističnu simulaciju. Visoke biljke održavaju smjer i neprestano osciliraju prema središtu između točaka centra ravnoteže otpora biljke, i konvergentne točke sile vjetra, ravnoteže i unutarnjeg otpora biljke. Niske biljke kod ovakvog vjetra djeluju kaotično i osciliraju velikom frekvencijom i amplitudom, sa središtem oscilacije koji vidno varira između konvergentne točke i centra ravnoteže biljke. Razlika u ponašanju je vidljiva na slikama 5.2 i 5.3



Slika 5.3: Stabilno ponasanje visoke biljke prilikom vjetra u valovima

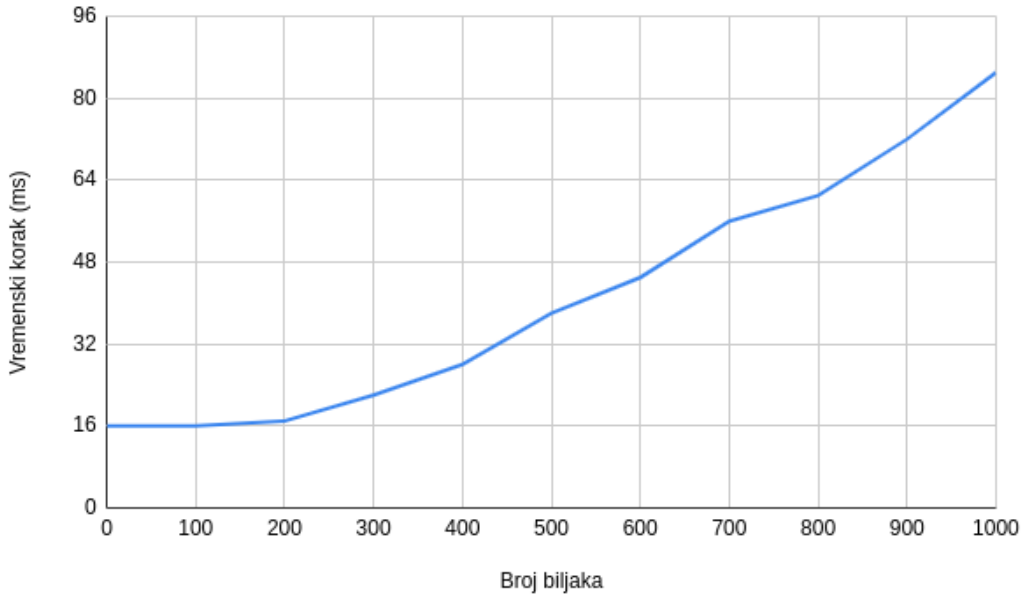
5.2.1. Poboljšanje realističnosti simulacije

Još realističnija simulacija se može postići dodavanjem većeg broja zglobova u fizički kostur biljke. Dodavanjem većeg broja zglobova dobili bi mogućnost simulacije pregiba u stabljci biljke. U prirodi kod jednostavnih biljaka ovaj fenomen nije uvijek jasno vidljiv, ali možemo ga uočiti kad je vjetar u rezonantnoj frekvenciji sa stabljkom biljke.

Dodatnu kontrolu možemo postići dodavanjem težina pojedinim zglobovima. Tako da na neke dijelove kostura vjetar ima jači utjecaj nego na druge. Primjer takve biljke je zvončić, kod kojeg je utjecaj vjetra puno jasnije vidljiv na cvijetu nego na stabljci biljke.

5.3. Brzina izvođenja

Brzina izvođenja linearno opada s brojem simuliranih biljaka. Najveći utjecaj na brzinu izvođenja ima fizička simulacija. Generiranje biljaka se odvija na samom početku i nema nikakvog utjecaja na brzinu izvođenja nakon početnog perioda generiranja. Iscrtavanje je zbog jednostavnosti prikaza vrlo jeftino i usporava izvođenje tek na velikom broju biljaka ili na vrlo visokoj razini detalja. Graf trajanja vremenkog koraka vidljiv je na 5.4.



