SVEUČILIŠTE U ZAGREBU

**FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA**

DIPLOMSKI RAD br. 574

# SIMULACIJA ELASTIČNIH OBJEKATA

Nikola Martinec

Zagreb, lipanj 2013.

# Sadržaj

# 1. Uvod

Upotreba 3D modela sveprisutna je u računalnoj grafici, no u većini slučajeva ti modeli su kruti, odnosno koristi se animacija kako bi se postigli razni efekti. Meka tijela imaju sposobnost promjene položaja svojih vrhova čime se njihov prvotni oblik deformira [1]. Kako bi se omogućila promjena položaja vrhova potrebne su računalne simulacije mekih tijela kojima se određuje novi oblik tijela. U ovom radu razrađene su dvije metode za simulaciju mekih tijela – metoda s oprugama i metoda usklađivanja oblika [2]. Prva metoda se temelji na fizikalnom modelu opruga koje svojim skupljanjem i rastezanjem deformiraju tijelo. Druga metoda je motivirana geometrijom modela gdje se deformacije ostvaruju otklonom vrhova modela od početnog oblika. Početni oblik služi kao ciljna pozicija te se otklonjeni vrhovi usmjeravaju prema njemu. Napravljene su programske implementacije navedenih simulacija mekih tijela te je dana njihova ocjena.

# 2. Usklađivanje oblika

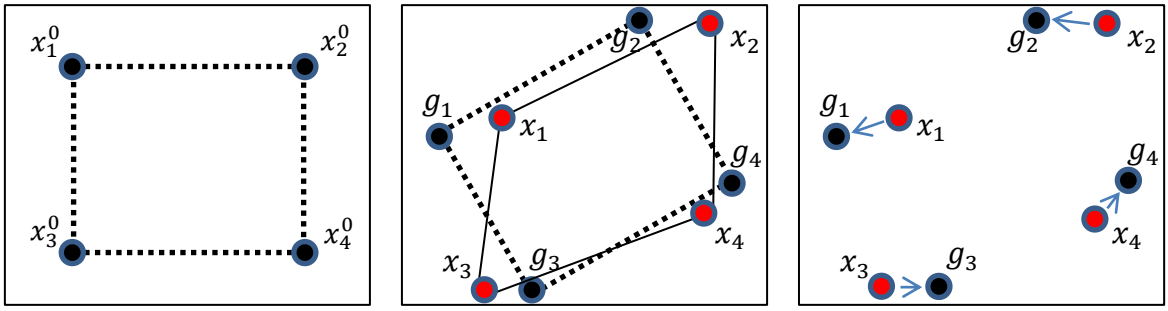
Djelovanjem sile na tijelo koje nije u potpunosti kruto dolazi do njegove deformacije. Djelovanjem unutarnjih sila, elastična tijela teže povratku u početno, ravnotežno stanje. Korištenjem tehnike prilagođavanja oblika (eng. shape matching) iz skupa točaka deformiranog tijela i skupa početnih položaja točaka tijela moguće je izračunati novi skup točaka koji predstavlja položaje točaka tijela na njegovu putu nazad u ravnotežni položaj. Unutarnje sile koje vraćaju tijelo u početni položaj zamijenjene su udaljenostima deformiranih položaja točaka od točaka dobivenih tehnikom prilagođavanja oblika. Svakoj deformiranoj točci odgovara točno jedna točka početnog oblika tijela, zbog čega se usklađivanje oblika zapravo svodi na pronalazak najbolje linearne transformacije kojom se želi uskladiti početni i deformirani oblik tijela. Metodom numeričke integracije određuju se nove vrijednosti brzine i položaja točaka tijela.

Postavlja se pitanje na koji način dobiti položaj deformiranog tijela. Jedna od mogućnosti je predstaviti tijelo sustavom čestica. Pod pretpostavkom da je tijelo opisano mrežom

vrhova na mjesto vrhova postavljaju se čestice koje ne sadrže nikakvu informaciju o povezanosti tijela. Čestice se modeliraju tako da imaju atribute koji pohranjuju njihov položaj, brzinu i masu. Kako bi se simuliralo ponašanje čestica u stvarnosti dodaje se gravitacijska sila te se novi položaj i brzina u vremenu računaju jednostavnim postupkom numeričke integracije. Pri prvom međudjelovanju čestica s podlogom dolazi do deformacije tijela (čestice se odbijaju od podloge) pa se deformirane pozicije koriste u prije navedenom postupku usklađivanja tijela.

## 2.1 Postupak usklađivanja oblika

Usklađivanje oblika je tehnika koja se koristi prilikom traženje sličnosti između zadanog i danog tijela. Npr. čovjek može za neku životinju s velikom sigurnošću reći da je ta životinja pas iako možda tu pasminu nikada u životu nije vidio. Dovoljna je općenita slika psa koju ima u sjećanju kako bi prepoznao da je promatrana životinja zaista pas. Uz pomoć tehnike usklađivanja oblika nastoji se postići automatsko prepoznavanje oblika u računalnom svijetu. Neki od problema te tehnike su nalaženje podudarnosti između dva tijela, transformiranje jednog tijela u drugo, mjerenje sličnosti itd. Na sreću, u prikazanoj metodi korištenja tehnike usklađivanja oblika ti problemi su uvelike pojednostavljeni. Naime, unaprijed je poznat položaj deformiranog tijela ($x_i$) kao i podudarnosti točaka početnog ($x_i^0$) i deformiranog tijela. Jedino što preostaje je pronaći transformaciju kojom se početan oblik tijela „usklađuje“ s deformiranim. Dobivene točke zovu se ciljni položaji (eng. goal positions, ($g_i$)). Na sljedećoj slici (Slika 1) prikazana su tri osnovna koraka za provođenje usklađivanja oblika.



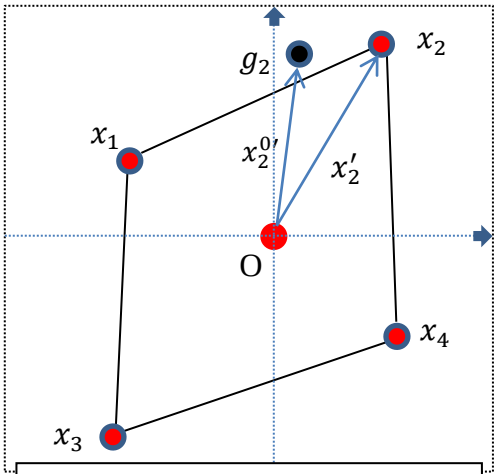

Slika 1. Prikaz rada algoritma temeljenom na usklađivanju oblika

Transformacija kojom se iz početnih i deformiranih točaka tijela računaju ciljne pozicije sastoji se od dva djela – translacije $tr$ i rotacije $R$. Pretpostavimo postojanje dva skupa točaka, početne točke $x_i^0$ i točke nakon deformacije $x_i$. Translacija $tr$ rastavlja se na dva vektora $t_0$ i vektor $t$. Kako bi se dobila najbolja moguća transformacija potrebno je minimizirati sljedeći izraz:

$$\sum_i^n w_i \, (R(x_i^0 - \, t_0) + t - \, x_i)^2,$$

gdje je $n$ ukupan broj vrhova.

Navedeni izraz predstavlja sumu kvadrata razlike vektora oblika $x_i^{0\prime} = \, R(x_i^0 - \, t_0)$ i $x_i' = x_i - t$. Vektori $x_i'$ predstavljaju translaciju u središte koordinatnog sustava pri čemu je težište tijela u ishodištu. Slično je i za vektore $x_i^{0\prime}$ uz dodatak rotacije. Na Slici 2 prikazana je jedna ciljna točka, deformirani oblik i navedeni vektori.




Slika 2. Translatirani položaji u ishodište koordinatnog sustava

Optimalna transformacija dobiva se upravo kada je $t_0$ centar mase skupa točaka $x_i^0$, a $t$ centar mase skupa točaka $x_i$. $w_i$ je težina kojom svaka točka sudjeluje u sumi te je logično za tu težinu postaviti masu čestice čiji se položaj promatra. Čestice mogu imati različite mase, no to nije nužno te su u implementaciji korištene jednake mase. Za izračun centra mase koriste se sve čestice $(n)$.

$$t_0 \;=\; x_{cm}^0 \;=\; \frac{\sum_i^n m_i\, x_i^0}{\sum_i^n m_i}$$

$$t \;=\; x_{cm} \;=\; \frac{\sum_i^n m_i x_i}{\sum_i^n m_i}$$

Rotacijska matrica $R$ može se izračunati iz optimalne linearne transformacije $A$ koja se dobiva minimizacijom sljedećeg izraza:

$$f(A) = \sum_i^n m_i (A q_i - p_i)^2$$

gdje je $p_i = x_i - x_{cm}$, a $q_i = x_i^0 - x_{cm}^0$. Deriviranjem se dobiva:

$$f(A)' = 2\sum_i^n m_i\,(A q_i - p_i) q_i$$

$$f(A)' = 2\sum_i^n m_i\,(A q_i - p_i) q_i$$

$$f(A)' = 2\sum_i^n (m_i\, A q_i q_i^T - m_i p_i q_i^T).$$

Izjednačavanjem izraza s nulom dobiva se optimalna linearna transformacija $A$:

$$\sum_i^n (m_i\, A q_i q_i^T - m_i p_i q_i^T) = 0$$

$$\sum_i^n (m_i\, A q_i q_i^T) = \sum_i^n (m_i p_i q_i^T)$$

$$A \sum_i^n (m_i\, q_i q_i^T) = \sum_i^n (m_i p_i q_i^T)$$

$$A = \sum_i^n (m_i p_i q_i^T) \sum_i^n (m_i\, q_i q_i^T)^{-1} = A_{pq} A_{qq}$$

Rotacijska matrica može se izlučiti iz matrice $A$ koristeći polarnu dekompoziciju $A = RS$. Simetrični dio te matrice je matrica $S$ te se on može izračunati:

$$S \;=\; \sqrt{A^t A}.$$

Iz toga slijedi da je rotacijska matrica $R$:

$$R = AS^{-1}.$$

Pri računanju matrice $S$ pojavljuje se potreba za računanjem korijena matrice. Korijen neke matrice $M$ može se izračunati pretvorbom matrice u oblik $M = QM'Q^{-1}$ gdje je $Q$ matrica svojstvenih vektora matrice $M$, a $M'$ dijagonalna matrica [3]. Neka je $M = N^2$, odnosno $N$ je korijen matrice $M$. Tada je $N^2 = QM'Q^{-1}$. Iz toga slijedi da je $N = QM'^{\frac{1}{2}}Q^{-1}$. Matrica $M$ može se svesti na navedeni oblik koristeći Jacobijeve rotacije.

