SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1565

ANIMACIJA TOKA FLUIDA

Jakov Fuštin

Zagreb, studeni 2005.

Sadržaj

1. Uvod 1
2. Dinamika fluida 2
2.1. Jednadžba kontinuiteta 2
2.2. Zakon o očuvanju količine gibanja 4
2.3. Zakon o očuvanju energije 5
2.4. Navier-Stokesove jednadžbe 6
3. Računalni model 7
3.1. Helmholtz-Hodgeova dekompozicija 7
3.2. Diskretizacija prostora 8
3.3. Diskretizacija vremena 10
3.4. Rješavanje Navier-Stokesovih jednadžbi 10
3.5. Gibanje supstanci kroz fluid 11
3.6. Prikaz rezultata 12
4. Simulacije vođene konačnim stanjem 14
4.1. Izraz za silu - $F(\rho,\rho^*)$ 15
4.2. Izraz za sprječavanje difuzije dima - $G(\rho,\rho^*)$ 16
4.3. Parametri simulacije 17
4.4. Rješavanje prepravljenih Navier-Stokesovih jednadžbi 18
5. Programsko rješenje 20
5.1. Strukture podataka i algoritmi 20
5.1.1. Primjena vodeće sile 23
5.1.2. Prigušenje količine gibanja 24
5.1.3. Vođenje brzine 25
5.1.4. Projekcija 26
5.1.5. Vođenje dima 28
5.1.6. Sprječavanje difuzije dima 28
5.2. Iscrtavanje na ekran 29
5.3. Grafičko korisničko sučelje 30
5.3.1. Konfiguracijska cjelina 31
5.3.2. Cjelina za učitavanje slika 31
5.3.3. Cjelina za određivanje iznosa parametara 32

5.3.4. Upravljačka cjelina ... 32
6. Rezultati ... 35
6.1. Utjecaj parametara na simulaciju ... 37
6.1.1. Parametar v_f ... 37
6.1.2. Parametar v_d ... 38
6.1.3. Parametar v_g ... 39
6.1.4. Parametar σ ... 41
6.2. Vremena izvođenja ... 43
7. Zaključak ... 46
8. Literatura ... 47
Dodatak: Konfiguracijska datoteka ... 48
Sažetak ... 49
Abstract ... 50

Popis slika

Slika 1. Scena iz filma Gospodar prstenova – brod i prsten od dima 1
Slika 2. Volumen V_0 3
Slika 3. Sve varijable su definirane u središtu ćelija 8
Slika 4. Vektorske vrijednosti su definirane na bridovima ćelija 9
Slika 5. Primjer 1 12
Slika 6. Primjer 2 13
Slika 7. Primjer 3 13
Slika 8. Prikaz ćelija oko komponente brzine *u* 21
Slika 9. Prikaz ćelija oko komponente brzine *v* 22
Slika 10. Grafičko korisničko sučelje 30
Slika 11. Konfiguracijska cjelina 31
Slika 12. Cjelina za učitavanje slika 31
Slika 13. Dijalog za učitavanje slika 32
Slika 14. Cjelina za određivanje iznosa parametara 32
Slika 15. Upravljačka cjelina 33
Slika 16. Prozor u kojem se odvija simulacija 33
Slika 17. Grafičko korisničko sučelje za vrijeme simulacije 34
Slika 18. Početno i konačno stanje (psi → omega) 35
Slika 19. Međustanja (prva slika je gore lijevo, a posljednja dolje desno) 35
Slika 20. Početno i konačno stanje (križić → kvačica) 36
Slika 21. Međustanja animacije 36
Slika 22. Simulacija uz parametar $v_f = 1.5$ 37
Slika 23. Početno i konačno stanje – utjecaj parametra v_d 38
Slika 24. Simulacija uz iznos parametra $v_d = 1$ 38
Slika 25. Simulacija uz iznos parametra $v_d = 5.5$ 39
Slika 26. Početno i konačno stanje – utjecaj parametra v_g 40
Slika 27. Simulacija uz iznos parametra $v_g = 0$ 40
Slika 28. Simulacija uz iznos parametra $v_g = 5 \cdot 10^{-6}$ 41
Slika 29. Početno i konačno stanje – utjecaj parametra σ 42
Slika 30. Simulacija uz iznos parametra $\sigma = 1$ 42
Slika 31. Simulacija uz iznos parametra $\sigma = 7$ 43
Slika 32. Izgled konfiguracijske datoteke 48

1. Uvod

Svjedoci smo sve češćeg pojavljivanja specijalnih efekata koji sadrže realistične animacije vatre, dima i tekućine u filmovima i računalnim igrama. Oni se obično koriste da film ili igra dobiju na uvjerljivosti, ali isto tako i da nas očaraju i ostave pod dojmom još dugo nakon gledanja, odnosno igranja. Međutim, ono što se većina ljudi rijetko ili skoro nikada ne zapita je kako ti efekti nastaju.

Slika 1. Scena iz filma Gospodar prstenova – brod i prsten od dima

Cilj ovog rada je pokazati kako se mogu postići realistične animacije toka fluida simulacijom na računalu. Prvo se daje teorijski uvod koji nije nužan za samu implementaciju, ali je koristan i ovdje se nalazi zbog kompletnosti. Potom se opisuje računalni model i metoda prezentirana od strane Jos Stama u radu "Stable Fluids" [2] nakon čega se opisuje metoda prezentirana u radu "Target-Driven Smoke Animation" [3] koja zapravo predstavlja nadgradnju Stamove metode. Na kraju je načinjena implementacija potonje metode te dan komentar rezultata koji se dobivaju njenim korištenjem.

2. Dinamika fluida

Mehanika fluida je grana mehanike koja se dijeli na dvije discipline, a to su statika i dinamika fluida. Statika fluida proučava fluide u mirovanju i kao takva nam nije od posebnog interesa u ovom radu. S druge strane, dinamika fluida se bavi proučavanjem gibanja fluida te ćemo se njome detaljnije pozabaviti.

Stanje svakog fluida koji se giba u trodimenzionalnom prostoru opisano je s točno pet fizikalnih veličina. To su tri komponente brzine te dvije termodinamičke veličine koje se odnose na fluid, kao što su npr. tlak i gustoća. Sve navedene veličine su funkcije položaja i vremena, odnosno:

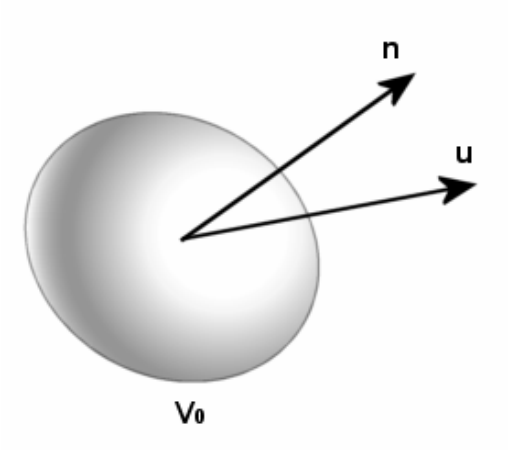
$$\mathbf{u} = u_x(\mathbf{x},t)\mathbf{i} + u_y(\mathbf{x},t)\mathbf{j} + u_z(\mathbf{x},t)\mathbf{k}$$
$$p = p(\mathbf{x},t)$$
$$\rho = \rho(\mathbf{x},t)$$

Određivanje stanja fluida svodi se na izračunavanje vrijednosti tih pet varijabli. Pri tom se služimo jednadžbom kontinuiteta, zakonom o očuvanju količine gibanja te zakonom o očuvanju energije.

2.1. Jednadžba kontinuiteta

Neka je s V_0 označen neki volumen u prostoru, takav da su mu granice ∂V_0 glatke i da se tijekom vremena ne mijenjaju. Neka je s **n** označena normala prema van u svakoj točki na ∂V_0, a s dA infinitezimalni element površine na ∂V_0. Zakon o očuvanju mase kaže da je brzina promjene mase unutar volumena V_0 jednaka brzini kojom masa prelazi granicu ∂V_0:

$$\frac{d}{dt}\int_{V_0} \rho dV = -\int_{\partial V_0} \rho \mathbf{u}\cdot\mathbf{n} dA \qquad (1)$$



Slika 2. Volumen V_0

Korištenjem teorema o divergenciji, desna strana jednadžbe (1) može se zapisati na drugi način:

$$-\int_{\partial V_0} \rho \mathbf{u} \cdot \mathbf{n} dA = -\int_{V_0} \operatorname{div}\left(\rho \mathbf{u}\right) dV$$

Uvrštavanjem dobivenog izraza u jednadžbu (1) dobiva se integralni oblik zakona o očuvanju mase:

$$\frac{d}{dt}\int_{V_0} \rho dV + \int_{V_0} \operatorname{div}\left(\rho \mathbf{u}\right) dV = \int_{V_0} \left[\frac{\partial \rho}{\partial t} + \operatorname{div}\left(\rho \mathbf{u}\right)\right] dV = 0 \qquad (2)$$

S obzirom da jednadžba (2) vrijedi za svaki volumen V_0, možemo eliminirati integral čime se dobiva jednadžba koja predstavlja diferencijalni oblik zakona o očuvanju mase:

$$\frac{\partial \rho}{\partial t} + \operatorname{div}\left(\rho \mathbf{u}\right) = 0$$

Ta jednadžba poznata je još i kao **jednadžba kontinuiteta**.

2.2. Zakon o očuvanju količine gibanja

Na volumen V_0 iz prošlog odlomka djeluju dvije vrste sila. To su sile poput tlačne koje djeluju na površinu, odnosno granicu ∂V_0 te vanjske sile poput gravitacijske koje djeluju na svaki komadić volumena.

Ukupna tlačna sila koja djeluje na površinu iznosi

$$\mathbf{F}_1 = -\int_{\partial V_0} p\mathbf{n}dA \tag{3}$$

a nakon primjene teorema o divergenciji dobiva se alternativni zapis:

$$\mathbf{F}_1 = -\int_{V_0} \text{grad}\, p dV \tag{4}$$

Iz jednadžbe (3) se vidi da je smjer tlačne sile u svakoj točki na ∂V_0 jednak kao i smjer normale. To znači da sve sile djeluju ortogonalno na površinu te samim time vrtloženje volumena V_0 nije moguće. Fluidi za koje je ovaj uvjet ispunjen nazivaju se idealnim fluidima. Njihova dinamička viskoznost jednaka je nuli. Koristimo ih u ovom razmatranju radi jednostavnosti.

Ako se uzme da je $\mathbf{a}(\mathbf{x},t)$ akceleracija uzrokovana nekom vanjskom silom, onda je ukupna vanjska sila na volumen V_0:

$$\mathbf{F}_2 = \int_{V_0} \rho\mathbf{a}dV \tag{5}$$

Kombiniranjem jednadžbi (4) i (5) može se dobiti jednadžba koja predstavlja **zakon o očuvanju količine gibanja** u diferencijalnom obliku:

$$\rho\frac{D\mathbf{u}}{Dt} = -\text{grad}\, p + \rho\mathbf{a} \tag{6}$$

Izraz $\frac{D\mathbf{u}}{Dt}$ predstavlja materijalnu derivaciju i vrijedi:

$$\frac{D\mathbf{u}}{Dt} = \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u}$$

Materijalna derivacija uzima u obzir činjenicu da čestice u fluidu tijekom vremena mijenjaju položaj. Izvod potražiti u [1]. Uvrštavanjem u jednadžbu (6) dobiva se drugi zapis zakona o očuvanju količine gibanja:

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla)\mathbf{u} - \frac{1}{\rho}\operatorname{grad} p + \mathbf{a}$$

2.3. Zakon o očuvanju energije

Ukupna energija fluida može se napisati kao zbroj kinetičke i unutarnje energije:

$$E_{uk} = E_k + E_{unutarnja}$$

Kod nestlačivih fluida unutarnja energija jednaka je nuli pa je ukupna energija isključivo kinetička. Vrijedi jednakost $\operatorname{div}\mathbf{u} = 0$. Uvrštavanjem u jednadžbu kontinuiteta i sređivanjem dobiva se $\frac{D\rho}{Dt} = 0$. Konačno, imamo sve tri **Eulerove jednadžbe** za nestlačive fluide:

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla)\mathbf{u} - \frac{1}{\rho}\nabla p + \mathbf{a}$$

$$\frac{\partial \rho}{\partial t} = -(\mathbf{u} \cdot \nabla)\rho$$

$$\nabla \cdot \mathbf{u} = 0$$

2.4. Navier-Stokesove jednadžbe

U odjeljku 2.2. spomenuto je kako se radi jednostavnosti promatraju idealni fluidi. U takvim fluidima nema molekularne viskoznosti. Međutim, u stvarnosti fluidi mogu biti jako viskozni pa je potrebno preurediti Eulerove jednadžbe da vrijede i za takve fluide. Rezultat tog preuređivanja su Navier-Stokesove jednadžbe. Izvodom iz [1] dobiva se:

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u}\cdot\nabla)\mathbf{u} - \frac{1}{\rho}\nabla p + \frac{\mu}{\rho}\nabla^2\mathbf{u} + \mathbf{a}$$

S desne strane je dodan izraz $\frac{\mu}{\rho}\nabla^2\mathbf{u}$ koji uračunava utjecaje viskoznosti. Koeficijent μ predstavlja dinamičku viskoznost. Dakle, Navier-Stokesove jednadžbe za nestlačive homogene fluide glase:

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u}\cdot\nabla)\mathbf{u} - \frac{1}{\rho}\nabla p + \frac{\mu}{\rho}\nabla^2\mathbf{u} + \mathbf{a} \tag{7}$$

$$\nabla\cdot\mathbf{u} = 0 \tag{8}$$

3. Računalni model

Simulacija toka fluida na računalu svodi se na numeričku integraciju Eulerovih, odnosno Navier-Stokesovih jednadžbi. Za potrebe ovog rada korištene su Navier-Stokesove jednadžbe. Njih je prvo potrebno zapisati u obliku koji je pogodan za implementaciju na računalu. Također je potrebno obaviti diskretizaciju prostora i vremena.

