SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 215

MODELIRANJE I SIMULACIJA DEFORMABILNIH OBJEKATA

Marko Vadlja

Zagreb, lipanj 2011.

Sadržaj

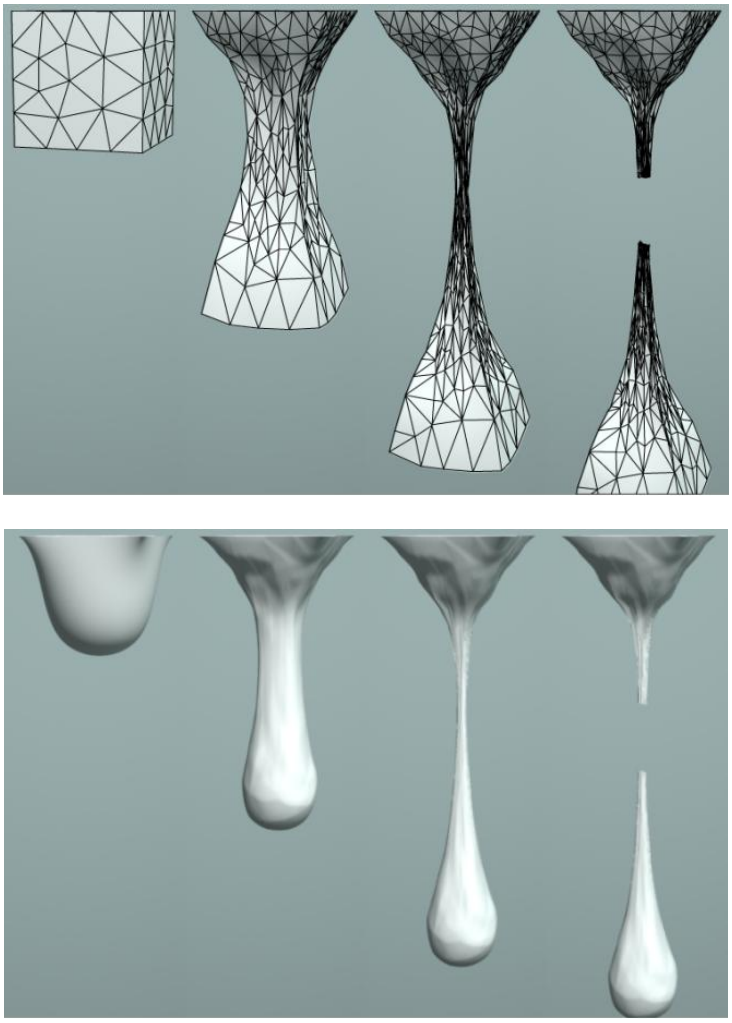
Uvod

Simulacija deformabilnih modela temeljenih na fizikalnim zakonitostima više se od dva desetljeća pojavljuje u računalnoj grafici. To interdisciplinarno polje kombinira Newtonovsku dinamiku, numeričku matematiku, diferencijalne izračune, računarsku znanost i mnoge druge kako bi se dobili što precizniji izračuni i što bolje simulacije. Polje je zanimljivo zbog svoje primjene u raznim granama industrije poput automobilske, filmske, industrija igara i ostalih. Unutar računalne grafike ovo područje se primjenjuje na modeliranje objekata, simulaciju lomova, plastičnosti, tkanina, fluida, raznih simulacija u stvarnom vremenu i ostalih.

U ovom radu bit će predstaljeni model deformabilnog objekta, kao i matematičke metode koje se koriste u njegovom modeliranju. Naglasak će biti stavljen na metodu konačnih elemenata. Implementacija spomenutih pojmova napravljena je u aplikaciji Deforma u kojoj će, uz spomenuti model, biti predstavljeni i neki od koncepata objektnog dizajna.

1. Model deformabilnog objekta u računalnoj grafici

U ovom poglavlju opisat ćemo karakteristike deformabilnog objekta, te matematičke modele na temelju kojih se oni mogu simulirati. Posebno će biti objašnjeno kako simulirati deformacije objekata koristeći metodu konačnih elemenata.

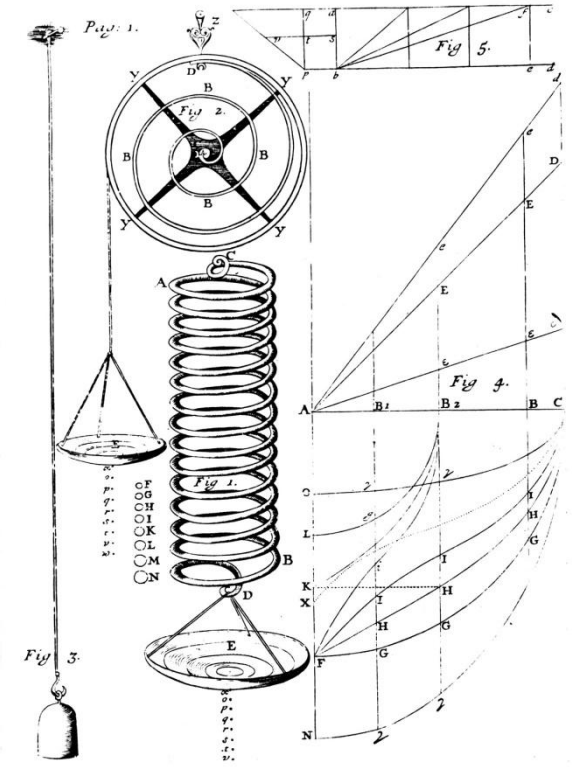


Slika 1. Simulacija kapanja deformabilne tvari

1.1. Plastičnost i elastičnost

Elastičnost je svojstvo materijala da se uslijed deformacije uzrokovanje djelovanjem vanjske sile vrati u svoj prvotni oblik po prestanku djelovanja sile. Za mnoge elastične materijale vrijedi linearna funkcija omjera primijenjenih sila i naprezanja materijala. Naprezanje materijala je svojstvo da uslijed primjene sile na materijal on stvara protusilu. Klasični model linearne elastičnosti je savršena opruga. Savršena oprugu kao pojam je prvi put predstavio Robert Hooke 1675. godine. Linearan odnos primijenjene sile i naprezanja materijala opisan je Hookeovim zakonom (1). Sila F koju opruga stvara obrnuto je proporcionalna pomaku opruge x iz ravnotežnog položaja uz neki faktor k koji je konstantan i zove se konstanta opruge. Mnoge elastične pojave se aproksimiraju Hookeovim zakonom.

$$F = -kx \tag{1}$$



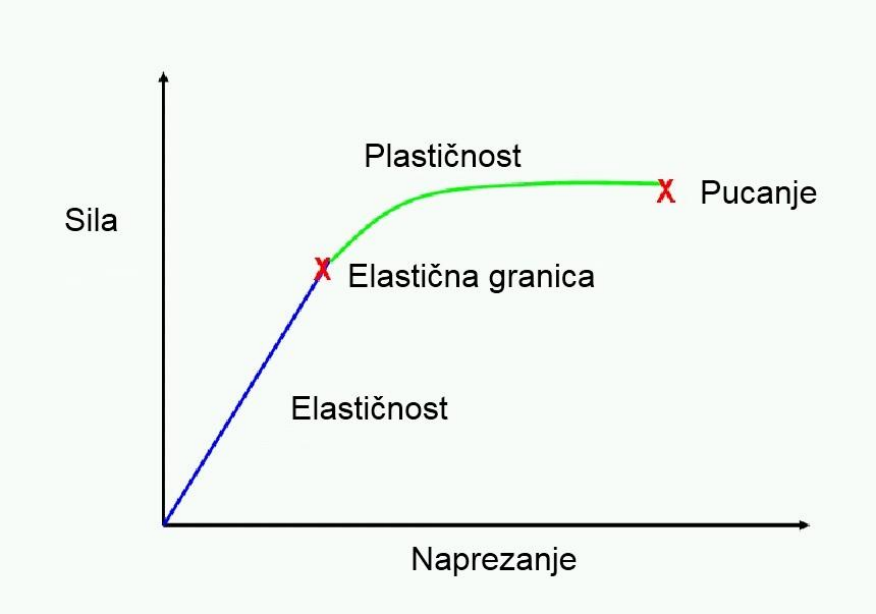
Slika 2. Hookeov zakon; izvor - *De Potentia Restitutiva 1678*

U fizici se plastičnost opisuje kao deformacija materijala uslijed djelovanja vanjskih sila na takav način da se on ne može vratiti u svoj prijašnji oblik. Plastične deformacije primijećene su u većini materijala poput kamenja, minerala, metala i sl. No razlozi zašto su pojedini materijali podložni plastičnoj deformaciji su različiti.

