Nemanja Milanović

Kreiranje aproksimativnog modela deformisanja krutih tela primenljivog za iterativno generisanje grafike sa prihvatljivom frekvencijom odziva

U ovom radu je istraživan aproksimativni model deformisanja krutih tela u realnom vremenu. Cilj istraživanja je bio da se nađe rešenje koje bi ovu vrstu simulacija izvršavalo tako što se svi proračuni vrše u toku rada programa generisanjem 30 ili više slika u sekundi. Za rešavanje je korišćen metod projektovanja zrakova (ray casting), uz primenu oktalnog stabla za lokalizaciju prostora. Osmišljena je funkcija opadanja sile kojom se računa uticaj sile na kružnu površinu u okolini direktne tačke sudara. Razmatran je sudar projektila koji ne podleže deformacijama i mete koja se deformiše. Rezultati su pokazali da osmišljeno rešenje radi u realnom vremenu kada se projektilom koji je aproksimiran tačkom u prostoru utiče na mali deo površine mete. Za kompleksnije slučajeve primećuje se pad u broju generisanih slika u sekundi pri udaru projektila u metu, a zatim i pad broja generisanih slika u sekundi u toku deformisanja tela, zavisno od broja tačaka na koje se utiče.

Uvod

Kruta tela su tela koja ne ispoljavaju elastičnost pod uticajem spoljašnjih sila, pa se može smatrati da deformisanje krutih tela predstavlja i trajnu promenu njihovog oblika. Vizuelne simulacije deformacija kojima se bavimo u ovom radu ne opisuju realan fizički proces, već ga samo oponašaju. Simulacije ove vrste se koriste u svrhe vizuelnih efekata u filmovima ili računarskim igrama. U ovom radu je razvijan metod kojim bi se u realnom vremenu izvršavala vizuelna simulacija deformisanja krutih tela, tj. svi proračuni bi se izvršavali u toku rada simulacije. Simulira se sudar mete koja se deformiše projektilom nepromenljivog oblika.

Simuliranje deformisanja se najčešće vrši korišćenjem metoda konačnih elemenata (finite element method) koji daje veoma precizne rezultate. Međutim, dati pristup je računarski jako zahtevan, i previše spor za simulaciju razvijanu u ovom radu (Irving *et al*. 2004). U metodu koji je razvijen u našem radu, mesto sudara se određuje na osnovu pravca i smera

Nemanja Milanović (2001), Petrovac na Mlavi, učenik 3. razreda Požarevačke gimnazije

MENTOR: Igor Šikuljak, student Fakulteta tehničkih nauka Univerziteta u Novom Sadu

kretanja projektila pomoću Meler-Trumborovog algoritma (Möller i Trumbore 1997). Kako nalaženje mesta sudara uključuje pretragu prostora, korišćeno je oktalno stablo, pomoću koga je prostor izdeljen i lokalizovan, što znatno ubrzava pretragu. Na osnovu unapred zadatih parametara materijala mete, računa se uticaj sile na površinu u okolini mesta sudara. U toku rada simulacije potrebno je generisati bar 30 slika u sekundi, da bi se stekao utisak da se simulacija neprekidno odvija.

Metod

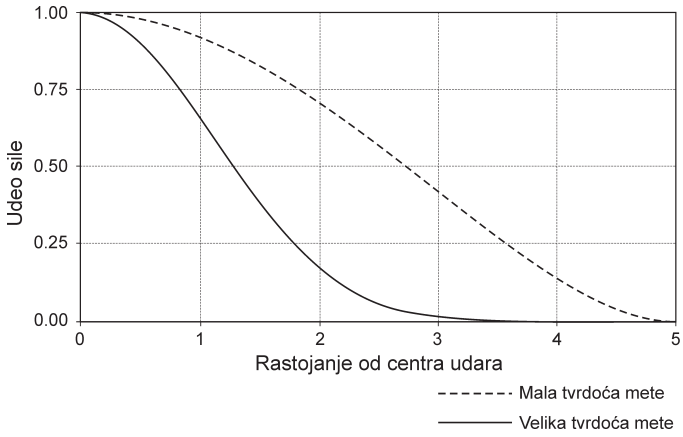
Teorijske osnove

Kada je u pitanju računarska grafika, svi trodimenzionalni objekti se najčešće predstavljaju kao skup tačaka koje su međusobno povezane u mrežu trouglova. Ovom mrežom se dobijaju složeni oblici i površine. Samim tim, sve deformacije tela su samo prostorne transformacije koordinata tačaka na površini mete.