3.1. Helmholtz-Hodgeova dekompozicija

Jednadžbe (7) i (8) se mogu objediniti koristeći teorem poznat pod nazivom Helmholtz-Hodgeova dekompozicija [1]. On kaže da se svako vektorsko polje **w** može zapisati na sljedeći način

$$\mathbf{w} = \mathbf{u} + \operatorname{grad} p$$

gdje je **u** takvo vektorsko polje za koje vrijedi $\operatorname{div}\mathbf{u} = 0$, a *p* je skalarno polje. Cilj je pronaći skalarno polje *p*, takvo da kada se od **w** oduzme $\operatorname{grad} p$, dobije se vektorsko polje **u** čija divergencija je nula. Pronalaženje takvog skalarnog polja *p* dobiva se rješavanjem jednadžbe:

$$\nabla \cdot \mathbf{w} = \nabla^2 p$$

Uvodi se linearni operator **P** [1] koji vrši projekciju vektorskog polja **w** na vektorsko polje **u**, za koje vrijedi $\operatorname{div}\mathbf{u} = 0$, odnosno:

$$\mathbf{P}(\mathbf{w}) = \mathbf{w} - \nabla p = \mathbf{u}$$

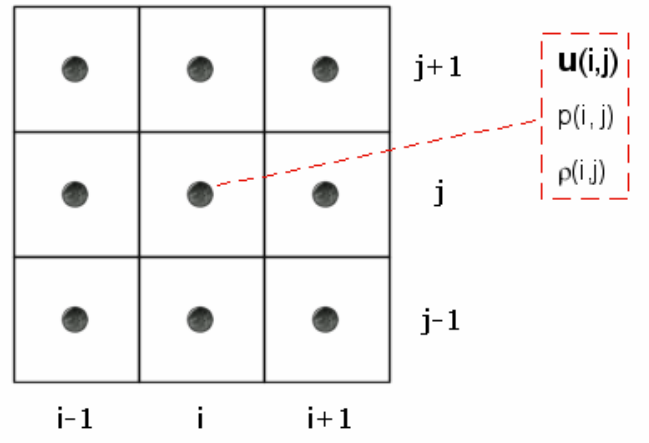
Primjenom operatora **P** na obje strane jednadžbe (7) dobiva se:

$$\frac{\partial \mathbf{u}}{\partial t} = \mathbf{P}\left(-(\mathbf{u}\cdot\nabla)\mathbf{u} + \frac{\mu}{\rho}\nabla^2\mathbf{u} + \mathbf{a} \right) \tag{9}$$

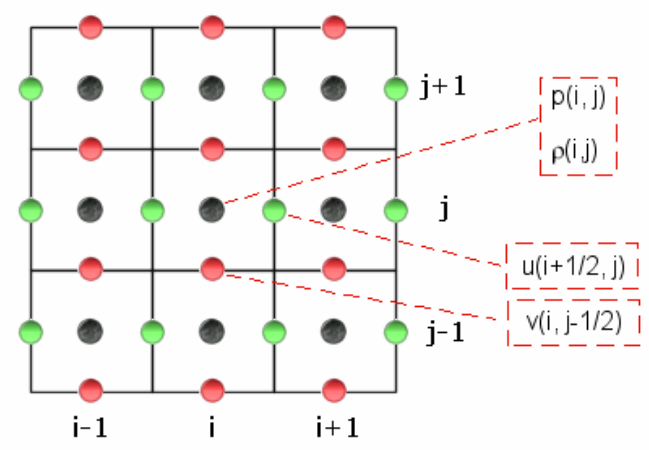
Za vektorsko polje **u** sada implicitno vrijedi $\operatorname{div}\mathbf{u} = 0$ pa se jednadžba (8) ne treba pisati. Za implementaciju u računalu koristit će se upravo ovaj oblik Navier-Stokesovih jednadžbi.

3.2. Diskretizacija prostora

Prostor na kojem se odvija simulacija dijeli se na jednake elemente volumena, tj. ćelije koje u općenitom slučaju imaju oblik kvadra. Varijable koje opisuju stanje fluida mogu se unutar ćelije pozicionirati na dva načina. Kod prvog se svih pet varijabli nalazi u središtu ćelije te je taj način podložan određenim nestabilnostima [4]. Kod drugog su u središtu samo tlak i gustoća, dok su komponente brzine pozicionirane na bridovima ćelije. Na slikama su radi jednostavnosti ćelije prikazane u dvije dimenzije.



Slika 3. Sve varijable su definirane u središtu ćelija



Slika 4. Vektorske vrijednosti su definirane na bridovima ćelija

Na ovako definiranom prostoru operator nabla poprima drugačiji oblik:

$$\nabla = \frac{\Delta}{\Delta x}\mathbf{i} + \frac{\Delta}{\Delta y}\mathbf{j} + \frac{\Delta}{\Delta z}\mathbf{k}$$

Tako će se za slučaj prikazan slikom 3 gradijent skalarnog polja *p* za ćeliju (i,j,k) računati na sljedeći način

$$(\nabla p)_{i,j,k} = \frac{p_{i+1,j,k} - p_{i-1,j,k}}{2xx}\mathbf{i} + \frac{p_{i,j+1,k} - p_{i,j-1,k}}{2yy}\mathbf{j} + \frac{p_{i,j,k+1} - p_{i,j,k-1}}{2zz}\mathbf{k}$$

gdje *xx*, *yy* i *zz* predstavljaju udaljenosti između dvije susjedne varijable *p* u smjeru **i**, **j** odnosno **k**. Slično vrijedi i za divergenciju vektorskog polja **u**:

$$(\nabla \cdot \mathbf{u})_{i,j,k} = \frac{u_{i+1,j,k} - u_{i-1,j,k}}{2xx} + \frac{v_{i,j+1,k} - v_{i,j-1,k}}{2yy} + \frac{w_{i,j,k+1} - w_{i,j,k-1}}{2zz}$$

S druge strane, za slučaj prikazan slikom 4 će se divergencija vektorskog polja **u** za ćeliju (i,j,k) računati ovako:

$$(\nabla \cdot \mathbf{u})_{i,j,k} = \frac{u_{i+\frac{1}{2},j,k} - u_{i-\frac{1}{2},j,k}}{xx} + \frac{v_{i,j+\frac{1}{2},k} - v_{i,j-\frac{1}{2},k}}{yy} + \frac{w_{i,j,k+\frac{1}{2}} - w_{i,j,k-\frac{1}{2}}}{zz}$$

3.3. Diskretizacija vremena

Umjesto da vrijeme protiče kontinuirano, ono je podijeljeno na vremenske korake Δt. To znači da se stanje fluida izračunava za trenutke koji su udaljeni od početnog za neki višekratnik od Δt. Preuređivanjem jednadžbe (9) dobiva se

$$\frac{\mathbf{u}^* - \mathbf{u}}{\Delta t} = \mathbf{P}\left(-(\mathbf{u} \cdot \nabla)\mathbf{u} + \frac{\mu}{\rho}\nabla^2 \mathbf{u} + \mathbf{a} \right) \tag{10}$$

gdje **u** predstavlja vektor brzine u trenutku t, a **u*** u trenutku $t+\Delta t$.

3.4. Rješavanje Navier-Stokesovih jednadžbi

Nakon što su obavljene potrebne predradnje, može se pristupiti rješavanju Navier-Stokesovih jednadžbi. Ideja je rastaviti desnu stranu jednadžbe (10) na zasebne izraze te za svaki izraz naći najprikladniju metodu rješavanja [2]. Neka je $\mathbf{u_0}$ brzina dobivena u trenutku t, a $\mathbf{u_1}$, $\mathbf{u_2}$ i $\mathbf{u_3}$ brzine dobivene u međukoracima. Za dobivanje brzine $\mathbf{u_4}$ u trenutku $t+\Delta t$ prvo je potrebno dodati vanjsku silu:

$$\mathbf{u}_1 = \mathbf{u}_0 + \Delta t \mathbf{a}$$

To je ujedno i najjednostavniji međukorak. Zatim je potrebno riješiti jednadžbu koja opisuje kako putuje brzina "nošena sama sobom":

$$\frac{\mathbf{u}_2 - \mathbf{u}_1}{\Delta t} = -(\mathbf{u}_1 \cdot \nabla)\mathbf{u}_1$$

To može biti prilično kompliciranо klasičnim računskim metodama. Međutim, Stam predlaže implicitnu metodu koja je bezuvjetno stabilna, a sastoji se od kombiniranja čestica i brzina. Detalje potražiti u [2]. Nakon toga na red dolazi jednadžba koja rješava utjecaje viskoznosti:

$$\frac{\mathbf{u}_3 - \mathbf{u}_2}{\Delta t} = \frac{\mu}{\rho}\nabla^2 \mathbf{u}_2$$

Moguće metode rješavanja potražiti u [2]. Konačno, posljednji korak je projekcija, odnosno primjena operatora **P**. Ona se svodi na rješavanje jednadžbe:

$$\nabla \cdot \mathbf{u}_3 = \nabla^2 p \tag{11}$$

Nakon što je skalarno polje *p* izračunato, potrebno je oduzeti njegov gradijent od vektorskog polja $\mathbf{u_3}$:

$$\mathbf{u}_4 = \mathbf{u}_3 - \nabla p$$

Dobiveno vektorsko polje $\mathbf{u_4}$ predstavlja brzinu u trenutku $t+\Delta t$. Postupak se dalje ponavlja za vremenske trenutke $t+2\Delta t$, $t+3\Delta t$ itd.

3.5. Gibanje supstanci kroz fluid

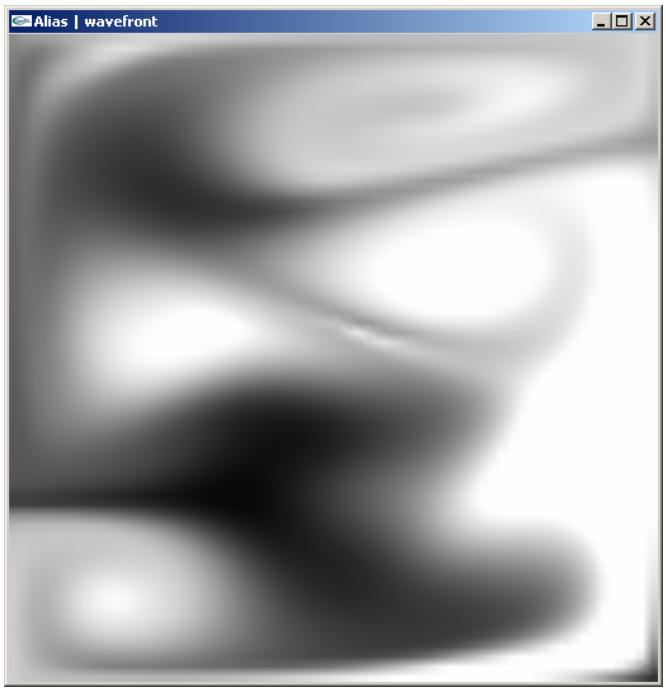
Kada se u fluid ubaci neka nereaktivna supstanca, kao npr. tinta u vodu, ona se počne gibati vođena brzinom fluida. Također dolazi i do difuzije. Općenito, ako je ρ neka skalarna veličina koja se giba fluidom, onda sljedeća jednadžba dobro opisuje to gibanje:

$$\frac{\partial \rho}{\partial t} = -\left(\mathbf{u}\cdot\nabla\right)\rho + \kappa_\rho \nabla^2 \rho - \alpha_\rho \rho + S_\rho \tag{12}$$

κ_ρ je difuzijska konstanta, α_ρ brzina disipacije, a S_ρ gustoća koja dolazi iz izvora. Jednadžba je jako slična Navier-Stokesovoj pa se svi izrazi na desnoj strani, osim $-\alpha_\rho\rho$ mogu riješiti na isti način kao što se to radilo s brzinom. Za detalje pogledati [2].