Slika 3. Deformirane limenke kao primjer plastičnih objekata

U prirodi se susrećemo sa materijalima koji su elastoplastični. Za njih vrijedi da se uz primjenu određene količine vanjske sile na materijal oni ponašaju kao elastični, no uslijed dovoljno velikog povećanja sile počnu se ponašati kao plastični. Prijelaz iz elastičnosti u plastičnost opisan je pojmom koji se zove elastična granica. Daljnjim povećanje sile na materijal dolazi se do pucanja materijala.

Također postoje i materijali koji se zovu pseudoelastični, odnosno superelastični. Takvi materijali se pod utjecajem neke vanjske sile ponašaju kao plastični predmeti, naizgled trajno mijenjajući svoj oblik. No, uslijed djelovanja nekog specifičnog vanjskog podražaja na materijal, poput topline ili električne struje, oni se vraćaju u svoj prvobitni položaj čime se ponašaju kao da su elastični.



Slika 4. Stanja elastoplastičnog objekta

1.2. Modeliranje deformabilnih objekata

Deformabilni objekt uobičajeno je definiran svojim nedeformiranim oblikom, koji još nazivamo ravnotežni ili inicijalni oblik, te parametrima koji su određeni prema materijalu od kojih je objekt napravljen. Inicijalni oblik možemo zamisliti kao skup kontinuiranih točaka M definiran na skupu realnih brojeva $\mathbb{R}^3$. Koordinate točaka $m \in M$ zovu se materijalne koordinate točke m.

Prilikom utjecaja neke vanjske sile točka **m** se pomakne u neke koordinate $x(m)$. Pomak se računa prema (2).

$$u(m) = x(m) - m \tag{2}$$

Iz $u(m)$ moguće je dobiti naprezanje ε točke m. U jednodimenzionalnom slučaju vrijedi da je $\varepsilon = \Delta l/l$, gdje je Δl relativni pomak točke m u zadnjem vremenskom intervalu a l ukupan pomak od inicijalnog položaja. U trodimenzionalnom slučaju popularno se naprezanje računa prema (3).

$$\varepsilon = \frac{1}{2}(\Delta u + (\Delta u)^T + \Delta u(\Delta u)^T)$$

$$u = (u, v, w)^T \text{ gdje je } \Delta u = \begin{bmatrix} u'_x & v'_x & w'_x \\ u'_y & v'_y & w'_y \\ u'_z & v'_z & w'_z \end{bmatrix} \tag{3}$$

Temeljem naprezanja točke m moguće je dobiti i silu koja djeluje na tu točku. Ona se u računalnoj grafici obično izražava pomoću Hookeovog zakona vrijedi (4) gdje je K matrica koja opisuje konstantu opruge.

$$F = K\varepsilon, \tag{4}$$

1.3. Uvođenje vremenske dimenzije

Kako bi simulirali dinamičke deformabilne objekte potrebno je dodati vremensku dimenziju pomaku $x(m,t)$ za sve točke skupa M. Uz poznavanje $\boldsymbol{x}(0), \boldsymbol{x}(\Delta t), \boldsymbol{x}(2\Delta t)$ itd. možemo animirati objekt. U ovom zapisu Δt je vremenski korak a $\boldsymbol{x}(t)$ predstavlja pomake za sve točke m.

Pomak možemo implicitno dobiti iz sila koje djeluju na sve točke i to temeljem drugog Newtonovog zakona (5).

$$\ddot{x} = F(\dot{x}, x, t) \tag{5}$$

Gdje $\ddot{x}$ predstavlja drugu derivaciju po vremenu od x, odnosno ubrzanje, a $\dot{x}$ prvu derivaciju po vremenu od x, odnosno brzinu. Uz (5) i (6) dobiva se (7).

$$\dot{x} = v \tag{6}$$

$$\dot{v} = F(v, x, t) \tag{7}$$

Definicija derivacije za $\dot{v}(t)$ i $\dot{x}(t)$ navedena je u (8).

$$\begin{aligned} \dot{v}(t) &= [v(t + \Delta t) - v(t)]/\Delta t \\ \dot{x}(t) &= [x(t + \Delta t) - x(t)]/\Delta t \end{aligned} \tag{8}$$

Uvrštavanjem derivacija pomaka i brzine (7) u (8) dobit ćemo (9).

$$\begin{aligned} v(t + \Delta t) &= v(t) + \Delta t F(v(t), x(t), t) \\ x(t + \Delta t) &= x(t) + \Delta t\, v(t) \end{aligned} \tag{9}$$

Uvođenje vremenske integracije korakom Δt do izražaja dolaze dva kriterija. Kriterij točnosti i kriterij stabilnosti. Kriterij točnosti određuje koliko se precizno temeljem odabranog parametra Δt mogu dobiti realno točni rezultati.

No, u računalnoj grafici češće je važno ispunjavanje kriterija stabilnosti. Izrazi iz (6) eksplicitno opisuju veličine u sljedećem vremenskom koraku, no uvjetno su stabilnije. Naime, ovisno o veličini koraka Δt, u ovisnosti s nekim pragom stabilnosti, moguće je da rješenje izraza u sljedećem vremenskom koraku postane iznimno nestabilno. Nestabilnost je omogućena zato jer desna strana jednadžbe opisuje samo buduće događaje bez obzira na prijašnje stanje. Na primjeru jednostavne opruge i dovoljno veliko izabrane veličine Δt u eksplicitnim izrazima zadanim u (6), dovodi do mogućnosti da se potpuno promaši trenutak u kojem se opruga vratila u svoje ravnotežno stanje i istegne više nego što je trebalo u određenom vremenskom trenutku. Time se stvara nestabilnost koja može dovesti do netočne simulacije.

Prijedlog za ispravljanje postojećih izraza (9), odnosno povećanja stabilnosti jest da se vremenski korak $t + \Delta t$ iskoristi s obje strane jednadžbe kao što je prikazano u izrazima (10).

$$v(t + \Delta t) = v(t) + \Delta t F(v(t + \Delta t), x(t + \Delta t), t)$$
$$x(t + \Delta t) = x(t) + \Delta t\, v(t + \Delta t) \qquad (10)$$

Ovakvi izrazi nazivaju se implicitni izrazi jer je nepoznata veličina Δt, dana kao argument za rješavanje izraza s desne strane jednakosti. Ovakvo rješavanje zove se i rješavanje “unazad“. Implicitno zadani izrazi stabilni su za proizvoljno velik Δt, dok ipak postoji niži prag za izabiranje Δt, koji ne predstavlja veliki problem. No, ovako se ne mogu predstaviti rješenja udaljena nekoliko vremenskih koraka, već se svaki korak mora izračunavati.

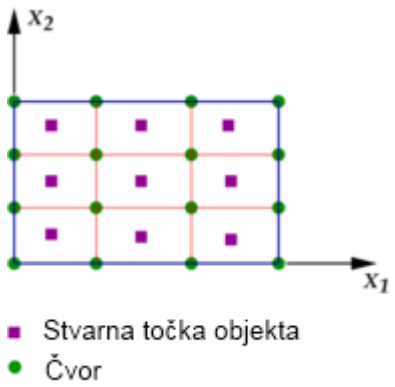
Postoji još jedno poboljšanje eksplicitne metode, u obliku rješavanja “unaprijed-unazad“, i dalje dajući eksplicitno, no stabilnije rješenje (11).

$$v(t+\Delta t) = v(t) + \Delta t F(v(t), x(t), t)$$
$$x(t+\Delta t) = x(t) + \Delta t\, v(t+\Delta t) \qquad (11)$$

2. Metode izračuna deformacije objekata zasnovane na Lagrangeovim mrežama

Integracija po vremenu deformabilnih elemenata računski je zahtjevna operacija na objektima koji ima beskonačno mnogo točaka. Uvođenjem diskretizacije i primjenom numeričkih metoda mogu se dobiti vrlo dobre aproksimacije koje ne zahtijevaju toliko resursa.

Lagrangeova mreža je zapravo rezultat metode diskretizacije tijela (engl. *meshing*). Metoda objekt podjeli u konačno mnogo elemenata, koji su obično predstavljeni nekim jednostavnim geometrijskim oblikom ili tijelom. Ti elementi se sastoje od čvorova i atributa. Čvorovi označavaju posebne točke elementa nad kojima se vrše vanjski utjecaji na objekt ili u kojima su elementi spojeni sa čvorovima drugih objekata. Atributi su zapravo svojstva materijala i stvarnih točaka koje opisuju objekt.