Način na koji telo reaguje na udarac projektila definisan je skupom parametara koji su nazvani parametrima materijala, kao i osmišljenom funkcijom kojom je definisano ponašanje površine u kružnoj okolini tačke udara. To su otpor mete, parametar udara (falloff), tvrdoća mete i minimalna sila potrebna za deformaciju tela. Otpor mete predstavlja vrednost u opsegu od 0 do 1, i određuje kolikim delom intenziteta sile će projektil delovati na metu. Dakle, ako je f intenzitet sile kojom bi projektil delovao bez otpora, a otpor mete R, onda je rezultujući intenzitet jednak $(1-R)\cdot f$.

Parametar udara (φ) i tvrdoća mete (r) određuju funkciju raspodele intenziteta sile udara po površini mete koja je za tačkasti udar data formulom:

$$f(x)=\left(1-\left(\frac{x}{\varphi}\right)^2\right)^r,\ x\in(0,\varphi), \tag{1}$$



Slika 1. Primer funkcije opadanja intenziteta sile u zavisnosti od rastojanja od tačke udara za dve različite vrednosti tvrdoće materijala u slučaju kada je vrednost parametra udara $\varphi = 5$

Figure 1. An example of the function of force falloff relative to the distance from the point of impact for two different values of material roughness when the collision parameter $\varphi = 5$

gde x predstavlja udaljenost tačke od mesta udara. Tvrdoća određuje izgled krive prema kojoj sila deluje, zavisno od rastojanja od tačke udara. Što je tvrdoća veća, kriva je „strmija" (slika 1).

Minimalna sila potrebna za deformaciju tela predstavlja donju granicu intenziteta sile u tački sudara ispod koje ne dolazi do deformacije mete.

Inicijalno rešenje

Na početku simluacije definišu se pozicije projektila i mete, početna brzina projektila koja je uvek 0, kao i vektor ubrzanja projektila na osnovu kojeg projektil ubrzava do trenutka sudara. Nakon podešavanja parametara, simulacija se izvršava u glavnoj petlji. U glavnoj petlji se vrši iscrtavanje objekata na ekran, kao i njihove transformacije (pomeranje, deformisanje, itd.) do kraja izvršavanja programa. Da bi se za datu simulaciju moglo reći da se izvršava u realnom vremenu, glavna petlja mora da se izvrši bar 30 puta u sekundi.

Projektili se mogu podeliti na dve grupe: na materijalne tačke (telo je aproksimirano tačkom koja deluje na metu) i na projektile koji imaju određeni oblik.

U slučaju materijalne tačke, kada se simulacija započne, prvo se računa tačka direktnog udara. Ovo se radi tako što se iz tačke projektila projektuje zrak sa pravcem i smerom njegove brzine. Zatim se za svaki trougao na meti proverava da li je došlo do preseka sa zrakom. Ovo se vrši uz pomoć Meler-Trumborovog algoritma (Möller i Trumbore 1997). Ako je došlo do preseka, tačka preseka se čuva, kao i udaljenost od projektila. Zatim se za svaku tačku mete prema formuli (1) računa sila u oblasti određenoj parametrom φ. U najgorem slučaju, potrebno je n provera preseka i m izračunavanja intenziteta sile, gde je n broj trouglova, a m broj tačaka kojima je definisana meta. Pri svakom narednom prolasku kroz petlju, distanca od tačke sudara se umanjuje za dužinu puta koji je prešao projektil. Kada distanca dođe do nule, započinje se pomeranje zahvaćenih tačaka u skladu sa udelom uticaja sile proračunatim funkcijom opadanja sile, ubrzanje projektila dobija vrednost negativnog inicijalnog ubrzanja, i shodno tome se brzina projektila smanjuje. Pomeraj zahvaćenih tačaka se računa na osnovu pomeraja samog projektila u vremenskom opsegu jednog izvršavanja glavne petlje. Simulacija prestaje sa izvršavanjem kada brzina projektila dođe do nule.

