SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 210

Modeliranje i simulacija mekih objekata sustavom masa i opruga

Davorin-Gordan Keserica

Zagreb, lipanj 2011

Tablica sadržaja

Popis slika

1 Uvod

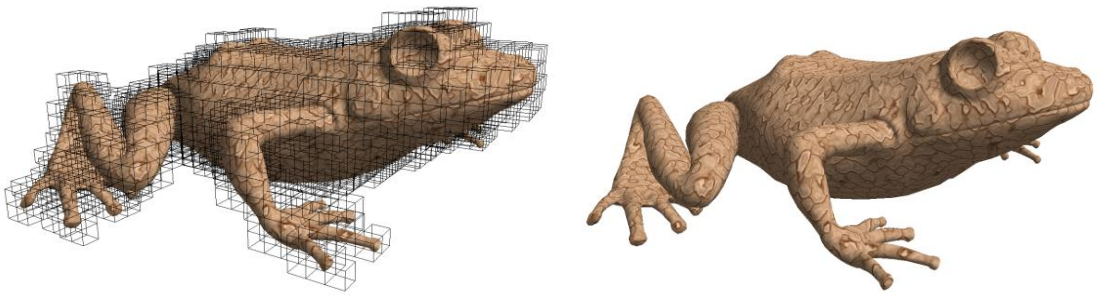
Jedna od zadaća računalne grafike jest prikazivanje i simulacija dinamičkih procesa poput modela dima, fluida, tkanine te mekih objekata o kojemu je riječ u ovome radu. Kako bi se preslikao proces stvaranog svijeta i prikazao pomoću računala, potrebno je obavljati simulaciju fizikalnog ponašanja nad tim dinamičkim procesom kreirajući tako njegov model. Simulacijom se preslikava realna domena svijeta u diskretnu domenu računala tako da ponašanje simuliranog modela bude što realnije.

Ova tehnologija je još u razvitku te će u budućnosti omogućiti prikaz složenih fizikalnih pojava u stvarnom vremenu. Razni dinamički procesi se na ovaj način mogu simulirati te tako koristiti u razne svrhe. Primarna grana njihovog korištenja je istraživanje i proučavanje dinamičkih pojava. Njima se također može povećati realnost prikaza unutar računalnih igara i filmske industrije.

Ovaj rad opisuje način kreiranja i prikaza mekih objekata, simulacije njihovog ponašanja uz mogućnost dinamičkog mijenjanja parametara koji ju definiraju. Simulacija mekih objekata ostvarena je u realnom vremenu uz mogućnost interakcije kroz djelovanje vanjskih sila na model mekog objekta. Koristi se OpenGL standard te je sama izvedba mekog objekta razrađena u programskom jeziku C++.

2 Model mekog objekta

Meki objekti *(eng. soft body)* predstavljaju računalne modele objekata koji za razliku od krutih objekata *(eng. rigid body)* omogućavaju deformiranje površine pod utjecajima vanjskih i unutarnjih sila. Deformacijom volumena objekta i njegove površine stvara se dojam realističnosti simuliranog objekta i približavamo se njegovom ponašanju u stvarnom svijetu. U ovom radu bit će predstavljen način kreiranja modela mekog objekta na temelju računalnih modela krutih objekata. Kruti objekti su sve prisutni u računalnoj grafici ali nedostaje im doza realnog ponašanja koja se pomoću mekih objekata može postići. Na slici 2.1 prikazan je model mekog objekta. S lijeve strane prikazana je njegova fizikalna struktura, a s desne strane sam model mekog objekta.



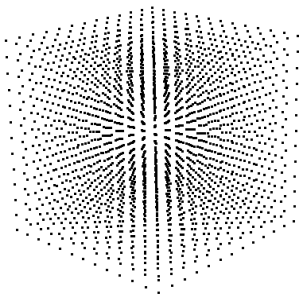
Slika 2.1 Model mekog objekta.

Da bi se kreirao model mekog objekta potrebno je proći kroz više faza. Na samom početku potrebno je odrediti fizikalni model koji će definirati fizikalnu pozadinu simulacije mekog objekta. Potrebno je zatim taj model strukturirati i modelirati kako bi se kreirala fizikalna struktura koja opisuje oblik objekta. Na slici gore se vidi kreirana i modelirana fizikalna struktura koja opisuje objekt žabe. Zatim je potrebno povezati kreiranu fizikalnu strukturu s modelom objekta kako bi deformacija fizikalne strukture, pod utjecajem izračuna jednadžbi stanja fizikalnog modela, utjecala i na deformaciju samog objekta. Svaka od ovih faza opisana je u nastavku. Nakon njihove provedbe bit će kreirana fizikalna struktura koja je povezana s modelom objekta i time tvori model mekog objekta. Nakon kreiranja modela mekog objekta prelazi se na samu reprezentaciju i prikaz kroz simulaciju njegovog ponašanja.

3 Fizikalni model

Da bi se izradila simulacija prvobitno je potreban fizikalni model koji ju opisuje. Objekti u stvarnom svijetu su sastavljeni od molekula povezani zajedno silama čineći na taj način relativno krute tvorevine. Primjenom vanjskih sila na objekte možemo vidjeti njihovu deformaciju, a konačno i njihovu mekoću. Svaki objekt je sastavljen od različite gustoće materijala pa su time željezni objekti kruti i potrebna je velika sila kako bi se objekt načinjen od željeza deformirao, a nakon njene deformacije on se neće vratiti u početno stanje. Za razliku od tvrdih objekata, mekši objekti se pod malim silama deformiraju i nakon prestanka djelovanja sile vraćaju se u svoje približno početno stanje. Svaki objekt u prirodi je po svojoj definiciji mekan jer ga je moguće deformirati primjenom sila.

Za reprezentaciju mekog objekta pomoću računala potrebna je njegova aproksimacija kako bi se smanjila kompleksnost. Radi toga prelazi se na diskretizaciju mekog objekta i kreiranja njegovog modela koji će na što vjerniji način simulirati njegovo stvarno ponašanje. Kako je nemoguće uzeti u obzir sve unutarnje sile u nekom objektu prelazimo na njegovu volumnu diskretizaciju. Na slici 3.1 možemo vidjeti primjer kako se objekt kocke diskretizira na raster čestica koje ga opisuju. Interakcijom tih čestica definira se fizikalni model. Ovim pristupom smanjuje se kompleksnost, veličina i sama složenost modela mekog objekta uz očuvanje njezine strukture i dojma realističnosti.

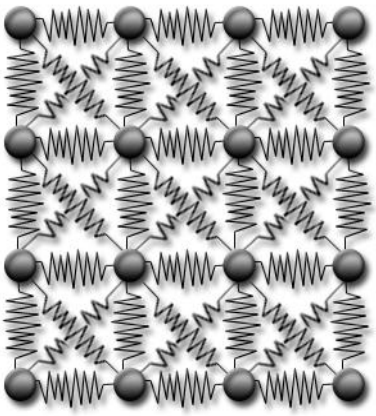


Slika 3.1 Reprezentacija kocke diskretizacijom na čestice.

Svaka diskretizirana čestica međudjeluje sa susjednim česticama, a njihova povezanost opisna je pomoću modela čestica povezanih oprugama. Izračun stanja fizikalnog modela provodi se nad svim česticama pomoću numeričke integracije.

3.1 Model čestica povezanih oprugama

Fizikalni model čestica povezanih oprugama prikazan je na slici 3.2. Svaka čestica povezana je okolnim česticama pomoću opruga čineći tako sustav čestica povezanih oprugama. Na pojedinu česticu djeluju sile okolnih opruga i vlastita sila teže.



Slika 3.2 Prikaz sustava čestica povezanih oprugama.

Prikazani sustav može se kreirati tako da opisuje unutarnji volumen objekta. Način povezivanja čestica unutar volumena bit će objašnjeno prilikom kreiranja fizikalne strukture mekog objekta. Povezanost čestica direktno utječe na čvrstoću i stabilnost kreiranog sustava, a u konačnici i na sam model mekog objekta.

Ponašanje čestice i njezino simuliranje je česta radnja u računalnoj grafici prilikom simuliranja dinamičkih procesa svijeta. Stvaranjem mreža čestica povezanih oprugama i implementiranjem fizikalno temeljenog ponašanja u pozadini, dobivamo složene dinamičke procese. Stoga, najvažniji dio mekog objekta jest njezina čestica (Slika 3.3) koja ima svoju masu, položaj u prostoru, brzinu i akceleraciju.

Slika 3.3 Reprezentacija čestice.

Položaj: $x(t)$

Brzina: $v(t) = \frac{dx(t)}{dt}$

Akceleracija: $a(t) = \frac{dv(t)}{dt} = \frac{d^2x(t)}{dt^2}$,

kada uzmemo u obzir konstantu akceleraciju tada dobivamo sljedeće izraze:

Brzina: $v(t) = \int a\, dt = v_0 + a_p t$

Položaj: $x(t) = \int v\, dt = x_0 + v_0 t + \frac{a_p t^2}{2}$

Slika 3.4 Prikaz djelovanja sila na česticu.

Sila opruge kojom su čestice povezane izračunava se pomoću Hookovog zakona. Hookov zakon elastičnosti je aproksimacija koja tvrdi da ako neku oprugu rastegnemo za udaljenost Δx od njenog položaja mirovanja, rezultirajuća sila Fo, koju stvara opruga, je proporcionalna udaljenosti Δx i konstanti opruge k, a njen smjer je suprotan od smjera pomaka:

$$\overrightarrow{F_0} = -k * \overrightarrow{\Delta x}\,,$$

gdje k predstavlja konstantu opruge, a $\overrightarrow{\Delta x}$ je razlika osnovne duljine opruge l_0 sa njezinom trenutnom l_i, odnosno:

$$\overrightarrow{F_0} = -k * \overrightarrow{(l_0 - l_i)}$$

Kada bi se implementirao i simulirao samo ovakav model opruga i čestica stvorilo bi se titranje koje bi osciliralo u beskonačnost jer nema gubitka energije u sustavu. Radi toga u stvarnom svijetu postoji sila prigušenja F_b koja prigušuje osciliranje ovakvoga modela.

$$\overrightarrow{F_b} = -b * \vec{v},$$

gdje b predstavlja faktor prigušenja, a v brzinu promjene položaja opruge. Brzinu dobivamo skalarnom projekcijom vektora brzine na jedinični vektor smjera opruge. Uvođenjem ove sile nad modelom postiže se prigušenje titranja i njegova stabilnost.