Korijen matrice $M'$ je jednostavno izračunati jer je ona dijagonalan matrica. Korijen dijagonalne matrice se lako računa tako da se korjenuju elementi na dijagonali matrice. No, postoji opasnost da su ti elementi negativni. Taj problem se može riješiti tako da se svi elementi matrice $M'$ manji od nule postave na nulu. Tijekom rada programa uočeno je da se taj slučaj pojavljuje kada se usklađivanje tijela provodi između originalnog trodimenzionalnog oblika te oblika koji je izgubio jednu dimenziju, odnosno svi su mu vrhovi u istoj ravnini.

Konačno, uz poznatu rotacijsku matricu $R$ mogu se izračunati ciljne pozicije tijela ($g_i$):

$$g_i = Rq_i + x_{cm}.$$

Računanje korijena tijekom izračunavanja matrice $S$ može se izostaviti. Tada u rubnim slučajevima (ekstremne vrijednosti parametara $\alpha$ i $\beta$ o kojima će biti uskoro riječ) sustav gubi stabilnost dok je u ostalima stabilan. Ciljne pozicije u slučaju kada je $\alpha$ nula (na čestice ne utječu ciljni položaji) počinju se jako razlikovati. S ispravno izračunatom rotacijskom matricom ciljne pozicije čestica odgovaraju izvornom za razliku od slučaja s ciljnim pozicijama izračunatim bez korištenja korjenovanja.

Slika 3. Na lijevoj slici prikazan je pravilno izračunat ciljni položaj za razliku od desne slike gdje nije korišteno korjenovanje

## 2.2 Sustav čestica

Svaki vrh tijela modeliran je kao zasebna čestica te se one kreću nezavisno. Djelovanjem gravitacije čestice ubrzavaju. Koristeći Eulerov postupak numeričke integracije u diskretnim vremenskim trenucima određuje se brzina i položaj svake čestice:

$$v_{k+1} = v_k + a\Delta T,$$

$$p_{k+1} = p_k + v_{k+1}\Delta T,$$

gdje je $a$ akceleracija čestice uzrokovana gravitacijskom silom, $v_k$ brzina i $p_k$ položaj u trenutnoj iteraciji, a $v_{k+1}$ i $p_{k+1}$ u sljedećoj. $\Delta T$ je vremenski korak numeričke integracije.

Očito je da ukoliko se na sustav čestica primijene navedene formule čestice će slobodno padati. Potrebno je odrediti granicu od koje će se čestice odbijati. Kad čestica prijeđe zadanu granicu, koja je radi jednostavnosti određena samo y koordinatom, čestica se smjesti na samu granicu te joj $y$ komponenta brzine promijeni predznak. Na taj način ostvaruje se efekt odbijanja od podloge. U tom slučaju $x$ i $z$ komponente brzine moraju se pomnožiti s nekim koeficijentom u rasponu od 0 do 1 kako bi se dodalo prigušenje. Kada ne bi bilo prigušenja (simulacija sile trenja) tijela bi se gibala jednolikom brzinom u $xz$ ravnini.

Kada čestica udari u podlogu te s toga promijeni smjer dolazi do nepodudaranja izvornog oblika tijela s trenutnim. Postupkom usklađivanja oblika računaju se ciljne pozicije te se čestice usmjeravaju prema njima. Ciljne pozicije računaju se nakon svake iteracije numeričke integracije sustava čestica. Položaj i brzina svake čestice mijenja se u ovisnosti o ciljnim položajima. Razlika ciljnog položaja i trenutnog položaja čestice daje vektor usmjerenja $u$. Svaka čestica se usmjerava koristeći njen vektor usmjerenja pomnožen s konstantom $\alpha$.

$$u_i = \alpha(g_i - x_i)$$

$\alpha$ je realan broj između 0 i 1 koji određuje koliko daleko se svaka čestica primiče svojoj ciljnoj poziciji. Ukoliko je $\alpha$ 1 sve čestice će odmah promijeniti svoj položaj u ciljni te tako simulirati ponašanje krutog tijela. Smanjenjem $\alpha$ u jednom vremenskom koraku tijelo neće moći poprimiti svoj originalan oblik te se na taj način simulira ponašanje mekih elastičnih tijela. Ako je $\alpha$ 0 ignoriraju se ciljne pozicije te se čestice kreću slobodno i nezavisno jer jedino ciljne pozicije sadrže informaciju o povezanosti. Nakon što se izračuna vektor $u$ neke čestice njen položaj se mijenja:

$$x_{i+k} = x_i + u_i.$$

Čestica prelaskom puta, koji je jednak $u_i$, dobiva brzinu i ta se brzina dodaje brzini u trenutku $i$:

$$v_{i+k} = v_i + \frac{u_i}{\Delta T}$$

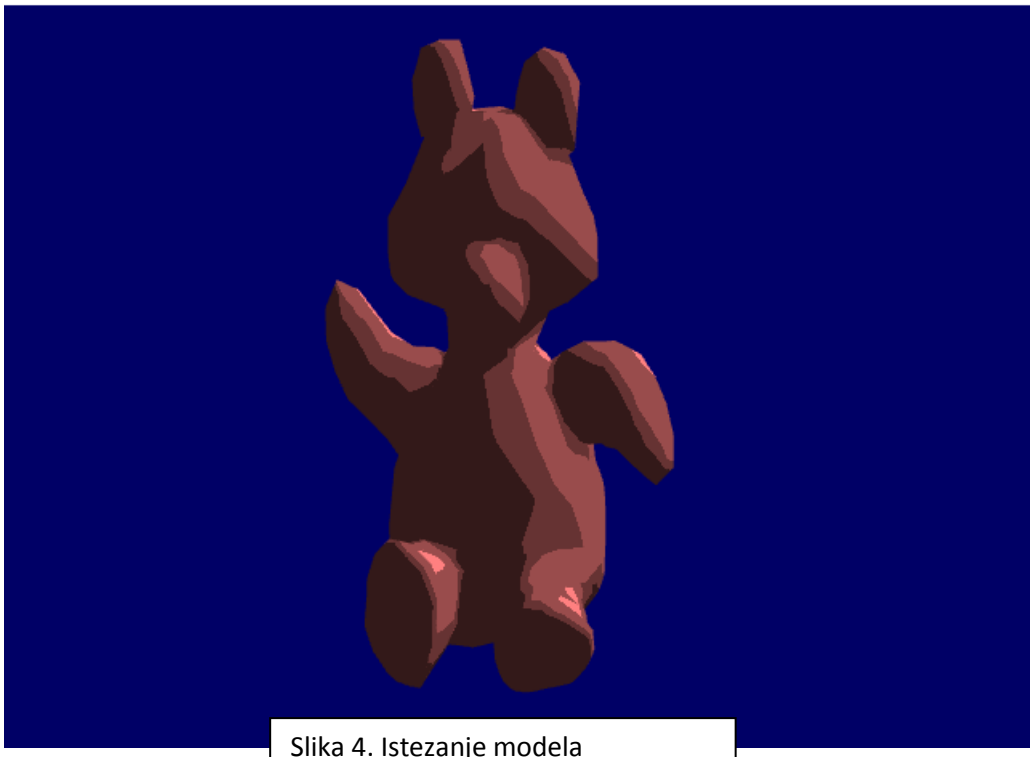
Ovdje valja primijetiti kako je udaljenost čestice od ciljne pozicije proporcionalna brzini koju će čestica poprimiti kada joj se primakne za iznos vektora $u_i$. Na taj je način iznos sile koji vraća tijelo u prvobitni oblik određen udaljenošću.

## 2.3 Vrste deformacija

Korištenjem samo matrice R kod računanja ciljnih položaja očuvan je početni oblik tijela jer matrica R sadrži samo rotaciju. Zbog toga je moguće ostvariti samo manje deformacije podešavanjem konstante $\alpha$. Zapravo se tada simulira kruto tijelo i moguće su vrlo mala odstupanja od početnog oblika. Uvrštavanjem u formulu za računanje ciljnih položaja

matrice $A$ dozvoljavaju se i linearne transformacije početnog oblika tijela (Slika 4). Ciljni položaji se tada računaju kao:

$$g_i = (\beta A + (1 - \beta) R) q_i + x_{cm} ,$$



Slika 4. Istezanje modela postignuto korištenjem i linearne transformacije $A$

gdje je parametar $\beta$ konstantan realan broj između 0 i 1. Kada je $\beta$ 0 koristi se samo rotacijska matrica $R$. Slučaj kada je $\beta$ 1 ne daje korisne rezultate, no u ostalim slučajevima postižu se veće deformacije u obliku rastezanja ili skupljanja početnog oblika.

## 2.4 Algoritam izvođenja simulacije

Nakon učitavanja modela i kreiranja sustava čestica od njegovih vrhova ponavljaju se četiri glavne faze:

a) Iteracija sustava čestica u vremenu $\Delta T$
b) Usklađivanje oblika
c) Korekcija položaja i brzine čestica u odnosu na ciljne pozicije u vremenu $\Delta T$
d) Iscrtavanje modela

## 2.5 Problemi

U promatranom radu *Meshless Deformations Based on Shape Matching* [2] za računanje rotacijske matrice koristi se matrica $A_{pq}$ navodeći kako ta matrica sadrži rotaciju koju je potrebno izlučiti. Pokazalo se da ta matrica, unatoč provedenom predlaganom postupku, sadrži i operaciju skaliranja. Koristeći matricu $A_{pq}$ dobivene ciljne pozicije odudaraju u velikoj mjeri od početnog oblika modela, a izvođenjem simulacije model pulsira. Eksperimentalnom metodom pokazalo se da je za računanje rotacijske matrice potrebno uzeti cijelu linearnu transformaciju $A$.