Kada su u pitanju projektili koji imaju konkretan oblik, proces je nešto složeniji. Prvo se projektuje po jedan zrak iz svake tačke projektila, a zatim se za svaki od zrakova vrši provera preseka sa svakim od trouglova mete. Međutim, ova informacija nije dovoljna, pošto se ne može precizno odrediti koji deo površine mete će biti zahvaćen projektilom. Zbog toga se sa mete projektuju zraci sa pravcem brzine projektila, ali suprotnog smera. Za svaki od ovih „inverznih" zrakova sa mete proverava se presek sa svakim od trouglova projektila. Ako dođe do preseka, znamo da će ovi trouglovi mete biti direktno zahvaćeni. Analogno slučaju materijalne tačke, za svaki zrak, bilo običan ili inverzni, čuva se udaljenost od tačke udara. Ove distance se

svakim prolaskom kroz petlju ažuriraju u skladu sa pomerajem projektila. Čim neka od distanci dostigne nulu, tačke zahvaćenog trougla, kao i okolne tačke pod indirektnim uticajem sile će se translirati u skladu sa pomerajem projektila. Vrednost intenziteta sile za svaku tačku se računa kao maksimum indirektnih uticaja iz svih tačaka direktnog udara koje su na rastojanju manjem od parametra udara materijala mete.

Glavna mana ovog pristupa rešavanju problema jeste činjenica da se sve operacije projektovanja zrakova, kao i sračunavanje mesta udara i funkcije opadanja sile izvršavaju u samo jednom prolasku kroz glavnu petlju, na početku izvršavanja simulacije. Ovo predstavlja problem zato što je potrebno izvršiti veliki broj kompleksnih operacija u roku od najviše 1/30 s da bi simulacija na posmatrača ostavila utisak neprekidnog odvijanja. Kada su u pitanju objekti sa velikim brojem trouglova, ovo nije moguće ostvariti. Drugi problem predstavlja činjenica da se ishod simulacije računa na samom početku. Ovo predstavlja ograničenje, jer u toku simulacije ne sme da dođe ni do kakvih promena na telima koja interaguju, što znači da ovaj model ne bi mogao da se koristi u sklopu većih sistema, gde figuriše više sporednih faktora.

Oktalno stablo

Oktalno stablo (eng. octree) je vrsta stabla u kojem svaki čvor ima po osmoro dece. Ova struktura podataka je korisna za organizovanje i lokalizaciju trodimenzionalnog prostora, jer se svaki čvor može predstaviti kao kocka sa osam kocki unutar nje, upola manjih ivica. Svaki čvor ovog stabla sadrži poziciju svog centra, osam čvorova dece i veličinu kocke koju obuhvata u prostoru. Listovi stabla sadrže tačke koje se u njima nalaze, kao i trouglove sa kojima se seku.

U toku učitavanja programa, generišu se dva ovakva stabla. Jedno stablo koje obuhvata metu, i jedno koje obuhvata projektil. Svako stablo ima definisanu dubinu – ako je dubina k, stablo se može predstaviti kao kocka sa 2^{3k} kocki u njoj. Koren stabla za centar uzima centar objekta, i generiše kocku sa najmanjom mogućom veličinom tako da obuhvata ceo objekat. Zatim se u svaki list stabla ubacuju sve tačke koje se nalaze unutar dela prostora koji taj list obuhvata, kao i svi trouglovi koji su sadržani ili koji seku deo prostora obuhvaćenog listom. Provera preseka trougla i lista, tj. kocke, vrši se pomoću teoreme razdvojne ose (separating axis theorem; Gottschalk 1996).

Radi bržeg pristupanja listovima stabla, stabla su prikazana pomoću trodimenzionalnog niza pokazivača. Pri kreiranju stabla, svaki list dobija svoj indeks u nizu na osnovu svoje pozicije. U tu svrhu je osmišljena funkcija pomoću koje je moguće i pristupanje listovima na osnovu proizvoljne pozicije u prostoru bez rekurzivnog traženja kroz stablo, što omogućava brži pristup listovima:

$$n_x = \frac{x + \frac{2^d - 1}{2} \cdot S - O_x}{S}, \tag{2}$$

gde je x pozicija centra lista na x osi, d dubina stabla, S veličina lista, a O_x pozicija centra stabla na x osi. Analogno se računaju i indeksi za y i z osu.

Kao i u prethodnom rešenju, projektili su podeljeni na materijalne tačke i na projektile koji imaju definisan oblik. Kod projektila koji predstavljaju materijalne tačke stablo se ne generiše, jer sadrže samo jednu tačku. Za metu se stablo generiše u oba slučaja.