Dakle nad jednom česticom mase m povezanom oprugom djeluje ukupna sila F_{uk} te ju možemo prikazati formulom:

Gravitacijska sila: $\overrightarrow{F_g} = m * \vec{g}$

Sila opruge: $\overrightarrow{F_0} = -k * \overrightarrow{\Delta x}$

Sila prigušenja: $\overrightarrow{F_b} = -b * \vec{v}$

Ukupna sila: $\overrightarrow{F_{uk}} = \overrightarrow{F_g} + \overrightarrow{F_0} + \overrightarrow{F_b}$

$$\overrightarrow{F_{uk}} = m * \vec{g} - k * \overrightarrow{\Delta x} - b * \vec{v}$$

Dani sustav čestica povezanih oprugama možemo prikazati sljedećom formulom, tako da računamo ukupnu silu za svaku česticu j, koju okružuje n opruga.

$$\overrightarrow{F_{j,uk}} = m * \vec{g} - \sum_{i=1}^{n} (k * \overrightarrow{\Delta x_i} + b * \overrightarrow{v_i})$$

Kada izračunamo silu koja djeluje na pojedinu česticu tada preko dvostruke integracije dobivamo akceleraciju a_i pomoću koje određujemo poziciju čestice. Kako je limitirano znanje o akceleraciji čestice tijekom cijele simulacije na trenutnu i prethodnu vrijednost, tada prelazimo na model numeričke integracije kako bi odredili približnu vrijednost sljedeće pozicije čestice.

$$x_i = \iint a_i dt\, dt$$

3.2 Metoda integracije

Kako je model koji izgrađujemo diskretan, odnosno vrijeme unutar simulacije nije kontinuirano, potrebno je uvesti model numeričke integracije. U ovom radu koristi se vrijeme osvježavanja izračuna postavljenog fizikalnog modela pod frekvencijom od 100Hz i većom (ovisno o brzini i snazi samog računala) što je dovoljno za glatku predodžbu simuliranja modela mekog objekta. Model mekog objekta se iscrtava brzinom od 30Hz koja je dovoljna frekvencija kojom ljudsko oko može vidjeti glatku animaciju tj. simulaciju. Duljim vremenom izračuna fizikalnog modela, odnosno smanjenjem vremena otipkavanja stvarnog svijeta, povećavamo stabilnost simuliranog modela, ali time djelujemo i na brzinu izvođenja cijelog programa koja se tada povećava. Ako taj broj prijeđe ispod granice od 30Hz (kojime osvježavamo model mekog objekta za prikaz) tada više ne možemo vršiti simulaciju u realnom vremenu jer se izračun fizikalno temeljenog ponašanja koji je u pozadini ne stigne u potpunosti izračunati. Predstavljena su dva modela integracije, njihove prednosti i mane.

3.2.1 Verlet metoda numeričke integracije

Verlet metoda numeričke integracije omogućava efikasnu aproksimaciju integracije. Njenim uvođenjem potrebno je pamtiti prošlu i sadašnju poziciju pojedine čestice. Ovoj metodi nije potrebna informacija o brzini čestice te time smanjujemo dodatno izračunavanje njene vrijednosti. Sljedeći izraz opisuje dobivanje nove pozicije čestice kojoj je za izračun potrebna akceleracija a_t, prethodna pozicija $x_{t-\Delta t}$ i vrijeme uzrokovanja Δt.

$$x_{t+\Delta t} = (2 - f)x_t - (1 - f)x_{t-\Delta t} + a_t(\Delta t)^2,$$

gdje f predstavlja djelić brzine koji se izgubi radi trenja, on se unosi kao broj između (0-1), u ovom radu za f je isprobana vrijednost od 0,01 kojom je se uspješno stabilizira simulacija. Bez parametra f sustav opruga ne bi gubio energiju te bi ušao u harmonično titranje. Ovaj parametar se koristi umjesto sile prigušenja nad oprugom i samim time stabilizira model mekog objekta. Verlet metoda djeluje globalnom silom prigušenja preko parametra f te tako umanjuje moć prikaza malih, za potrebe simulacije, utjecajnih sila.

Radi toga je potrebno smanjivati vrijeme otipkavanja Δt kako bi i te sile došle do izražaja. Smanjivanjem Δt simulacija ima popratni efekt radi kojega je potrebna veća količina osvježavanja fizikalnih parametara, što na kraju dovodi do njene slabe iskoristivosti u realnom vremenu.

3.2.2 Eulerova metoda integracije

Klasičan pristup je Eulerova metoda numeričke integracije, a ona glasi:

Izračun brzine: $\dot{v} = a \qquad v_{n+1} = v_n + a_n \Delta t$

Izračun pozicije: $\dot{x} = v \qquad x_{n+1} = x_n + v_{n+1} \Delta t,$

gdje je Δt vrijeme između dvije iteracije izračunavanja fizikalnog ponašanja čestice, odnosno ako se izračun izvodi pod frekvencijom od 100Hz, tada Δt iznosi 10 milisekundi.

Za implementaciju Eulerove metode potrebno je pamtiti brzinu pojedine čestice i njezinu poziciju. Ova metoda nema u sebi ugrađenu stabilizaciju kao verlet metoda. Kako ne postoji unutarnji mehanizma stabilizacije moguća je veća fleksibilnost i mogućnost djelovanja slabih sila koje neće biti prigušene kao kod verlet metoda numeričke integracije. Za prigušenje je potrebno izračunati brzinu pojedine čestice te implementirati prigušenje nad oprugama samog modela u smjeru obrnutom od brzine gibanja čestice (sila prigušenja). Povećavanjem dodatnih izračuna unutar Eulerove metode je jeftinije i isplativije od veće količine osvježavanja fizikalnih proračuna unutar verlet metode. Za isti broj čestica verlet metodom, koja je naizgled praktičnija varijanta, dobivamo i do par puta manju isplativost od Eulerove metode integracije.

3.3 Stabilizacija fizikalnog modela opruga

Jedan od najvećih problema koje se javlja unutar simulacije modela rada čestica jest njezina stabilizacija, a razlog tomu jest diskretizacija informacija i vremena stvarnog svijeta. Ovom problemu prilazi se na razne načine ali sve se one svode na uvođenje gubitka energije u sustav kojim osiguravamo konačno prigušenje.

Kada koristimo verlet metodu integracije nije potrebna dodatna stabilizacija kako bismo izbjegli osciliranje fizikalnog modela mekih objekata. Verlet metoda, na svaku iduću iteraciju izračuna fizikalnih parametara, umanjuje doprinos te iteracije pa time zapravo imitira globalnu silu prigušenja koja se može gledati kao sila otpora zraka. Primjenom Eulerove metode integracije moramo koristiti silu prigušenja koje djeluje nad domenom opruga. Prigušivanjem oscilacija opruge dobivamo stabilizaciju samog modela.

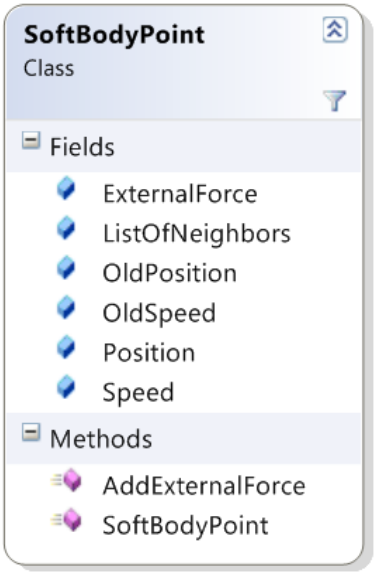
Povećavanjem broja čestica unutar simulacije fizikalno temeljenog modela dolazi do predugog vremena izračuna. Samim time vrijeme otipkavanja izračuna Δt prelazi granicu od 10 milisekundi unutar koje je simulacija stabilna, dulje vrijeme otipkavanja unosi nestabilnost unutar samog izračuna. Ovaj problem se može riješiti na više načina, jedan od način jest da se ograniči maksimalna sila ili brzina koja može djelovati na pojedinu česticu. Ograničavanjem sila ili brzina dovodimo model simulacije do nekonzistentnosti pa tako i do ne realističnog ponašanja. Drugi način koji je i implementiran u ovom radu jest ograničavanje brzine otipkavanja, pa tako vrijeme otipkavanja može biti samo manje ili jednako 10 milisekundi, a u situacijama da vrijeme izračuna fizikalnih parametara prijeđe ovu granicu vrijeme otipkavanja ostaje fiksirano na toj definiranoj gornjoj granici. Na taj način je unesena sigurnost od nepredviđenih osciliranja unutar izračuna kojom bi se mogla „razletiti“ simulacija mekog objekta.

4 Definiranje fizikalna strukture

Nakon što je određen fizikalni model koji opisuje ponašanje i međudjelovanje čestica potrebno je definirati fizikalnu strukturu. Fizikalnom strukturom definiramo oblik sustava čestica povezanih oprugama koji će opisivati unutarnju strukturu mekih objekata. Čestica je u računalu predstavljena kao objekt točke fizikalnog modela te će se nad njome vršiti sve daljnje operacije. Točka fizikalnog modela predstavlja osnovni element fizikalne strukture, a kako bi se kreirala i modelirala fizikalna struktura volumena objekta potrebno ih je grupirati u građevne jedinice volumena.

4.1 Točka fizikalnog modela

Točka fizikalnog modela mekog objekta predstavljena je klasom `SoftBodyPoint` (Slika 4.1). Svaki objekt tipa `SoftBodyPoint` u sebi sadrži poziciju `Position`, brzinu `Speed` te njihove vrijednosti u prethodnoj iteraciji izračuna. Masa pojedine točke definirana je globalno za svaku točku pa je masa jednoliko unificirana.



Slika 4.1 Klasa točke fizikalnog modela, njezini atributi i metode.

Nad točkama mekog objekta se obavlja izračun jednadžbi fizikalnog modela. Svaka točka ima definiranu listu susjeda `ListOfNeighbors` s kojima je povezana oprugama.

Susjedi su definirani unutar klase u kojoj je zapisana njihova inicijalna udaljenost i pokazivač na susjednu točku fizikalnog modela. Također je moguće definirati i različite koeficijente opruga nad susjedstvom unutar klase susjedstva, no u ovom radu se koriste jednake konstante za sve kreirane opruge.

Točka fizikalnog modela ima definiranu logičku varijablu `ExternalForce` koja putem metode `AddExternalForce` omogućava dodavanje vanjske sile na pojedinu točku fizikalnog modela. Način djelovanja vanjske sile biti će detaljnije objašnjeno u nastavku rada.

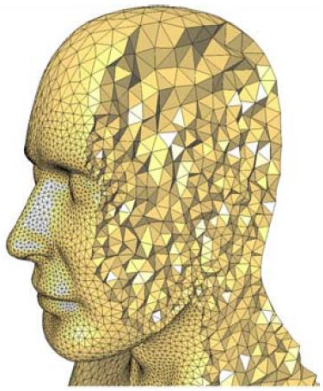
Fizikalna struktura je sagrađena od točaka fizikalnog modela, a kako bi ju izgradili potrebno je načiniti osnovnu građevnu jedinicu volumena. Osnovna građevna jedinica volumena sastavljena je od više točaka fizikalnog modela.

4.2 Građevna jedinica volumena

Osnovna građevna jedinca volumen grupira više točaka fizikalnog modela u strukturu kojom se ispunjava fizikalna struktura unutarnjeg volumena mekog objekta. Građevni volumen može biti bilo koji volumen koji svojim nadograđivanjem može činiti prostor. Radi toga sferama ne možemo ispuniti prostor jer će se njihovim nadograđivanjem kreirati šupljine unutar većeg volumena. Najčešći građevne jedinice kojima se može ispuniti prostor su tetraedri i kocke.

4.2.1 Tetraedar kao građevna jedinica volumena

Pravilnim tetraedrima kao građevnim element ne može se ispuniti prostor [1]. Razlog tome je njihov prostorni unutarnji kut od ~72 stupnjeva koji onemogućava njihovo kombiniranje i izgradnju volumena. Tetraedrima se može popuniti volumen na način da se lijepe različite veličine i oblici tetraedara i na taj način popunjavaju prostor, pa tako i volumen objekta kao što je prikazano na slici 4.2. Trokuti definiraju stranicu tetraedara, pa se kreiranje volumena nekog objekta vrši od njegove površine prema unutrašnjosti.



Slika 4.2 Volumen popunjen tetraedrima različitih veličina.