Računanje korijena matrice predstavlja složen problem koji se može riješiti korištenjem Jacobijevih rotacija [2]. S Interneta je preuzeta gotova funkcija [4], napisana u programskom jeziku C, koja rješava navedeni problem uz dodane modifikacije kako bi se spriječilo pojavljivanje negativnih brojeva tijekom korjenovanja elemenata dobivene dijagonalne matrice.

## 3. Iscrtavanje

Podaci o modelu kao što su vrhovi, normale i sl. pohranjuju se u posebne spremnike (eng. array buffer). Informacije o povezanosti vrhova u trokute pohranjuju se u spremnike nazvane spremnici elemenata (eng. element array buffer). U njima se nalaze indeksi vrhova pohranjenih u spremnicima vrhova koji čine poligone – za svaki trokut tri indeksa. Na taj način smanjuje se potrebni memorijski prostor za pohranu modela jer je svaki vrh pohranjen samo jednom. U slučaju kada se ne bi koristio spremnik elemenata spremnik vrhova morao bi sadržavati kopije vrijednosti položaja vrha za svaki trokut u kojem se taj vrh nalazi. Slično je i s normalama.

Položaj vrhova i normala mijenja se tijekom izvođenja programa te je spremnike vrhova i normala potrebno osvježavati prije izmjene spremnika boje koji će se iscrtavati. U implementaciji programa koriste se navedeni spremnici, umjesto uobičajenog poziva naredbi *glBegin* i *glEnd,* ne samo zbog boljih performansi nego i jer su u korištenoj verziji OpenGL-a te naredbe u potpunosti izbačene kao i neke druge poznate i često korištene naredbe (*glTranlate, glRotate…*). Također potrebno je napisati sjenčare (eng. shader) koji će obrađivati podatke o vrhovima.

## 4. Djelovanje vanjskih sila

Kako bi se omogućila interakcija korisnika s modelom potrebno je uvesti djelovanje vanjskih sila na model. Tu postoji više pristupa na koji način upravljati modelom. Jedan od načina je promjena vektora brzine čestica modela. Svim česticama modela može se dodati vektor brzine koji predstavlja djelovanje vanjske sile. Moguće je vektor brzine dodijeliti i samo određenom podskupu čestica modela, npr. česticama koje čine jedan trokut, no u tom slučaju dolazi ili do prevelike deformacije modela ili do zanemarivog djelovanja sile.

## 4.1 Odabir modela u sceni

Za potrebe upravljanja modelom u sceni implementiran je postupak kojim je moguće odabrati model klikom na miš [5]. Postupak se sastoji od dva djela. U prvom se svi trokuti modela iscrtaju u različitim bojama (Slika 4). Odabir boje temelji se na vrijednosti varijable koja sadrži tri komponente. Po jednu za crvenu, zelenu i plavu boju. Varijabla se povećava za jedan nakon svakog pridruživanja boje trokutu. Ne povećavaju se sve komponente varijable istovremeno jer bi tada na raspolaganju bilo samo 256 različitih boja. Umjesto toga prva komponenta se popunjava sve dok ne dođe do broja 255. U tom trenutku ta komponenta se resetira, a sljedeća se poveća za jedan. Npr. ako je varijabla u trenutku $k_0$ sadržavala vrijednosti (254, 0, 0) u trenutku $k_1$ biti će (255, 0, 0), zatim $k_2$ (0, 1, 0), $k_3$ (1, 1, 0), itd. Time je moguće iskoristiti $256^3$ različitih boja. To omogućava razlikovanje svih trokuta te u konačnici odabir pojedinog trokuta, odnosno samog modela. Trokut se odabire na temelju boje slikovnog elementa na kojem se nalazi kursor miša u trenutku pritiska na tipku miša. Inverznom metodom onoj za dodjeljivanje boja trokutima lako se dođe do indeksa odabranog trokuta.

Slika 4. Korištenje jedinstvene boje za svaki trokut modela

Korišteni ispitni model se ne iscrtava na zaslon, već se podaci o boji slikovnog elementa uzimaju samo iz odgovarajućeg spremnika boje. Nakon toga spremnik se briše i iscrtava se model koji je namijenjen prikazivanju na zaslonu.

## 4.2 Interakcija s modelom

Poželjno je omogućiti korištenje miša za interakciju, odnosno pomicanje modela u sceni. Vektor brzine koji će biti dodan svim česticama modela izračunava se kao promjena $x$ i $y$ koordinata miša, $\Delta x$ i $\Delta y$

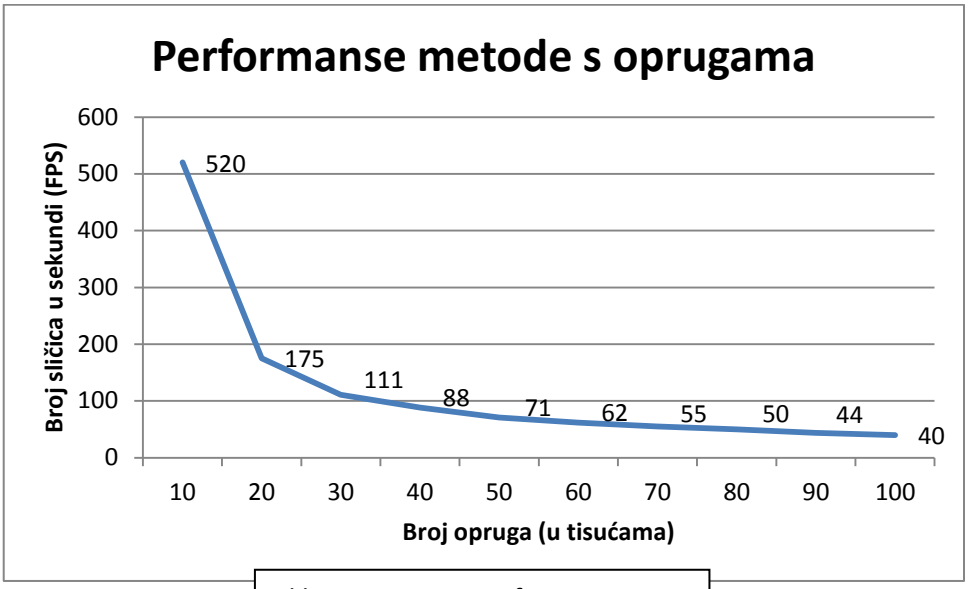
$$b = (\Delta x, \Delta \mathrm{y}, 0).$$

Zbog mogućnosti promjene položaja kamere u sceni nije dovoljno taj vektor jednostavno pridodati brzinama čestica modela. Vektor $b$ potrebno je pomnožiti s inverznom matricom položaja kamere (eng. view matrix). Na taj način model se može pomicati u svim smjerovima, ovisno o položaju kamere, npr. ako se scena promatra iz ptičje perspektive model se može pomicati u $xz$ ravnini. Kako se brzina kretanja modela ne bi mijenjala ovisno o broju sličica u sekundi (end. FPS) korisno je pomnožiti komponente vektora $b$ s umnoškom vremena proteklim između iscrtavanja dvije uzastopne scene i željene brzine.

# 5. Usporedba performansi

Testiranje brzine izvođenja mjereno je kao broj sličica u sekundi iscrtanih na zaslonu. Testiranje je izvršeno na računalu s procesorom AMD Phenom II 945 3.0 GHz i grafičkom karticom AMD Radeon HD6850. Valja napomenuti da je program preveden uz korištenje optimizacije brzine izvođenja (/O2) pri čemu se broj sličica u sekundi povećao za više od 400%.

Izvođenje simulacije mekih tijela upotrebom metode usklađivanja oblika je matematički znatno manje zahtjevno u usporedbi s metodom masa i opruga. Razlozi za to su brojni. Iako je računanje sila opruga fizikalno jednostavno, korištenje velikog broj opruga usporava se vrijeme računanja zbog kvadratne složenosti izračuna simulacije. Da bi se postigla čvrstina potrebno je koristiti velik broj opruga. Veća složenost modela (više vrhova i poligona) negativno utječe na performanse zbog potrebe za korištenjem više opruga.



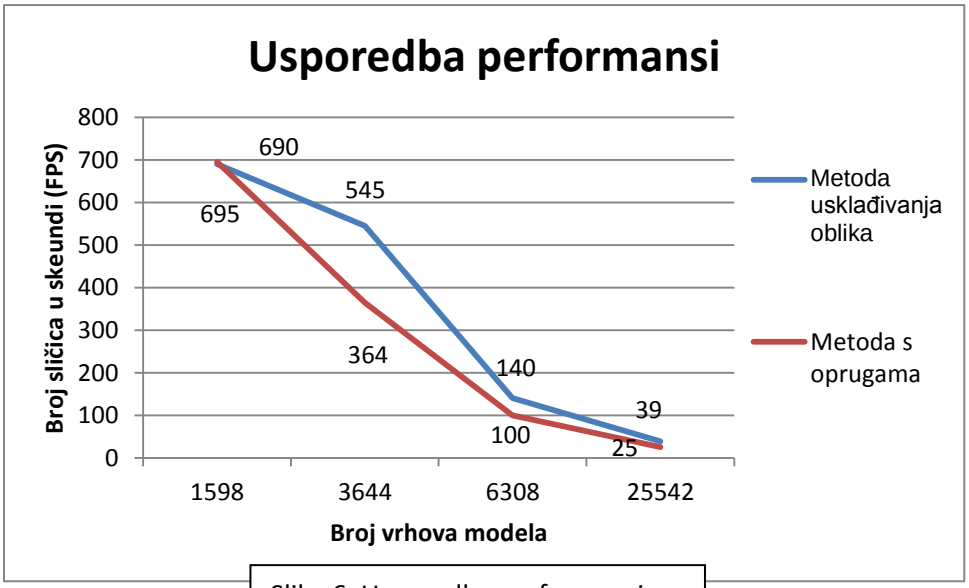

Slika 5. Kretanje performansi u ovisnosti o broju opruga

Na gornjem grafu prikazan je rezultat testiranja simulacije koristeći opruge. Test je proveden koristeći jednostavniji model s 1598 vrhova. Povećanjem broja opruga brzina izvođenja naglo pada. Količina opruga uvjetuje mekost ili tvrdoću modela i ovakav pad performansi uzrokuje uzak dio iskoristivosti u praktičnoj primjeni.