Kod materijalnih tačaka se svakim prolaskom kroz glavnu petlju projektuje zrak sa pravcem i smerom brzine projektila na list stabla mete u delu prostora koji je u neposrednoj blizini projektila. Ako je meta predaleko, ne vrši se provera. Ako je došlo do preseka zraka sa određenim listom stabla, proverava se presek zraka sa svakim trouglom koji seče taj list. U slučaju preseka trougla sa zrakom saznaje se da će u sledećem prolasku kroz glavnu petlju doći do sudara, i sračunava se sila u svim listovima stabla koji su dovoljno blizu da bi bili zahvaćeni u skladu sa parametrom udara materijala. Od sledećeg prolaska kroz glavnu petlju započinje deformacija mete, sve njene zahvaćene tačke se pomeraju u skladu sa pomerajem projektila i prethodno izračunatim intenzitetom na osnovu funkcije opadanja sile (1). Simulacija se završava kada se projektil zaustavi.

U slučaju projektila koji predstavlja trodimenzionalni objekat, svakim prolaskom se vrši identičan proces projektovanja zrakova kao kod materijalne tačke, samo što sada zrakova ima onoliko koliko je potrebno tačaka da se projektil definiše. Kao i u inicijalnom rešenju, projektuju se zraci sa mete na projektil, radi provere da li će površina između udarnih tačaka takođe biti zahvaćena udarcem. Razlika je u tome što sada projektil ima svoje stablo koje ga obuhvata i koje se pomera zajedno sa njim, tako da se, slično direktnim zracima, inverzni zraci projektuju u istom pravcu ali suprotnom smeru brzine projektila, a provere preseka se izvršavaju u neposrednoj blizini mete. Zatim se u okolnim listovima stabla svake zahvaćene tačke sračunava intenzitet sile na osnovu funkcije opadanja sile, po istom principu kao kod materijalne tačke. Ako ima više indirektnih uticaja, uzima se maksimalna vrednost. Svaka zahvaćena tačka se dodaje u niz zahvaćenih tačaka, i pri svakom prolasku kroz glavnu petlju niz tačaka se po potrebi ažurira dodavanjem novih tačaka koje će se sudariti; tačke koje su već pod uticajem udara pomeraju se u skladu sa pomeranjem projektila i udela sile sračunatog funkcijom opadanja sile.

Navedeni pristup ima prednosti u odnosu na inicijalno rešenje. Broj provera u ovom slučaju je dosta manji u odnosu na inicijalno rešenje, jer se pri proverama preseka ne proverava svaki trougao mete. Takođe, proračuni se izvršavaju pri svakom prolasku kroz glavnu petlju, a ne na početku simulacije, tako da se simulacija ne predviđa na početku, pa nema ni prevelikog opterećenja na početku simulacije. Ovaj pristup takođe ima manu, jer trouglovi koji su ubačeni u određeni list stabla zauvek ostaju u tom listu, iako se deformacijom mogu prostorno izmestiti u neki drugi list.

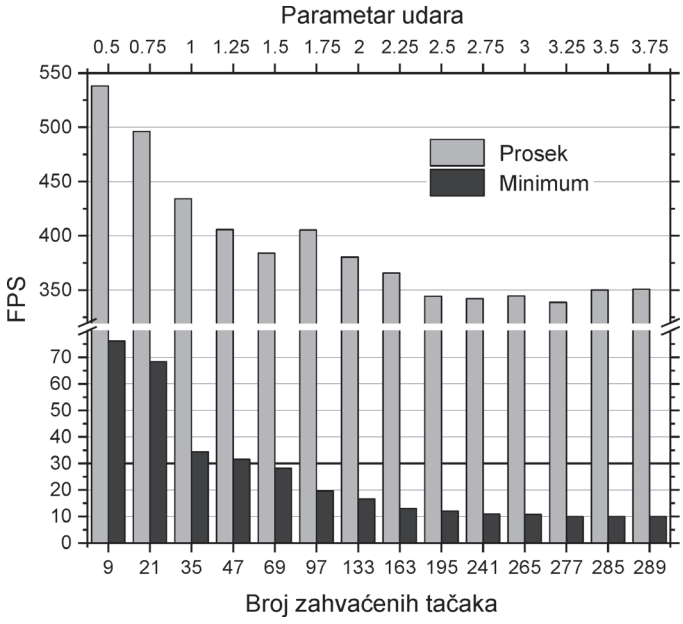
Ovo bi predstavljalo problem kada bi se posle prve deformacije izvršila još jedna deformacija, jer podaci u stablu više nisu važeći.