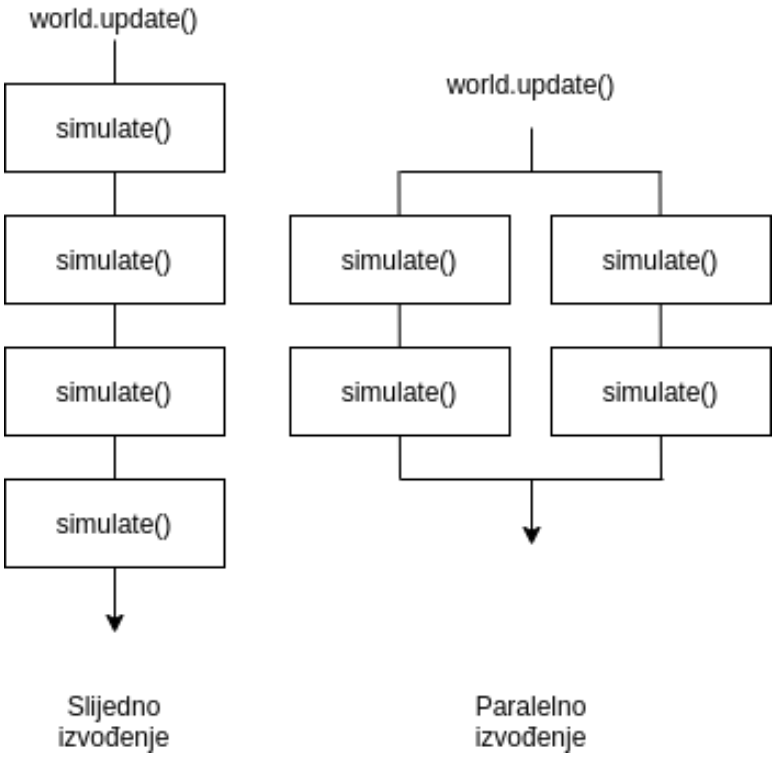
Slika 5.4: Trajanje vremenskog koraka u ovisnosti o broju biljaka u simulaciji

5.3.1. Pristupi poboljšanju brzine izvođenja

Fizička simulacija ima najveći utjecaj na brzinu izvođenja pa su prijedlozi za ubrzanje programskog rješenja fokusirani na taj dio.

Paralelno izvođenje fizičke simulacije

Trenutna programska implementacija u obzir uzima i kolizije samih biljaka. Fizička simulacija se odvija slijedno za svaku od biljaka i eventualni dodir nakon simulacije neke od narednih biljaka moće imati utjecaj na onu prvu. Ako ne marimo za interakciju između samih biljaka i odlučimo je zanemariti, jednostavan način za ubrzanje fizičke simulacije je paralelizam. Na 5.5 je prikazan dijagram slijednog i paralelnog izvođenja (na dvije dretve) i vidljivo je da paralelni primjer završava u dva koraka dok slijedni algoritam završava u jednom koraku.



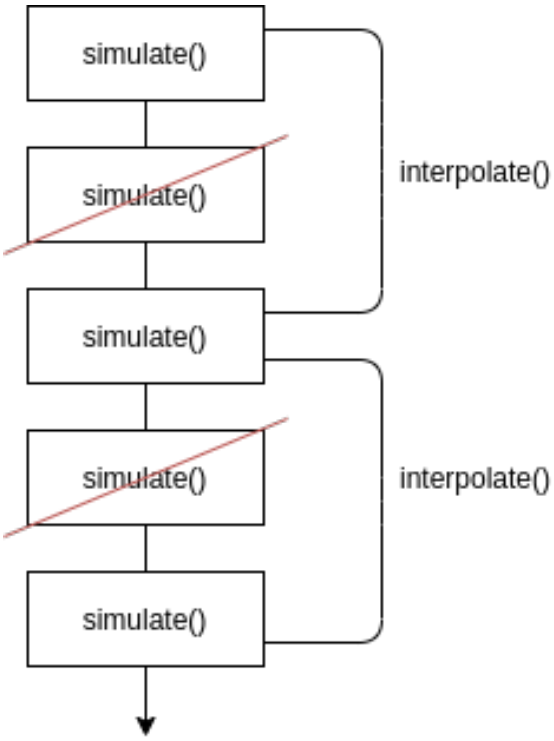
Slika 5.5: Dijagram slijednog i paralelnog izvođenja

Na ovaj način svaka biljka (ili grupa biljaka) može imati svoj fizički procesor koji će se brinuti o njezinoj simulaciji i cijeli proces se završava u manje koraka.