Sa složenijim modelom (6308 vrhova) rezultati su još gori. Zbog povećanja broja vrhova vrijeme iscrtavanja se povećava, a istovremeno se ne može postići slična tvrdoća – veliki dio od zadanog broja opruga iskorištava se za popunjavanje bridova poligona pa stoga ostane manji dio za unutrašnjost modela.

Metoda usklađivanja tijela znatno je jednostavnija. Na složenost simuliranog modela utječe samo broj vrhova jer su informacija o povezanosti vrhova nepotrebne za provođenje simulacije. Računanje Jacobijevih rotacije ne ovisi o složenosti modela jer se uvijek izvršava nad matricom istih dimenzija. Također, složenost izračuna je linearna.

Nije lako dati usporedbu performansi ovih metoda jer one ovise o različitim parametrima te su u srži vrlo različite. Metoda usklađivanja oblika ovisi samo o broju vrhova modela dok metoda s oprugama ovisi o broju vrhova modela, ali i broju opruga. Stoga je moguće da će, inače sporija, metoda s prugama u nekim slučajevima čak i biti brža. U prikazu usporedbe performansi (Slika 6.) za početni broj vrhova broj opruga je sveden na minimum kako bi se dobila početna točka usporedbe gdje su performanse gotovo jednake. Porastom broja vrhova i dalje je broj opruga na minimumu te metoda s oprugama jedva drži korak s metodom usklađivanja oblika. Bitno je i napomenuti da s minimalnim brojem opruga model nema nikakvu čvrstinu tako da se model ponaša kao tkanina.




Slika 6. Usporedba performansi dviju metoda simulacije mekih tijela

# 6. Metoda s oprugama i masama

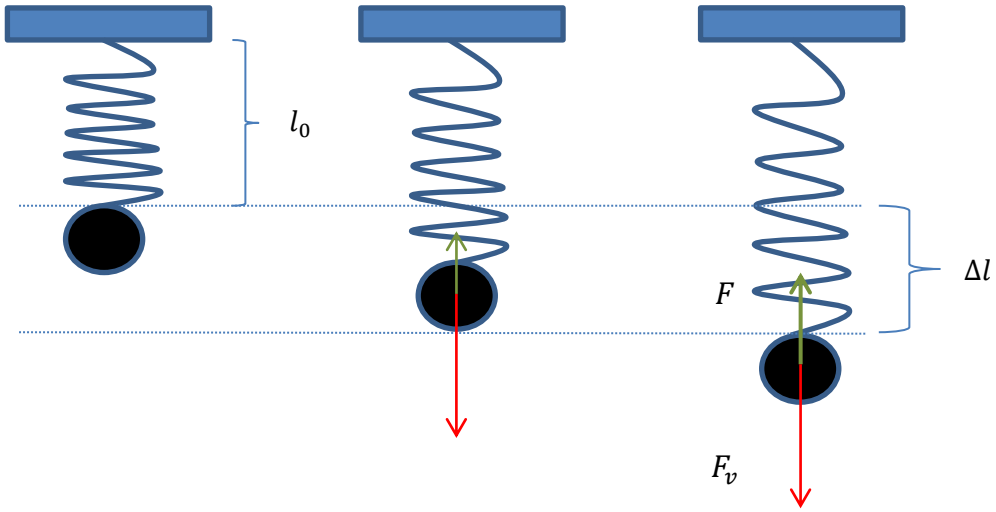
Elastičnost tijela može se simulirati koristeći jednostavan fizikalan model opruge [6]. Oprugama se popuni površina modela određena trokutima tako da se na mjesto bridova trokuta postave opruge. Zatim se popunjava unutrašnjost modela kako bi se dobilo na čvrstini. Za ovu metodu od velike je važnosti odabir metode numeričke integracije kojom se računaju novi položaji i brzine vrhova modela. Korištena je Eulerova metoda integracije zbog brzine, ali je kao posljedica stabilnost sustava slabija. Razlog slabije stabilnosti leži u mogućem krivom računanju brzina i pozicija pri čemu se narušava zakon o očuvanju energije te se vrhovi modela raspršе.

## 6.1 Eulerova metoda integracije

Prije analize Eulerove metode integracije valja prikazati model opruge koji će biti korišten. Najvažniji podatak o opruzi je konstanta opruge koja ovisi o dimenziji, obliku i materijalu opruge. Na kraju opruge nalazi se kuglica zanemarivih dimenzija mase $m$. U sljedećem razmatranju pretpostavka je da je jedan kraj opruge nepomičan dok se u implementaciji na svakom kraju opruge nalazi masa. Kada se tijelo nalazi u ravnotežnom položaju kuglica miruje. Ako se djelovanjem sile $F_v$ kuglica nađe izvan ravnotežnog položaja opruga djeluje silom u suprotnom smjeru od promjene. Navedeno je poznato kao Hookov zakon

$$F = -k\Delta x,$$

koji kaže da je sila proporcionalna umnošku promjene dužine opruge i konstanti opruge (Slika 7).




Slika 7. Hookeov zakon opruge

U nekom trenutku $t$ kuglica se nalazi na položaju $x(t)$ te je elastična sila jednaka $F = -k(x(t) - l_0)$. Koristeći modificirani Eulerov postupak brzina u sljedećem vremenskom odsječku $\Delta T$ računa se kao

$$v(t + \Delta T) = v(t) + \Delta T \frac{-k((x(t) - l_0)}{m}.$$

Brzina je izračunata eksplicitnim Eulerovim postupkom i ta brzina se koristi kao procjena u implicitnom Eulerovom postupku kojim se računa položaj

$$x(t + \Delta T) = x(t) + \Delta T v(t + \Delta T).$$

Problem kod Eulerovog postupka je preciznost. Ako se uzme preveliki vremenski korak $\Delta T$ energija sustava se naglo povećava te raste ukupna energija sustava. Ako je $\Delta x = x(t + \Delta T) - x(t)$ razlika između dva uzastopna pomaka slijedi

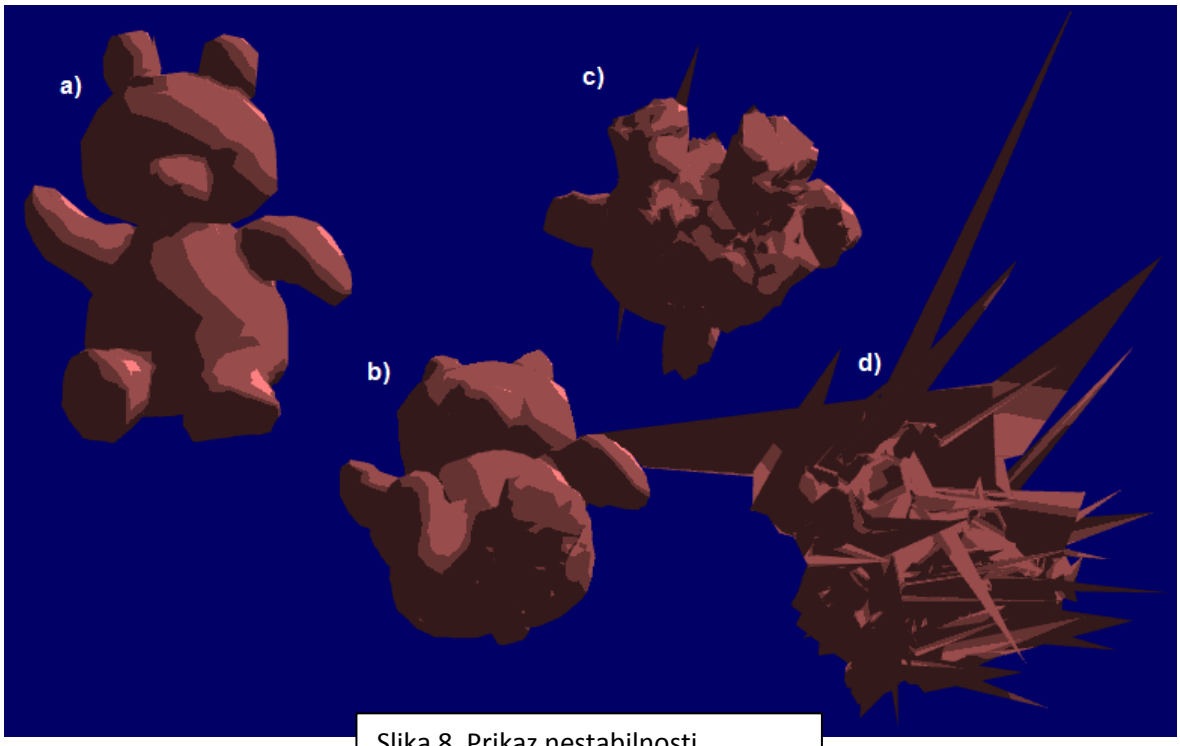
$$\Delta x = \Delta T v(t + \Delta T)$$

$$\Delta x = \Delta T(v(t) + \Delta T \frac{-k((x(t) - l_0)}{m})$$

Uz pretpostavku da je u trenutku $t$ brzina, odnosno $v(t) = 0$

$$\Delta x = \Delta T^2 \frac{-k((x(t) - l_0)}{m}$$

Uz preveliki $\Delta T$ ili $k$, ili premali $m$, može se dogoditi da je promjena duljine opruge veća nego u prethodnom trenutku. U skladu s tim, sila koja pokušava vratiti oprugu u ravnotežno stanje biti će veća od sile u prethodnom trenutku. Navedeno stanje pokreće lančanu reakciju u kojoj sila opruge iz iteracije u iteraciju raste, kao i pomaci tijela na opruzi (Slika 8).