Slika 5. Primjer dvodimenzionalne Lagrangeova mreže

2.1. Metoda konačnih elemenata

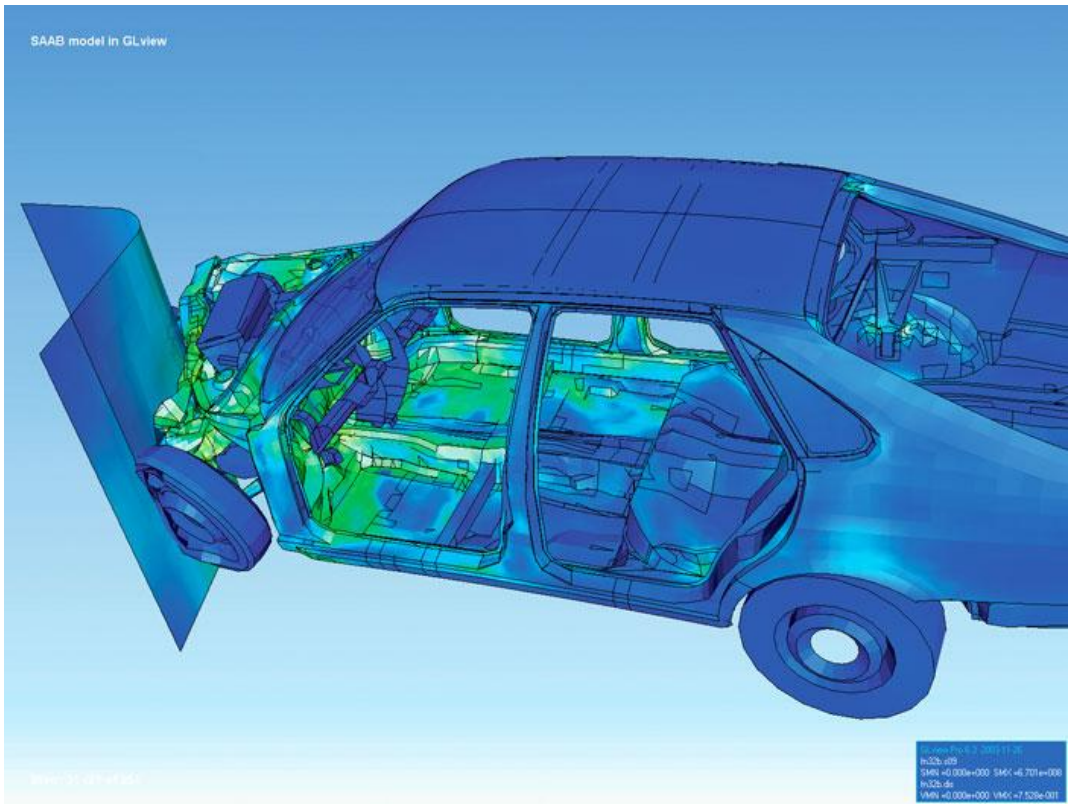
Metoda konačnih elemenata je numerička metoda za rješavanje rubnih vrijednosti koja se temelji na fizičkoj diskretizaciji. Razmatrano područje dijeli se na konačni broj manjih područja, podskupova početnog područja, te ih nazivamo konačni elementi. Elementi su međusobno povezani u točkama koje nazivamo čvorovi. Za svaki element se pretpostavlja rješenje zadane diferencijalne jednadžbe. Diferencijalne jednadžbe imaju oblik interpolacijskih funkcija koje povezuju zavisne varijable s njihovim vrijednostima u čvorovima. Za svaki element se izvodi lokalni algebarskih jednadžbi čije su nepoznanice čvorne veličine. Nakon toga se odgovarajućim postupcima formira globalni sustav jednadžbi za cijeli diskretizirani model, u kojemu su nepoznanice čvorne vrijednosti svih elemenata diskretiziranog područja.

Razlikuju se jednodimenzijski, dvodimenzijski i trodimenzijski konačni elementi. Također, postoje elementi za rješavanje posebnih geometrijskih oblika poput pločastih i ljuskastih elemenata, no oni neće biti predstavljeni u ovom radu.

Korištenje metode konačnih elemenata, kao i mnoge druge numeričke metode, omogućuju izbjegavanje integriranja koje je računski vrlo zahtjevno, a ponekad i nemoguće izvesti eksplicitno. Ono što metodu konačnih elemenata čini vrlo povoljnim za probleme unutar računalne grafike je što se jednadžbe konačnih elemenata mogu zapisati u obliku velike matrice koja sadrži globalnu jednadžbu konačnih elemenata. Matrice su iznimno povoljne za implementaciju paralelizma, čijom primjenom rješavamo vječni problem računalne grafike, a to su resursi.

Postupak određivanja tražene matrice započinje određivanjem problema koji se može riješiti konačnim elementima. Nakon toga slijedi odabir oblika elementa koji određuje način na koji će čvorne varijable i svojstva elementa utjecati jedno na drugo. Rješavanjem problema za jedan općeniti element, elementi se spajaju u globalnu cjelinu i tvore sustav konačnih elemenata.

Temeljem njihovih veza stvara se matrica globalne jednadžbe konačnih elemenata.



Slika 6. Simulacija sudara autombila korištenjem metode konačnih elemenata

2.1.1. Primjer linearnog konačnog elementa

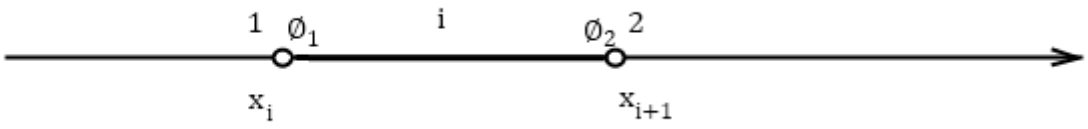
U postupku određivanja jednadžbe svih konačnih elemenata potrebno je prvo izvesti jednadžbu jednog konačnog elementa. Na sljedećem primjeru opisat ćemo postupak izvođenja gdje je neki razmatrani problem opisan jednodimenzijskom Poissonovom diferencijalnom jednadžbom (12).

$$-\alpha \frac{d^2\emptyset}{dx^2} = f(x) \tag{12}$$

Diferencijalna jednadžba je u intervalu $[a, b]$ s Dirichletovim i Neumannovim rubnim uvjetima (13).

$$\emptyset(a) = \emptyset_a$$
$$\alpha \frac{d\emptyset}{dx} |_b = h_b \tag{13}$$

Pretpostavlja se da je α konstantna veličina. Zadani interval podijelit ćemo u manje dijelove, podintervale $[x_i, x_{i+1}]$ gdje je $i = 1, \dots, n-1$. Svaki podinterval čini jednodimenzijski konačni element s dva čvora s nepoznatim vrijednostima funkcije $\emptyset$. Jedan takav element prikazan je na slici 7.



Slika 7. Jednodimenzijski element

Duž elementa pretpostavlja se linearna raspodjela funkcije ∅ (14).

$$\emptyset = a_1 + a_2 x \tag{14}$$

Pomoću rubnih uvjeta (15)

$$x = x_i \rightarrow \emptyset = \emptyset_1$$
$$x = x_{i+1} \rightarrow \emptyset = \emptyset_2 \ , \tag{15}$$

izračunava se vrijednost funkcije u čvorovima $\emptyset_1$ i $\emptyset_2$. Iz (14) i dobivaju se izrazi iz (16).

$$\emptyset_1 = a_1 + a_2 x_i$$
$$\emptyset_2 = a_1 + a_2 x_{i+1} \tag{16}$$

Iz (16) slijede izrazi za koeficijente a_1 i a_2 (17).

$$a_1 = \frac{\emptyset_1 x_{i+1} - \emptyset_2 x_i}{x_{i+1} - x_i}$$
$$a_2 = \frac{\emptyset_2 - \emptyset_1}{x_{i+1} - x_i} \tag{17}$$

Nakon uvrštavanja (17) u (14) dobiva se (18).

$$\emptyset = \frac{x_{i+1} - x}{x_{i+1} - x_i}\emptyset_1 + \frac{x - x_i}{x_{i+1} - x_i}\emptyset_2 \tag{18}$$

Jednostavnom supstitucijom vrijedi (19).

$$N_1(x) = \frac{x_{i+1} - x}{x_{i+1} - x_i}$$

$$N_2(x) = \frac{x - x_i}{x_{i+1} - x_i}$$

$$\emptyset = N_1(x)\emptyset_1 + N_2(x)\emptyset_2 \tag{19}$$

Sljedeći korak je izvod jednadžbe konačnog elementa u kojem će se primijeniti Galerkinova metoda težinskog reziduala pobliže opisana u []. Za područje konačnog elementa pretpostavit će se rješenje diferencijalne jednadžbe (12) u obliku funkcije (19). Prema Galerkinovoj metodi dobivamo težinske funkcije (20).

$$W_1 = N_1(x)$$

$$W_2 = N_2(x) \tag{20}$$

Težinska prosječna vrijednost za područje konačnog elementa je (21).