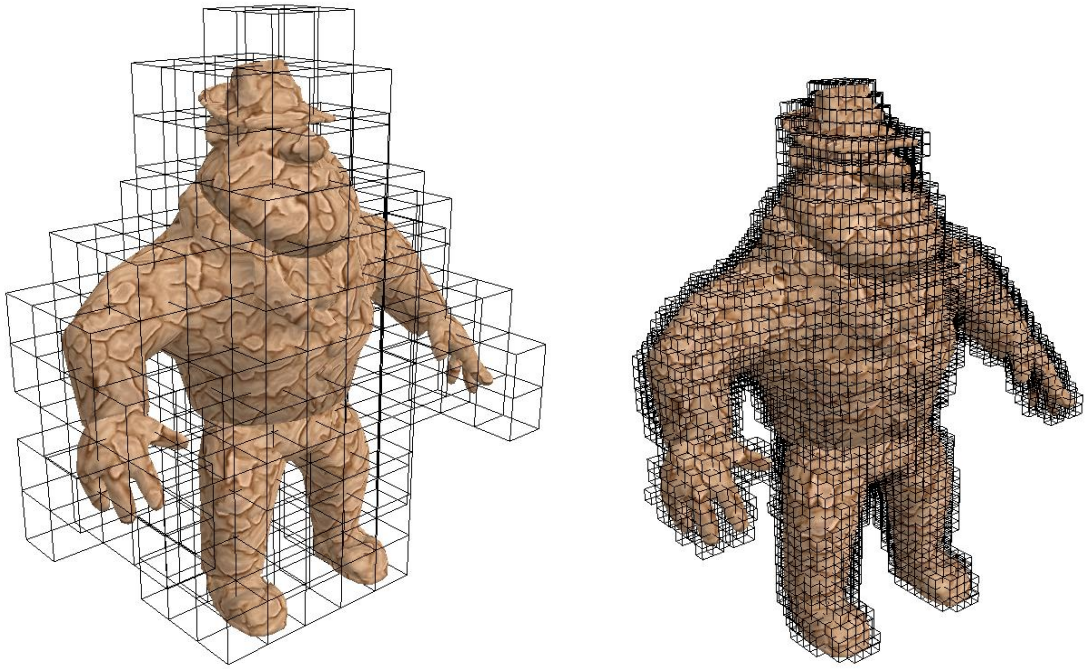
Problem kreiranja ovako popunjenog objekta je u tome što se lijepe različite veličine i oblici tetraedara kojima se pokušava optimirati njihovo popunjavanje. Stabilnost ovakvog modela ovisi o najlošije kreiranom tetraedru. Najlošiji tetraedar u smislu stabilnosti simulacije fizikalnog ponašanja predstavljaju oni tetraedri koji imaju najmanji unutarnji kut. Definiranje volumena na ovaj način zahtjeva prolaženje kroz niz koraka optimiranja, zamjene i generiranja novih i prikladnijih tetraedara kojima se u konačnici pokušava osigurati što veća stabilnost.

Na ovaj način popunjen volumen se koristi u strojarstvu prilikom metode konačnih elemenata, a za potrebe kreiranja simulacije mekog objekta u realnom vremenu predstavlja previše kompleksan model i koji je gotovo nemoguće simulirati u stvarnom vremenu.

4.2.2 Kocka kao građevna jedinica volumena

Kocka predstavlja savršenu građevnu jedinicu volumena. Prostornim nadovezivanjem kocaka kreiramo veću strukturu kojom popunjavamo prostor. Kockama se na taj način mogu popuniti različiti oblici volumena pa tako i model nekog objekta. Prilikom njihovog generiranja ne moramo optimirati oblike jer je kocka sama po sebi jedinstvena. Veličina građevne jedinice kocke, odnosno njezine stranice, definira raster pod kojim ćemo ispuniti i diskretizirati prostor. Kada bi se stranica kocke približavala nuli tada bi prostor bio maksimalno popunjen.

Na slici 4.3. prikazan je model objekta ispunjen kockama kao građevnim jedinicama volumena. Na lijevoj strani slike odabrana je veća stranica kocke, a sa desne strane manja stranica. Stranicom kocke definira se raster kocka kojime ispunjavamo volumen objekta. Što je stranica manja to je volumen objekta egzaktnije ispunjen.



Slika 4.3 Volumen popunjen kockama kako građevnim jedinicama.

Tetraedri kao građevne jedinice omogućavaju savršeno ispunjavanje volumena objekta, dok se kockama aproksimira njegova površina. Vrijeme popunjavanja prostora je znatno brže korištenjem kocaka, a stabilnost ovisi samo o veličini rastera građevnih kocaka. Za potrebe kreiranja mekog objekta odabrana je kocka jer je s njome moguće lako baratanje, kreiranje i popunjavanje prostora.

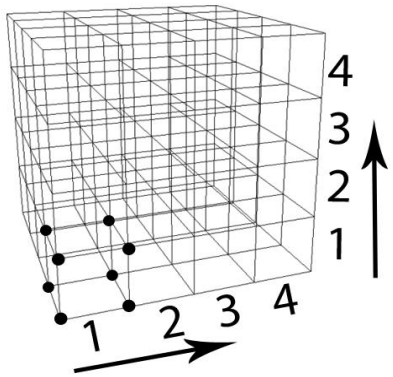
5 Izgradnja fizikalne strukture

Nakon definiranja građevne jedinice volumena moguće je sagraditi fizikalnu strukturu. Fizikalna struktura se gradi unutar volumena jedinične kocke kako bi se kreirao generički sustav koji se tada može modelirati prema oblicima različitih objekata. Izgradnjom fizikalne strukture stvara se povezanost između točaka fizikalnog modela i građevnih jedinica volumena.

Točke fizikalnog modela unutar izgrađene fizikalne strukture potrebno je povezati sa oprugama. Povezivanjem stvaramo fizikalno strukturiran sustav čestica povezanih oprugama. Povezivanje se vrši nakon modeliranja fizikalne strukture prema učitanom objektu kako bi smanjili kreiranje nepotrebnih poveznica i ubrzali postupak generiranja modela mekog objekta.

5.1 Popunjavanje fizikalne strukture kockama

Za izgradnju fizikalne strukture potrebno je ispuniti volumen jedinične kocke s odabranim građevnim jedinicama (kockama). Prilikom popunjavanja volumena jedinične kocke odabire se raster `gridSize`, odnosno veličina stranice građevne jedinice. Ako je veličina građevne jedinice 1 tada će se volumen jedinične kocke popuniti samo jednom kockom. Na slici 5.1 prikazana je rasterizacija kockama veličine 0.25, čime dobivamo 4*4*4 građevnih kocaka unutra volumena jedinične kocke. Dakle što je manja veličina rasterizacije to će prostor biti popunjen sa sve više kocaka.

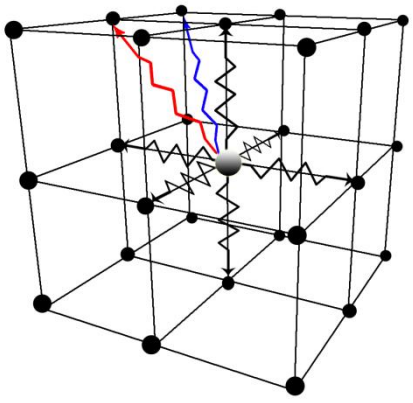


Slika 5.1 Rasterizirana jedinična kocka na građevne jedinice (kocke).

Prilikom kreiranja rastera kocka, kreira se i raster točaka fizikalnog modela. Svaka kocka sadrži referencu na 8 točaka fizikalnog modela koje definiraju njene vrhove. Povezivanje kocke s točkama fizikalnog modela postiže se njihovim indeksiranjem unutar kreiranog rastera i kopiranjem reference njihovog objekta.

5.2 Povezivanje fizikalne strukture oprugama

Ovako kreiranu strukturu kocaka s referenciranim točkama fizikalnog modela potrebno je povezati sa strukturom opruga. Svakoj točki fizikalnog modela potrebno je definirati njezino susjedstvo, odnosno točke s kojime je ona povezana. Kako su svi objekti kreirani u rasteru moguće ih je lako indeksirati i na jednostavan način povezati u razne strukture. Odabrana je maksimalna povezanost sa svim prostornim susjedima točke koji se mogu vidjeti na slici 5.2 prikazane kružnicama crne boje. Na taj način svaka točka fizikalnog modela povezana je sa 26 susjednih točaka, od toga 9 na gornjoj prostornoj razini, 9 na donjoj i 8 na razini unutar koje se nalazi. Na slici je prikazano 6 opruga crnom bojom koje predstavljaju strukturne opruge. Kako bi struktura bila čvršća koriste se dijagonalne opruge. Postoje dva tipa dijagonalnih opruga, prostorne dijagonale volumena (opruga prikazana plavom bojom) i dijagonale stanica (opruga prikazana crvenom bojom).



Slika 5.2 Povezanost točaka fizikalnog modela s oprugama.

Ovako definirana i povezana struktura spremljena je u pripadajuće liste točaka fizikalnog modela i liste građevnih kocaka s referencama na točke fizikalnog modela. Njome možemo simulirati meke objekte koji izgledaju kao kocka. Kako mi želimo kreirati meki objekt ovisno o izgledu modela krutog objekta potrebno je modelirati dobivenu fizikalnu strukturu prema njegovom obliku volumena.

6 Modeliranje fizikalne strukture

Dobivena fizikalna struktura je u obliku kocke rasteritzirana manjim kockama njezinim građevnim jedinicama. Potrebno je modelirati fizikalnu strukturu kako bi ona odgovarala modelu odabranog objekta. Prva faza u ovom postupku je učitavanje željenog modela krutog objekta i njegovog normiranja na jediničnu kocku kako bi ga prilagodili kreiranoj fizikalnoj strukturi. Druga faza je određivanje kocaka unutar kreirane fizikalne strukture koje pripadaju unutrašnjosti učitanog objekta. Sve ostale kocke i pripadajuće točke fizikalnog modela se mogu obrisati iz sustava jer nam one više nisu potrebne i predstavljaju vanjski volumen objekta.

Određivanje unutarnjeg volumena objekta samo po sebi djeluje intuitivno, no računalni modela objekta ima zapis samo o trokutima i vrhovima koji ga čine. Potrebno je na neki način iz te strukture odrediti unutarnji volumen. U ovom radu su isprobane dvije metode i prikazane njihove prednosti i mane. Prva metoda je napravljena pomoću praćenja zrake. Druga metoda koristi logičko određivanje volumena pomoću rasteriziranog prostora.

6.1 Učitavanje i prilagođavanje objekta

Prilikom učitavanja objekta iz datoteke spremaju se njegovi trokuti i vrhovi u pripadajuće liste. Svaki objekt prilikom njegovog modeliranja je napravljen u svom koordinatnom sustavu i u različitim veličinama. Potrebno je translatirati i skalirati objekt u jediničnu kocku kako bi se napravio generički sustav za povezivanje objekata s fizikalnom strukturom.

6.1.1 Izračun normala objekta

Prilikom učitavanja modela, normale pojedinog vrha mogu biti zadane, a ako nisu potrebno ih je izračunati. Normale poligona statičkog objekta unutar računalne grafike potrebno je izračunati samo jedanput i to na samome početku. Da bi odredili normalu vrha potrebno je poznavati sve normale poligona koji ga okružuju. Za svaki poligon se prvo

izračuna normala njegove površine. Sumirajući normale okružnih poligona pojedinog vrha i njihovim normiranjem dobivamo vektor normale dotičnog vrha.

6.2 Pronalaženje unutarnjeg volumena algoritmom praćenja zrake

Zadatak je pronaći koje se građevne kocke nalaze unutar objekta, a koje izvan. Najjednostavnija metoda za pronalazak unutarnjeg volumena nekog zatvorenog objekta je metoda praćenja zrake *(eng. ray-casting)*.