Slika 8. Prikaz nestabilnosti simulacije zbog nepovoljno postavljenih parametara

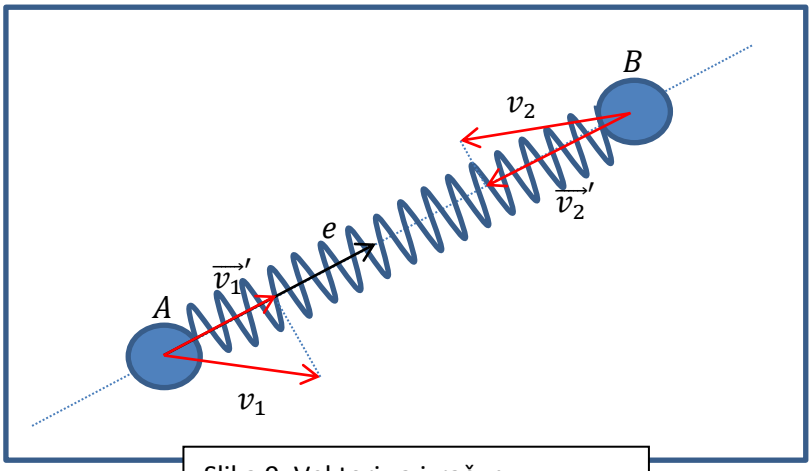
## 6.2 Fizikalni model simulacije

Sam Hookeov zakon nije dovoljan kako bi se napravio fizikalni model simulacije. Na kuglice na krajevima opruga djeluje i gravitacijska sila. Kako bi se simuliralo prigušenje s

ciljem da opruge ne titraju bez prestanka uvodi se i sila prigušenja. Sila prigušenja definira se kao

$$F_b = -b(\overrightarrow{v_2}' - \overrightarrow{v_1}')$$

gdje su vektori $\overrightarrow{v_2}'$ $i$ $\overrightarrow{v_1}'$ vektorske projekcije vektora $\overrightarrow{v_2}$ $i$ $\overrightarrow{v_1}$ na jedinični vektor smjera $\overrightarrow{e}$ (Slika 9). Jedinični vektor smjera jednak je razlici vektora položaja kuglice $B$ $i$ $A$.




Slika 9. Vektori za izračun prigušenja

Rezultantna sila koja djeluje na neki vrh, odnosno kuglicu, može se izračunati kao zbroj svih sila koje djeluju na njega — sve sile opruga kojima je povezan, pripadnih sila prigušenja opruga i gravitacijska sila

$$F_r = mg + \sum_{i=1}^{n}[k\Delta l_i + b(v'_{2_i} - v'_{1_i})].$$

Nakon što se izračuna rezultantna sila u svim vrhovima računa se novi položaj vrhova Eulerovim postupkom numeričke integracije

$$v_{k+1} = v_k + a_n\Delta T,$$

$$p_{k+1} = p_k + v_{k+1}\Delta T,$$

gdje je $a_n$ akceleracija promatranog vrha te se može izraziti preko ukupne sile koja djeluje na taj vrh i njegove mase.

$$a_n = \frac{F_{r_n}}{m_n}.$$

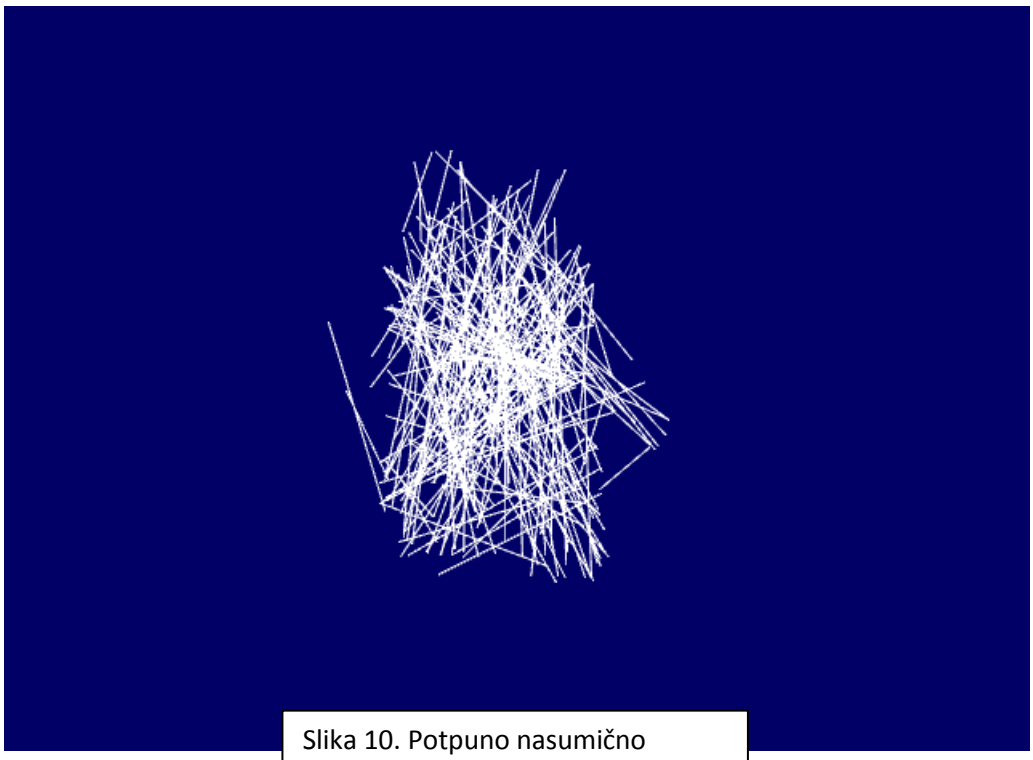
Važno je napomenuti da je potrebno koristiti dva seta vrhova prilikom izračuna novih brzina i položaja. Jedan set čuva stare vrijednosti tijekom izračuna novih. Nove vrijednosti ne zamjenjuju stare sve dok nije završeno računanje novih vrijednosti za sve vrhove. Pretpostavimo da nije tako i nove vrijednosti odmah zamijene stare. Promatramo objekt s tri vrha. Vrh $A$ koji je oprugama povezan s vrhovima $B$ i $C$. Pretpostavimo da se u diskretnom trenutku $k$ računaju novi položaj i brzina za vrh A, pri čemu nove vrijednosti zamjene stare. Za proračun rezultantne sile također u trenutku k za vrh $B$ koristit će se podaci o vrhu $A$ u trenutku $k+1$ i podaci o vrhu $C$ u trenutku $k$. Upravo kako bi se spriječilo miješanje podataka iz različitih vremenskih odsječaka uvedeno je korištenje dva skupa vrhova.

## 6.3 Raspoređivanje opruga

Način na koji će se rasporediti opruge i količina opruga ima veliku ulogu u stabilnosti, ali i vizualnom rezultatu simulacije. Opruge se moraju postaviti na mjesto svih bridova trokuta od kojih je model napravljen. To je ujedno i minimalan broj opruga koji se za neki model mora definirati. U prikazima različitih načina postavljanja opruga nisu prikazane opruge koje su po površini modela jer su one uvijek jednake i njihovo prikazivanje samo bi otežalo promatranje opruga unutar modela.

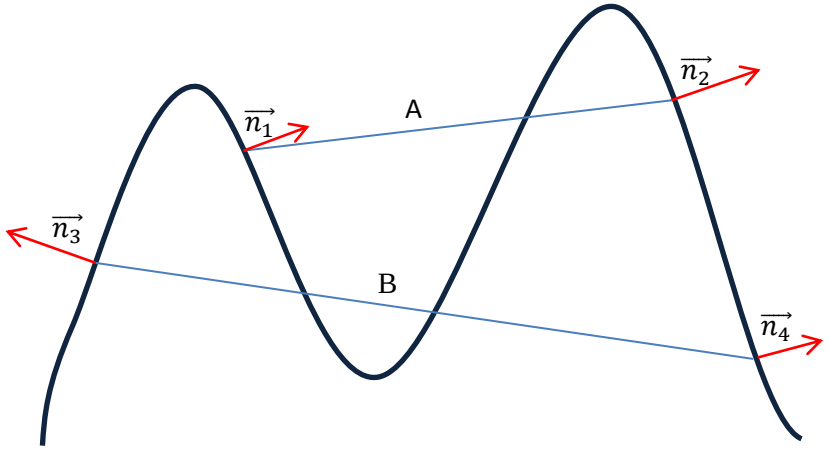
Najjednostavnije je opruge rasporediti nasumično. Na taj način postiže se kompaktnost modela jer se zbog raspršene mreže opruga svi dijelovi modela drže zajedno (Slika 10). Mijenjanjem vrijednosti konstante opruga tijelo postaje kruće ili mekše, no zbog nasumične povezanosti oprugama nemoguće je postići da se pojedini dijelovi modela kreću donekle samostalno. Npr. udovi modela medvjedića, za kojeg su prikazane opruge, ne mogu se slobodno pomicati. Kako bi se taj problem riješio potrebno je opruge

rasporediti uzimajući u obzir oblik tijela, odnosno ne dopustiti povezivanje oprugama izvan volumena modela.