3.6. Prikaz rezultata

Za prikaz rezultata koji se mogu dobiti ovom metodom korišten je program koji je načinio Jos Stam u sklopu rada "Stable Fluids" [2]. Program omogućuje korisniku da upotrebom miša primjenjuje silu na fluid preko izraza **a** s desne strane jednadžbe (9). Upotrebom miša je također omogućeno dodavanje novih količina dima u fluid i to preko izraza S_ρ s desne strane jednadžbe (12).



Slika 5. Primjer 1

Slika 6. Primjer 2

Slika 7. Primjer 3

4. Simulacije vođene konačnim stanjem

Koristeći metodu opisanu u prošlom poglavlju mogu se dobiti uvjerljive simulacije gibanja fluida i supstanci u njemu (npr. dim u zraku). Međutim, teško je ili gotovo nemoguće postići da taj dim dobije željeni oblik (npr. da korisnik određivanjem iznosa vanjskih sila natjera dim u oblik kvadrata). Zbog toga su od nedavno razvijeni postupci koji omogućavaju da se zada željeni oblik, a sile se onda tijekom simulacije izračunavaju tako da dim zaista poprimi taj željeni oblik. Ti su postupci od iznimne važnosti za animatore u filmovima, jer im omogućuju stvaranje uvjerljivih specijalnih efekata kakve prije nije bilo moguće vidjeti.

U ovom radu fokus je stavljen na metodu razvijenu od strane dvojice profesora s The Hebrew University of Jerusalem [3]. Oni u Navier-Stokesove jednadžbe uvode dva nova izraza. Prvi je izraz za silu koja vodi dim prema željenom obliku i nalazi se u jednadžbi koja predstavlja zakon o očuvanju količine gibanja. Drugi je izraz za sprječavanje difuzije dima zbog numeričke disipacije, a nalazi se u jednadžbi koja opisuje gibanje supstanci u fluidu.

Neka je $\rho(\mathbf{x},t)$ skalarno polje koje označava gustoću dima u vremenskom trenutku *t*, a neka je $\rho^*(\mathbf{x})$ skalarno polje koje predstavlja željenu gustoću dima. U svakom trenutku tijekom simulacije dim se mora kretati tako da poprimi oblik zadan s ρ^*, a jednom kada u tome uspije, treba se u tom stanju zadržati. Navier-Stokesove jednadžbe, prepravljene da ispunjavaju zadani uvjet, sljedećeg su oblika:

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u}\cdot\nabla)\mathbf{u} - \nabla p + v_f \mathbf{F}(\rho,\rho^*) - v_d\mathbf{u} \qquad (13)$$

$$\nabla\cdot\mathbf{u} = 0$$

Izraz $\mathbf{F}(\rho,\rho^*)$ predstavlja silu koja vodi dim prema željenom obliku. Točna definicija tog izraza dana je niže u tekstu. Također je dodan i izraz $-v_d\mathbf{u}$ koji

služi kao prigušivač količine gibanja. Osim ovih jednadžbi, prepravljena je i jednadžba (12) te sada ima oblik:

$$\frac{\partial \rho}{\partial t} = -(\mathbf{u} \cdot \nabla)\rho + v_g \mathbf{G}(\rho, \rho^*)$$

Na desnoj strani je dodan izraz $\mathbf{G}(\rho, \rho^*)$ koji sprječava difuziju dima. Točna definicija navedena je niže u tekstu. Utjecaj svakog od tri dodana izraza određuje se pomoću odgovarajućih nenegativnih parametara v_f, v_d i v_g.

4.1. Izraz za silu - F(ρ,ρ)*

Već je spomenuto kako je namjena ove sile da "natjera" dim da poprimi oblik zadan s ρ^*, a kada u tome uspije trebala bi iščeznuti tako da fluid dođe u stanje mirovanja $\mathbf{u} = \frac{\partial \mathbf{u}}{\partial t} = 0$. U tom stanju jednadžba (13) poprima sljedeći oblik:

$$v_f \mathbf{F}(\rho^*, \rho^*) = \nabla p$$

To znači da sila treba biti gradijent nekog potencijalnog polja kako bi se omogućilo da ju hidrostatski tlak poništi u trenutku kada gustoća dima poprimi željeno stanje. Tada će fluid biti u stanju mirovanja.

Vektorsko polje $\nabla \rho^*$ u svakoj točki ima smjer prema višim koncentracijama ρ^*, a upravo to je smjer koji želimo. Zbog toga silu usmjeravamo tamo gdje pokazuje $\nabla \rho^*$, odnosno:

$$\mathbf{F}(\rho, \rho^*) \propto \nabla \rho^*$$

Problem nastaje na mjestima gdje je $\rho*$ konstantna pa vrijedi $\nabla\rho* = 0$, a samim time i $\mathbf{F}(\rho, \rho*) = 0$. Da bi se doskočilo ovome, konvoluira se $\rho*$ s Gaussovom jezgrom

$$g(\mathbf{x}) = e^{-\frac{\mathbf{x}^2}{\sigma^2}}$$

te se dobiva zaglađena verzija gustoće dima koju označavamo s $\tilde{\rho}*$. Za nju u svakoj točki vrijedi $\nabla\tilde{\rho}* \neq 0$. Sada možemo napisati konačan izraz za vanjsku silu [3], a on glasi:

$$\mathbf{F}(\rho, \rho*) = \tilde{\rho}\frac{\nabla\tilde{\rho}*}{\tilde{\rho}*} \tag{14}$$

U trenutku kada dim postigne željeno stanje, ta jednadžba će se pretvoriti u

$$\mathbf{F}(\rho*, \rho*) = \tilde{\rho}*\frac{\nabla\tilde{\rho}*}{\tilde{\rho}*} = \nabla\tilde{\rho}*$$

i time je zadovoljen uvjet u stanju mirovanja, a to je da je $\mathbf{F}(\rho, \rho*)$ gradijent potencijalnog polja.

4.2. Izraz za sprječavanje difuzije dima - G(ρ,ρ)*

S obzirom da zbog numeričke disipacije i drugih razloga [3] u općenitom slučaju nije moguće postići da dim poprimi željeno stanje, uvodi se izraz za sprječavanje difuzije. Neka je na sljedeći način predstavljeno odstupanje trenutnog stanja dima od konačnog:

$$e(\mathbf{x}, t) = \rho(\mathbf{x}, t) - \rho*(\mathbf{x})$$

Potrebno je zadovoljiti sljedeću jednadžbu:

$$\frac{\partial e}{\partial t} = \nabla^2 e \tag{15}$$

uz von Neumannove granične uvjete $\frac{\partial e}{\partial \mathbf{n}} = 0$. Proticanjem vremena odstupanje *e* postaje nula na cijeloj domeni te vrijedi:

$$e = \rho - \rho^* = 0$$
$$\nabla e = \nabla\rho - \nabla\rho^* = 0$$

Uvrštavanjem dobivenog u jednadžbu (15) slijedi:

$$\frac{\partial \rho}{\partial t} = \nabla^2 \left(\rho - \rho^*\right)$$

te se može pisati:

$$\mathbf{G}\left(\rho, \rho^*\right) = \nabla^2 \left(\rho - \rho^*\right)$$

Međutim, dobiveni izraz se preuređuje tako da vrijedi samo u blizini ciljnog stanja i tamo gdje uopće ima dima. Postupkom iz [3] dobiva se:

$$\mathbf{G}\left(\rho, \rho^*\right) = \nabla \cdot \left[\rho \tilde{\rho}^* \nabla\left(\rho - \rho^*\right)\right]$$

4.3. Parametri simulacije

Ova metoda simulacije toka fluida upravljiva je pomoću četiri parametra. To su parametar zaglađenja sile σ, koeficijent pokretačke sile v_f, koeficijent prigušenja v_d te koeficijent prikupljanja dima v_g.

σ - Što je koeficijent veći, sila je glađa te dim bez rasipanja putuje prema cilju, ali teže postiže oblik zadan s ρ^*.

v_f - Služi za pojačanje ili oslabljenje vanjske sile. Što je koeficijent veći, simulacija brže napreduje prema cilju.

v_d - Povećanjem ovog koeficijenta smanjujemo mogućnost da dim zbog brzine "preskoči" ciljno stanje.

v_g - Ovaj koeficijent određuje brzinu kojom se dim skuplja prema ρ^*. Ako se postavi previsoko, simulacija može izgledati nerealno jer ovaj dio nije fizikalno utemeljen.

4.4. Rješavanje prepravljenih Navier-Stokesovih jednadžbi

Postupak rješavanja Navier-Stokesovih jednadžbi, prepravljenih da omogućuju vođenje simulacije k željenom konačnom stanju, vrlo je sličan postupku opisanom u odjeljku 3.4. Ako uzmemo da je $\mathbf{u_0}$ brzina dobivena u trenutku *t*, a $\mathbf{u_1}$, $\mathbf{u_2}$ i $\mathbf{u_3}$ brzine dobivene u međukoracima, onda je za dobivanje brzine $\mathbf{u_4}$ u trenutku $t+\Delta t$ prvo potrebno dodati vanjsku silu

$$\mathbf{u}_1 = \mathbf{u}_0 + v_f \mathbf{F}(\rho, \rho^*)\Delta t$$

nakon čega slijedi prigušivanje nastalog momenta:

$$\mathbf{u}_2 = \mathbf{u}_1 - v_d \mathbf{u}_1 \Delta t$$

Sljedeća dva koraka jednaka su kao u odjeljku 3.4. pa se mogu rješavati istim postupcima.

$$\frac{\mathbf{u}_3 - \mathbf{u}_2}{\Delta t} = -(\mathbf{u}_2 \cdot \nabla)\mathbf{u}_2$$

$$\mathbf{u}_4 = \mathbf{P}(\mathbf{u}_3)$$

Postupak se dalje ponavlja za vremenske trenutke $t+2\Delta t$, $t+3\Delta t$ itd.

Jednadžba koja opisuje gibanje supstanci unutar fluida slična je Navier-Stokesovoj jednadžbi za očuvanje količine gibanja te se za nju mogu upotrijebiti isti postupci. Neka je ρ_0 gustoća dima u trenutku t, a ρ_1 međurezultat. Za dobivanje gustoće dima ρ_2 u trenutku $t+\Delta t$ potrebno je izvršiti sljedeće korake:

$$\frac{\rho_1 - \rho_0}{\Delta t} = -\left(\mathbf{u}_4 \cdot \nabla\right)\rho_0$$

$$\rho_2 = \rho_1 + v_g \mathbf{G}(\rho, \rho^*)\Delta t$$

Brzina $\mathbf{u_4}$ je brzina u trenutku $t+\Delta t$ dobivena na već opisan način. Postupak se dalje ponavlja za vremenske trenutke $t+2\Delta t$, $t+3\Delta t$ itd.

5. Programsko rješenje

Za potrebe ovog rada implementiran je postupak opisan u poglavlju 4. i to u dvodimenzionalnom prostoru. Kao baza za implementaciju poslužio je izvorni kod Stamove aplikacije čiji su rezultati prikazani u odjeljku 3.6. Korišten je programski jezik C++.