6.2.1 Algotiram praćenja zrake (Ray-casting)

Princip ove metode je taj da se odabere jedna zraka u prostoru i pomoću nje pokuša odrediti da li je neka točka unutar volumena samog objekta. Točka za koju se provjerava pripadnost je u ovom slučaju središte građevne kocke. Za odabranu zraku i točku u prostoru potrebno je preispitati njezin presjek sa svim trokutima modela objekta. Za svaki trokut koji je u presjeku sa zrakom se provjerava da li je taj trokut tj. njegova ravnina iznad ili ispod točke u prostoru nad kojom provjeravamo da li je unutar volumena. Radi same definicije trokuta tj. redoslijeda njegovih vrhova moguće je razlikovati njegovu gornju odnosno donju stranu. Ako je ravnina presječenog trokuta iznad točke tada povećavamo brojač, a ako je ispod tada brojač smanjujemo. Točka je unutar volumena objekta ako je konačni brojač za tu točku jednak nuli.

Ovaj postupak potrebno je napraviti za sva središta kocaka unutar jediničnog volumena. Svakoj kocki za koju se odredi da je unutar volumena dodjeljuje joj se varijabla pobrojanog tipa `InsideVulume` kojom definiramo njezinu pripadnost volumenu objekta. Sve ostale kocke su time izvan volumena objekta te se mogu obrisati iz sustava zajedno sa svim točkama fizikalnog modela koje se nalaze unutar tih kocaka.

Problem korištenja ove metode jest njezina nepreciznost i sporost pri izvođenju. Nepreciznost se javlja u trenutku kada se ne ustanovi presjek zrake s trokutom. Razlog tome je taj da je zraka koju koristimo paralelna ili gotovo paralelna s trokutom. Algoritam će odrediti da zraka ne presijeca trokut te će se u konačnici krivo odrediti pripadnost točke unutar volumena objekta. Ovaj tip greške se može smanjiti i poboljšati na način da se

kreira neparan broj zraka te se metodom glasovanja odredi konačna pripadnost točke koju provjeravamo.

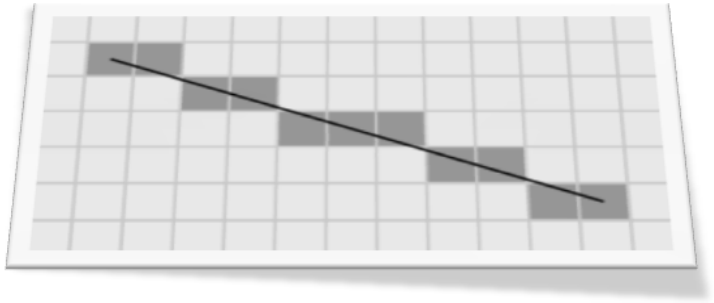
Drugi problem uz nepreciznost određivanja je sporo izvođenje. Ono se javlja jer je potrebno napraviti jako puno računskih operacija koje u ovom slučaju ovise o broju kocaka unutar jediničnog volumena i broju poligona učitanog objekta. Za velike objekte (objekti sa velikim brojem poligona) određivanje pripadnosti pojedine kocke zahtjeva puno vremena, a konačni rezultat ne mora nužno biti ispravan.

6.3 Pronalaženje unutarnjeg volumena pomoću rastera kocaka

Kao i prethodnom metodom pokušava se pronaći unutarnji volumen ali ovaj put pomoću definiranog rastera fizikalne strukture kocaka i znanjem da je objekt skaliran unutar tog istog prostora. Ova metoda započinje tako da se prvo odrede rubne kocke učitanog modela objekta, a nakon ispravno definiranih kocaka koje čine oplošja objekta određuje se kocke njegove unutrašnjosti.

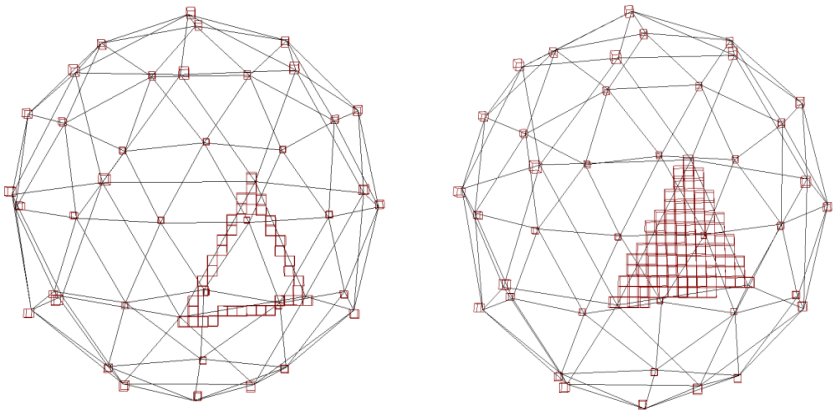
6.3.1 Određivanje rubnih kocaka pomoću 3D Bresenham algoritma

Za određivanje rubnih kocaka koristi se prilagođen Bresenham 3D algoritam [5]. Bresenham algoritam služi za iscrtavanje povezanih slikovnih elemenata, odnosno linije između dvije točke. Veličina rasterizacije 2D prostora je u tom slučaju veličina slikovnog elementa. Algoritam je proširiv i na 3D varijantu koja se ovdje i koristi. Na slici 6.1 prikazan je primjer Bresenham 2D algoritma.



Slika 6.1 Primjer Bresenham 2D algoritma.

Kako je raster fizikalne strukture definiran veličinom stranice kocke tako je i algoritam prilagođen na način da se pomiče po veličini duljine stranice. Algoritam je promijenjen tako da umjesto da iscrtava slikovni element na određenoj koordinati on postavlja vrijednost pobrojanog tipa kocke na vrijednost `BordeOfVolume`. Određivanje kojoj kocki treba odrediti njezinu vrijednost obavlja se indeksiranjem dobivenih koordinata unutar rastera fizikalne strukture. Na slici 6.2 je prikazano djelovanje algoritma na jednom trokutu objekta sfere. Na slici s lijeve strane prilagođen algoritam prvo popunjava kockama rubove odabranog trokuta, a s desne strane se algoritam proširuje kako bi popunio unutrašnjost trokuta. Popunjavanje unutrašnjosti trokuta se obavlja također pomoću Bresenham 3D algoritmom na način da se za svaki trokut s vrhovima `A`, `B` i `C` prvo provede Bresenham od `A` do `B` te se za sve dobivene kocke napravi po jedan Bresenham od te točke do vrha `C`. Rezultat popunjavanja se može vidjeti na desnoj strani slike.



Slika 6.2 Određivanje kocaka koje čine oplošje volumena objekta.

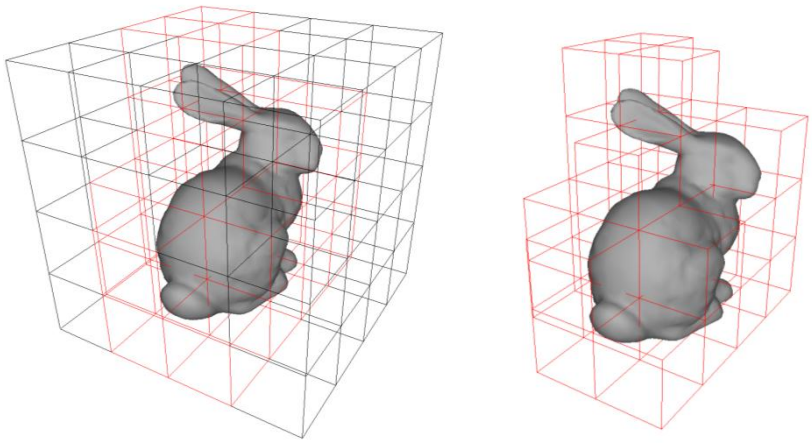
Ovaj postupak se obavlja za sve trokute unutar modela. U konačnici dobivamo ispravno popunjeno oplošje modela. Nakon dobro definiranih granica moguće je odrediti kocke unutarnjeg volumen objekta.

6.3.2 Određivanje unutarnjih kocaka

Prijašnjim postupkom pronađene su kocke koje se nalaze na oplošju modela objekta i označene su sa pobrojanim tipom `BordeOfVulume`, sve ostale kocke unutar fizikalne strukture imaju pobrojani tip `NotSpecified`. Realizacija pronalaženja unutrašnjosti

modela riješena je na način da se kreira stog objekata tipa kocke. U taj stog se inicijalno umetnu objekti tipa kocke koji se nalaze na oplošju jedinične kocke i koji su pobrojanog tipa `NotSpecified`. Svim kockama unutar tog stoga dodjeljuje se pobrojani tip `OutsideOfVolume` kojime definiramo da se te kocke nalaze izvan volumena objekta. Nakon kreiranja stoga ulazi se u petlju koja se izvršava sve dok se stog ne isprazni. U jednoj iteraciji petlje miče se prvi element sa stoga, a na kraj se stavljaju svi njegovi susjedi koji su tipa `NotSpecified` i postavlja im se vrijednost na `OutsideOfVolume`.

Nakon što je stog ispražnjen, fizikalna struktura i dalje sadrži sve kocke ali s različitim varijablama pripadnosti. Postoje kocke koje imaju varijablu pripadnosti `BordeOfVulume` koje su određene prilikom pronalaženja oplošja objekta, zatim kocke koje imaju tip `OutsideOfVolume` koje su određene pomoću stoga i kocke tipa `NotSpecified` koje se sada zapravo nalaze unutar objekta i definiraju njegovu unutrašnjost. Kocke s `NotSpecified` preimenujemo u `InsideOfVolume` i time završava postupak određivanja unutrašnjosti objekta. Na slici 6.3 prikazana je grafička reprezentacija odstranjivanja kocaka vanjskog volumena (kocke obojane crnom bojom). Raster kocaka u ovom primjer je namjerno velik kako bi se građevne kocke mogle lakše prikazati.



Slika 6.3 Odstranjivanje kocaka vanjskog volumena.

Nakon definiranja rubnih i unutarnjih kocaka mogu se obrisat sve ostale kocke i pripadajuće točke fizikalnog modela jer one definiraju vanjski volumen objekta i nisu više potrebne. Ovime završava modeliranje fizikalne strukture i moguće ju je simulirati. Sljedeći korak je povezivanja fizikalne strukture s modelom objekta.

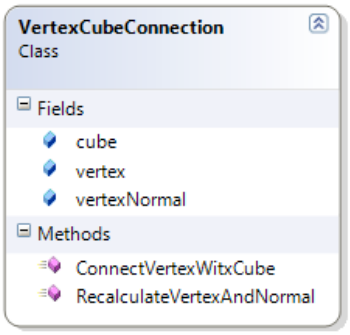
7 Povezivanje fizikalne strukture s modelom objekta

Izgrađena i modelirana fizikalna struktura nije povezana s poligonima objekta, odnosno njegovim vrhovima, pa bi se prilikom simulacije simulirala samo fizikalna struktura. Radi toga potrebno ju je povezati s vrhovima objekta kako bi se deformacija fizikalne strukture odražavala i na model objekta.

7.1 Kreiranje strukture povezanosti

Kao što smo rekli model objekta se sastoji od trokuta i vrhova. Njegov sastavni dio koji je potrebno deformirati jesu njegovi vrhovi, a struktura i raspored trokuta ostaje nepromijenjen. Ako želimo vršiti sjenčanje potrebne su nam i normale pojedinog vrha, koje se također mijenjaju u ovisnosti o deformaciji objekta. Dakle potrebno je definirati i kreirati strukturu koja povezuje fizikalnu strukturu sa vrhovima i normalama objekta.

Za svaki vrh modela potrebno je načiniti poveznicu s fizikalnom strukturom. Njihovo povezivanje obavlja se pomoću najbliže građevne kocke unutar fizikalne strukture. Struktura povezanost je definirana klasom `VertexCubeConnection`.