Za realizaciju predstavljenog rešenja korišćen je programski jezik C++ sa grafičkom bibliotekom OpenGL.

Rezultati i diskusija

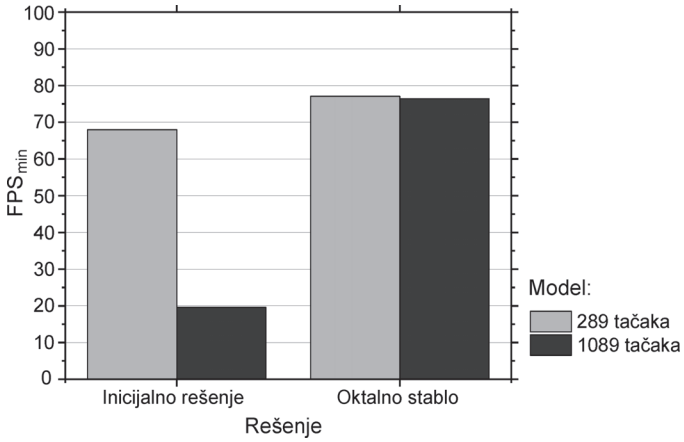
Eksperimentalno testiranje je obuhvatalo četiri različite konfiguracije. Prvi test je ispitivao uticaj povećanja parametra udara materijala na performanse implementacije oktalnog stabla. Drugi test je ispitivao uspešnost implementacije strukture oktalnog stabla, tj. lokalizaciju prostora. Pristupi sa i bez oktalnog stabla su upoređeni u slučaju kada je udarom bilo zahvaćeno devet tačaka na meti sastavljanoj od 289 tačaka i na meti sastavljenoj od 1089 tačaka. U trećem testu su ispitivane performanse udara projektila zadatog oblika u varijanti sa oktalnim stablom. U četvrtom testu je ispitivan uticaj računanja funkcije opadanja sile na performansu programa pri udaru projektila koji ima definisan oblik. Upoređena su dva slučaja: slučaj kada se funkcija opadanja sile računa, ali je parametar udara zanemarljivo mali, pa ne dolazi do indirektnog uticaja, i slučaj kada je sračunavanje funkcije opadanja sile potpuno isključeno.

Iz svih rezultata dobavljan je minimalni broj prolazaka kroz glavnu petlju u jedinici vremena, tj. najmanji broj iscrtanih slika u jedinici vremena (eng. framerate) izražen u FPS vrednostima (broj frejmova u sekundi, eng. frames per second), jer se na osnovu minimuma može pokazati da li je simulacija ostvariva u realnom vremenu ili ne.



Slika 2. Uticaj promene parametra udara na performanse (vrednosti su dobijene na osnovu 140 merenja)

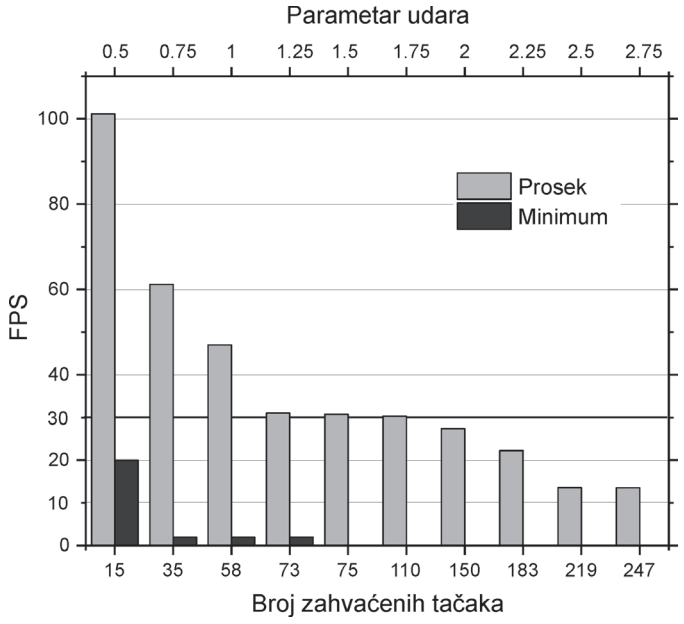
Figure 2. The effect of varying the falloff parameter on minimum (black bars) and average (grey bars) FPS values (values estimated based on 140 tests)