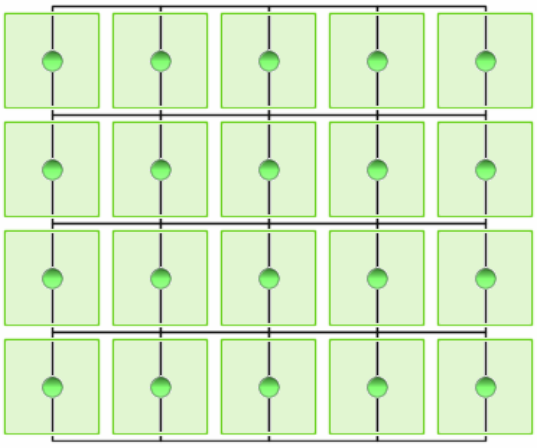
5.1. Strukture podataka i algoritmi

Definiran je razred `TDSmokeSolver` koji implementira navedeni postupak. Stanje fluida je zbog dvodimenzionalnog prostora opisano s četiri članske varijable, a to su `u`, `v`, `r`, `p`, odnosno brzina u smjeru osi x, brzina u smjeru osi y, gustoća dima i tlak. Željeno stanje dima predstavljeno je varijablom `rs`. Gustoća dima konvoluirana s Gaussovom jezgrom predstavljena je varijablama `rt` i `rts`. Sve navedene varijable su tipa `std::vector <double>`. Parametri simulacije označeni su varijablama `vf`, `vd`, `vg` i `sigma` koje su tipa `double`. Ostale varijable su većinom pomoćne. Ključna je metoda `void MakeStep ( bool fst_order )` koja računa stanje fluida za sljedeći vremenski trenutak $t+\Delta t$. Zaglavlje cijelog razreda izgleda ovako:

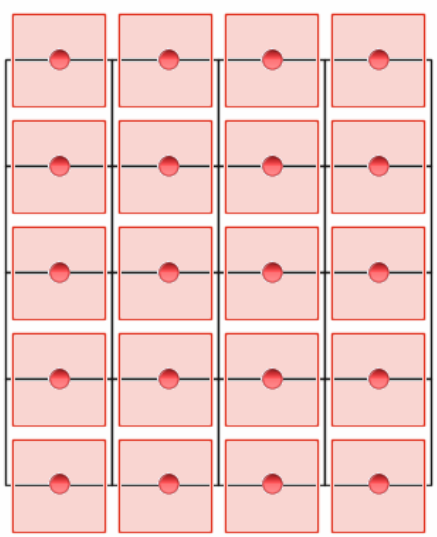
```
class TDSmokeSolver
{
protected:
    std::vector <double> u, v, r, p; // stanje fluida
    std::vector <double> rs;  // zeljeno stanje dima
    std::vector <double> rt, rts; // zagladjene verzije dima
    double vf, vd, vg, sigma; // parametri simulacije

    int N;
    double dt, delta;
    std::vector <double> uv_temp, u_approx, v_approx;
    std::vector <double> r_gather, r_gather2;
    std::vector <double> div, kernel, r_temp;

    double mml ( double Q1, double Q2, double Q3 );
    double hyperbolicSolveU (
                std::vector <double> &x, double s1, double s2,
                int i, int j, bool b, bool fst_order );
    double hyperbolicSolveV (
                std::vector <double> &x, double s1, double s2,
                int i, int j, bool fst_order );
```

```
    void gaussianSmooth ( bool b );
    void applyDrivingForce ( );
    void attenuateMomentum ( );
    void advectMomentum ( bool fst_order );
    void project ( );
    void advectSmoke ( bool fst_order );
    void gatherSmoke ( );

public:
    TDSmokeSolver ( int n, double v_f, double v_d, double v_g,
            double sgm, double d_t, std::vector <double> &r1,
            std::vector <double> &r2 );
    ~TDSmokeSolver ( );

    void MakeStep ( bool fst_order );
    std::vector <double> GetSmoke ( ) { return r; };
    void GetVelocity (
            std::vector <double> &ux, std::vector <double> &uy );
};
```

Korištene su ćelije s rasporedom varijabli prikazanim na slici 4 kako bi se izbjegle nestabilnosti uzrokovane osnovnim rasporedom. Vidi se da je broj crnih krugova manji od broja zelenih, odnosno crvenih krugova. To znači da je za veličine koje se nalaze na bridovima ćelija potrebno više varijabli nego za one veličine koje se nalaze u središtu ćelije. Zbog toga ranije spomenuti vektori `u` i `v` imaju više elemenata od vektora `r` i `p`. Ćelije oko elemenata na bridovima možemo ilustrirati na sljedeći način:



Slika 8. Prikaz ćelija oko komponente brzine *u*



Slika 9. Prikaz ćelija oko komponente brzine *v*

Kao što je već rečeno, metoda `MakeStep` izračunava stanje fluida u sljedećem vremenskom trenutku. To čini u koracima opisanim u odjeljku 4.4. Svaki korak predstavlja po jedna od sljedećih metoda sa zaštićenim pravom pristupa:

```
void applyDrivingForce ( );
void attenuateMomentum ( );
void advectMomentum ( bool fst_order );
void project ( );
void advectSmoke ( bool fst_order );
void gatherSmoke ( );
```

Niže u tekstu opisana je implementacija svake od metoda, ali za početak je potrebno definirati neke diskretne operatore kako bi se pojednostavio zapis [3]. Tako ćemo prvu derivaciju brzine u smjeru osi x označavati na sljedeći način:

$$D_x(u)_{i,j} = \frac{u_{i+\frac{1}{2},j} - u_{i-\frac{1}{2},j}}{\Delta x} \tag{16}$$

Ona se aproksimira na pola puta između varijabli $u_{i+\frac{1}{2},j}$ i $u_{i-\frac{1}{2},j}$. Druga derivacija brzine aproksimira se na mjestima gdje su definirane varijable, odnosno:

$$D_{xx}(u)_{i+\frac{1}{2},j} = \frac{D_x(u)_{i+1,j} - D_x(u)_{i,j}}{\Delta x} = \frac{u_{i+\frac{3}{2},j} - 2u_{i+\frac{1}{2},j} + u_{i-\frac{1}{2},j}}{(\Delta x)^2} \tag{17}$$

Prva derivacija gustoće dima aproksimira se na pola puta između varijabli $\rho_{i,j}$ i $\rho_{i,j+1}$ glasi:

$$D_y(\rho)_{i,j+\frac{1}{2}} = \frac{\rho_{i,j+1} - \rho_{i,j}}{\Delta y}$$

Ako nam zatreba bilo koja varijabla i to na mjestu koje se nalazi na pola puta između mjesta gdje je željena varijabla definirana, onda ćemo za dobivanje tražene vrijednosti koristiti aritmetičku sredinu:

$$A_y(\rho)_{i,j+\frac{1}{2}} = \frac{\rho_{i,j+1} + \rho_{i,j}}{2}$$

Sada možemo opisati način implementacije svakog koraka algoritma.

5.1.1. Primjena vodeće sile

Prva metoda izračunava vodeću silu $\mathbf{F}(\rho, \rho^*)$ te ju primjenjuje na fluid. Sila je definirana na bridovima ćelija, odnosno na istim mjestima gdje je definirana i brzina. Jednadžba (14) poprima sljedeći oblik:

$$F^u_{i+\frac{1}{2},j} = A\left(\tilde{\rho}\right)_{i+\frac{1}{2},j} \frac{D_x\left(\tilde{\rho}^*\right)_{i+\frac{1}{2},j}}{A\left(\tilde{\rho}^*\right)_{i+\frac{1}{2},j}}$$

$$F^{v}_{i,j+\frac{1}{2}} = A\left(\tilde{\rho}\right)_{i,j+\frac{1}{2}} \frac{D_y\left(\tilde{\rho}*\right)_{i,j+\frac{1}{2}}}{A\left(\tilde{\rho}*\right)_{i,j+\frac{1}{2}}}$$

Za svaku ćeliju sa slika 8 i 9 je potrebno izračunati odgovarajuću silu F^u, odnosno F^v te ju primijeniti na sljedeći način:

$$u^{(1)}_{i+\frac{1}{2},j} = u^{(0)}_{i+\frac{1}{2},j} + v_f F^u_{i+\frac{1}{2},j} \Delta t$$

$$v^{(1)}_{i,j+\frac{1}{2}} = v^{(0)}_{i,j+\frac{1}{2}} + v_f F^v_{i,j+\frac{1}{2}} \Delta t$$

Slijedi zapis u pseudokodu:

```
// horizontalna komponenta - u
za ( svaku_celiju u[i,j] )
{
    D_rts = ( rts[i,j] - rts[i-1,j] ) / delta_x;
    A_rt  = ( rt [i,j] + rt [i-1,j] ) / 2.;
    A_rts = ( rts[i,j] + rts[i-1,j] ) / 2.;
    Fu    = A_rt * D_rts / A_rts;

    u[i,j] += dt * vf * Fu;
}

// vertikalna komponenta - v
za ( svaku_celiju v[i,j] )
{
    D_rts = ( rts[i,j] - rts[i,j-1] ) / delta_y;
    A_rt  = ( rt [i,j] + rt [i,j-1] ) / 2.;
    A_rts = ( rts[i,j] + rts[i,j-1] ) / 2.;
    Fv    = A_rt * D_rts / A_rts;

    v[i,j] += dt * vf * Fv;
}
```

5.1.2. Prigušenje količine gibanja

Druga metoda prigušuje količinu gibanja i predstavlja najjednostavniji korak u implementaciji. Za svaku ćeliju sa slika 8 i 9 potrebno je primijeniti odgovarajuće prigušenje:

$$u^{(2)}_{i+\frac{1}{2},j} = u^{(1)}_{i+\frac{1}{2},j} - v_d u^{(1)}_{i+\frac{1}{2},j} \Delta t$$

$$v^{(2)}_{i,j+\frac{1}{2}} = v^{(1)}_{i,j+\frac{1}{2}} + v_d v^{(1)}_{i,j+\frac{1}{2}} \Delta t$$

Slijedi zapis u pseudokodu:

```
// horizontalna komponenta - u
za ( svaku_celiju u[i,j] )
    u[i,j] -= dt * vd * u[i,j];

// vertikalna komponenta - v
za ( svaku_celiju v[i,j] )
    v[i,j] -= dt * vd * v[i,j];
```

5.1.3. Vođenje brzine

Treća metoda računa utjecaje izraza $-(\mathbf{u}\cdot\nabla)\mathbf{u}$. To je moguće napraviti na implicitni način kako je opisano u [2] te tako osigurati bezuvjetnu stabilnost postupka bez obzira na odabir vremenskog koraka Δt, ali autori ove metode predlažu hiperbolični rješavač drugog reda čija stabilnost ovisi o Δt. Izraz $-(\mathbf{u}\cdot\nabla)\mathbf{u}$ se rješava nezavisno za svaku os, odnosno u našem slučaju za komponente brzine u i v. To se manifestira pozivom dvije pomoćne metode:

```
double hyperbolicSolveU (
            std::vector <double> &x, double s1, double s2,
            int i, int j, bool b, bool fst_order );
double hyperbolicSolveV (
            std::vector <double> &x, double s1, double s2,
            int i, int j, bool fst_order );
```

Slijedi pseudokod:

```
// aproksimiraj brzine u sredistima celija
za ( svaku_celiju [i,j] )
{
    u_approx[i,j] = ( u[i,j] + u[i+1,j] ) / 2.;
    v_approx[i,j] = ( v[i,j] + v[i,j+1] ) / 2.;
}

// horizontalna komponenta - u
```

```
za ( svaku_celiju u[i,j] )
    u[i,j] -= dt * hyperbolicSolveU ( ... ) / delta;

// vertikalna komponenta - v
za ( svaku_celiju v[i,j] )
    v[i,j] -= dt * hyperbolicSolveV ( ... ) / delta;
```

Za detalje o postupku pogledati u [3].

5.1.4. Projekcija

Četvrta metoda vrši projekciju vektorskog polja dobivenog u prethodnom koraku. Jednadžba (11) se pretvara u sljedeći zapis:

$$D_x\left(u^{(3)}\right)_{i,j} + D_y\left(v^{(3)}\right)_{i,j} = D_{xx}\left(p\right)_{i,j} + D_{yy}\left(p\right)_{i,j}$$

Primjenjujući jednadžbe (16) i (17) dobiva se:

$$\frac{u^{(3)}_{i+\frac{1}{2},j} - u^{(3)}_{i-\frac{1}{2},j}}{\Delta x} + \frac{v^{(3)}_{i,j+\frac{1}{2}} - v^{(3)}_{i,j-\frac{1}{2}}}{\Delta y} = \frac{p_{i+1,j} - 2p_{i,j} + p_{i-1,j}}{\left(\Delta x\right)^2} + \frac{p_{i,j+1} - 2p_{i,j} + p_{i,j-1}}{\left(\Delta y\right)^2}$$

S obzirom da u našem slučaju vrijedi jednakost $\Delta x = \Delta y$, lako se dobiva:

$$p_{i,j} = \frac{1}{4}\left(p_{i+1,j} + p_{i-1,j} + p_{i,j+1} + p_{i,j-1} - \left(u^{(3)}_{i+\frac{1}{2},j} - u^{(3)}_{i-\frac{1}{2},j} + v^{(3)}_{i,j+\frac{1}{2}} - v^{(3)}_{i,j-\frac{1}{2}} \right)\Delta \right)$$

gdje Δ predstavlja Δx, odnosno Δy. Dakle, skalarno polje *p* se za svaku ćeliju računa na upravo opisan način. Razlike se pojavljuju jedino kod rubnih ćelija. Tako se npr. za ćelije koje se nalaze na lijevoj granici vrijednost tlaka računa:

$$p_{0,j} = \frac{1}{3}\left(p_{i+1,j} + p_{i,j+1} + p_{i,j-1} - \left(u^{(3)}_{i+\frac{1}{2},j} - u^{(3)}_{i-\frac{1}{2},j} + v^{(3)}_{i,j+\frac{1}{2}} - v^{(3)}_{i,j-\frac{1}{2}} \right)\Delta \right)$$