$$\int_{x_i}^{x_{i+1}} N_j \left(\alpha \frac{d^2\emptyset}{dx^2} + f(x) \right) dx = 0 \quad , \qquad j = 1{,}2 \tag{21}$$

Parcijalnom integracijom relacija (21) se transformira u (22).

$$\alpha \int_{x_i}^{x_{i+1}} \frac{dN_j}{dx} \frac{d\emptyset}{dx} dx = \int_{x_i}^{x_{i+1}} N_j f(x) dx + \left(N_j \alpha \frac{d\emptyset}{dx} \right) |_{x_i}^{x_{i+1}} \tag{22}$$

Uz zadanu supstituciju izvodimo (23).

$$\emptyset = N_k \emptyset_k, \frac{d\emptyset}{dx} = \frac{dN_k}{dx}\emptyset_k \quad , \quad k = 1{,}2$$

$$\left(\alpha \int_{x_i}^{x_{i+1}} \frac{dN_j}{dx}\frac{dN_k}{dx}dx\right) d\emptyset_k = \int_{x_i}^{x_{i+1}} N_j f(x)dx + \left(N_j \alpha \frac{d\emptyset}{dx}\right)|_{x_i}^{x_{i+1}}, \quad uz \begin{cases} j = 1{,}2 \\ k = 1, 2 \end{cases} \tag{23}$$

Zbrajanjem po indeksu k dobiva se (24). Drugi član na desnoj strani je oznaka za Neumannov rubni uvjet za funkcije oblika (19).

$$j = 1, \qquad N_1 \alpha \frac{d\emptyset}{dx}|_{x_i}^{x_{i+1}} = N_1(x_{i+1})\alpha \frac{d\emptyset}{dx}|_{x=x_{i+1}} - N_1(x_i)\alpha \frac{d\emptyset}{dx}|_{x=x_i} = -\alpha \frac{d\emptyset}{dx}(x_i)$$

$$j = 2, \qquad N_2 \alpha \frac{d\emptyset}{dx}|_{x_i}^{x_{i+1}} = N_2(x_{i+1})\alpha \frac{d\emptyset}{dx}|_{x=x_{i+1}} - N_2(x_i)\alpha \frac{d\emptyset}{dx}|_{x=x_i} = \alpha \frac{d\emptyset}{dx}(x_i) \tag{24}$$

Relacija (23) sada se može zapisati u matričnom obliku (25).

$$\begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix} \begin{bmatrix} \emptyset_1 \\ \emptyset_2 \end{bmatrix} = \begin{bmatrix} F_{S1} \\ F_{S2} \end{bmatrix} + \begin{bmatrix} -\alpha \frac{d\emptyset}{dx}(x_i) \\ \alpha \frac{d\emptyset}{dx}(x_i) \end{bmatrix}$$

$$k_{ij} = \alpha \int_{x_i}^{x_{i+1}} \frac{dN_j}{dx}\frac{dN_k}{dx}dx \ , uz \begin{cases} j = 1{,}2 \\ k = 1, 2 \end{cases}$$

$$F_{Sj} = \int_{x_i}^{x_{i+1}} N_j f(x)dx \, , uz \; j = 1{,}2 \tag{25}$$

Jednadžba konačnog elementa zadana je u kao matrični oblik (26).

$$\boldsymbol{k^i \emptyset^i} = \boldsymbol{F_S^i} + \begin{bmatrix} -\alpha \frac{d\emptyset^i}{dx}(x_i) \\ \alpha \frac{d\emptyset^i}{dx}(x_{i+1}) \end{bmatrix} \tag{26}$$

Za simbole vrijedi da je $\boldsymbol{k^i}$ matrica krutosti elementa, $\emptyset^{\boldsymbol{i}}$ vektor nepoznatih veličina u čvorovima, a $\boldsymbol{F_S^i}$ vektor opterećenja. Poznavanje $\boldsymbol{k^i}$ i $\boldsymbol{F_S^i}$ dobivamo vektor nepoznatih veličina što smo zapravo i tražili.

Kako se u metodi konačnih elemenata tražena pojava sastoji od više elemenata koji su međusobno spojeni zajedničkim čvorovima $\emptyset^{\boldsymbol{i}}$, poznavanjem susjedstva elemenata i jednadžbom svakog od elementa može se dobiti globalna jednadžba konačnih elemenata (27)**.** Poznavanjem **K** i $\boldsymbol{R_S}$, te rubnih uvjeta može se odrediti vrijednost funkcija $\emptyset$.

$$\boldsymbol{K\emptyset = R_S} \tag{27}$$

2.2. Metoda konačnih razlika

Metoda konačnih razlika je numerička metoda koja pomoću jednadžbi konačnih razlika aproksimira derivacije elemenata unutar diferencijalnih jednadžbi. Metoda se zasniva na aproksimaciji formule za derivaciju funkcije (28).

$$f'(x) = \lim_{t \to 0} \frac{f(x+t) - f(x)}{t} \tag{28}$$

Vrijednost t se zamjenjuje proizvoljnim, dovoljno malim parametrom t te se dobiva aproksimacija zadana u (29).

$$f'(x) \approx \frac{f(x+t) - f(x)}{t} \tag{29}$$

2.3. Metoda konačnih volumena

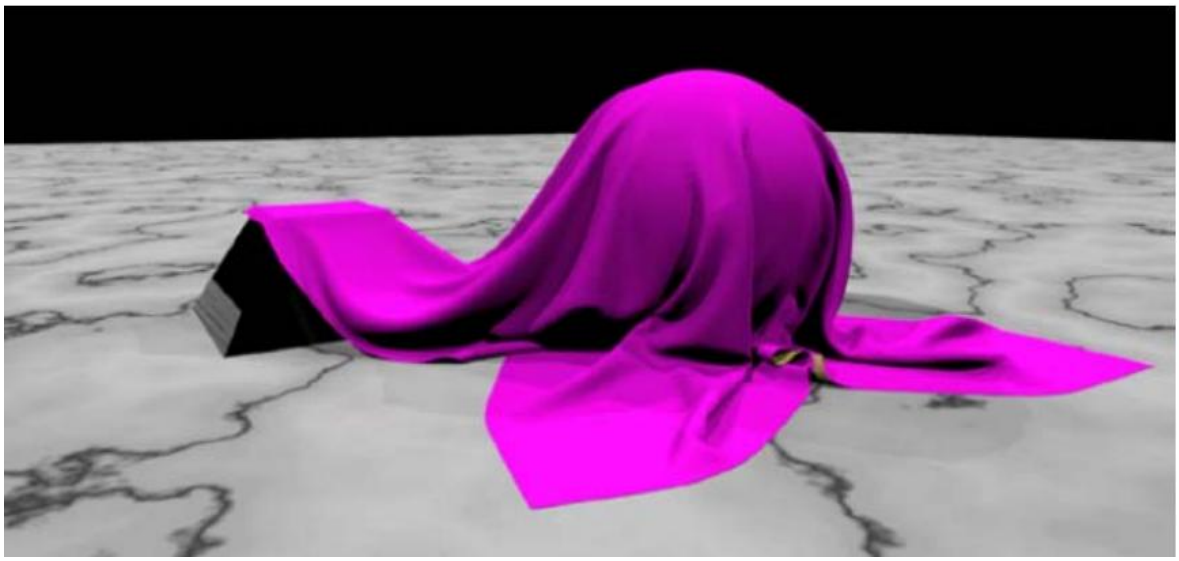
Metoda konačnih volumena je numerička metoda za aproksimaciju diferencijalnih jednadžbi pomoću jednadžbi na sličan način poput metode konačnih elemenata i metode konačnih razlika. Volumen se dijeli na velik broj manjih volumena koji okružuju svaki čvor na Lagrangeovoj mreži. Volumni integrali u parcijalnoj diferencijalnoj jednadžbi pretvoreni su u integrale po površini koristeći Gaussov teorem divergencije. Zatim se izračunavaju tokovi, primjerice energije ili sile iz jednog volumnog elementa u susjedni. Na temelju tokova mogu se odrediti vrijednosti čvorova. Metoda je povoljna za nedovoljno dobro strukturirane mreže, pa se često koristi pri izračunavanju problema kod dinamike fluida.

2.4. Metoda konačnih granica

Metoda konačnih granica je zanimljiva alternativa rješavanju diferencijalnih jednadžbi pomoću metode konačnih elemenata, kada se o elementima razgovara kao trodimenzionalnim entitetima. Umjesto da se element gleda kroz njegov volumen, slično kao i kod metode konačnih volumena, svi izračuni se obavljaju na površini elementa. Time se problem iz tri dimenzije spušta na problem dvije dimenzije. No, ovakav pristup moguć je samo za elemente koji se sastoje od homogenog materijala. Također, topološke promjene objekta, poput lomova, teško se implementiraju, odnosno zahtijevaju veće promjene u postojećem matematičkom modelu.