Slika 3. Uticaj primene strukture oktalnog stabla na minimalni broj slika u sekundi sa 9 zahvaćenih tačaka za različite veličine mete (289 i 1089 tačaka)

Figure 3. The effect of using the octree structure on the minimal framerate with 9 affected points for different target sizes (289 and 1089 points)

U prvom testu se primećuje pad minimalne vrednosti broja frejmova u sekundi sa povećanjem parametra udara mete (slika 2). Ovaj pad je u slučaju malog broja zahvaćenih tačaka primećen samo u jednom prolasku kroz glavnu petlju: u trenutku udarca projektila u metu. U slučaju većeg broja zahvaćenih tačaka, uočava se i pad prosečne FPS vrednosti posle sudara. Pad broja frejmova u sekundi u testovima sa manjim brojem zahvaćenih tačaka uzrokovan je određivanjem tačaka na koje treba uticati i sračunavanjem funkcije opadanja sile. Mali broj frejmova u sekundi u slučaju više zahvaćenih tačaka uzrokovan je brojem vektorskih operacija i operacija projektovanja zrakova koje treba izvršiti pri svakom prolasku kroz petlju.



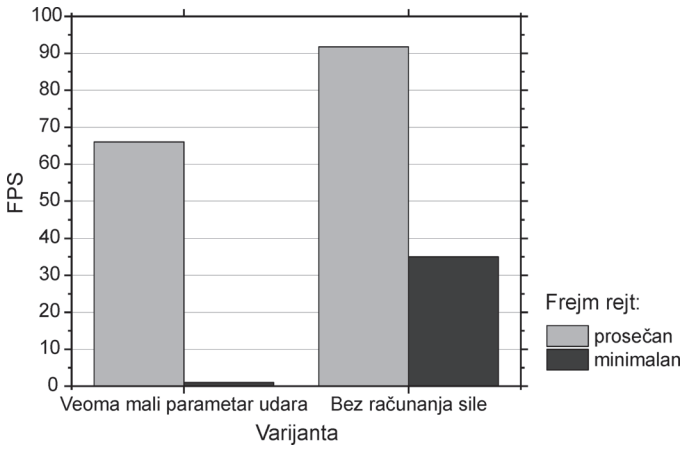
Slika 4. Uticaj usložnjavanja geometrije na performansu pri sudaru sa projektilom datog oblika

Figure 4. The effect of increasing geometrical complexity on minimum (black bars) and average (grey bars) FPS values when colliding with a projectile that has a specific shape

U drugom testu je inicijalno rešenje u slučaju modela visoke rezolucije davalo lošije performanse u odnosu na model niske rezolucije. Pri implementaciji oktalnog stabla, minimalni broj iscrtanih slika u sekundi je jednak u oba slučaja i približno je jednak inicijalnom rešenju u slučaju modela niske rezolucije.

Iz rezultata trećeg testa, tj. u slučaju projektila zadatog 3D oblika sa implementacijom oktalnog stabla, uočavaju se neprihvatljivo loše performanse, u nekim slučajevima potrebno je više sekundi za računanje i iscrtavanje samo jedne slike (slika 4). Objašnjenje za ovakav ishod je to što ima mnogo zahtevnih operacija projektovanja zrakova, kao i sračunavanje funkcije opadanja sile za sve tačke zahvaćene udarom.

U četvrtom testu, u slučaju kada je računanje indirektne sile isključeno, primećen je skok u minimalnom broju slika u sekundi u odnosu na slučaj gde je računanje indirektne sile uključeno sa zanemarljivo malim parametrom udara (slika 5).



Slika 5. Uticaj sračunavanja funkcije opadanja sile na performanse programa

Figure 5. The effect of calculating the force falloff function on the performance of the program

Iz prvog (slika 2) i četvrtog (slika 5) testa jasno se vidi da računanje indirektne sile ima najveći negativan uticaj na performanse programa, pogotovo kada su u pitanju slučajevi sudara mete sa projektilom zadatog oblika. Uzrok ovog negativnog uticaja jeste taj što sračunavanje funkcije opadanja sile podrazumeva veliki broj provera i vektorskih transformacija. Za svaku tačku za koju se računa uticaj indirektne sile, vrši se pretraživanje okolnog prostora, i nad svakom tačkom u tom prostoru se primenjuju vektorske transformacije. Broj provera i proračuna raste eksponencijalno u slučaju projektila definisanog oblika, što znatno utiče na vreme potrebno za izvršavanje svih proračuna, a samim tim i vreme potrebno za iscrtavanje slike na ekranu. Iz drugog testa se zaključuje da je implementacija oktalnog stabla efikasna za lokalizaciju prostora, što omogućava znatno bolje performanse ako se utiče samo na manji segment mete. Iz trećeg testa (slika 4) se jasno vidi uticaj velikog broja proračuna na performanse. Kod projektila sa konkretnim oblikom, broj vektorskih operacija i provera raste eks-