Poseban slučaj su još i četiri ćelije koje se nalaze na gornjoj lijevoj, gornjoj desnoj, donjoj lijevoj te donjoj desnoj granici. Za npr. donju lijevu ćeliju vrijednost tlaka iznosi:

$$p_{0,0} = \frac{1}{2}\left(p_{i+1,j} + p_{i,j+1} - \left(u^{(3)}_{i+\frac{1}{2},j} - u^{(3)}_{i-\frac{1}{2},j} + v^{(3)}_{i,j+\frac{1}{2}} - v^{(3)}_{i,j-\frac{1}{2}} \right) \Delta \right)$$

Nakon što je *p* izračunat, potrebno je od brzine oduzeti njegov gradijent. Zapis u pseudokodu izgleda ovako:

```
// izracunaj izraz koji se pojavljuje u sva 3 slucaja
za ( svaku_celiju div[i,j] )
    div[i,j] = ( u[i+1,j] - u[i,j] + v[i,j+1] - v[i,j] ) * delta;

// izracunaj tlak
za ( k=0; k<20; ++k )
    za  ( svaku_celiju p[i,j] )
    {
        // cetiri rubne celije
        ako ( donja_lijeva_granica )
            p[i,j] = ( p[i+1,j] + p[i,j+1] - div[i,j] ) / 2.;
        inace ako ( gornja_lijeva_granica )
            p[i,j] = ( p[i+1,j] + p[i,j-1] - div[i,j] ) / 2.;
        inace ako ( donja_desna_granica )
            p[i,j] = ( p[i-1,j] + p[i,j+1] - div[i,j] ) / 2.;
        inace ako ( gornja_desna_granica )
            p[i,j] = ( p[i-1,j] + p[i,j-1] - div[i,j] ) / 2.;

        // cetiri granice
        inace ako ( lijeva_granica )
          p[i,j] = (p[i+1,j] + p[i,j+1] + p[i,j-1] - div[i,j]) / 3.;
        inace ako ( donja_granica )
          p[i,j] = (p[i+1,j] + p[i-1,j] + p[i,j+1] - div[i,j]) / 3.;
        inace ako ( desna_granica )
          p[i,j] = (p[i-1,j] + p[i,j+1] + p[i,j-1] - div[i,j]) / 3.;
        inace ako ( gornja_granica )
          p[i,j] = (p[i+1,j] + p[i-1,j] + p[i,j-1] - div[i,j]) / 3.;

        // sve unutarnje celije
        inace p[i,j] = ( p[i+1,j] + p[i-1,j] + p[i,j+1] + p[i,j-1] -
                                                    div[i,j] ) / 4.;
    }

// oduzmi gradijent tlaka od brzine
// horizontalna komponenta - u
za ( svaku_celiju u[i,j] )
    u[i,j] -= ( p[i,j] - p[i-1,j] ) / delta;

// vertikalna komponenta - v
za ( svaku_celiju v[i,j] )
```

```
    v[i,j] -= ( p[i,j] - p[i,j-1] ) / delta;
```

5.1.5. Vođenje dima

Peta metoda rješava utjecaj izraza $-(\mathbf{u}\cdot\nabla)\rho$. U tu svrhu se koristi isti postupak kao i u odjeljku 5.1.3. Izraz se rješava neovisno po osima, a pseudokod izgleda ovako:

```
za ( svaku_celiju r[i,j] )
{
    r[i,j] -= dt * hyperbolicSolveU ( ... ) / delta +
              dt * hyperbolicSolveV ( ... ) / delta;
}
```

5.1.6. Sprječavanje difuzije dima

Šesta metoda rješava utjecaj izraza $\nabla\cdot\left[\rho\tilde{\rho}^{*}\nabla\left(\rho-\rho^{*}\right)\right]$. Kao u odjeljcima 5.1.3., 5.1.5. i ovdje se koristi hiperbolični rješavač i to u dva koraka. U prvom koraku se za svaku ćeliju izračunava sljedeća vrijednost te se pohranjuje u privremenu varijablu ρ_g

$$\rho_g = \nabla\left(\rho-\rho^{*}\right)$$

nakon čega se u drugom koraku dobiva konačan rezultat:

$$\nabla\cdot\left[\rho\tilde{\rho}^{*}\rho_g\right]$$

Detalje o ovoj metodi moguće je pronaći u [3] te pregledom izvornog koda programa.

5.2. Iscrtavanje na ekran

Iscrtavanje rezultata na ekran se obavlja pomoću dijelova koda preuzetih iz programa koji je načinio Jos Stam. Program koristi GLUT sučelje čime se sve maksimalno pojednostavljuje. Ukratko će biti opisane dvije najvažnije funkcije korištene u ovom dijelu programa, a to su:

```
void idle_func ( );
void display_func ( );
```

Prva funkcija predstavlja tzv. *callback idle* funkciju. Ona se poziva od strane GLUT-a uvijek kada prozor nema drugog posla, a upravo je to pogodno vrijeme za izračunavanje stanja fluida za sljedeći vremenski trenutak. Uz pretpostavku da su `td` i `r` globalne varijable tipa `TDSmokeSolver*`, odnosno `std::vector <double>`, implementacija te funkcije je sljedeća:

```
void idle_func ( )
{
    if ( simulating )
    {
        td->MakeStep ( false ); // izracunaj stanje fluida
        r = td->GetSmoke ( ); // spremi trenutno stanje dima
    }
}
```

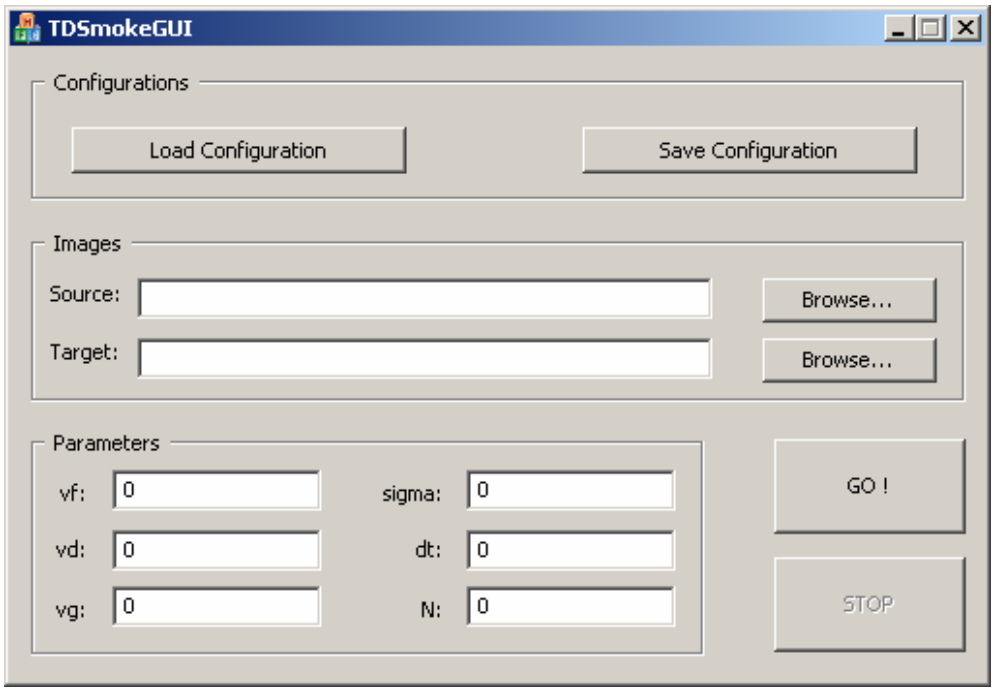
Druga funkcija predstavlja tzv. *callback display* funkciju. Ona se poziva od strane GLUT-a svaki put kada je u prozor potrebno nešto nacrtati. Njena implementacija izgleda ovako:

```
void display_func ( )
{
    pre_display ( ); // posao prije crtanja
    draw_density ( ); // crtanje dima
    post_display ( ); // posao nakon crtanja
}
```

Funkcija `void draw_density ( );` uzima ranije dobivene vrijednosti iz varijable `r` te ih crta na ekran. Pri tome se vrijednost svake ćelije interpolira s tri susjedne ćelije, a iscrtava se rezultat. Za detalje pogledati u izvorni kod.

5.3. Grafičko korisničko sučelje

Za izradu grafičkog korisničkog sučelja korišten je MFC (Microsoft Foundation Classes). Njime se omogućuje relativno jednostavna izrada prozora s uobičajenim Windows kontrolama poput polja za unos teksta, dijaloga za učitavanje i spremanje datoteka itd. Grafičko korisničko sučelje aplikacije izrađene u okviru ovog rada sastoji se od jednog prozora koji izgleda ovako:



Slika 10. Grafičko korisničko sučelje

Prozor je podijeljen na četiri logičke cjeline. To su konfiguracijska cjelina, cjelina za učitavanje slika, cjelina za određivanje iznosa parametara te upravljačka cjelina. Sve cjeline osim upravljačke uokvirene su uz odgovarajući natpis (*eng. Configurations, Images, Parameters*). Slijedi opis svake od njih.

5.3.1. Konfiguracijska cjelina

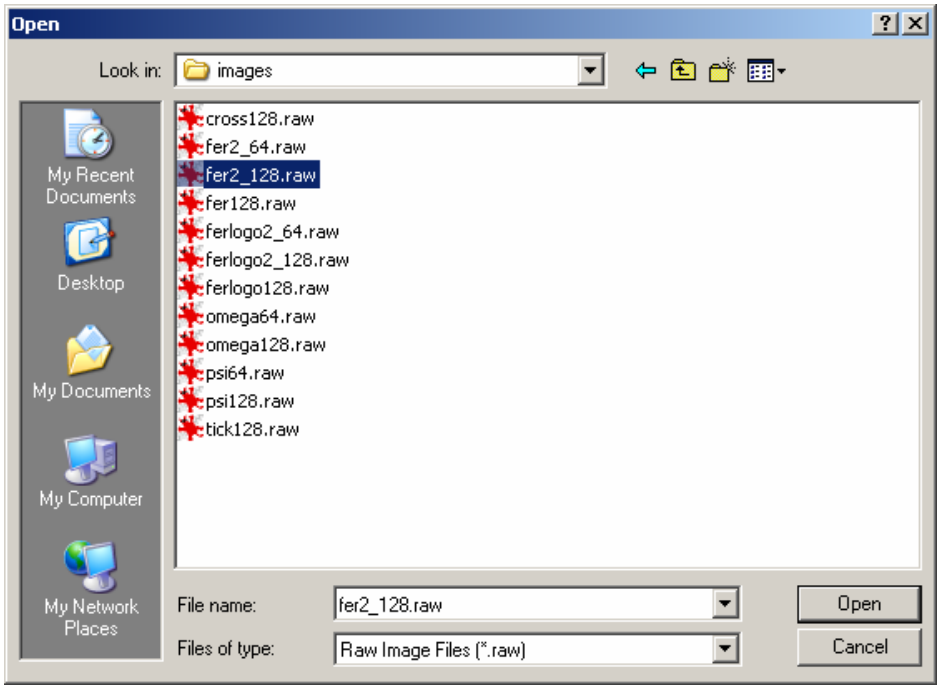
Na vrhu prozora nalazi se konfiguracijska cjelina s dva pripadna gumba – Učitaj konfiguraciju (*eng. Load Configuration*) i Sačuvaj konfiguraciju (*eng. Save Configuration*). Pomoću njih se, kao što im naziv govori, može sačuvati postojeća konfiguracija ili učitati ranije sačuvana konfiguracija. Obje radnje obavljaju se preko standardnog Windows dijaloga za pretraživanje datoteka. Time se pojednostavljuje korištenje programa. Detaljniji opis konfiguracijske datoteke potražiti u dodatku.

Slika 11. Konfiguracijska cjelina

5.3.2. Cjelina za učitavanje slika

Odmah ispod se nalazi cjelina za učitavanje početne i konačne slike. Početna slika predstavlja početnu gustoću dima, a konačna predstavlja željenu gustoću dima. Put do slike može se unijeti ručno ili pomoću standardnog Windows dijaloga za pretraživanje datoteka kao na slici 13. Obje slike moraju biti u RAW formatu koji je detaljnije opisan u dodatku.

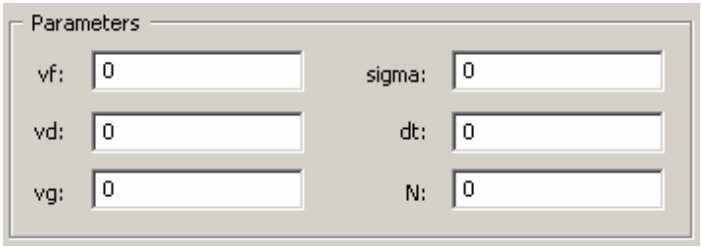
Slika 12. Cjelina za učitavanje slika



Slika 13. Dijalog za učitavanje slika

5.3.3. Cjelina za određivanje iznosa parametara

Sastoji se od šest kontrola za unos brojeva. Pored svake kontrole je naveden naziv parametra kojeg ona predstavlja. Brojevi se mogu unositi isključivo ručno.