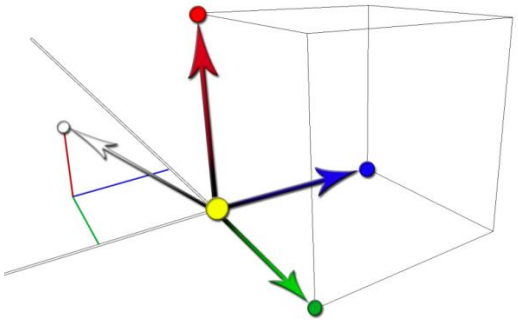


Slika 7.1 Klasa strukture povezanosti.

Na slici 7.1 prikazana je struktura povezanosti koja sadrži referencu na građevnu kocku, poveznicu s vrhom i normalom objekta. Prilikom kreiranja ove strukture poziva se `ConnectVertexWithCube` kojoj se šalje koordinata vrh i vektor njegove normale. Oni se tada povezuje s lokalnim koordinatnim sustavom kocke kakao bi deformacija građevne kocke utjecala na deformaciju vrha i normale sa kojom je povezana.

7.1.1 Definiranje lokalnog koordinatnog sustava kocke

Lokalni koordinatni sustav kocke predstavlja osnovnu poveznicu fizikalne strukture s modelom objekta. Na slici 7.2 prikazana je reprezentacija povezanosti kocke s jednim vrhom modela, povezivanje s normalom vrha nije prikazana jer se povezuje na gotovo identičan način. Kako je građevna jedinica kocka, tako je vrlo jednostavnom pronaći njen koordinatni sustav. Koordinatni sustav kocke definiran je vrhovima kocke (vrhovi su na slici predstavljeni crvenom, plavom, zelenom i žutom bojom). Rješenje je napravljeno na način da donji lijevi vrh (na slici točka žute boje) predstavlja ishodište koordinatnog sustava, a strjelice plavom, zelenom i crvenom čine redom njegove osi `X`, `Y` i `Z`. Unutar strukture `VertexCubeConnection` za varijable `vertex` i `vertexNormal` potrebno je spremiti vrijednosti njihovih odsjeka po osima lokalnog koordinatnog sustava. Na taj načine one postaju relativno povezane s građevnom kockom i ovise samo o njenom koordinatnom sustavu.



Slika 7.2 Povezivanje lokalnog koordinatnog sustava sa vrhom modela objekta.

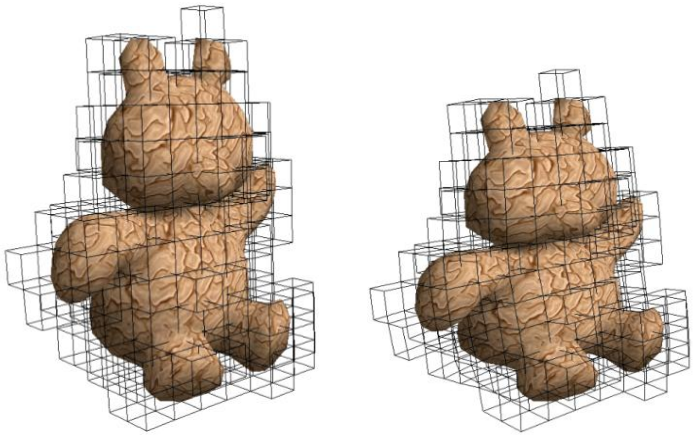
Prilikom računanja odsjeka potrebno je pronaći vektor od odabranog ishodišta lokalnog koordinatnog sustava (na slici označen žutom bojom) do vrha modela objekta (na slici točka označena bijelom bojom). Nad tim vektorom potrebno je načiniti skalarnu projekciju po svim osima te njihove rezultate spremiti u strukturu povezanosti. Za povratak vrha u globalni koordinatni sustav potrebno je prvo izračunati njegovu lokaciju u lokalnom koordinatnom sustavu kocke i zbrojiti je s vektorom ishodišta tog sustava.

Za svaki vrh i njegovu normalu načini se ovako predstavljena procedura i time dobivamo njihove odsjeke u lokalnom koordinatom sustavu kocke. Popunjena struktura povezanosti sa svim vrhovima i normalama modela spremna je na primjenu deformacije te će se simuliranjem fizikalne strukture simulirat i model objekta.

7.1.2 Primjena deformacije fizikalne strukture nad modelom objekta

. Fizikalna struktura se osvježava prilikom izračuna jednadžbi fizikalnog modela, a osvježavanje deformacije vrhova i normala objekta potrebna je samo neposredno prije njegovog iscrtavanja jer nam ona služi isključivo za vizualizaciju. Prije iscrtavanja modela mekog objekta poziva se metoda `RecalculateVertexAndNormal`, kojom se osvježavaju vrhovi i normale.

Svaka kocka je povezana s točkama fizikalnog modela unutar kreirane fizikalne strukture nad kojima djeluje simulacija fizike. Deformacijom točaka fizikalnog modela deformiraju se i građevne kocka. Kocke tada izgledaju više nalik kvadra i prilikom simulacije se rotiraju i translatiraju u ovisnosti o točkama fizikalnog modela koje ih definiraju. Radi deformacije kocaka potrebno je nanovo izračunati njihov lokalni koordinatni sustav.



Slika 7.3 Primjer utjecaja deformacije fizikalne strukture na model objekta.

Struktura povezanosti ima referencu na kocku i odsjeke po koordinatnim osima kojima su povezani vrhovi i normale. Za izračun deformacije vrhova i normala potrebno je izračunati njihovu novu poziciju u promijenjenom koordinatnom sustavu kocke. Ovaj izračun je napravljen tako da što kraće traje kako ne bi utjecao na brzinu simulacije. Izračun se obavlja na način da se pomnože skalarne vrijednosti odsjeka s osima novo izračunatog koordinatnog sustava kocke. Kako bi se izračunati vrh nalazio na točnom mjestu u prostoru potrebno ga je zbrojiti s ishodištem koordinatnog sustava kocke. Za normalu se obavlja isti postupak ali bez zbrajanja s ishodištem jer normala predstavlja vektor smjera te joj translacija lokacija nije potrebna.

8 Primjena sila na model

Kako bi se prikazala simulacija i interakcija s modelom mekog objekta potrebno je omogućiti djelovanje vanjskih sila na nju. Jedna od vanjskih sila koja je uvijek prisutna je gravitacijska sila. Kada se pokrene simulacija, možemo vidjeti kako gravitacijska sila djeluje na model mekog objekta tako što se počinje deformirati te se pri djelovanju sila opruga i sila prigušenja stabilizira.

U ovom radu implementirane su dvije vrste sila, lokalna sile koje djeluju nad odabranom skupinom točaka fizikalnog modela i globalna sila koja djeluje nad svim točkama fizikalnog modela. Svakoj od njih možemo mijenjati jačinu i smjer njihovog djelovanja.

Određivanje područja djelovanja lokale sile moguće je naviše načina. Jedan od pristupa je takoreći klasični pristup određivanja područja djelovanja vanjske lokalne sile. On se postiže projiciranjem pravca iz pozicije i smjera sile te određivanjem udaljenosti između tog pravca i pojedine točke fizikalnog modela unutar fizikalne strukture. Najbliže točke fizikalnog modela s obzirom na pravac su tada pod utjecajem sile. Ovaj pristup zahtijeva puno kalkulacije i izračuna, što sve zajedno otežava prikaz i simulaciju mekog modela u realnom vremenu. Radi toga u ovom radu se koristi metoda selektiranja objekata na prizoru kako bi se odredilo nad kojim točkama fizikalnog modela trebamo primijeniti vanjsku silu.

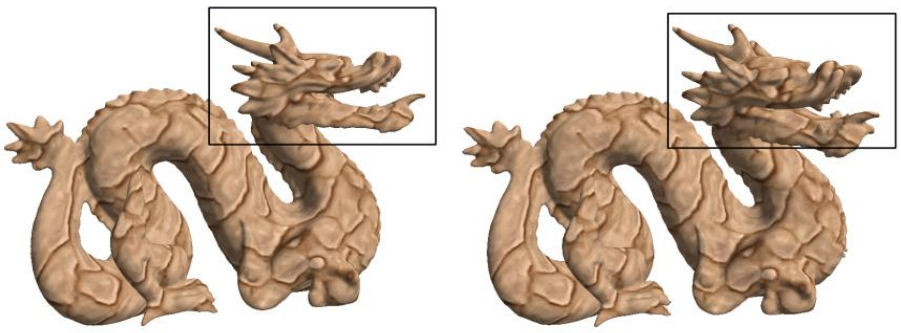
8.1 Metoda selektiranja

Postoje ugrađene funkcije unutar OpenGL-a koje omogućavaju imenovanje objekata na sceni te ekstrakciju imena vidljivih objekata unutar prikaza. Ova tehnika naziva se metoda selektiranja [6].

Za provedbu ove metode potrebno je iscrtavanje točaka fizikalnog modela. Radi bržeg i efikasnijeg iscrtavanja odbacuju se sve ostale nepotrebne informacije sa scene. Prilikom generiranja takvog prikaza obavlja se jednostavno iscrtavanje točaka fizikalne strukture pozivom `glBegin(GL_POINTS)`. Potrebno je prilikom slanja pozicije vrhova pojedinog

objekta OpenGL-u postaviti i njegovu identifikaciju (u ovom slučaju naš objekt je zapravo točka fizikalnog modela) to se obavlja funkcijskim pozivom `glPushName(i)`, gdje ime predstavlja indeks točke. Kada smo definirali imena svim točkama fizikalne strukture obavlja se iscrtavanje funkcijskim pozivom `glRenderMode(GL_SELECT)`. Ovaj način iscrtavanja nije vidljiv korisniku već se iscrtava samo radi brze ekstrakcije imena vidljivih objekata unutar definiranog prizora.

Kako bi izvukli, odnosno spremili podatke o imenima objekata, iz prikaza moramo definirati spremnik, `glSelectBuffer(selectBufferSize, selectBuffer)`, u koji će se pohraniti imena dohvaćena ovim pristupom iscrtavanja kao i njihove udaljenosti od ravnine promatranja. Ovim pristupom dobivamo imena objekata koji se nalaze na cijelom prizoru no ako želimo ograničiti prostor djelovanja metode selektiranja tada moramo definirati `PickMatrix`. Pomoću `PickMatrix` vršimo rezanje trenutnog prikaza pravokutnikom visine `selectHight` i širine `selectWidth` te određujemo poziciju toga pravokutnika unutar prikaza. Za poziciju pravokutnika unutar prikaza koriste se koordinate miša `(x, Height - y)`. Time ograničavamo metodu selektiranja tako da prihvaća samo pronađene objekte unutar pravokutnika koji svojom visinom i širinom okružuje koordinate kursora miša.



Slika 8.1 Prikaz djelovanje vanjske sile na području pravokutnika.