3. Sustav masa i opruga

Sustav masa i opruga jedan je od najjednostavnijih načina za modeliranje deformabilnih objekata. Za razliku od metode konačnih elemenata gdje se objekt predstavlja beskonačnim skupom točaka, pa se zatim diskretizira, ovakva metoda već započinje s diskretnim modelom. Svaka od masa povezana je sa susjednim masama s oprugom. Stanje sustava u vremenu t zadano je položajem masa x i brzinom v. Sila f koja se vrši nad masom određuje se pomoću opruga koje su spojene s masom i položajem susjednih masa, te vanjskim silama poput gravitacije ili trenja koje djeluju na masu. Ubrzanje mase, a po njemu i brzina i položaj, određuju se pomoću drugog Newtonovog zakona (5).



Slika 8. Modeliranje tkanine pomoću sustava masa i opruga

4. Programsko rješenje - Deforma

U ovom radu cilj je bio proučiti modele deformabilnih elemenata i implementirati programsko rješenje koje će simulirati ponašanje navedenih entiteta. Programsko okruženje u kojem je aplikacija *Deforma* razvijana je Visual Studio 2010. Izabrani programski jezik je C#. Ovaj izbor učinjen je kako bi se što lakše predstavio zadani koncept.

Za iscrtavanje je izabrana OpenTK biblioteka. Ona implementira aplikacijsko programsko sučelje (engl. *Application programming interface, API*) OpenGL te omogućava korištenje u programskom jeziku C#. OpenGL je otvorena standardna biblioteka za pisanje aplikacija koje koriste 2D i 3D računalnu grafiku.

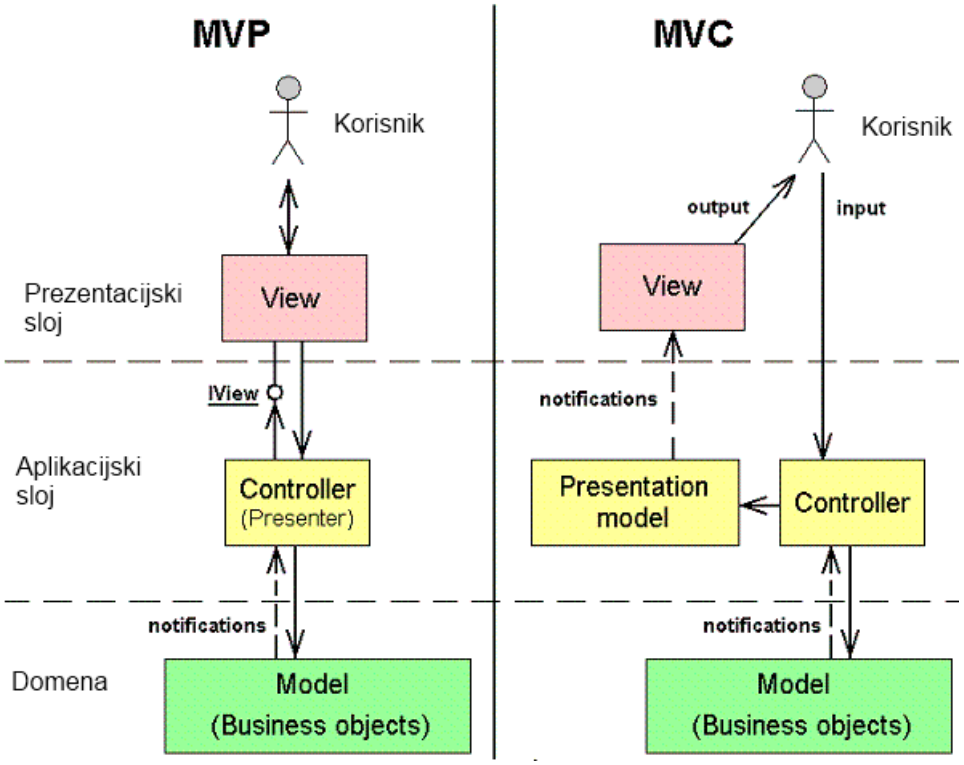
Deforma je napravljena prema praksi dobrog objektnog oblikovanja. Takav način omogućuje da se postojeće rješenje može lako nadograditi novim modulima, dodavati nove, te prenijeti dio ili u potpunosti model domene u novu aplikaciju. Aplikacija je napisana pomoću oblikovnog obrasca MVP (engl. *Model View Presenter*).

4.1. MVP oblikovni obrazac

Model Pogled Prezenter je oblikovni obrazac koji je izveden iz obrasca Model Pogled Kontroler (engl. *Model View Controller, MVC*). MVP je za razliku od poznatijeg oblikovnog obrasca MVC bolje prilagođen radu sa Windows Formama pomoću kojih je napravljena aplikacija.

Oblikovni obrazac Model Pogled Prezenter već u nazivu sadrži građevne jedinice na temelju kojih se grade aplikacije koje ga primjenjuju. *Model* označava model domene, odnosno dio aplikacije u kojem se nalazi logika problema koji se

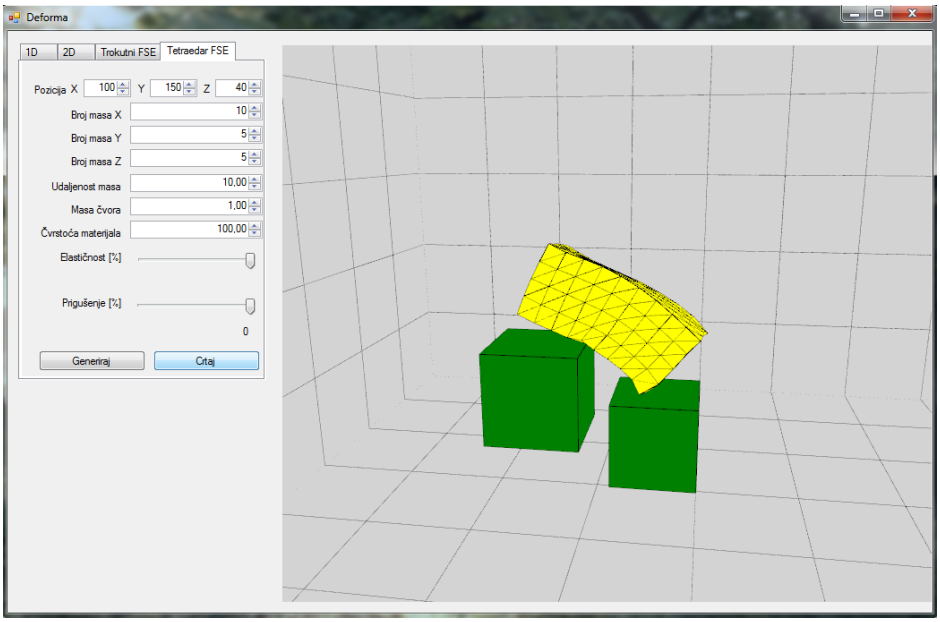
pokušava riješiti. *Pogled* je dio koji je zaslužan za interakciju sa korisnikom. U njemu se nalazi sučelje, te je to jedini dio aplikacije kojeg korisnik vidi. *Prezenter* je zadužen za povezivanje modela i pogleda. Na temelju korisnikovih akcija nad pogledom, kontroler odlučuje koje će ulaze poslati pojedinim dijelovima modela. Model šalje izlaze kontroleru, a kontroler bira koji će dijelovi pogleda biti prikazani korisniku za najbolji mogući prikaz dobivenih rezultata.



Slika 9. Prikaz oblikovnih obrazaca MVP i MVC

4.2. Ideja aplikacije

Deforma je osmišljena kako bi se pomoću konačnih elemenata modelirali elastični i plastični objekti. Aplikacija je također odijeljena na omogućavanje dvodimenzionalnih i trodimenzionalnih simulacija. Na scenu svake od tih simulacija moguće je dodavati deformabilne objekte, te jednostavne nedeforrmabilne objekte koji služe kako bi se omogućili sudari i simulirale deformacije. U aplikaciji je implementirana i sila gravitacije koja uvodi dinamiku u sustav i u svakom trenutku simulacije pomiče deformabilne objekte.



Slika 10. Prikaz sučelja Deforme

4.3. Modeliranje konačnih elemenata

Konačni elementi su središnji dio aplikacije. Dizajn aplikacije omogućuje dodavanje različitih vrsta konačnih elemenata. Elementi mogu biti različitih oblika i rješavati različite diferencijalne jednadžbe. Konačni se elementi osim svog oblika sastoje i od dva ili više čvorova.