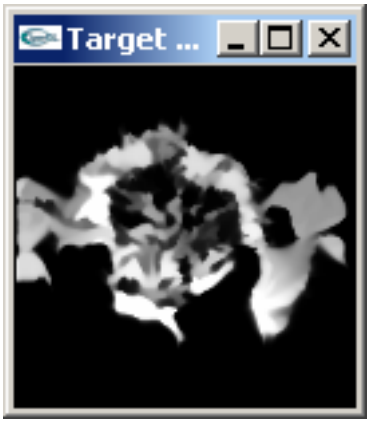
Slika 14. Cjelina za određivanje iznosa parametara

5.3.4. Upravljačka cjelina

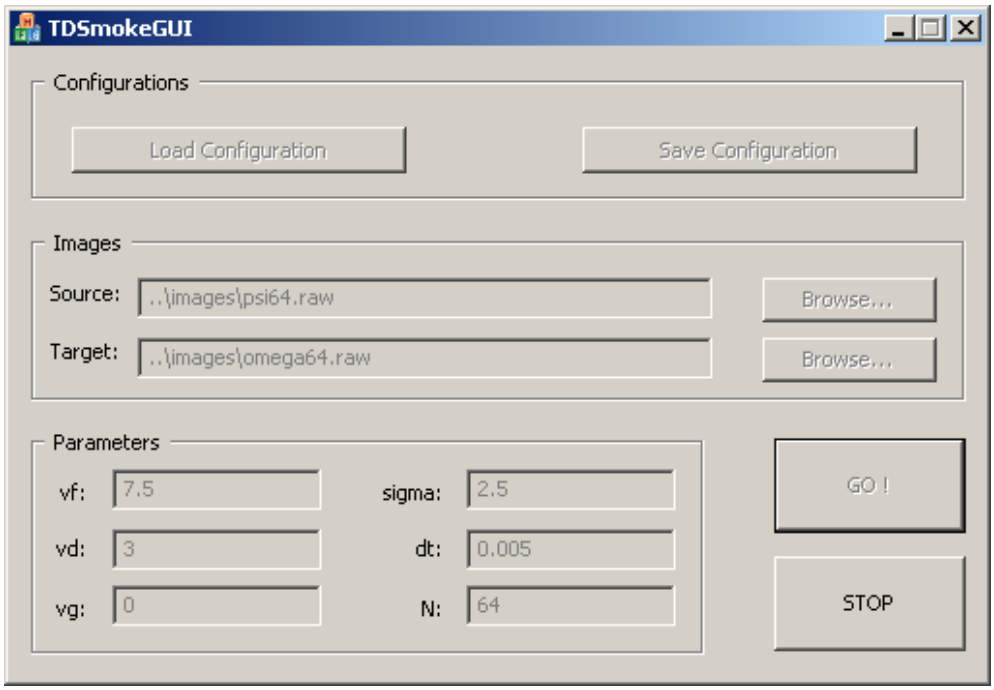
Nalazi se u donjem desnom dijelu prozora i sastoji od dva gumba – Pokreni (*eng. Go*) i Zaustavi (*eng. Stop*). Pritiskom na "Pokreni" provjerit će

se ispravnost parametara te, ako je sve u redu, pokrenuti simulacija. Simulacija se pokreće u novom prozoru kao na slici 16. Dok simulacija traje, cijelo grafičko korisničko sučelje osim gumba "Zaustavi" biti će onemogućeno što se može vidjeti na slici 17. Nakon pritiska na gumb "Zaustavi", sučelje se vraća u početno stanje.

Slika 15. Upravljačka cjelina



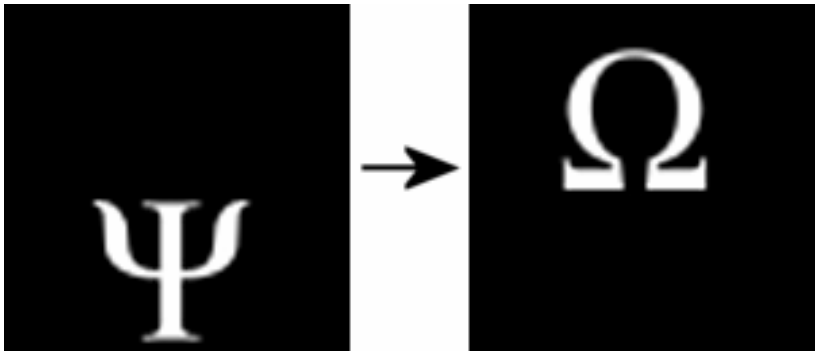
Slika 16. Prozor u kojem se odvija simulacija



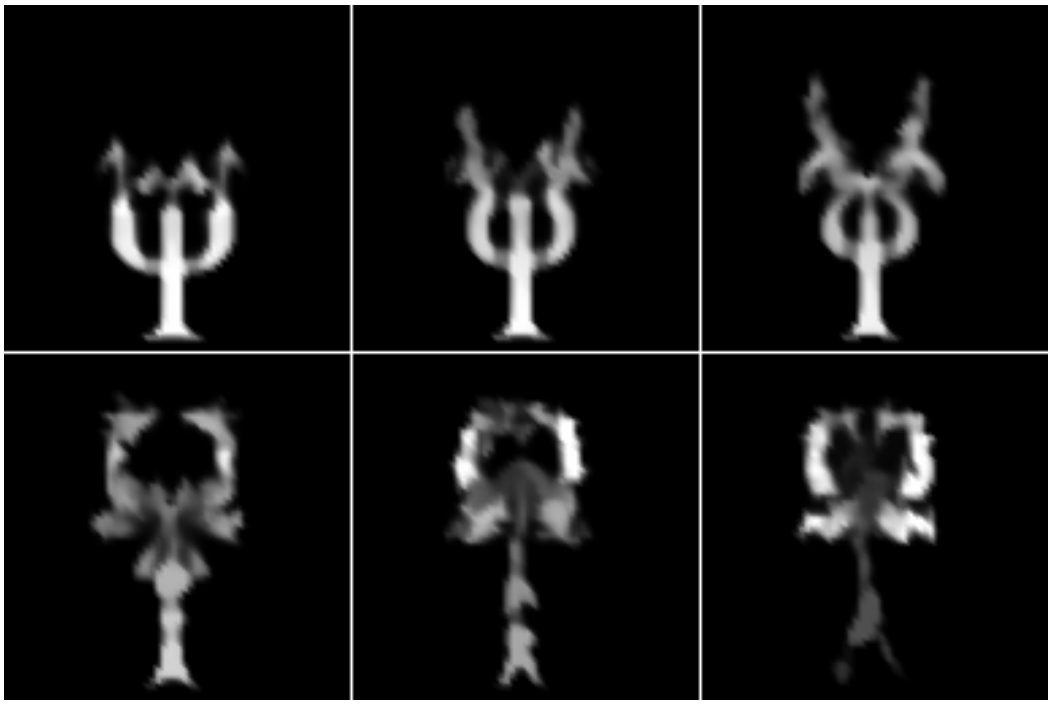
Slika 17. Grafičko korisničko sučelje za vrijeme simulacije

6. Rezultati

U ovom poglavlju biti će prikazani rezultati dobiveni programom čija je implementacija opisana u poglavlju 4. Posebice će se prikazati utjecaj svakog od parametara simulacije navedenih u odjeljku 4.3. Za početak slijedi primjer koji je naveden i u [3]:

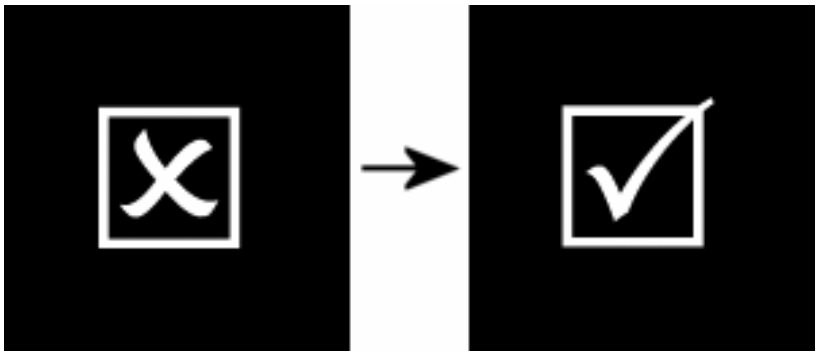


Slika 18. Početno i konačno stanje (psi → omega)

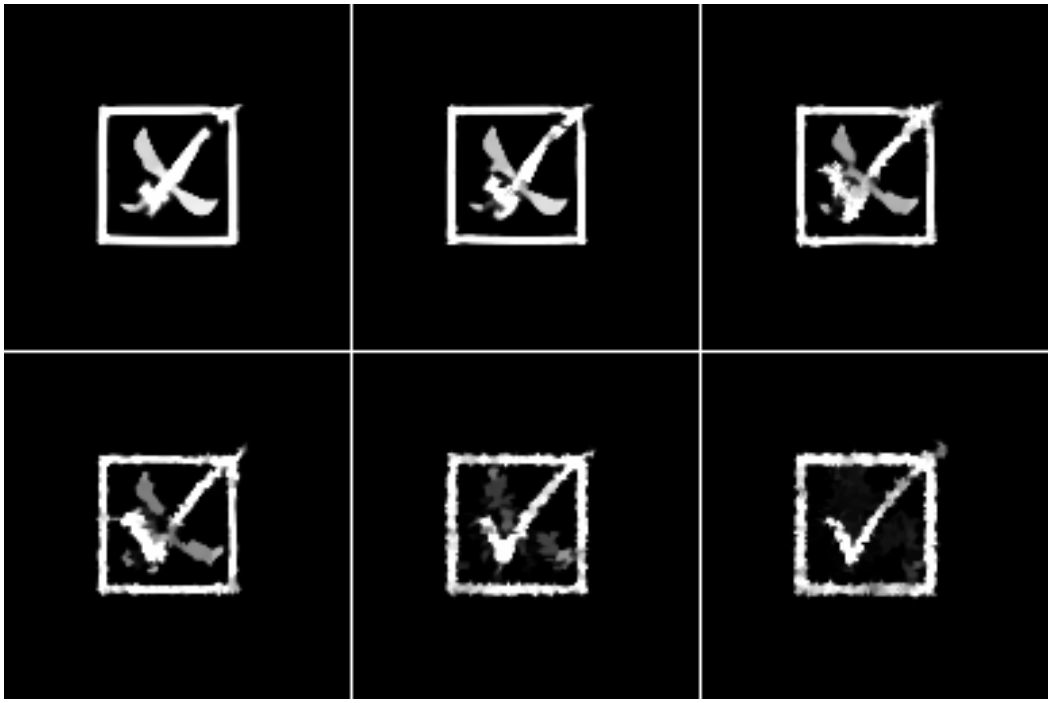


Slika 19. Međustanja (prva slika je gore lijevo, a posljednja dolje desno)

Simulacija sa slike 18 dobivena je na polju veličine 64 x 64 uz sljedeće iznose parametara: $v_f = 7.5$, $v_d = 3$, $v_g = 0$, $\sigma = 2.5$. Vremenski korak iznosi $dt = 0.005$. Slijedi još jedan primjer:



Slika 20. Početno i konačno stanje (križić → kvačica)



Slika 21. Međustanja animacije

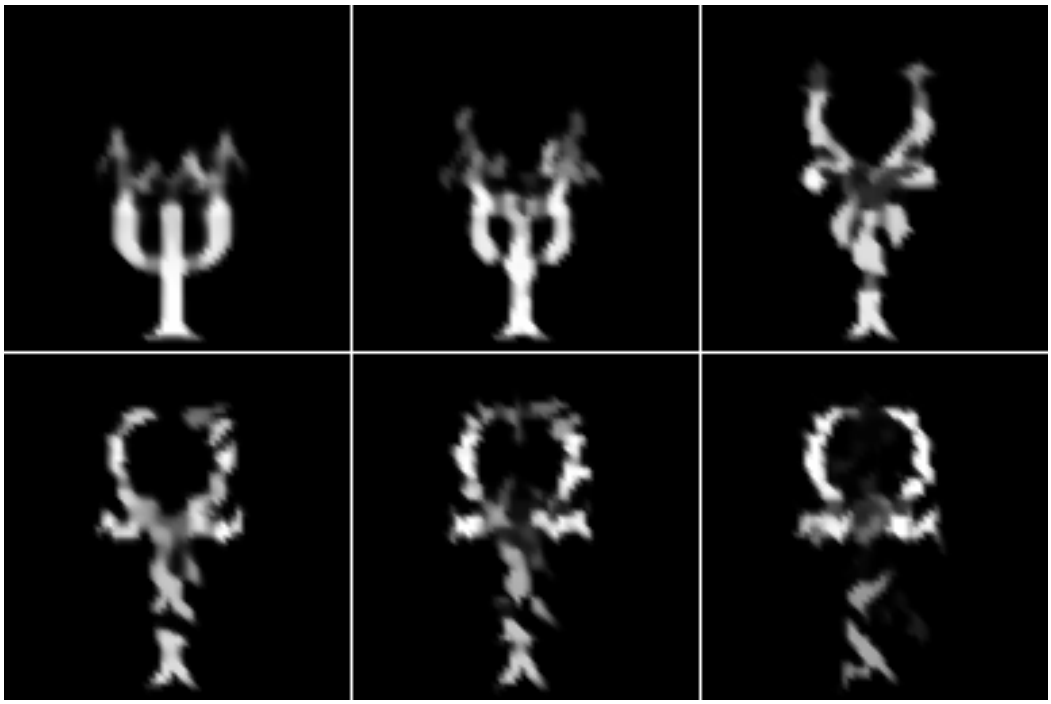
Simulacija sa slike 21 dobivena je na polju veličine 128 x 128 uz vremenski korak $dt = 0.005$. Ostali parametri iznose: $v_f = 1.25$, $v_d = 2.5$, $v_g = 0$, $\sigma = 1$. Odmah se može primijetiti da je animacija sa slike 19 mutnija od animacije sa

slike 21. To je zbog rezolucije polja na kojem se simulacija odvija (64 x 64 odnosno 128 x 128).