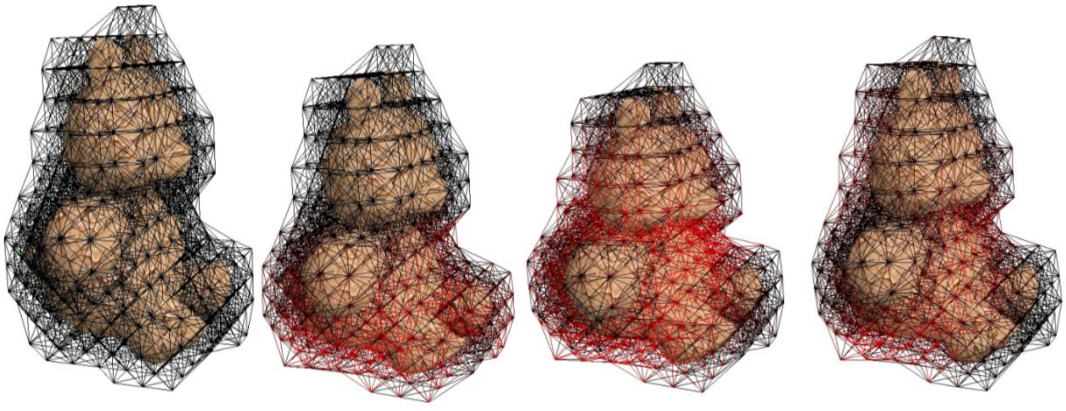
Pozivom `numberOfHits=gLRenderMode(GL_RENDER)` dohvaćamo broj pogodaka odnosno broj spremljenih imena unutar `selectBuffer`, te pomoću tog broja možemo proći kroz `selectBuffer` i dohvatiti indekse točaka nad kojima trebamo primijeniti vanjsku silu. Djelovanje lokalne sile prikazano je na slici 8.1, prikazan je model mekog objekta, pravokutnik selektiranja i utjecaj primjene sile.

8.2 Simulacija djelovanja vanjskom silom

Prethodno definiranom metodom određujemo nad kojim ćemo točkama fizikalnog modela primijeniti silu u slučaju korištenja lokalnog djelovanje sile. Za djelovanje globalne sile odabiru su sve točke fizikalnog modela. Smjer lokalne sile je projekcija pogleda na model mekog objekta. Za određivanje smjera sile potrebni su nam vektori očista i gledišta trenutne pozicije i orijentacije kamere unutar scene. Oduzimanjem vektora očišta od vektora gledišta dobivamo vektor koji je okomit na trenutni prizor i određuje njezin smjer.

Slika 8.2 Primjer djelovanja sile na model mekog tijela.

Moguće je povećavati ili smanjivati jačinu sile i mijenjati smjera njezinog djelovanja. Sile su izražene u Newtonima te se lako mogu pribrajati ukupnoj sili koja djeluje na pojedinu točku fizikalnog modela. Vanjska sila na točku fizikalnog modela sprema se u vektor `ExternalForce`, koja se prilikom ažuriranja stanja modela zbraja sa ukupnom silom.



Slika 8.3 Prikaz deformacije fizikalne strukture.

Na gornjoj slici se može vidjeti utjecaj sila na deformaciju fizikalne strukture, gdje su crnom bojom označe opruge fizikalnog modela a crvenijom bojom opruge koje se nalaze u stanju napetosti te je sila između njih veća.

9 Simulacija mekog objekta

Kreiran model mekog objekta spreman je za simulaciju. Za provedbu simulacije potrebno je proći kroz izračun jednadžbi fizikalnog modela, osvježavanje deformacije modela objekta, osvjetljenje i sjenčanje, teksturiranje i konačno njegov prikaz unutar scene.

9.1 Izračun jednadžbi fizikalnog modela

Računanje jednadžbi fizikalnog modela obavlja se frekvencijom od 100Hz kojom dobivamo glatku reprezentaciju realnog svijeta preslikanu u računalni model. Na početku se definira inicijalno stanje modela mekog objekta postavljanjem u sustav scene te definiranjem brzine pojedine točke na nulu. Model mekog objekta je interno povezan oprugama kroz definiranu fizikalnu strukturu kojoj se može odabirati gustoća kojom želim predstaviti odnosno simulirati model mekog objekta.

Postoje razni atributi koji definiraju ponašanje model mekog objekta, a među njima su gravitacija, masa čestice, konstanta opruge, prigušenje opruge i raster fizikalne strukture.

Izračun jednadžbi fizikalnog modela svodi se na zbrajanje doprinosa svih sila koje okružuju pojedinu točku kroz njezino k susjedstvo, gravitacijska sila i utjecaj vanjskih sila. Algoritam započinje zbrajajući izračunate sile opruga okružujućih točaka te njihovih prigušenja. Dobivenoj sumi dodamo gravitacijsku silu $F_{grav,i}$ trenutne točke i utjecaj vanjskih sila definiranom od strane korisnika $F_{van,i}$ te dobivamo ukupnu silu F_{ui} točke i

$$F_{ui} = \sum_{j=0, j\neq i}^{k} (F_{opruge\ ,j} + F_{prigusenja\ \ ,j}) + F_{grav,i} + F_{van,i}$$

Nakon što su sve sile koje djeluje na trenutnu točku fizikalnog modela izračunate tada preko Eulerove integracije dobivamo novu brzinu, a pomoću nje novu poziciju. Te vrijednosti spremamo u objekt točke prepisivanjem prethodno izračunatih vrijednosti. Ovaj

postupak obavlja se za sve točke fizikalnog modela mekog objekta. Model mekog objekta je potrebno ažurirati kako bi odgovarao fizikalnoj strukturi koja ga opisuje.

9.2 Osvježavanje deformacije mekog objekta

Vrhovi i normale dinamičkog objekta se deformiraju prilikom simulacije. Za prikaz je potrebno osvježiti vrhove i normale objekta kako bi se simulirala deformacije njegove površina i volumena. Osvježavanje se vrši neposredno prije prikaza modela mekog objekta.

9.3 Osvjetljenje i sjenčanje mekog objekta

Kada bi model mekog objekta bio jednobojan tada se ne bi mogle točno primijetiti deformacije jer ne bi imali osjećaj dubine samog predmeta. Unutra scene se definira globalni sustav osvjetljenja s definiranom jačinom utjecaja ambijentalne, difuzne i spektralne komponente svjetla.

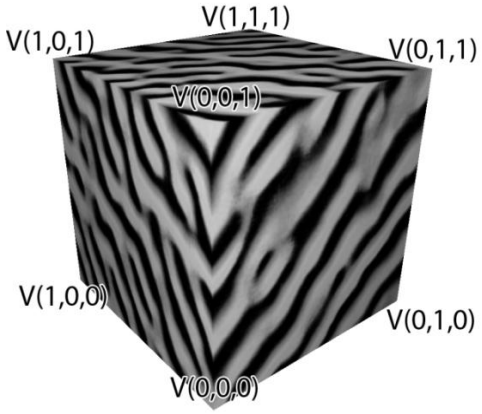
Sjenčanje nad vrhovima obavlja se prilikom iscrtavanja trokuta modela objekta. Prilikom funkcijskog poziva OpenGL-a za iscrtavanje trokuta se uz koordinate vrhova trokuta šalju i vektori normala pojedinog vrha. OpenGL automatski primjenjuje Phongov model osvjetljenja s Gouraudovim postupkom sjenčanja.

9.4 Teksturiranje mekog objekta

Tekstura daje dozu realnosti mekom objektu te je u ovom radu omogućeno dinamičko učitavanje teksture. Kako je napravljen generički sustav za učitavanje različitih modela mekih objekata, potrebno je napraviti i sustav za njegovo teksturiranje.

Teksturiranje je napravljeno pomoću volumnih tekstura [4]. Volumna tekstura je definirana s tri koordinate skalirane na jediničnu kocku. Vrijednosti koordinata se kreče u intervalu [0,1], a dohvat boje slikovnog elementa teksture na određenoj koordinati dobiva

se funkcijom $V(x, y, z)$ pomoću trilinearne interpolacije. Primjer volumne teksture kojom će se vršiti teksturiranje prikazano je na slici 9.1.



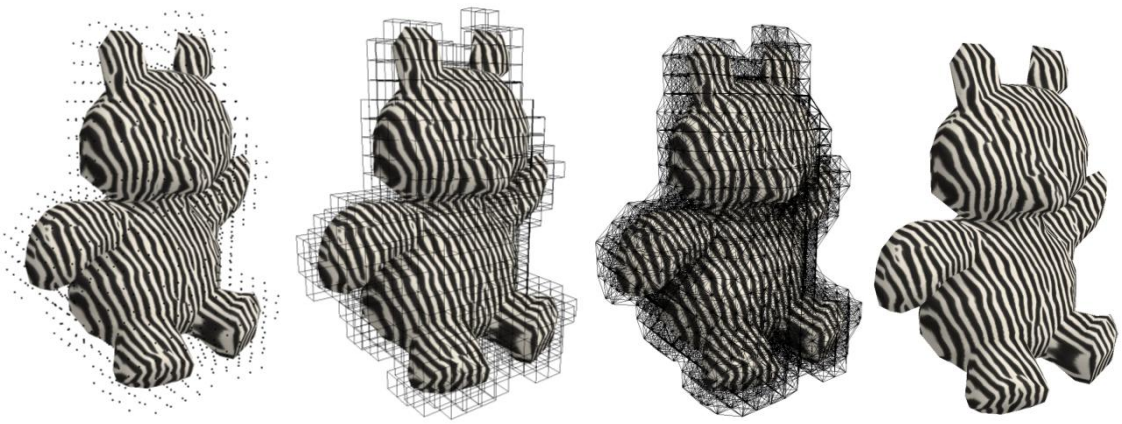
Slika 9.1 Primjer volumne teksture.

Tekstura se preslikava na model mekog objekta tako da se za vrijednosti funkcije dohvaćanja slikovnog elementa volumne teksture zada koordinata nedeformiranih vrhova modela mekog objekta. Tekstura koja se učitava je ponavljajuća u svim smjerovima, pa se radi toga mogu teksturirati veliki objekti sa relativno malom teksturom. Primjer nekoliko teksturiranih modela mekih objekta se može vidjeti na slici 9.2.

Slika 9.2 Primjer teksturiranja mekih objekata volumnom teksturom.

9.5 Prikaz mekog objekta

Kada su sve pripremne faze gotove moguće je provesti i sam prikaz simulacije mekog objekta. Moguće je odabrati između 4 vrste iscrtavanja. Svaki od tih prikaza daje uvid u različite aspekte modela mekog objekta, a mogu se vidjeti na slici 9.3.



Slika 9.3 Primjer različitih načina iscrtavanja mekog objekta.

Prva tri načina iscrtavanja prikazuju utjecaj djelovanja sila na fizikalnu strukturu mekog objekta, na prvoj slici prikazane su točke fizikalnog modela. Na drugoj slici prikazane su građevne kocke fizikalne strukture, a trećim prikazom predstavljena je povezanosti opruga unutar fizikalne strukture. Zadnji način iscrtavanja prikazuje sam model mekog objekta bez fizikalne pozadine koja ga definira.

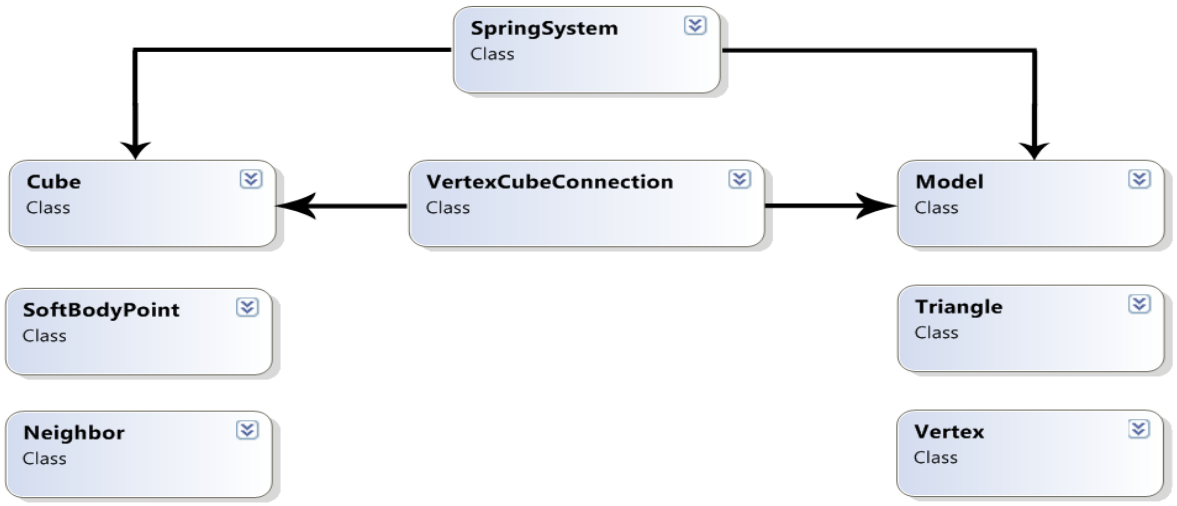
Svaki od ovih prikaza daje uvid u prijenos i djelovanje sile unutar fizikalne strukture prilikom njene simulacije. Iscrtavanjem fizikalne pozadine uz samu deformaciju mijenja se i boja u ovisnosti o silama koja djeluje na nju. Crna boja je boja mirovanja, ona prelazi u crveniju što je utjecaj sila na pojedini segment fizikalne strukture veći. Kako bi mogli nesmetano promatrati samo fizikalnu pozadinu moguće je isključiti iscrtavanje modela mekog objekta.