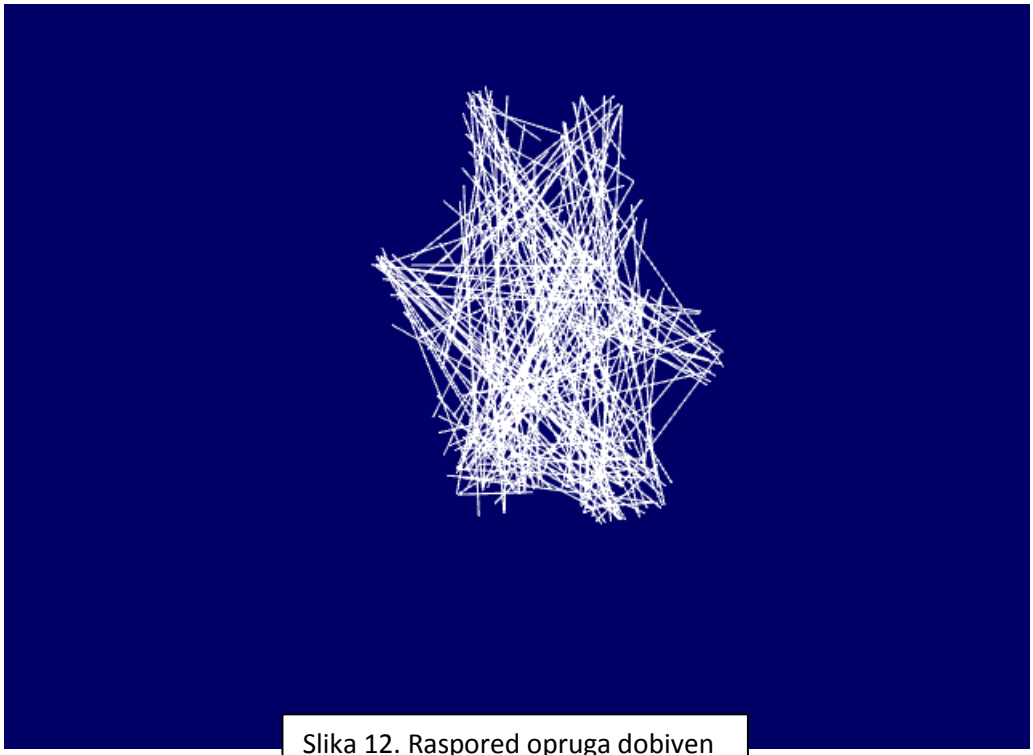


Slika 10. Potpuno nasumično raspoređivanje opruga

Nije trivijalno provjeriti za svaku oprugu prolazi li ili ne samo kroz unutarnji dio modela. Zbog toga se koristi dovoljno dobra metoda koja i dalje ne može spriječiti pojavu opruga koje su djelom ili u potpunosti izvan modela. Ideja se zasniva na tome da se opruge postavljaju samo između vrhova čije normale gledaju u suprotnim smjerovima. Npr. za dvodimenzionalni sustav ako normala početnog vrha opruge leži u prvom kvadrantu očekuje se da će normala krajnjeg vrha ležati u III. kvadrantu. Na taj način mogu se izbjeći neki slučajevi u kojima se model povezuje oprugama izvana, no i dalje postoje situacije kada ova metoda neće dati uspjeha (Slika 11). Unatoč tome, poboljšanje u odnosu na nasumično raspoređivanje bez provjere normale početnog i krajnjeg vrha je primjetno (Slika 12). Dok se kod nasumičnog odabira opruga gotovo ne može pretpostaviti oblik izvornog modela, u ovom se slučaju već naziru konture medvjedića.

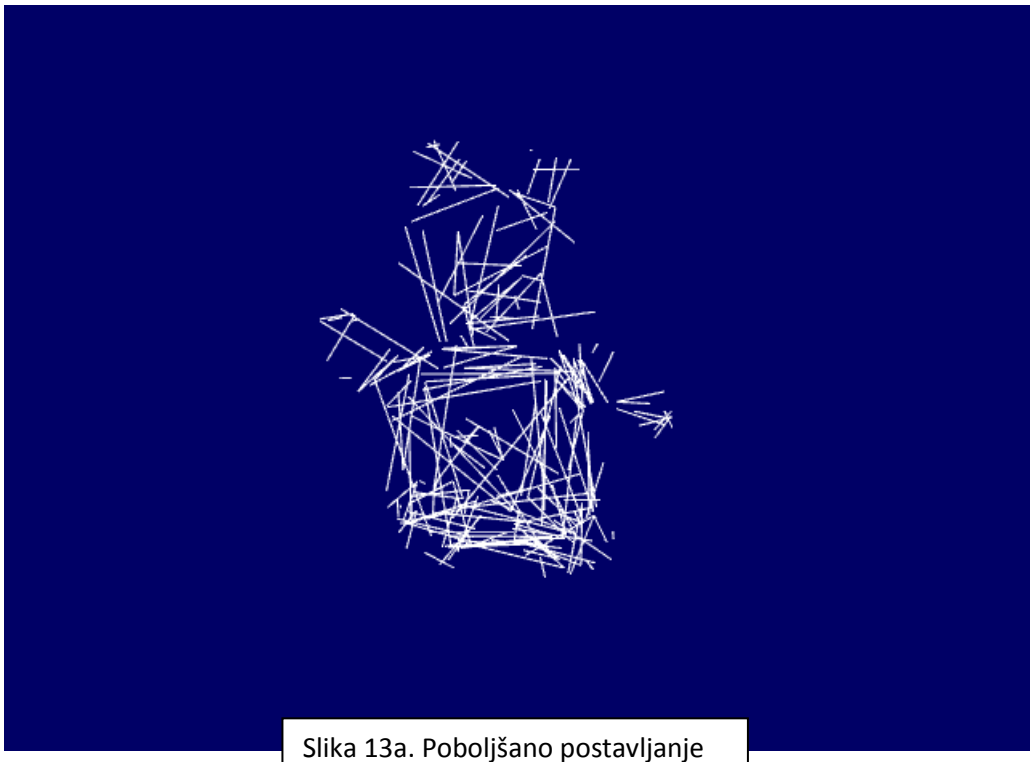



Slika 11. S obzirom na smjer normala opruga a je ispravno odbačena, a opruga B pogrešno prihvaćena



Slika 12. Raspored opruga dobiven uzimajući u obzir normale krajnjih točaka opruga

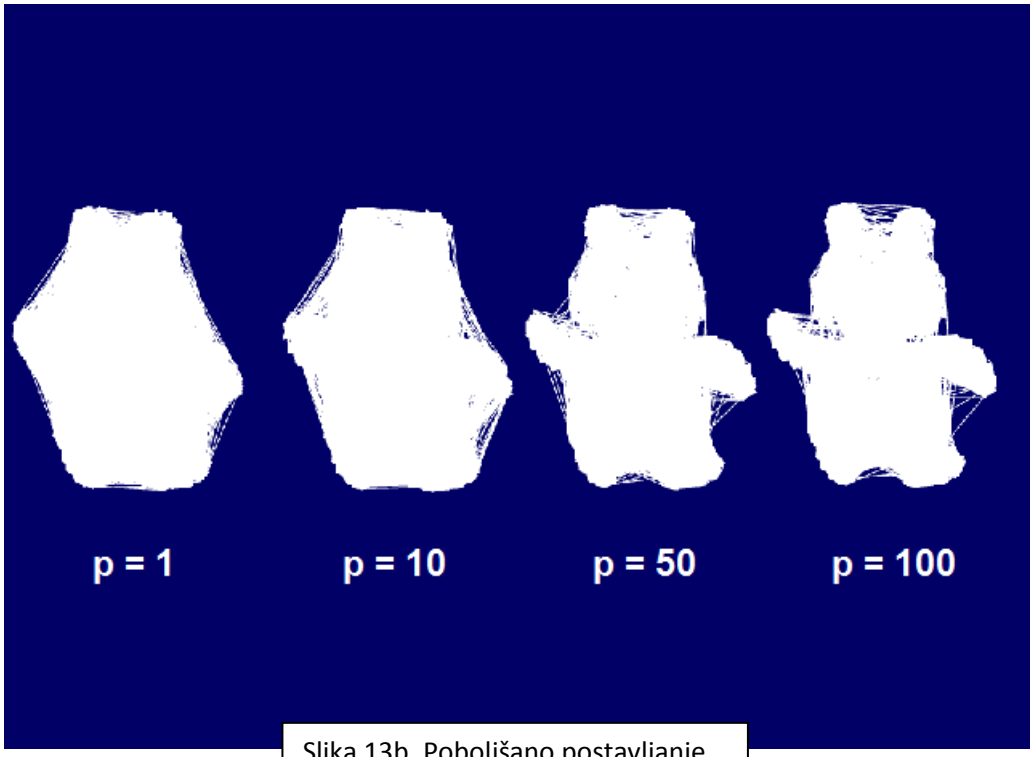
Ova tehnika može se modificirati tako da daje još bolje rezultate. Kada se za neki vrh traži drugi vrh kojim će se spojiti s oprugom nije potrebno uzeti prvi koji zadovoljava uvjet s normalama. Moguće je pamtiti $p$ potencijalnih vrhova te između njih odabrati najpovoljniji. Za najpovoljniji vrh uzima se onaj koji je najbliže početnom vrhu. Razlog tome je pretpostavka da kraće opruge imaju veću vjerojatnost da povezuju dva vrha unutarnjim putem. Ovakav pristup ne garantira da se neće dogoditi prihvaćanje pogrešno smještene opruge (primjer kao na slici 11), ali se ta vjerojatnost smanjuje povećanjem $p$. Što je veći $p$ biti će više opruga koje su unutar modela, ali će i opruge biti kraće (Slika 13a).