6.1. Utjecaj parametara na simulaciju

6.1.1. Parametar v_f

Kao što je već ranije rečeno, parametar v_f određuje brzinu kojom simulacija napreduje prema ciljnom stanju. Promatrat ćemo simulaciju istu kao na slici 19, ali će parametar v_f biti postavljen na 1.5 umjesto na 7.5. Dobiva se sljedeći rezultat:



Slika 22. Simulacija uz parametar v_f = 1.5

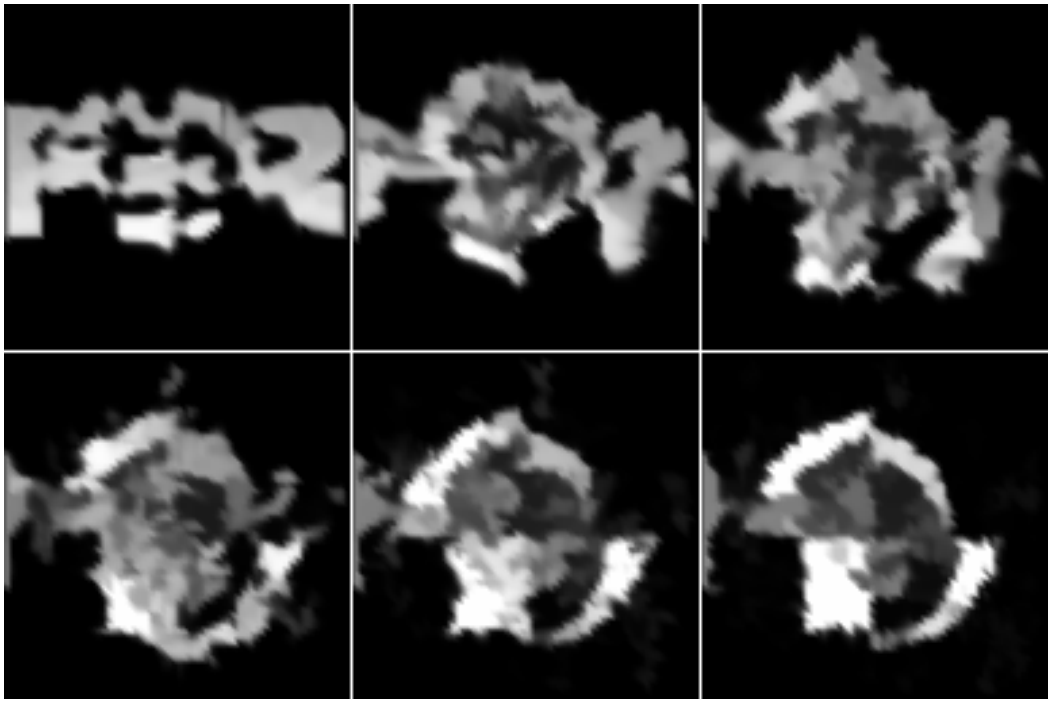
Može se primijetiti da se stvari nisu značajno promijenile. Dim ima slično gibanje kao i na slici 19 i više-manje uspješno postiže ciljni oblik kao i na slici 19. Međutim, ono što se je promijenilo je brzina kojom se dim giba. Dok simulacija sa slike 22 dođe do stanja prikazanog četvrtom sličicom (prva u

drugom redu), simulacija sa slike 19 već se nalazi u ciljnom stanju. Za detaljniji uvid potrebno je pokrenuti program priložen uz ovaj rad.

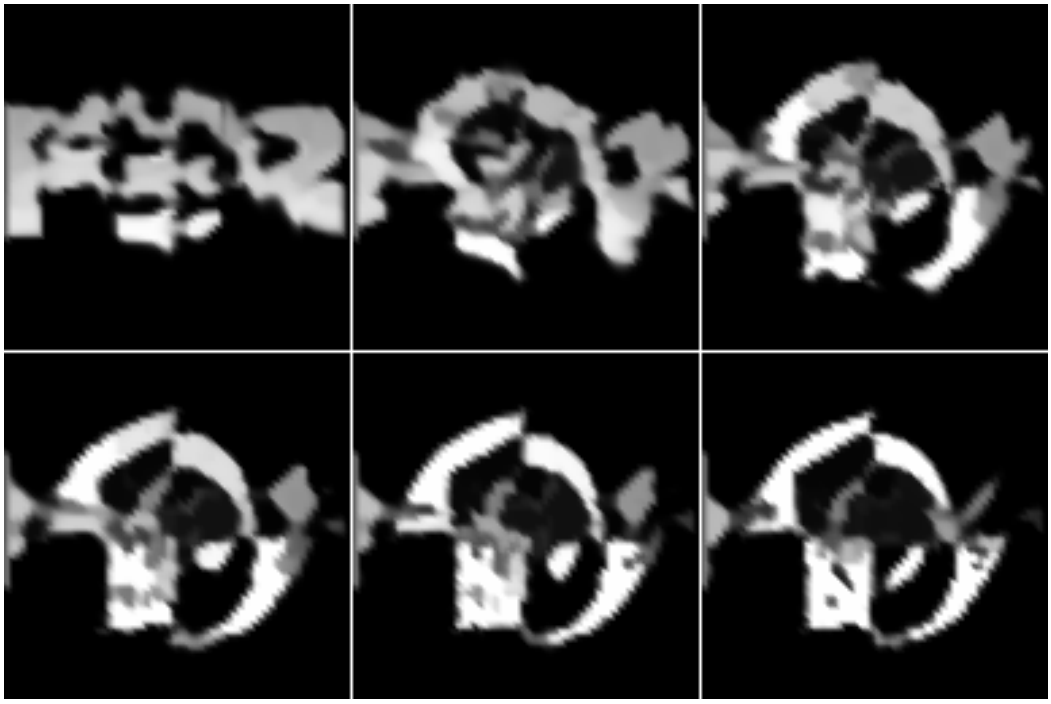
6.1.2. Parametar v_d

Parametar v_d određuje prigušenje količine gibanja. Njegov utjecaj vidjet ćemo na sljedećoj simulaciji.

Slika 23. Početno i konačno stanje – utjecaj parametra v_d



Slika 24. Simulacija uz iznos parametra v_d = 1

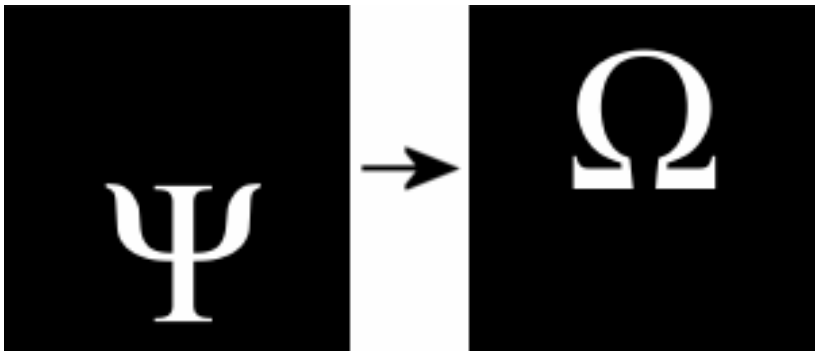


Slika 25. Simulacija uz iznos parametra v_d = 5.5

Ostali parametri simulacije su $v_f = 3.5$, $v_g = 5 \cdot 10^{-5}$, $\sigma = 1$ uz vremenski korak $dt = 0.001$. Polje na kojem se odvija proračun je veličine 64 x 64. Može se primijetiti da dim na slici 24 lako "preskače" ciljno stanje. Premda je njegova gustoća blizu željene, sile su još uvijek dovoljno jake da ga izbace iz ciljnog položaja i tako nastave gibanje. S druge strane, na slici 25 je situacija drugačija. Zbog 5.5 puta većeg parametra v_d, prigušenje količine gibanja je dovoljno da se dim lakše zadrži u ciljnom stanju.

6.1.3. Parametar v_g

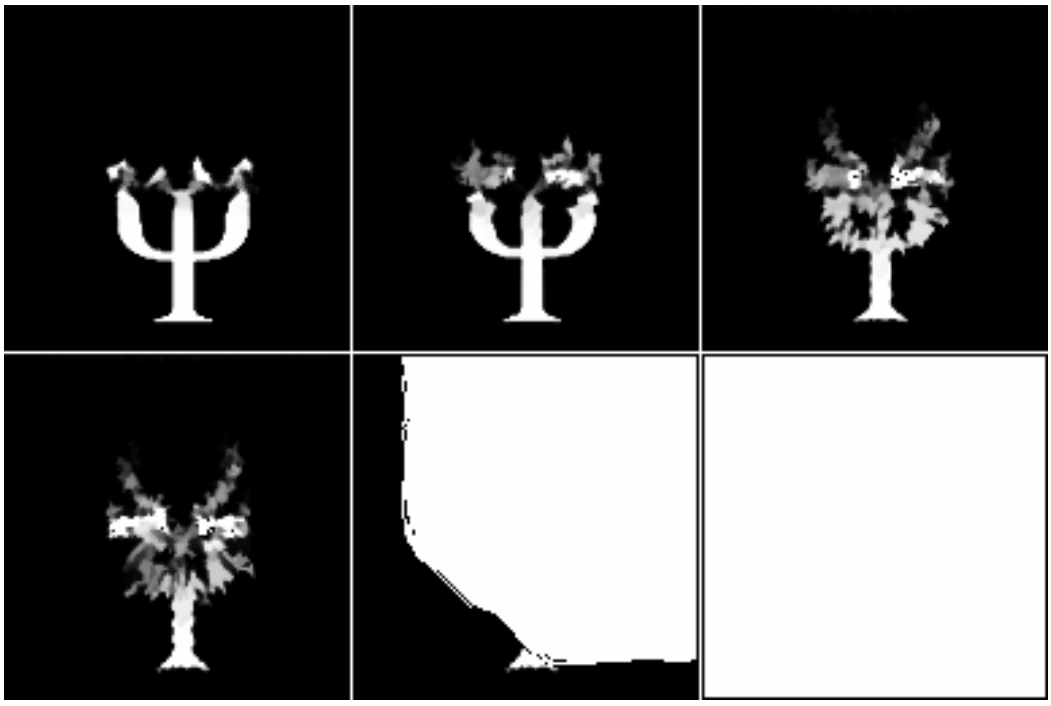
Parametar v_g sprječava difuziju dima u blizini ciljnog stanja. Njegov utjecaj promatrati će se na simulaciji sa slike 26. Ta simulacija nije ista kao ona sa slike 18 jer se izvodi na polju veličine 128 x 128 dok se prethodna izvodila na polju veličine 64 x 64.