10 Pregled programske implementacije

Programska implementacija napisana je u jeziku C++ korištenjem objektno orijentirane paradigme uz OpenGL standard. Koristi se GLUI proširenje koje je potrebno za stvaranje korisničkog grafičkog sučelja [7]. Ono omogućava kreiranje sučelja s tipkama, manipulatorima miša i prostorima za unos teksta. U ovom poglavlju biti će opisana struktura programskog koda, grafičko sučelje programa i mogućnosti interakcije korisnika.

10.1 Struktura programa

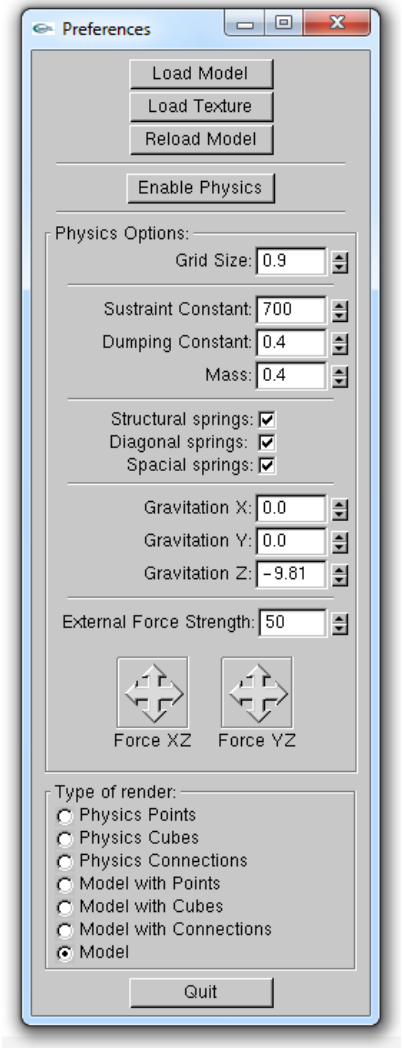
Na slici 10.1 prikazan je dijagram važnijih klasa i njihova međusobna povezanost. `SpringSytem` predstavlja glavnu klasu koja sadrži cijeli sustav mekog modela. U njoj se nalaze metode za kreiranje mekog modela, izračun fizikalno temeljenog modela i iscrtavanje modela. Ona sadrži fizikalnu strukturu koja je sačinjena od građevnih jedinica kocaka. Svaka kocka povezana je s točkama fizikalnog modela i definiranim susjedstvom pojedine točke. S druge strane model objekta je definiran trokutima i njegovim vrhovima. Klasa `VertexCubeConnection` predstavlja strukturu povezanosti kojom se povezuje model objekta sa fizikalnom strukturom. Osvježavanjem strukture povezanosti dobiva se konačni model mekog objekta.



Slika 10.1 Dijagram važniji klasa kreiranog sustava.

10.2 Sučelje programa

Sučelje programa napravljeno pomoću GLUI proširenja te je prikazano na slici 10.2. Program kroz sučelje omogućava dinamičko učitavanje modela i mijenjanje njegove teksture. Unutar panela `Type of render` moguće je mijenjati način iscrtavanja modela mekog objekta.



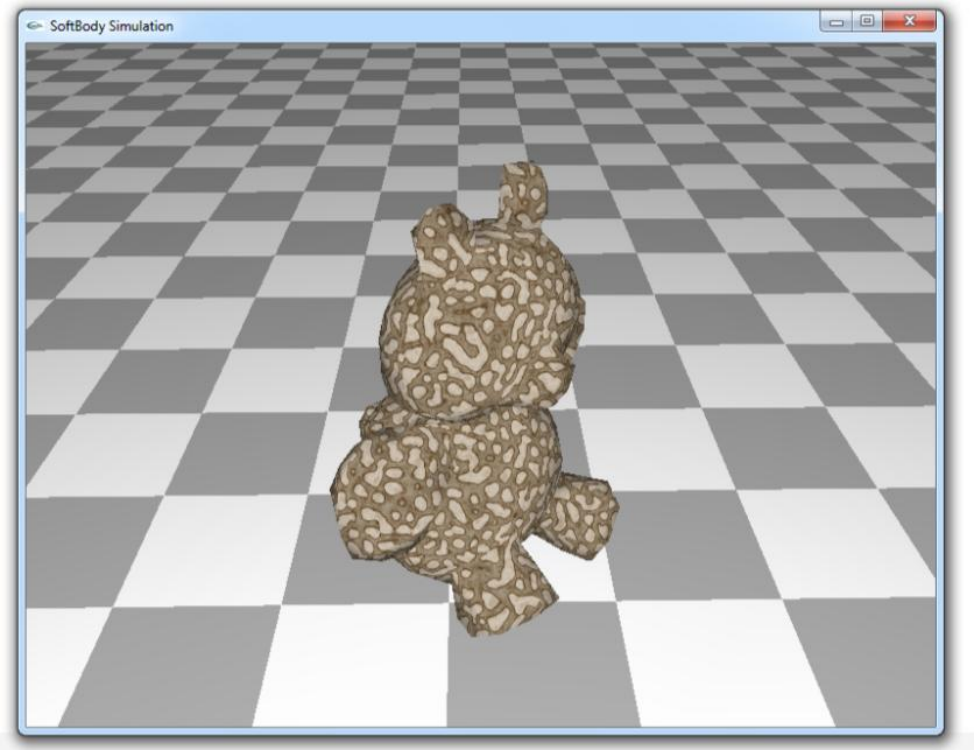
Slika 10.2 Prikaz grafičkog sučelja za manipulaciju nad simulacijom.

Parametre simulacije mekog modela moguće je dinamički mijenjati, a njihova promjena se automatski primjenjuje na sam model. Parametri koji se mogu mijenjati u stvarnom vremenu su: broj opruga, konstanta opruge, konstanta prigušenja, masa čestice,

gravitacijska sila i jačina djelovanja vanjskih sila. Veličina rastera `gridSize` odnosno gustoća fizikalne strukture je parametar koji se ne može mijenjati u stvarnom vremenu, kako bi se vidio utjecaj njegove promijene potrebno je ponovno kreirati sustav ili osvježiti model tipkom `Reload Model`.

Panel `External Force` omogućava djelovanje globalnom silom. Pomoću kreiranih strelica unutar panela te pritiskom i povlačenjem miša nad njima definiramo smjer djelovanja globalne sile. Predstavljena su dvije mogućnosti prilikom odabira smjera djelovanja sile i to u `XZ` i `YZ` ravnini. Uz korištenje strelica moguće je koristiti i tipke na tastaturi `Y`, `X` kojima djelujemo na objekt vertikalnom silom kojom se može simulirati pritisak na meki objekt.

Uz manipulaciju kroz grafičko sučelje GLUI moguća je i interakcija kroz prozor iscrtavanja objekta prikazanog na slici 10.3. Kombinacijom tipaka na tastaturi (`W`, `S`, `A` i `D`) te pomakom pritisnutog miša moguće je gibati se kroz prostor u kojemu se nalazi meki objekt. Promjenom lokacije unutar scene dobiva se uvid u simulaciju iz različitih pogleda.



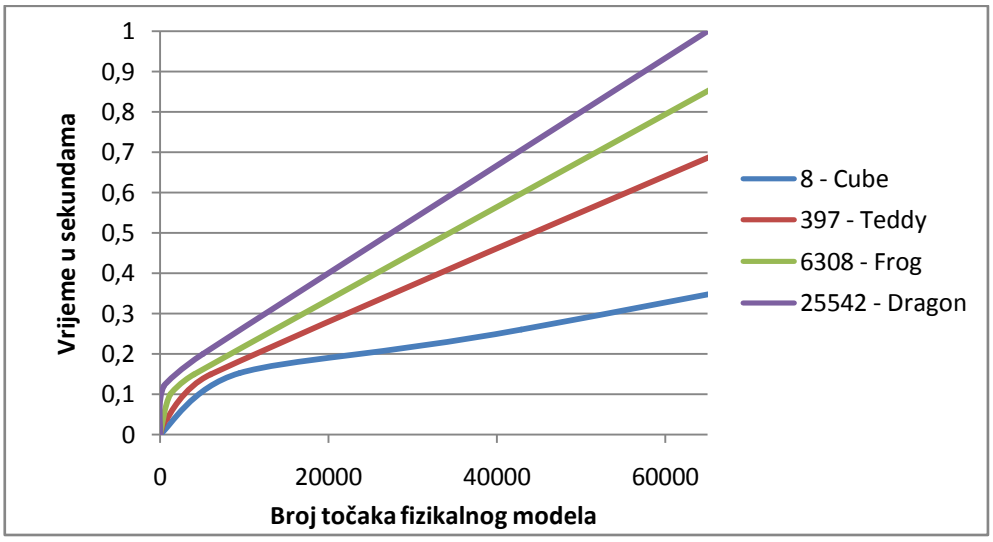
Slika 10.3 Prikaz grafičkog sučelja namijenjenog za iscrtavanje i interakciju.

Za djelovanje lokalne sile na model mekog objekta koriste se tipke na tastaturi *F* i *G*. Tipkom *F* djelujemo pozitivnom silom, a sa *G* negativnom odnosno u suprotnom smjeru. Smjer djelovanja sile definirano je pogledom, a područje djelovanja je pravokutnik oko lokacije kursora miša. Također je omogućeno i brisanje fizikalne strukture modela pritiskom tipke *H* koja će obrisati dio fizikalne strukture na koju pokazujemo kursorom miša.

Manipulacija parametara i interakcija omogućava korisniku da na razne načine djeluje na samu simulaciju mekog modela. Najveći utjecaj na simulaciju postiže se kroz interaktivno djelovanje vanjskim silama na sustav.