ponencijalno zbog računanja funkcije opadanja sile, i zbog činjenice da se projektovanje zrakova vrši i sa mete na projektil. Zbog toga, u slučaju kada se koriste modeli sa više trouglova, broj iscrtanih slika u sekundi jako brzo pada na neprihvatljivo nizak nivo.

Zaključak

Iz dobijenih rezultata se uočava da je broj frejmova u sekundi iznad 30 kada deformacije na meti zahvataju par desetina do par stotina tačaka pod uticajem materijalne tačke. U tom slučaju se simulacija izvršava u realnom vremenu. U slučaju kada se utiče na veći broj tačaka, primećuje se nizak broj frejmova u sekundi u trenutku sudara projektila i mete. Ako se nizak broj frejmova u sekundi u tom trenutku zanemari, moguće je uticati na više tačaka i smatrati da se simulacija izvršava u realnom vremenu. Kada se koriste projektili sa definisanim oblikom, performanse su znatno lošije nego u slučaju materijalne tačke. Takođe se vidi da je primenom oktalnog stabla prostor uspešno lokalizovan, što znači da se prostor efikasnije pretražuje sa ovom implementacijom. U kontekstu modernih računarskih igara koje često imaju više stotina hiljada trouglova na ekranu predloženo rešenje u većini slučajeva ne bi bilo primenljivo.

Dalji rad može se baviti usavršavanjem oktalnog stabla u sklopu ovog rešenja. Mogu se ispitati načini za efikasno premeštanje trouglova kroz listove stabla na osnovu njihove pozicije. Takođe se može ispitivati način da se stablo dinamički deli na osnovu gustine trouglova u određenom delu prostora, uz implementaciju efikasne pretrage takve strukture i pretrage listova u okolini datog lista. Rešavanju problema se može i pristupiti računanjem prostornih transformacija na grafičkoj kartici, pošto se na taj način proračuni mogu izvršavati paralelno. Zatim se može ispitati ishod takve simulaciije u odnosu na simulaciju koja se izvršava na procesoru. Može se ispitati performansa ovakvog modela uz primenu uprošćenih 3D modela meta i projektila nad kojima bi se vršile transformacije. Na osnovu deformacija uprošćenih modela moguće je aproksimirati i ponašanje modela sa više tačaka.

Literatura

Möller T., Trumbore B. 1997. Fast, Minimum Storage Ray-Triangle Intersection. *Journal of Graphics Tools*, **2** (1): 21-28.

Gottschalk S. 1996. Separating axis theorem. Technical Report TR96-024. Department of Computer Science, UNC Chapel Hill.

Irving G., Teran J., Fedkiw R. 2004. Invertible finite elements for robust simulation of large deformation. U *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animatio*n (ur. R. Boulic i D. K. Pai). Aire-la-Ville: Eurographics Association, str. 131-140.

Nemanja Milanović

Creation of an Approximate Model for the Deformation of Rigid Bodies Applicable to Iterative Graphics Generation with an Acceptable Response Frequency

This paper covers the research of an approximate model for simulating the deformation of rigid bodies in real time. The aim of this research was to find a solution that would execute all the calculations that are necessary for the simulation while simultaneously generating 30 or more images per second. The approach used for solving this problem uses ray tracing for collision detection using the Möller-Trumbore ray-triangle intersection algorithm (Möller and Trumbore 1997), an implementation of the octree structure for space localization, and a falloff function for calculating the indirect effect of the impact as well as surface smoothing. The solution consists of two objects: the projectile, which can be either a single point in space or a rigid body that cannot be deformed, and a target which is deformed by the projectile according to its material parameters.

Results have shown that this solution is applicable for targets that have a low triangle count (in the range of hundreds) with a projectile represented by a single point in space. The octree implementation has proven to be effective when it comes to space localization. The main negative effect on the performance of the program is caused by calculating the falloff function for applying indirect force on the target, because the function is calculated on each point in a radius according to each ray-triangle intersection between the target and the projectile.