Slika 26. Početno i konačno stanje – utjecaj parametra v_g

Slika 27. Simulacija uz iznos parametra v_g = 0

Postavljanjem parametra v_g na nulu isključeno je sprječavanje difuzije dima. Ostali parametri iznose $v_f = 1$, $v_d = 2.5$, $\sigma = 1.3$ uz vremenski korak $dt = 0.005$. Simulacija više manje uspješno napreduje prema željenom stanju. Povećanjem parametra v_g na iznos $5 \cdot 10^{-6}$ dobiva se sljedeći rezultat:



Slika 28. Simulacija uz iznos parametra $v_g = 5 \cdot 10^{-6}$

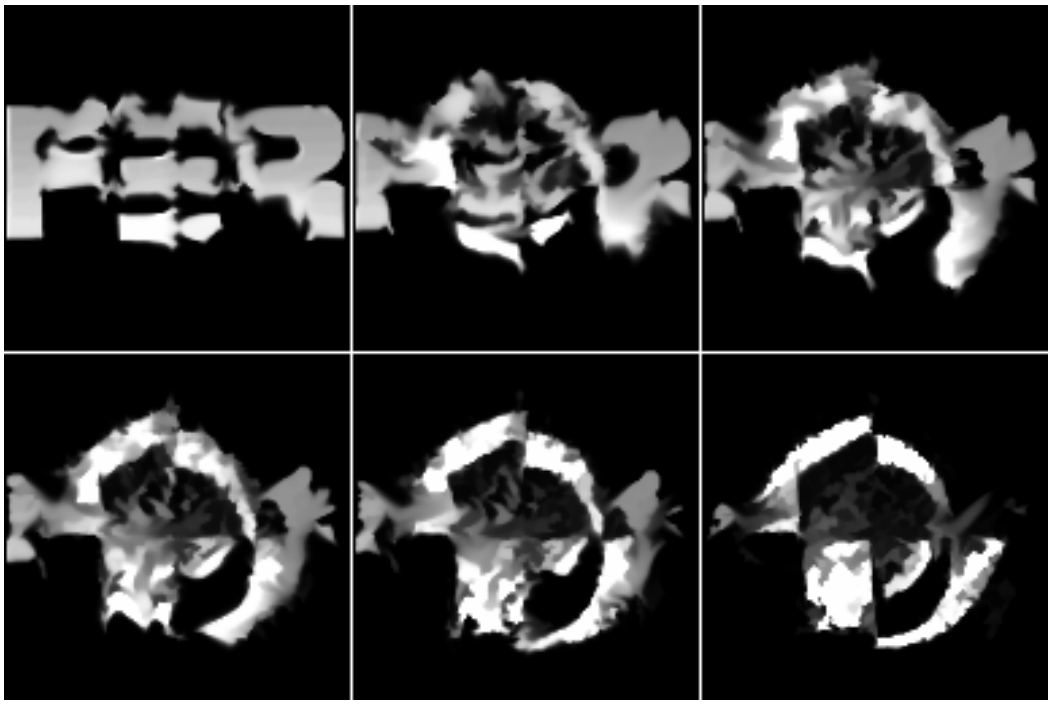
Prve četiri sličice sa slike 28 ne razlikuju se od onih sa slike 27. Međutim, posljednje dvije sličice prikazuju kako se simulacija "razletila" pod utjecajem parametra v_g. Smanjenje iznosa tog parametra nema smisla jer se i ovako na prve četiri sličice njegov utjecaj ne primjećuje, a smanjenje koraka integracije *dt* također se nije pokazalo uspješnim. S obzirom da u [3] nigdje nije naveden opseg parametara, parametar v_g nije dao željene rezultate već je samo bio uzrok nestabilnosti simulacije.

6.1.4. Parametar σ

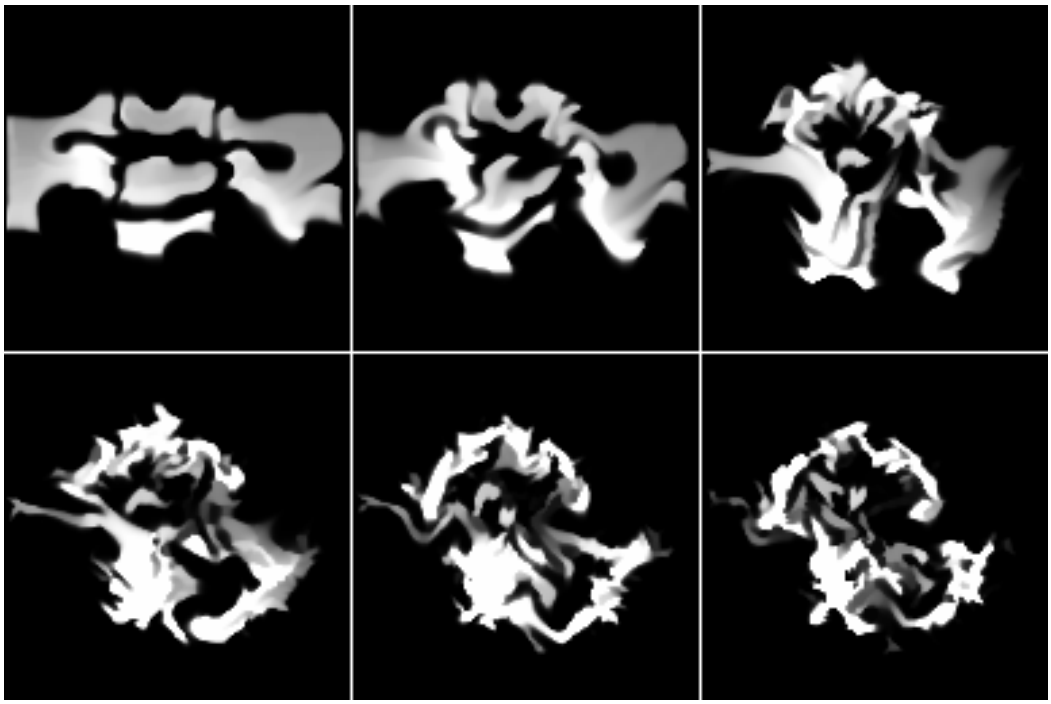
Parametrom σ zaglađuje se vodeća sila. Njegov utjecaj prikazat ćemo na sljedećoj simulaciji.

Slika 29. Početno i konačno stanje – utjecaj parametra σ

Ova simulacija nije ista kao sa slike 23 jer se odvija na polju veličine 128 x 128 dok se simulacija sa slike 23 odvija na polju veličine 64 x 64. Prikazat će se dva slučaja – kada parametar σ iznosi 1 i kada iznosi 7. Slijede rezultati:



Slika 30. Simulacija uz iznos parametra σ = 1



Slika 31. Simulacija uz iznos parametra σ = 7

Ostali parametri simulacije su $v_f = 20$, $v_d = 6.5$, $v_g = 5 \cdot 10^{-5}$ uz vremenski korak $dt = 0.0002$. Slika 30 prikazuje simulaciju uz parametar $\sigma = 1$. Vidimo da dim prilično grubim kretnjama putuje do cilja. Također dolazi i do rasipanja po putu. Za razliku od toga, slika 31 prikazuje simulaciju uz parametar $\sigma = 7$. Kretnje dima su finije, a rasap je smanjen. Međutim, uočljivo je da se povećanjem parametra σ smanjuje mogućnost da dim postigne željeni oblik.

6.2. Vremena izvođenja

Sva testiranja obavljena su na osobnom računalu PC s procesorom AMD Athlon 64 3000+ (1800 MHz) i 512 MB radne memorije. Testirane su simulacije koje se odvijaju na polju veličine 64 x 64 te 128 x 128 i to na način da se mjerilo koliko vremenskih koraka, odnosno poziva metode `MakeStep`, je moguće obaviti unutar 5 sekundi. Taj broj je podijeljen s vremenom da se dobije broj poziva unutar 1 sekunde (*eng. FPS – frames per second*).

Za simulaciju na polju veličine 64 x 64 dobiveni su sljedeći rezultati:

Br.	Br. poziva unutar 5 s	unutar 1 s
1	493	98.6
2	465	93
3	471	94.2
4	476	95.2
5	474	94.8
6	475	95
7	476	95.2
8	474	94.8
9	474	95.4
10	476	95.2

Tabela 1. Rezultati na polju veličine 64 x 64

S obzirom na činjenicu da je ljudskom oku dovoljno 25 sličica po sekundi da dobije osjećaj animacije, iz tablice možemo ustvrditi da simulacija protiče i više nego glatko. Srednja vrijednost mjerenja iznosi 95.14 sličica po sekundi.

Za simulaciju na polju veličine 128 x 128 dobiveni su sljedeći rezultati:

Br.	Br. poziva unutar 5 s	unutar 1 s
1	116	23.2
2	121	24.2
3	116	23.2
4	109	21.8
5	118	23.6
6	116	23.2
7	113	22.6
8	114	22.8
9	113	22.6
10	117	23.4

Tabela 2. Rezultati na polju veličine 128 x 128

Iz tablice možemo vidjeti da su rezultati malo ispod donje granice od 25 sličica po sekundi. Srednja vrijednost mjerenja iznosi 23.06 sličica po sekundi.

Usporedbe radi, na osobnom računalu PC s procesorom Pentium IV 2400 MHz i 512 MB radne memorije proizvedeno je u prosjeku 15.5 sličica po sekundi za simulaciju na polju 128 x 128 te 65.6 sličica po sekundi za simulaciju na polju 64 x 64. Računalo s procesorom Pentium IV 1800 MHz uspjelo je u prosjeku proizvesti 11.91 sličicu po sekundi za simulaciju na polju 128 x 128 te 44.81 sličicu po sekundi za simulaciju ja polju 64 x 64.

7. Zaključak

Za potpuno razumijevanje postavljenog zadatka bilo je potrebno prisjetiti se osnova dinamike fluida i vektorske analize. Premda nije nužna, dobra razrada teorijskog dijela ipak je omogućila izbjegavanje nekih grešaka u kasnijoj fazi projekta. Implementacija zadanog algoritma nakon toga je uspješno obavljena. Dobiveni rezultati prilično su uvjerljivi što dokazuje da je računalni model opisan u poglavlju 3 valjan. Korištena metoda predstavlja prilično zanimljiv i jednostavan način na koji animatori mogu postići željene rezultate. Za to je potrebno utrošiti malo vremena na podešavanje parametara, ali se brzo nauči. Konačno, ključ uspješne simulacije ponajviše leži u iskustvu.

8. Literatura

[1] A.J. CHORIN, J.E. MARSDEN, *A Mathematical Introduction to Fluid Mechanics*, Springer-Verlag, New York, 2000.

[2] J. STAM, Stable Fluids, *SIGGRAPH 99 Conference Proceedings, Annual Conference Series*, August 1999, pp 121-128

[3] R. FATTAL, D. LISCHINSKI, *SIGGRAPH 2004 Conference Proceedings, Annual Conference Series*, August 2004, pp 441-448

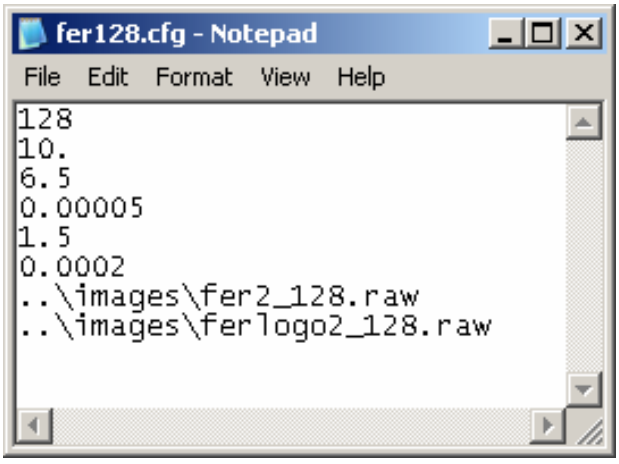
[4] U. TROTTENBERG, C.W. OOSTERLEE, A. SCHÜLLER, *Multigrid*, Academic Press, 2001.

[5] http://amronasr.com/site/projects.htm

[6] http://www.cc.gatech.edu/~sarah/smoke.htm

Dodatak: Konfiguracijska datoteka

Konfiguracijska datoteka je obična tekstualna datoteka s nastavkom cfg. U njoj se nalaze iznosi parametara i putevi do izvorne i odredišne slike. Svaki podatak nalazi se u svom redu. Otvaranjem jedne takve datoteke u programu poput Notepada dobivamo sljedeće:



Slika 32. Izgled konfiguracijske datoteke

Prvi podatak je veličina polja, u ovom slučaju 128 x 128. Zatim slijede parametri i to ovim redom: v_f, v_d, v_g, σ nakon čega dolazi vremenski korak *dt*. Posljednji su putevi do slika: prvo do izvorne slike, a zatim do odredišne.

Sažetak

U radu je dan kratak pregled teorijskih osnova dinamike fluida te razrađen model za simulaciju na računalu. Posebna pažnja posvećena je simulacijama vođenim konačnim stanjem, točnije metodi opisanoj u radu "Target-Driven Smoke Animation" [3]. Načinjena je implementacija navedene metode u dvije dimenzije koristeći programski jezik C++. Prikazani su dobiveni rezultati te utjecaj svakog parametra zasebno na tijek simulacije.

Abstract

Title: Animation of Fluid Flow

In this paper we briefly outline theoretical fundamentals of fluid dynamics as well as present a model for computer based simulation. Of special interest are target driven smoke simulations, specifically the method presented in "Target-Driven Smoke Animation" [3]. The method was implemented in two dimensions using programming language C++. The results obtained are presented alongside parameters that affect the progress of simulation.