4.3.1. Čvorovi

Čvorovi su, kao što je zadano metodom konačnih elemenata, objekti koji omogućavaju spajanje konačnih elemenata u sustav konačnih elemenata. Za aplikaciju su čvorovi veoma bitni jer njihov položaj zapravo označava vrhove poligona za iscrtavanje, a modeliranje sudara i djelovanje sila također se vrši na temelju atributa svakog čvora.

Čvorovi sadrže atribute položaja, brzine, ubrzanja, mase, inicijalnog položaja i položaja u prošlom vremenskom trenutku. Čvorovi su modelirani u klasi *Node* (programski odsječak 1).

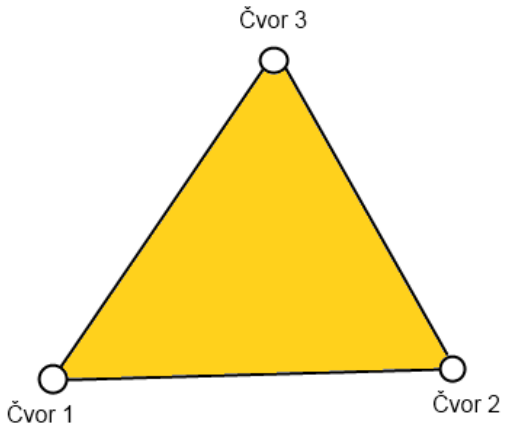
```
public class Node
{
   public Vector3 Position { get; set; }
   public Vector3 RestPosition { get; set; }
   public Vector3 PositionOld { get; set; }
   public Vector3 Speed { get; set; }
   public Vector3 Acceleration { get; set; }
   public float Mass { get; set; }

   public Node(Vector3 position, Vector3 speed, Vector3 acceleration, float mass);
}
```

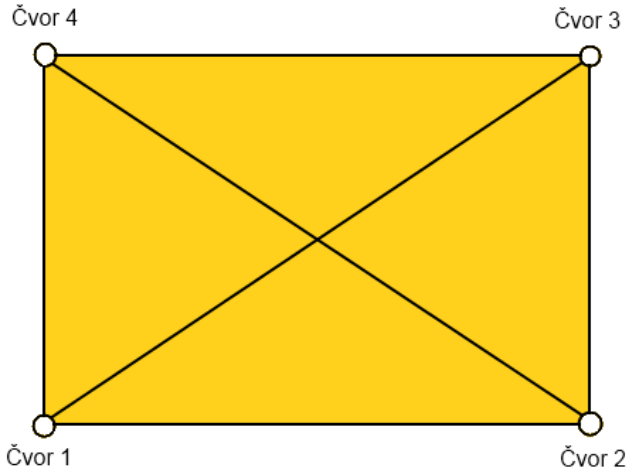
Programski odsječak 1.

4.3.2. Konačni elementi

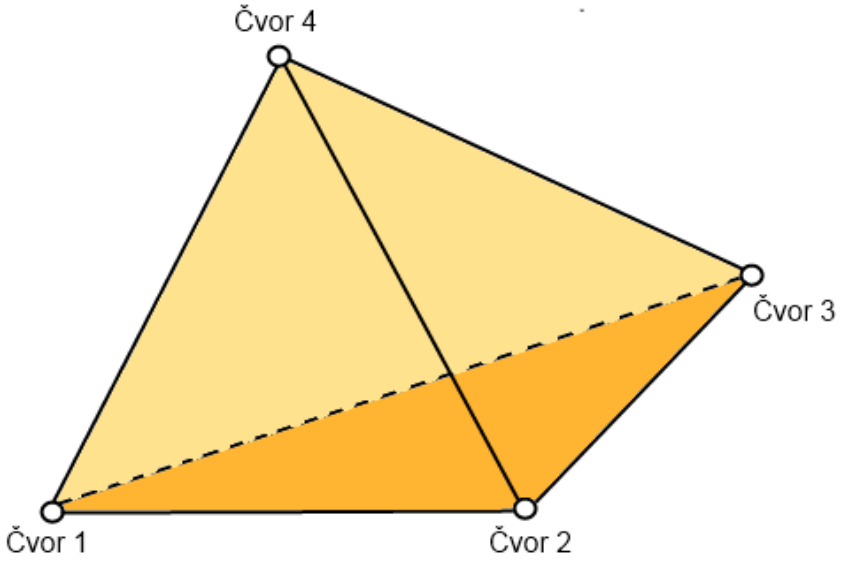
Deforma ima implementirana tri oblika konačnih elemenata. Dvodimenzionalni elementi koji su implementirani su trokutni (slika 11) i četverokutni (slika 12) konačni elementi, dok je za trodimenzionalni element izabran tetraedarski konačni element (slika 13).



Slika 11. Trokutni konačni element



Slika 12. Četverokutni konačni element



Slika 13. Tetraedarski konačni element

Konačni elementi implementirani su kako bi rješavali Hookeovu diferencijalnu jednadžbu (30), u kojoj je m masa tijela, x pomak tijela, t vrijeme, k konstanta opruge, a c konstanta prigušenja.

$$m\frac{d^2x}{dt^2} = -kx - c\frac{dx}{dt} \tag{30}$$

Hookeov konačni element riješava difirencijalnu jednadžbu za sve susjedne čvorove u elementu, gdje se pretpostavlja da se između svakog od čvorova nalazi opruga konstante k koja je definirana nad cijelim konačnim elementom. Računa se sila koja se vrši nad svakim od čvorova sa obzirom na susjedni čvorove. Na temelju dobivenih sila pomoću (5) se izračunava vektor ubrzanja čvora koji se dodaje na već postojeći vektor ubrzanja čvora. Pseudokod za izračunavanje akceleracije koja je uzrokovana silama unutar konačnog elementa za jedan par susjednih čvorova elementa prikazan je u programskom odsječku 2, gdje je x vektor pomaka iz ravnotežnog položaja za sve koordinatne osi.

```
Izračunaj Slijedeći Trenutak(čvor1, čvor2, Δt)
       x = IzračunajPomakIzRavnotežnogPolozaja();
      sila = - k*x;
      čvor1.Akceleracija += sila/čvor1.masa - c * (sila/čvor1.masa)* Δt;
      čvor2.Akceleracija += -sila/čvor2.masa - c*(-sila/čvor1.masa)* Δt;
```

Programski odsječak 2.

Klase koje modeliraju sustav konačnih elemenata sastoje se od tipova konačnih elemenata koji odgovaraju njihovom imenu. Implementirane su tri klase sustava konačnih elemenata, trokutni, četverokutni i tetraedarski sustav konačnih elemenata. Oni nasljeđuju apstraktnu klasu *FiniteElementSystem* prikazanu u programskom odsječku 1. Apstraktna klasa sadrži listu konačnih elemenata, listu čvorova i metodu *AppendSystem* koja dodaje vremenski odsječak, pomoću kojeg se računa sljedeće stanje svakog od elemenata.

```
public abstract class FiniteElementSystem
{
    public List<Node> SystemNodes { get; set; }
    public List<FiniteElement> SystemElements { get; set; }

    public virtual void AppendSystem(float timeInterval)
    {
        foreach (FiniteElement systemElement in SystemElements)
        {
           systemElement.AppendNodes(timeInterval);
        }
    }
}
```

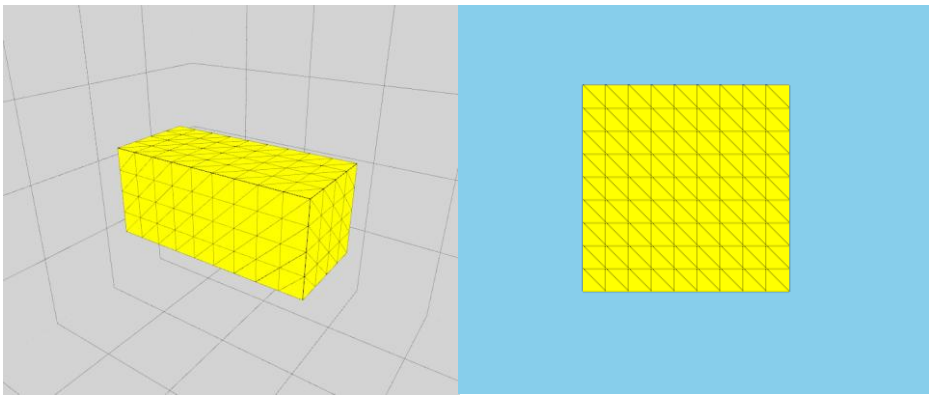
Programski odsječak 3.

4.3.3. Sustav konačnih elemenata

Sustav je matematički model nekog promatranog objekta. Sustav sadrži u sebi jedan ili više elemenata nekog tipa, kao i dva ili više čvora. U ovom radu sustav je zamišljen kao skup konačnih elemenata koje predstavljaju jedno homogeno tijelo, istih ponašajnih karakteristika u svojoj cijelosti. No, moguće je na temelju postojećeg modela domene modelirati i nehomogene sustave konačnih elemenata.