Slika 13a. Poboljšano postavljanje opruga

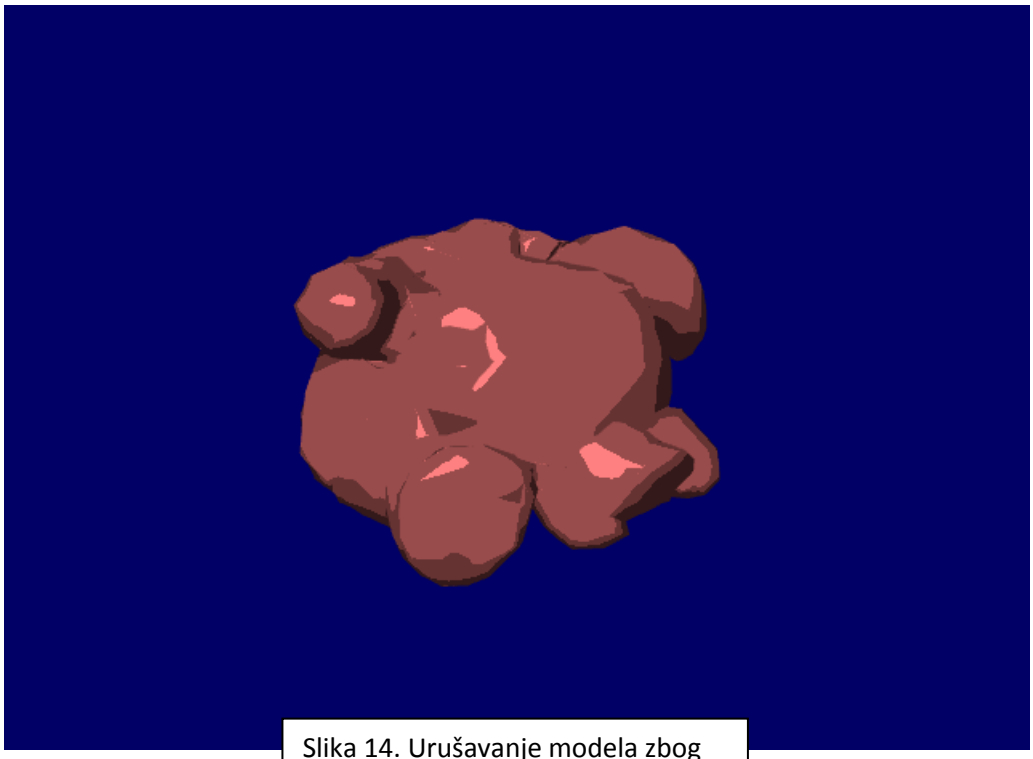
S većim brojem opruga lakše je vidjeti učinak ovog algoritma (Slika 13b) uz različit parametar $p$. Može se uočiti da razlika u vrijednosti parametara $p$ ne odgovara očekivanoj razlici u kvaliteti razmještaja opruga. Razlog tome je optimizacije algoritma koja je

rezultirala time da parametar $p$ ne mora nužno biti konstantna vrijednost. Za neke vrhove može se dogoditi da pronalazak točno $p$ opruga kandidata nije uopće moguć, a da bi se došlo do te spoznaje potrebno je proći kroz sve vrhove modela što rezultira značajnim usporavanjem. Rješenje ovog problema je u uvođenju parametra koji će poslužiti kao granica za broj vrhova koji se može ispitati zadovoljava li uvjet s normalama. Npr. ako je $p = 5$ i granica 10 te ako se unutar 10 ispitivanja vrhova ne nađe 5 potencijalnih opruga uzet će se u obzir samo do tada pronađene opruge kandidati.




Slika 13b. Poboljšano postavljanje opruga

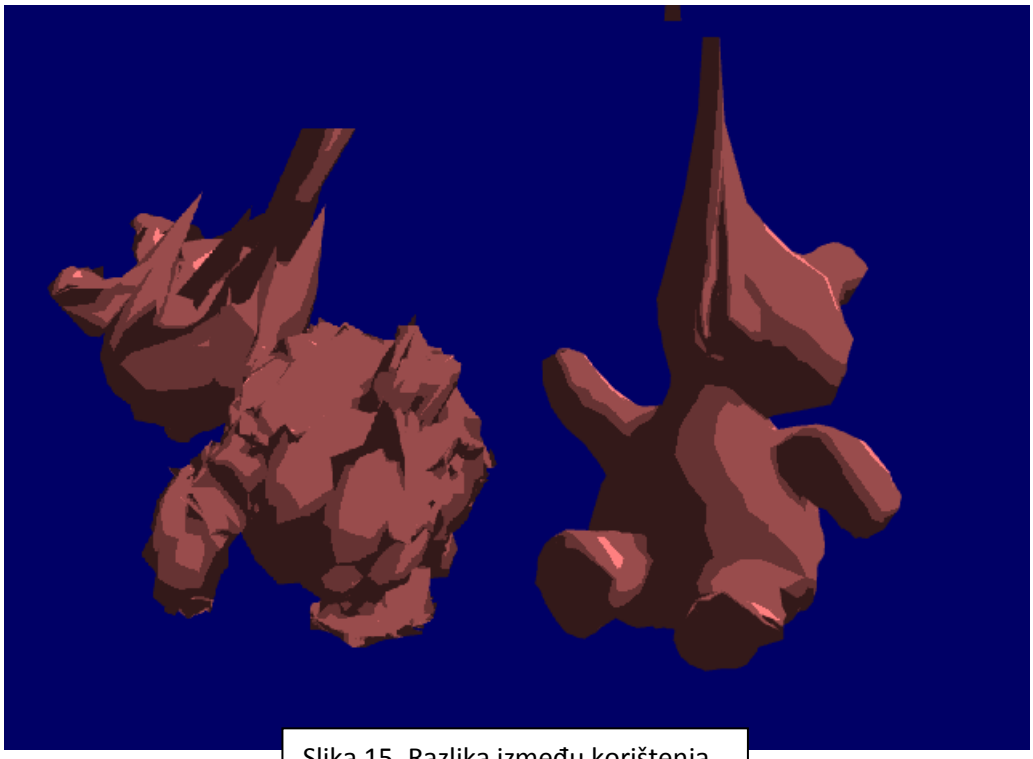
Zbog većeg broja kratkih opruga gubi se globalna povezanost. Posljedica toga može biti i urušavanje modela (Slika 14). Naime, upravo duge opruge koje se protežu modelom daju modelu čvrstoću. Stoga je potrebno vrijednost $p$ postaviti dovoljno malim kako bi ipak ostao dovoljan broj dužih opruga koje prolaze unutar ili izvan modela te se zadržala njegova kompaktnost. Nabrojane metode raspoređivanja opruga nisu preuzete iz postojećih radova već su izvorne autorove ideje.



Slika 14. Urušavanje modela zbog prevelikog broja kratkih opruga

## 6.4 Odabir vrijednosti konstante opruge $k$

Deformiranjem tijela stvaraju se mnoge sitne udubine i izbočine na površini modela. Svaki vrh je povezan s više opruga i svaka deformacija uzrokuje lančanu reakciju koja može vrlo jako utjecati na vrhove koji su prostorno daleko od mjesta gdje je deformacija nastala. Nastala nazubljenost se može u velikoj mjeri smanjiti uvođenjem površinske napetosti. Oprugama koje su smještene po površini modela dodjeljuje se višestruko veća konstanta opruge od onih u unutrašnjosti. Na taj način unutrašnje opruge neće moći tako lako deformirati površinske vrhove te će kao rezultat površina modela biti glađa. Nadalje, konstanta opruge ovisi i o duljini opruge pa se tako konstanta svake opruge još množi s faktorom koji je obrnuto proporcionalan duljini opruge. Utjecaj konstanti opruge najbolje se može vidjeti kad je jedan vrh modela fiksiran te ostatak visi na njemu (Slika 15).



Slika 15. Razlika između korištenja jednake vrijednosti konstante opruge (lijevo) i različitih (desno)

## 6.5 Algoritam izvođenja simulacije

Prije izvođenja simulacijske petlje potrebno je učitati i pripremiti model. Glavni korak u pripremi je raspoređivanje opruga. Kada je postavljen željeni broj opruga simulacije može započeti. Simulacija se sastoji od tri koraka:

1) Izračun sila, odnosno brzine i položaja vrhova modela

2) Zamjena starih vrijednosti novima

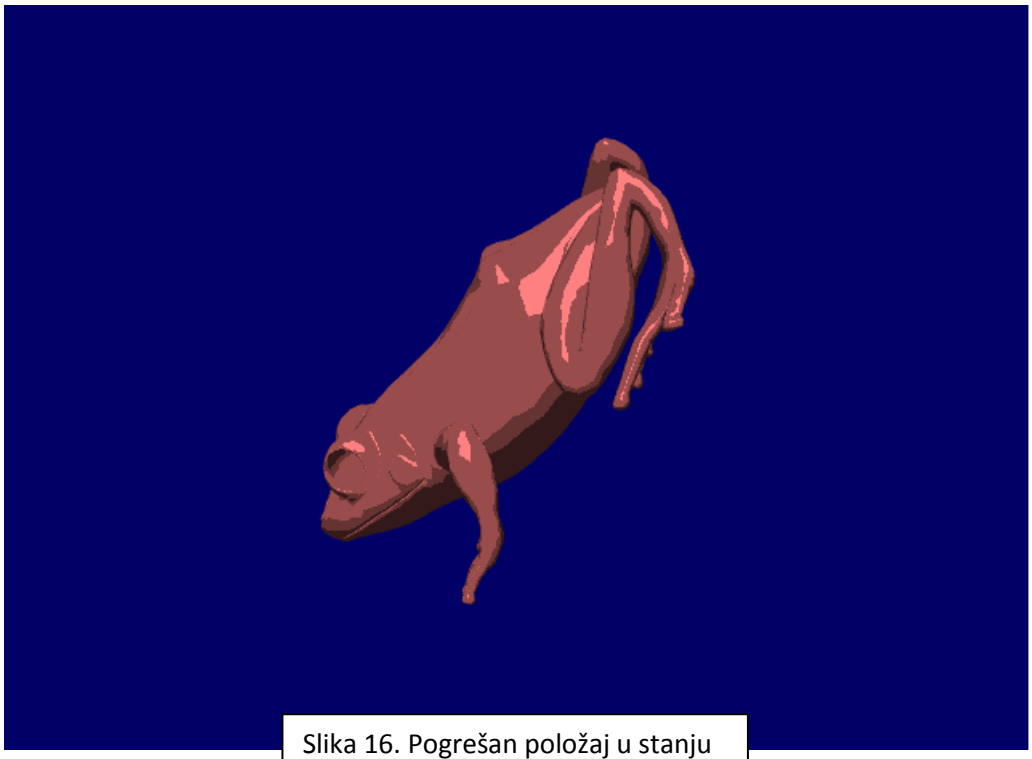
3) Iscrtavanje

## 6.6 Problemi

Mekoća modela ovisi o broju opruga, masi sadržanoj u vrhovima i konstantama opruga. Promjena bilo kojeg od tih parametara može uzrokovati nestabilnost simulacije stoga je potrebno pažljivo izabrati njihove vrijednosti. Kako bi se postigao jednaki efekt mekoće dvaju modela s različitim brojem vrhova model s  više vrhova mora imati i više opruga. Više opruga uzrokuje usporavanje simulacije što uvelike ograničava primjenu ove metode na modele manje složenosti.