11 Performanse sustava

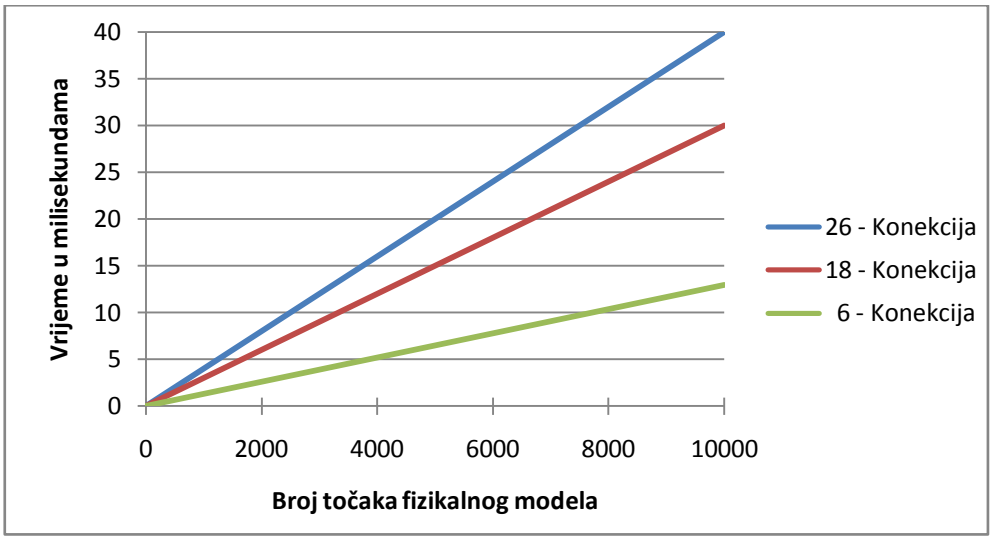
Kreiran sustav omogućava prikaz i simulaciju modela mekih objekata u stvarnom vremenu. Na slici 11.1 predstavljeno je vrijeme koje je potrebno za kreiranje različitih tipova objekata. Objekti nad kojima se mjerilo vrijeme kreiranja imaju različiti broj vrhova kako je pokazano na legendi grafa. Najviše vremena koje je potrebno za kreiranje mekog modela predstavljaju oni objekti koji imaju najviše vrhova. Razlog porasta vremena kreiranja je u činjenici da deformacija fizikalnog modela mora utjecati i na deformaciju vrhova učitanog objekta. Za model objekta kocke koji ima samo 8 vrhova dovoljno je odabrati 8 kocaka u rasteru koje će definirati povezanost između fizikalne strukture i vrhova objekta. Veličina strukture povezanosti je prema tome ista broju vrhova objekta. Dovoljan broj točaka fizikalnog modela za simuliranje mekog objekta je 500, a vrijeme potrebno za generiranje takvog sustava iznosi manje od 20 milisekundi za sve isprobane objekte.



Slika 11.1 Vrijeme potrebno za kreiranje sustava mekog objekta

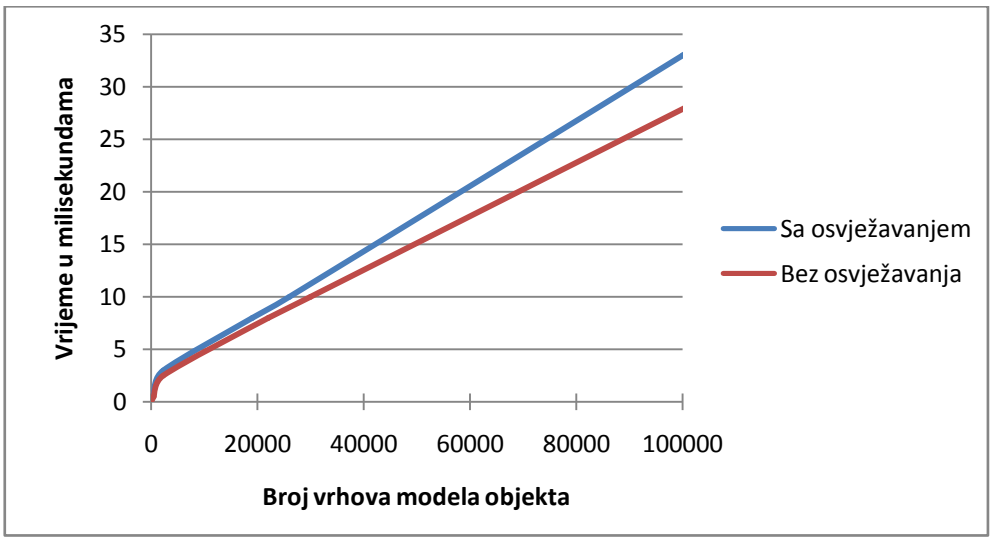
Na slici 11.2 prikazana je ovisnost brzine izračuna stanja fizikalnog modela u odnosu na broj točaka fizikalnog modela. Predstavljene su tri krivulje koje definiraju različiti broj susjedstva nad točkama fizikalnog modela. Zelena krivulja predstavlja model koji je načinjen s prosječnim brojem od 6 susjedstva, crvena sa 18, a plava predstavlja 26 susjedstva koji je i korišten u ovome radu. Vrijeme potrebno za kreiranje stabilne simulacije iznosi 10 milisekundi, na grafu se može vidjeti kako je za simulaciju mekog

modela moguće kreirati oko 2000 točaka fizikalnog modela kojom i dalje ne prelazimo vrijeme od 10 milisekundi. Za simulaciju mekih objekata dovoljno je koristiti samo 500 točaka fizikalnog modela kojom dobivamo lijepe rezultate i visoku realističnost same simulacije.



Slika 11.2 Ovisnost broja točaka i poveznica fizikalnog modela o vremena izvođenja.

Brzina iscrtavanja u računalnoj grafici ovisna je o vremenu jer postoji usko grlo prilikom slanja velikog broja poligona u grafički protočni sustav. Prilikom simulacije mekog objekta potrebno je u fazi iscrtavanja osvježavati vrhove trokuta i njihovih normala.



Slika 11.3 Ovisnost broja vrhova objekta o vremenu iscrtavanja.

Na slici 11.3 prikazana je ovisnost broja vrhova objekta o vremenu potrebnom za njegovo iscrtavanje. Crvenom krivuljom predstavljeno je vrijeme koje je potrebno za iscrtavanje objekta bez osvježavanja deformacije vrhova i normala. Plavom krivuljom predstavljeno je ukupno vrijeme zajedno sa osvježavanjem vrhova i normala objekta. Može se primijetiti kako je povećanje potrebnog vremena linearno i kako osvježavanje deformacije ne ugrožava mogućnost prikaza modela u stvarnom vremenu. Pogotovo ako je riječ o relativno malim objektima koji imaju ispod 30000 vrhova.

Ovakav sustav je moguće poboljšati prebacivanjem izračuna stanja fizikalnog modela unutar GPU-a grafičke kartice. Korištenjem `Vertex Shader` moguće je prebaciti čitavi izračun stanja fizikalnog modela na način da se polje točaka fizikalnog modela spremi u odgovarajuću teksturu. Svaka točka fizikalnog modela pomoću prenamijenjene teksture u paraleli izvršava izračun promijene brzine i pozicije. Osvježavanje vrhova modela objekta bi se također radilo pomoću prilagođenih tekstura unutar kojih bi bili zapisani indeksi njihove povezanosti sa fizikalnog strukturom. Proširenjem na GPU dobilo bi se strahovito ubrzanje jer su sve komponente razdvojive i mogu se paralelizirati.

12 Zaključak

Simulacija mekih objekta predstavlja budućnost računalne grafike. Ona daje objektima dozu realnosti koja se pomoću krutih i na taj način statičnih objekata ne može postići. Korištenje sustava čestica povezanih oprugama otvara razne mogućnosti primjene unutar računalne grafike. Stvaranjem fizikalne strukture modela mekog objekta je jedna od njih. Kreiranjem i povezivanjem fizikalne strukture sa različitim modelima objekta mogu se kreirati razni efekti poput: simulacije gibanja krošnje drveta, simulacija utjecaja vjetra na raslinje i nešto sasvim ne intuitivno kao što je kreiranje trodimenzionalnog savitljivog teksta.

Modeli mekih objekata su iskoristivi na različite načine. Ponajprije u računalnim igrama i filmskoj industriji za prikaz realističnih objekata kojima se deformira oblik prilikom djelovanja vanjskih i unutarnjih sila. Druga velika grana njihove primjene je unutar medicine kroz simulaciju rada pojedinih organa. Simuliranje dinamičkih pojava iz stvarnog svijeta unutar računalnog sustava otvara nove horizonte računalne grafike i animacije. Primjene sustava modela čestica povezanih oprugama su raznolike, potrebna je samo inspiracija.

13 Literatura

[1] Alliez, P., Cohen-Steiner, D., Yvinec, M., Desbrun, M.:Variational Tetrahedral Meshing, *Proceedings of ACM SIGGRAPH 2005, Volume 24 Issue 3, July 2005.*, <http://www.cs.illinois.edu/class/fa05/cs598anh/slides/Bell2005_AlCoYvDe2005.pdf>, [lipanj 2011].

[2] Mosegaard, J.: Cardiac Surgery Simulation, PhD thesis, Department of Computer Science, University of Aarhus, Denmark, 2006., <http://jespermosegaard.dk/Publications/Mosegaard2006Phd.pdf>, [lipanj 2011].

[3] Czilli, G.: Particle-based simulation of deformable bodies, Department of compueter science, Friedrich Alexander Universitat, Nurnberb, Germany, 2010., <http://www10.informatik.uni-erlangen.de/Publications/Theses/2010/Czilli_BA_10.pdf>, [lipanj 2011].

[4] Kopf, J., Fu, C.W., Cohen-Or, D., Deussen, O., Lischinski, D., Wong, T.T.: Solid Texture Synthesis from 2D Exemplars, *Proceedings of ACM SIGGRAPH 2007, Volume 26 Issue 3, July 2007.* <http://johanneskopf.de/publications/solid/index.html>, [lipanj 2011].

[5] Bresenham's line algorithm, Wikipedia, <http://en.wikipedia.org/wiki/Bresenham's_line_algorithm>, [lipanj 2011].

[6] OpenGL Picking Made Easy, <http://web.engr.oregonstate.edu/~mjb/cs553/Handouts/Picking/picking.pdf>, [lipanj 2011].

[7] GLUI User Interface, <http://www.cs.unc.edu/~rademach/glui/>, [lipanj 2011].

14 Sažetak

Modeliranje i simulacija mekih objekata sustavom masa i opruga

Ovaj rad opisuje način kreiranja i simulacije prikaza modela mekih objekata u stvarnom vremenu. Na početku je opisan i definiran fizikalni model sustava čestica povezanih oprugama. Sustav čestica se tada pakira u kocke kao građevne jedinice volumena kojima se izgrađuje fizikalna struktura. Izgrađena fizikalna struktura se zatim modelira kako bi opisivala oblik dinamički učitanog krutog objekta. Građevne kocke modelirane fizikalne strukture se tada povezuju sa vrhovima mekog objekta kako bi deformacija fizikalne strukture utjecala i na deformaciju njegove površine.

Nakon kreiranja modela mekog objekta prelazi se na opis njegove reprezentacije i simulacije. Predstavljena je implementacije programa i njegovo korisničko sučelje. Kroz korisničko sučelje omogućena je manipulacija parametara mekog objekta i simulacija djelovanja vanjske sile nad njime. Zadnji dio rada daje pregled performansi i mogućnosti buduće nadogradnje.

Ključne riječi: simulacija modela mekih objekata, sustav čestica povezanih oprugama, metoda selektiranje, volumna tekstura

Abstract

Modeling and simulating soft body objects using spring mass system

This paper describes creation and simulation of soft body objects in real time. In the first part of paper physical model is introduced through the spring-mass system. Spring-mass system is then represented with cubes as construction units of physical structure. Built physical structure is then modeled to describe a form of rigid object. Construction cubes of physical structure are then connected with object vertices so that deformation of physical structure affects deformation of its surface.

Next step is description of soft body object representation and simulation. Afterwards paper summarizes program implementation and user interface. The user interface enables manipulation of soft body parameters and simulation of outside force acting upon it. The last part of this paper provides an overview of performances and possibilities for future updates.

Key words: soft body simulation, spring mass system, selection method, volume textures