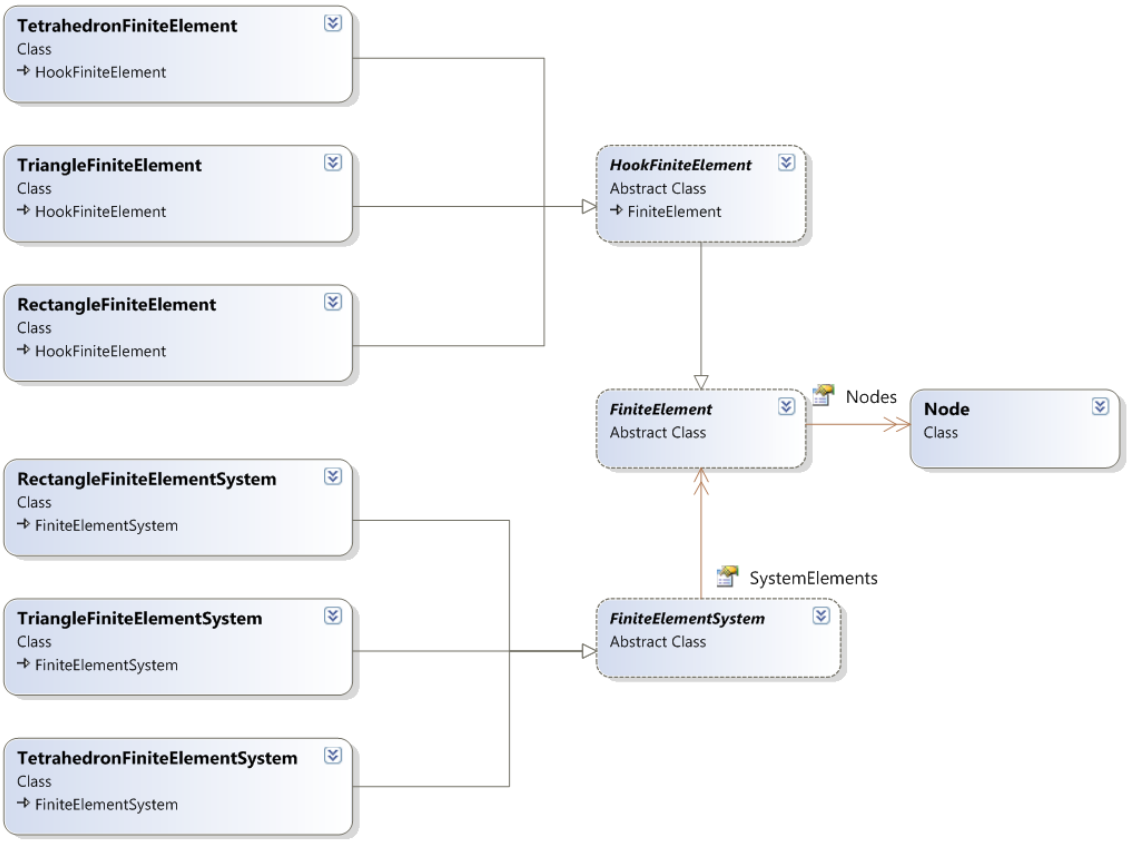
Sustavi konačnih elemenata grade se na temelju nekog zadanog algoritma, pomoću kojeg se na neki način sustavu dodaju čvorovi, te se iste povezuje sa jednim ili više konačnih elemenata.

U ovom radu primijenjeni su jednostavni algoritmi za izgradnju sustava u obliku pravokutnika u dvodimenzionalnim sustavima konačnih elemenata, odnosno u obliku kvadra u trodimenzionalnim sustavima konačnih elemenata. Neki od ulaznih parametara koji se koriste su početna pozicija sustava, broja čvorova po svakoj od osi Kartezijevog koordinatnog sustava, međusobna udaljenost čvorova po osima i masa čvora.



Slika 14. Simulirani 3D i 2D elementi u aplikaciji Deforma

Algoritmi za popunjavanje već postojećih dvodimenzionalnih i trodimenzionalnih objekata postoje, no nisu primjenjivani zbog njihove kompleksnosti, te bi njihova implementacija nadišla temu ovoga rada.

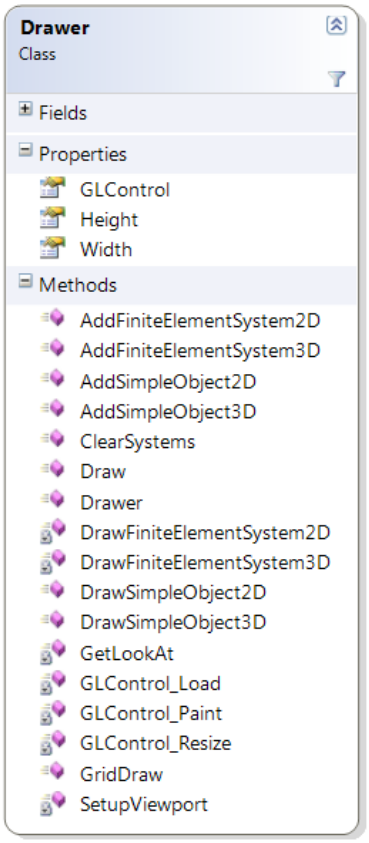


Slika 15. Dijagram klasa konačnih elemenata

4.4. Iscrtavanje

Pri iscrtavanju primjenjivala se već spomenuta biblioteka OpenTK. OpenTK sadrži Windows Forms kontrolu zvanu *GLControl* koju se može implementirati u vizualno sučelje forme. Ta kontrola omogućava stvaranje scene i njeno iscrtavanje na području kontrole pomoću OpenGL naredbi.

Klasa koja je zadužena za iscrtavanje sustava konačnih elemenata kao i ostalih objekata u sceni poput koordinatnog sustava ili jednostavnih objekata zove se *Drawer* (slika 16).



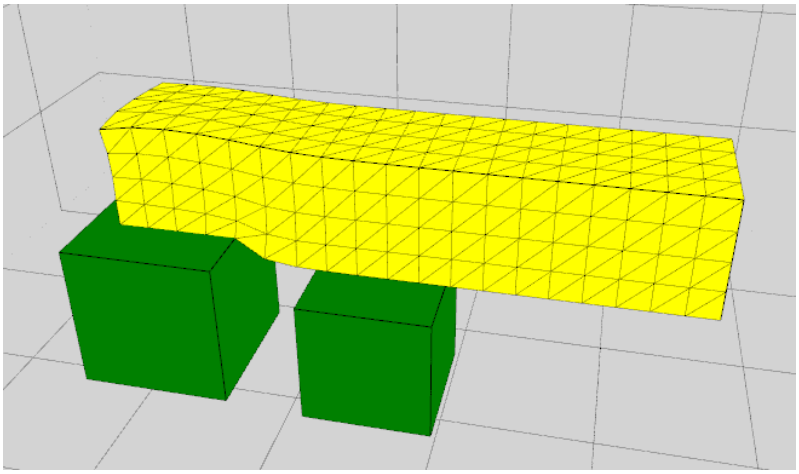
Slika 16. Klasa Drawer

4.5. Detekcija sudara

Za uspješno modeliranje deformabilnih objekata vrlo je važno dobro detektirati sudare i omogućiti primjenjivanje različitih sila.

Za detekciju sudara u ovom je radu zadužena klasa *CollisionDetector* koja pomoću raznih statičkih metoda koje primaju razne tipove objekata gleda hoće li se objekti na temelju svoga položaja, brzine i akceleracije u sljedećem vremenskom trenutku sudariti.

U ovom je radu primijenjen jednostavan, ali spor algoritam za detekciju sudara. On prilikom svakog vremenskog trenutka za svaki čvor svakog sustava konačnih elemenata provjerava hoće li se sustav sudariti sa nekim drugim objektom u sceni.