# 7. Usporedba metode s oprugama i metode usklađivanja oblika

Obje metode imaju svoje prednosti i mane, stoga njihova primjena ovisi o namjeni. Najveća prednosti metode usklađivanja je njezina stabilnost i brzina izvođenja. Ova metoda je prikladna za simulaciju mekih objekata u igrama gdje je udio iskorištenog procesorskog vremena jako bitan, kao i stabilnost same simulacije. Razina mekoće lako se podešava promjenom parametara $\alpha$ i $\beta$ bez promjene performansi simulacije. Nedostatak ove metode je vizualna vjerodostojnost. Iako je površina modela uvijek glatka neovisno o deformaciji, deformacije su manje realistične nego kod metode s oprugama. S određenim modelima tijekom rada simulacije uočeno je neočekivano ponašanje modela, npr. tijelo zauzme položaj u stanju mirovanja koji fizikalno ne bi trebalo poprimiti (Slika 16)  ili se neprestano odbija od podloge.



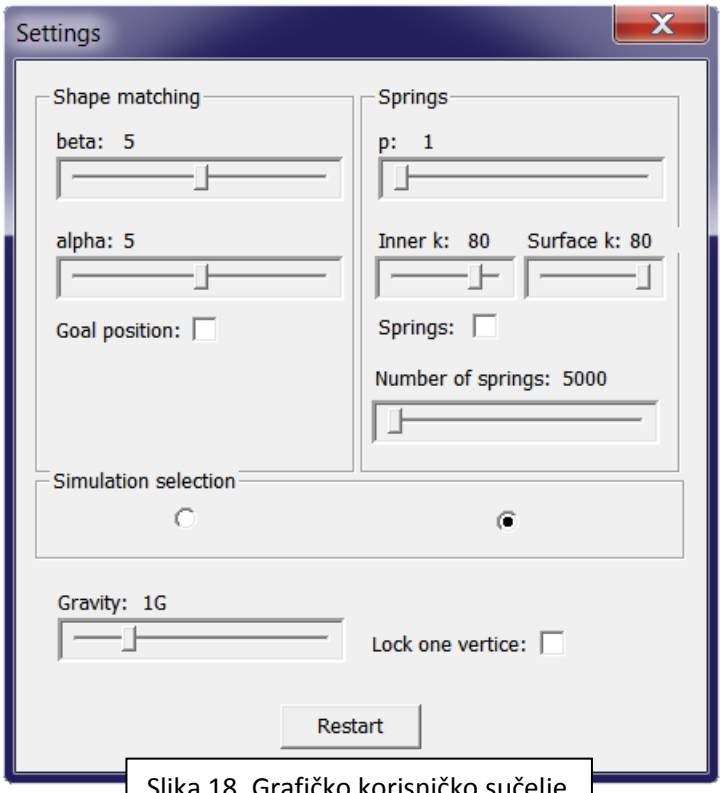
Slika 16. Pogrešan položaj u stanju mirovanja

Metoda s oprugama je pogodnija za simulacije mekih tijela jer je temeljena na fizikalnom modelu. Realističnost ponašanja simulacije je veća, ali uz osjetno manje performanse. Prilikom deformacija površina modela postane hrapava zbog različitih djelovanja sila na vrhove (Slika 17). Mijenjanje čvrstoće modela zahtijeva povećanje broja opruga ili konstanta opruga što može dovesti do prevelikog pada performansi za složene modele te neupotrebljivosti simulacije u stvarnom vremenu. Krivim odabirom parametara stabilnost simulacije može biti narušena što može predstavljati veliki problem kod aplikacija u stvarnom vremenu.

Slika 17. Površinske nepravilnosti zbog nagle deformacije modela

## 8. Korisničko sučelje

Tijekom izvođenja simulacije korisno je omogućiti promjenu raznih parametara o kojima ovise pojedine metode simulacije mekih tijela. Korisničko sučelje sastoji se od jednog prozora na kojemu su smještene kontrole za podešavanje vrijednosti parametara (Slika 18). Za metodu usklađivanja tijela omogućeno je mijenjane parametara $\alpha$ i $\beta$ te uključivanje prikaza ciljnih pozicija. Promjene parametara odmah se primjenjuju na simulaciju. Za metodu s oprugama moguće je mijenjati konstante unutrašnjih i vanjskih opruga, broj opruga te parametar koji utječe na odnos broja opruga koje se nalaze unutar modela i onih koje vrhove modela povezuju izvana ili sijeku model. Također je moguće uključiti prikazivanje opruga modela. Za razliku od metode usklađivanja tijela, kako bi promjena parametara djelovala na simulaciju potrebno je pritisnuti gumb za ponovno pokretanje simulacije.




Slika 18. Grafičko korisničko sučelje za promjenu parametara simulacije

Za obje vrste simulacija moguće je mijenjati vrijednosti gravitacijske sile. Još jedna zajednička opcija omogućava zaključavanje trenutnog položaja jednog vrha modela (Slika 15). Unutar sučelje može se odabrati koja simulacije će se pokrenuti.

## 9. Implementacija

Program za simulaciju je napisan u programskom jeziku C++ koristeći razvoji alat Microsoft Visual Studio 2010. Korištena verzija programskog sučelja OpenGL-a je 3.3 [7]. Za učitavanje OpenGL ekstenzija, kreiranje prozora, pristup mišu i tipkovnici korištene su biblioteke GLEW i GLWF. Za razne matematičke operacije zadužena je bila biblioteka GLM koja je napisana po uzoru na GLSL. Još jedna matematička biblioteka pod nazivom Armadillo korištena je u funkciji za računanje Jacobijevih rotacija. Grafičko sučelje napisano je uz pomoć Microsoftove biblioteke MFC.

# 10. Zaključak

U industriji igara sve se više posvećuje pažnja simulacijama mekih tijela. Razvojem algoritama te povećanjem računalne moći otvoren je put simulacijama mekih tijela u područjima gdje su se do sada koristili isključivo kruti objekti. Ovim radom pokazano je da je moguće koristiti simulacije mekih tijela i pritom postići vrlo dobre rezultate. Posebno se to odnosi na metodu usklađivanja oblika koja nije zahtjevna za izračunavanje te je pogodna za igre. Metoda s oprugama je više namijenjena simulacijama u kojima je važnija realističnost prikaza od brzine izvođenja. Postoji dosta prostora za daljnji razvoj ovog rada, posebice u vidu ugradnje detekcije kolizije. Jedan od velikih vizualnih nedostataka je nemogućnost interakcije vrhova modela što ne predstavlja problem kada je čvrstoća modela velika te se dijelovi modela ne mogu previše udaljavati od početnih pozicija te doći u doticaj s drugim dijelovima. Detekcija kolizije bi bila veliki korak prema realističnijoj simulaciji.

# 11. Literatura


[1] Wikipedija, "Soft body dynamics", http://en.wikipedia.org/wiki/Soft_body_dynamics#Spring.2Fmass_models, 8. travnja 2013.

[2] Matthias Müller, Bruno Heidelberger, Matthias Teschner, Markus Gross, "Meshless Deformations Based on Shape Matching", Proceedings of ACM SIGGRAPH 2005

[3] Arturo Magidin, "Find the square root of a Matrix", 24. kolovoza 2011., http://math.stackexchange.com/questions/59384/find-the-square-root-of-a-matrix, 5. travnja 2013.

[4] D. V. Fedorov, "Eigenvalues and eigenvectors", Numerical methods 2013, http://users-phys.au.dk/fedorov/nucltheo/Numeric/12/eigen.pdf, 5. travnja 2013.

[5] "OpenGL Selection Using Unique Color IDs", http://content.gpwiki.org/index.php/OpenGL_Selection_Using_Unique_Color_IDs, 10. sibnja 2013.

[6] Wikipedija, "Opruga", http://hr.wikipedia.org/wiki/Opruga, 20. svibnja 2013.

[7] "Tutorials for modern OpenGL (3.3+)", http://www.opengl-tutorial.org/, 1. travnja 2013.

# 12. Sažetak

Simulacije mekih tijela neizostavan su dio realističnog prikaza stvarnog svijeta u različitim virtualnim okruženjima. U ovom radu obrađene su dvije metode simulacije mekih tijela – metoda s oprugama i metoda usklađivanja oblika. Napravljene su programske implementacije obiju metoda te je prikazana usporedba u obliku brzine izvođenja i vizualne kvalitete simulacija. Metoda uspoređivanja oblika napravljena je po uzoru na rad *Meshless Deformations Based on Shape Matching* [2] dok su kod metode s oprugama, ponajviše za algoritme raspoređivanja opruga, korištena autorova rješenja. U radu su opisani karakteristični dijelovi potrebni za implementaciju kao i problemi koji su se javljali tijekom izrade ovoga rada.

**Puni naslov zadatka rada:** Simulacija elastičnih objekata

**Ključne riječi:** meka tijela, simulacija, usklađivanje oblika, opruge i mase

# 13. Abstract

In today's virtual environments, soft bodies simulation has become one of the major parts in presenting a realistic scene. This paper presents two methods for simulating soft bodies – the spring/mass model and the method based on shape matching. Those two methods were implemented and the comparison was given based on the speed and visual quality of the of implementations. Shape matching method was implemented, for the most part, using the work Meshless Deformations Based on Shape Matching, while mass/spring method was implemented mainly using the authors solutions. This work describes common steps needed for soft bodies simulation implementation the problems that occurred while writing this work.

**Full title of the task:** Soft bodies simulation

**Key words:** soft bodies, simulation, shape matching, mass/spring

# 14. Privitak

Programska potpora s detaljnim uputama za korištenje nalazi se na kompaktnom disku priloženom uz ovaj rad.