Smanjenje rezolucije simulacije i zaglađivanje rezultata

Brzinu izvođenja možemo i povećati na način da ne ažuriramo fizičku simulaciju biljke u svakom koraku. Kod ovakvog pristupa stanje svake biljke izračunavamo nakon nekoliko koraka umjesto na svakom, a rezultate u međukoracima zagladimo. U primjeru 5.6 vidimo slijed simulacije jedne biljke kroz vremenske korake. Prilikom simulacije trebamo osigurati da je vremenski razmak za koji simuliramo točno onoliko koraka koliko će trajati do sljedeće simulacije i to uzrokuje smanjenje reyolucije simulacije. Ako bi vremenski korak simulacije ostao jednak kao da simuliramo biljku u svakom simulacija bi se odvijala usporeno.

Bitno je ostvariti da se nikad istovremeno ne simuliraju i zaglađuju svi modeli nego da je postupak naizmjeničan. Za primjer sa dvije biljke, dok se jedna biljka simulira druga se zaglađuje i obratno.



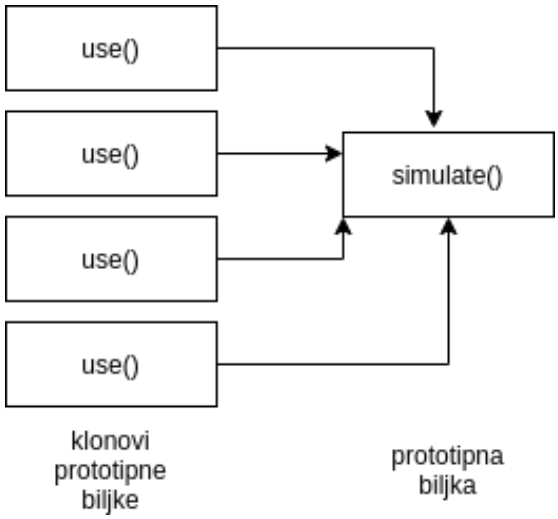
Slika 5.6: Dijagram simulacije sa zaglađivanjem

Problem kod ovakvog pristupa je kašnjenje prikaza nad stvarnim stanjem fizičke simulacije. Fizička simulacija je uvijek barem jedan korak ispred prikaza. Ovo može uzrokovati naizgledno podrhtavanje ukoliko želimo zadržati simulaciju interakcija između biljaka, a biljka se pomaknula nakon što je dobivena zadnja točka za interpola-

ciju. Ako ne marimo za interakciju između biljaka, ovaj problem možemo zanemariti. Iako će animacija i dalje kasniti za stvarnim fizičkim stanjem, promatraču to neće biti vidljivo.

Izračunavanje sličnosti modela i grupiranje izračuna

Ukoliko nam nije bitna interakcija između biljaka i želimo drastično smanjiti utjecaj fizičke simulacije na brzinu izvođenja možemo izdvojiti prototipne biljke iz simulacije i simulirati samo njih. Sve ostale biljke, ovisno o sličnosti će koristiti podatke simulacije te biljke kao svoje. Primjer takvog sustava vidljiv je na slici 5.7



Slika 5.7: Dijagram simulacije sa zaglađivanjem

Na ovaj način potrebno je simulirati samo nekoliko biljaka umjesto svih ali ovakav pristup će rezultirati time da simulacija izgleda nerealistično jer će sličnosti između ponašanja biljaka biti identične u isto vrijeme i na istim uvjetima što narušava dojam realizma. Potencijalno rješenje ovog problema je da kod nekih biljaka unesemo kašnjenje nad prototipnom biljkom. Na taj način smo eliminirali problem dojma da se sve ponavlja, ali smo uveli problem da simulacija ne djeluje toliko responzivno.

6. Zaključak

POPIS SLIKA

POPIS ISJEČAKA KODA

Proceduralno generiranje trave i niskog raslinja

Sažetak

Sažetak na hrvatskom jeziku.

Ključne riječi: Ključne riječi, odvojene zarezima.

Procedural generation of grass and low vegetation

Abstract

Abstract.

Keywords: Keywords.