Slika 17. Detekcija sudara deformabilnog objekta sa jednostavnim objektom

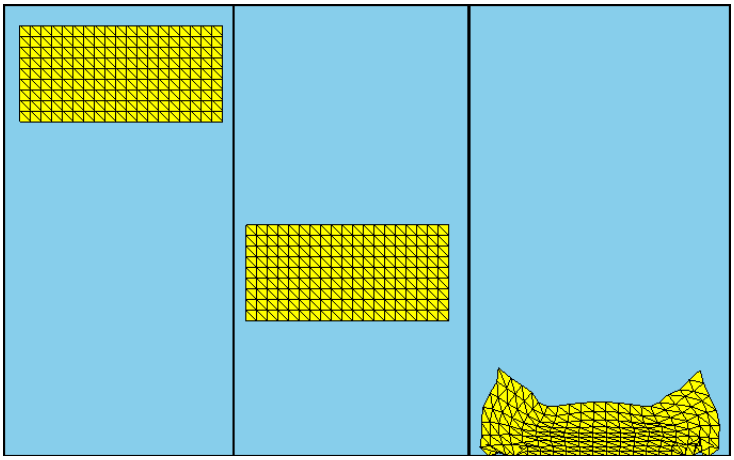
Za jednostavne objekte, prikazane na slici 17 zelenom bojom, primjenjuje se *MinMax* metoda detekcije. *MinMax* metoda odredi minimalnu i maksimalnu vrijednost za svaku koordinatnu os, koje predstavljaju maksimalnu udaljenost

točaka objekta. Ukoliko za neki čvor vrijedi da su koordinate njegovog položaja veće od minimalne vrijednosti, odnosno manje od maksimalne vrijednosti objekta, respektivno po koordinatnim osima, tada je čvor unutar objekta.

Ako će postojati sudar vrši se obrada sudara, gdje se na neki od načina, prikladnih vrsti sudara, izmijenjuju već spomenuti parametri brzine, akceleracije i položaja kako bi se izbjegao sudar sustava elemenata i objekta.

4.6. Simuliranje sila

Sve sile možemo opisati prema drugom Newtonovom zakonu (5). Njihovo djelovanje na neko tijelo očituje se sa dodavanjem određene akceleracije koja je proporcionalna iznosu sile i obrnuto proporcionalna masi tijela na koje sila djeluje. Na tom principu implementirano je sučelje za dodavanje sila koje vrše svoj utjecaj na deformabilna tijela, odnosno čvorove tijela.



Slika 18. Djelovanje gravitacije

Zaključak

Prvotna ideja ovoga rada bila je i ostala napraviti aplikaciju koju će drugi moći shvatiti kako bi nastavili sa istraživanjem ovog tipa u računalnog grafici. Predstavljeni su osnovni koncepti za modeliranje deformabilnih objekata i njihovu simlaciju, kao i dobra praksa u objektnom dizajnu, a sve u svrhu lakšeg rada na unaprijeđenju postojećih implementiranih temelja.

Kompleksnosti i zahtjevnosti modeliranja i simuliranja deformabilnih objekata autor je pokušao doskočiti korištenjem i objašnjavanjem metode konačnih elemenata, a da su pritom prikazane njene prednosti i mane.

Mjera uspješnosti ovog rada bit će broj ljudi koje je zainteresiralo za ovo područje, po njegovu čitanju i korištenju aplikacije Deforma koja uz rad dolazi. Najveća želja autora je da osobe koje će raditi na unaprijeđenju ovog rada uvaže sve dobre strane, a primjete za sve loše strane, promašaje i pogreške. A nakon toga sve te loše strane, promašaje i pogreške pretvore u izvrsne radove, pogotke i uspjehe.

Literatura

[1] Sorić J., *Uvod u numeričke metode u strojarstvu*, Zagreb : Fakultet strojarstva i brodogradnje 2009.

[2] Nealen A., Müller M., Keiser R., Boxerman E., Carlson M., *Physically Based Deformable Models in Computer Graphics*, EuroGraphics 2005, Dublin, Republia Irska, 2005., stranice 71 - 94

[3] Desburn M., Schröder P., Barr A., *Interactive Animation of Structured Deformable Objects*, EuroGraphics 1999, Milan, Italija, 1999.

[4] Brown J., Sorkin S., Bruyns C., Latombe J. -C., Montgomery K., Stephanides M., *Real-Time Simulation of Deformable Objects: Tools and Aplikaction*, The Fourteenth Conference on Computer Animation, Seul, Južna Koreja, 2001, stranice 228 – 259

[5] Gissler M., *Simulation and Visualisation of Topology-changing Plastic Material*, Central European Seminar on Computer Graphics, Budemerice, Slovačka, stranice 121 – 128

[6] *Introduction to Finite Element Methods, Participants workbook*, ASME International

[7] Gusev A. A., *Finite Element Mapping for Spring Network Representation of the Mechanics of Solids,* Institute of Polymers, Department of Materials, ETH Zürich, Švicarska

[8] Molino N., Bridson R., Fedkiw R., *Tetrahedral Mesh Generation for Deformable Bodies*, 20st Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH, 1993.

[9] Teschner M., Kimmerle S., Heidelberger B., Zachmann G., Raghupathi L., Fuhrmann A., Cani M. -P., Faure F., Magnenat-Thalmann N., Strasser W., Volino P. *Collision Detection for Deformable Objects*, EuroGraphics, Norköping, Švedska, stranice 119 – 139

[10] Lizier M. A. S., Shepherd J. F., Nonato L. G., Comba J. L. D., Silva C. T., *Comparing Techniques for Tetrahedral Mesh Generation*, Inaugural International Conference of the Engineering Mechanics Institute

[11] Wicke M., Ritchie D., Klinger B. M., Burke S., Shewchuk J. R., O'Brien J.M., *Dynamic local Remeching for Elastoplastic Simulation,* SIGGRAPH 2010, Los Angeles, Sjedinjene Američke Države, 2010., stranice 49:1 - 11

[12] Galoppo N., Otaduy M. A., Mecklenburg P., Gross M., Lin M. C., *Fast Simulation of Deformable Models in Contact Using Dynamic Deformation Textures,*ACM SIGGRAPH Symposium on Computer Animation, Boston, Massachusetts, Sjedinjene Američke Države, 2006., stranice 73-82

[13] Labelle François, Shewchuk J. R., *Isosurface Stuffing: Fast Tetrahedral Meshes with Good Dihedral Angels*, SIGGRAPH 2007, San Diego, California, Sjedinjene Američke Države, 2007.

[14] Wikipedia, *Elasticity (physics)*, http://en.wikipedia.org/wiki/Elasticity_(physics), 04. travnja 2011.

[15] Wikipedia, *Plasticity (physics),* http://en.wikipedia.org/wiki/Plasticity_(physics), 04. travnja 2011.

[15] Wikipedia, *Linear Elasticity*, http://en.wikipedia.org/wiki/Linear_elasticity, 04. travnja 2011.

[16] Wikipedia, *Pseudoelasticity*, http://en.wikipedia.org/wiki/Pseudoelasticity, 04. travnja 2011.

[17] Wikipedia, *Finite element method*, http://en.wikipedia.org/wiki/Finite_element_method, 04. travnja 2011.

[18] Wikipedia, *Partial differential equation,* http://en.wikipedia.org/wiki/Partial_differential_equation, 04. travnja 2011.

Sažetak

Modeliranje i simulacija deformabilnih objekata

U ovo radu predstavljen su metode za stvaranje raznih modela deformabilnih objekata. Uz pojašnjenje plastičnosti i elastičnosti koji su osnovni koncepti pri shvaćanju deformabilnih tijela, prikazani su i matematički modeli na kojem se temelji simulacija takvih tijela. Poseban naglasak stavljen je na matematičke metode koje koriste Lagrangeove mreže, sa posebnim pojašnjenjem metode konačnih elemenata i spominjanje ostalih sličnih metoda, poput metode konačnih volumena, konačnih razlika i sličnih. Predstavljena je i metoda modeliranja deformabilnih tijela korištenjem sustava masa i opruga.

U nastavku rada, opisana je aplikacija Deforma napisana pomoću programa C# koja koristi poznato aplikacijsko programsko sučelje OpenGL za modeliranje objekata u 2D i 3D računalnoj grafici implementiranih u OpenTK biblioteci.. Aplikacija je namjenjena simulaciji predstavljenih koncepata iz teoretskog dijela rada.

Ključne riječi: računalna grafika, deformabilni objekti, elastičnost, plastičnost, Lagrangeova mreža, metoda konačnih elemenata, C#, OpenGL, OpenTK, MVC, MVP

Summary

Modeling and simulation of deformable objects

This paper presents the methods for creating and understanding the diverse models of deformable objects. The basic concepts for understanding the basics of deformble objects, elasticity and plasticity are presented. Also, mathematical methods on which such objects can be represented are also contained within this papaer.

Mathematical methods which include Lagrangian meshes are presented more thoughrouly in the form of finite element method, finite differences method, finite volume methods and others. There is also mention of another method for modeling deformable objects called mass spring system.

An application called Deforma is also presented. It's purpose is to simulate the concepts shown in the theoretical part of the paper. Deforma uses the programming language C# and the popular application programming interface OpenGL, for modeling in computer graphics, implemented in the OpenTK library.

Key words: computer graphics, deformable objects, elasticity, plasticity, finite element method, Lagrangian mesh, C#, OpenGL, OpenTK, MVC, MVP