Gymnázium, Praha 7, Nad Štolou 1

Maturitní práce

Úvod do simulace tekutin

Autor práce: Samuel Grulich
Vedoucí práce: Martin Sourada
Třída: 4.A

Praha 2023

Maturitní práce z Informatiky a programování

Téma: Počítačová simulace fyzikálního děje či experimentu

Autor práce: Samuel Grulich, 4. A

Vedoucí práce: Martin Sourada

Žák dodrží předem dohodnuté termíny konzultací (alespoň tři konzultace – do 31. prosince 2022, do 10. února 2023 a do 24. března 2023) a práci, která je součástí maturitní zkoušky z informatiky a programování, předloží nejpozději do 31. března 2023. Tuto práci odevzdá elektronicky ve formátu PDF v systému *Moodle* a ve dvou podepsaných výtiscích, oba s přiloženou praktickou částí (na CD či flash kartě) a kopií tohoto zadání. Maturitní práce musí mít délku nejméně 10 normostran.

Teoretická část

Žák teoreticky rozebere a vysvětlí problematiku proudění tekutin. Dále navrhne a popíše algoritmy, které budou sloužit k výpočtu a simulaci. Provede rešerši existujících technologií, které může využít. U technologií, které použije, pak odůvodní tento výběr a popíše, jakým způsobem je při tvorbě práce využil. Navrhne grafické prostředí pro vizualizaci výpočtů/simulací a rozebere implementaci nejdůležitějších částí tohoto prostředí.

Praktická část

Žák vytvoří grafické prostředí pro praktickou vizualizaci a simulaci proudění tekutin. Součástí prostředí bude možnost snadného uživatelského nastavení všech parametrů simulace. Dále napíše uživatelskou příručku (návod k použití), kde vytvořené testovací prostředí popíše. Tato příručka bude tvořit samostatnou kapitolu v maturitní práci.

Při tvorbě kódu bude využívat systém pro sledování verzí (např. git) a změny kódu bude pravidelně publikovat v repozitáři dostupném (přinejmenším) vedoucímu i oponentovi práce (např. prostřednictvím služby github či gitlab). Adresu repozitáře uvede v maturitní práci.

Datum: 23. října 2022

Podpis žáka: _____

Podpis vedoucího práce: _____

Abstrakt

Cílem této práce je úvod simulace tekutin. Letmé představení často používaných metod a technik pro simulaci tekutin a její části. Bližší prozkoumání některých odvození, nebo detailnější představení zmíněných metod. Na konci práce představíme jednoduchou implementaci vybudovanou pomocí metod zmíněných v práci. Následující text vychází z excelentní práce [1].

Prohlášení

Prohlašuji, že jsem maturitní práci vypracoval samostatně, použil jsem pouze podklady uvedené v přiloženém seznamu a postup při zpracování a dalším nakládání s prací je v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů.

V Praze dne 30. března 2023 _____

Poděkování

Chtěl bych poděkovat vedoucímu práce, Martinu Souradovi, za jeho čas, trpělivost a ochotu. Při vedení mé maturitní práce. Za jeho vřelý přístup k mým dotazům. Jeho rady mi pomohly k opravení chyb a posunuly mne o krok blíže k finálnímu výsledku.

Obsah

Úvod

Simulace tekutin je v počítačové grafice technika sloužící pro simulaci tekutin ve virtuálním prostředí. Dnes je často využívána i v komerčně dostupných produktech, jako je například Houdini nebo Blender. Simulace mají širokou škálu uplatnění od vizuálních efektů ve filmech přes design až po vědecký výzkum (například proudění krve).

Potřebná úroveň detailu a přesnosti výsledku simulace se většinou liší podle případu použití. Například simulace využitá pro vědecký výzkum může vyžadovat vyšší úroveň detailu a přesnosti než simulace použitá v animaci. Jenže simulace tekutin jsou hardwarově a časově náročné, proto je každá implementace optimalizována pro dané použití. Podkladné modely mohou simulovat například teplotu tekutiny, viskozitu, vířivost nebo spoustu dalších veličin.

1 Navier-Stokesovy rovnice

Soustava Navier-Stokesových rovnic, vyvinutá Claudem-Louisem Navierem v roce 1822 a Georgem Gabrielem Stokesem v roce 1845, se prokázala jako nejspolehlivější a nejprominentnější model pro simulaci tekutin. Jedná se o soustavu parciálních diferenciálních rovnic, které popisují pohyb částic způsobený externími a interními silami. Externími silami mohou být síla gravitační, síly způsobené kolizí s okolními objekty a jiné. Interními silami mohou být například viskózní síla nebo tlaková síla. Nicméně rovnice jsou notoricky známé jako obtížně řešitelné. Soustava vychází z Newtonova 2. pohybového zákona o síle:

$$F = ma$$

Dále pak z prvního zákona termodynamiky o zachování hmoty. Následující zápis Navier-Stokesových rovnic je poměrně častý:

$$\rho \frac{Dv}{Dt} = \nabla p + \mu \nabla^2 v + f_{ext} \tag{1}$$

$$\nabla \cdot v = 0 \tag{2}$$

V rovnicích je hmotnost zaměněna za hustotu. [1] vysvětluje, že důvod tohoto zápisu je čistě historický. Zápis je platný protože všechny částice reprezentují stejný a konstantní objem. Nicméně o rovnici (1) můžeme uvažovat jako o Newtonově 2. pohybovém zákoně. Derivace rychlosti podle času vyjadřuje zrychlení částice. Zrychlení vynásobené hmotností, v našem případě hustotou, musí být ekvivalentní silám působícím na danou částici kapaliny. Rovnice (2) říká, že rychlosti částic v jakémkoliv okolí nemohou směřovat, nebo vycházet z jednoho bodu. Takže v žádném bodě spontánně nevzniká ani nezaniká energie soustavy.

1.1 Metody řešení Navier-Stokesových rovnic

Jak už bylo výše zmíněno, řešení Navier-Stokesových rovnic je obtížné, obzvláště pro komplexní proudění tekutin. Zatím neexistuje jednotné a obecné analytické řešení rovnic. V této práci byly použity pouze postupy vycházející z Lagrangeovy metody, ale další populární metodou je Eulerova, pro bližší informace odkazujeme čtenáře na [2].

Lagrangeova metoda sleduje individuální částice napříč časem a prostorem. Lagrangeova metoda je zejména užitečná pro simulaci nestabilních proudění a proudění s vysokou mírou deformace. [2]

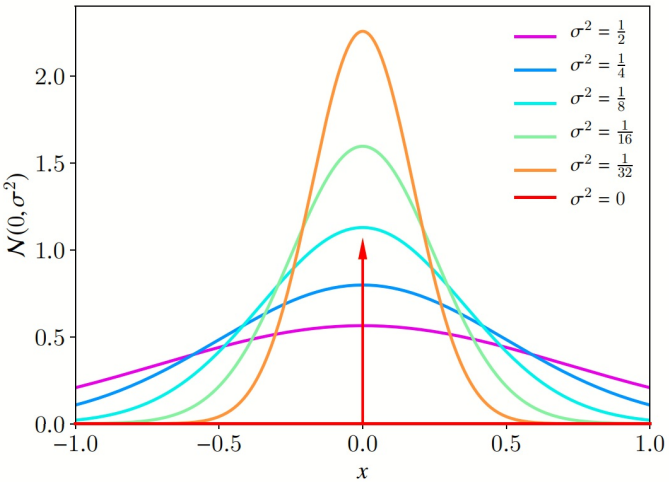
Eulerova metoda uvažuje o výchozí tekutině jako o kontinuu. Látce složené z nekonečně mnoha nekonečně malých částic. Tekutinu následně rozděluje do stejně velkých kontrolních kvadrantů. Každý kvadrant sleduje množství a vlastnosti tekutiny, kterou obsahuje. Tato metoda je často využívána pro aerodynamické problémy. [2]

1.2 Hydrodynamika vyhlazených částic (Smoothed Particle Hydrodynamics - SPH)

SPH je numerická metoda vycházející z Lagrangeovy metody. SPH byla poprvé představena Lucy [3], Monaghanem a Gingoldem [4] pro aplikaci v astronomii. Později byla využita i pro simulaci tekutin a dnes je poměrně rozšířená v simulaci tekutin. Je důležité zmínit, že SPH částice nereprezentují elementární částice, ale jakýsi soubor částic. Obecně můžeme o SPH uvažovat jako o reprezentaci pole veličiny, které je nahrazeno částicemi. Zároveň SPH umožňuje vyhodnotit veličinu v jakémkoliv bodě výchozí tekutiny, takže můžeme zrekonstruovat původní pole. Částice tedy slouží jako referenční body pro jakýkoliv bod kapaliny. Podle [1] diskretizací Navier-Stokesových rovnic můžeme přibližnou hodnotu pole dané veličiny v jakémkoliv bodě v kapalině vyjádřit následovně.

$$A(x_i) = \sum_j m_j \frac{A_j}{\rho_j} W(x_i - x_j, h) \tag{3}$$

Kde x_i je hledaný bod, j jsou částice v okolí h, m_j je hmotnost částice j, A_j je hodnota v částici j, rho_j je hustota částice j a W je vyhlazovací jádro. Vyhlazovací jádro je funkce o dvou proměnných, r a h, kde h je podpůrný poloměr funkce a r je vektor mezi dvěmi částicemi. Jádro je odvozeno z Diracovy-delta distribuce a musí splňovat následující podmínky. [1]



Obrázek 1: Normalizovaný graf vyhlazovacího jádra, [1]

$\forall r \in \mathbb{R}^d$	$h \in \mathbb{R}^+$
$\int_d W(r', h)dv' = 1$	(Normalizační podmínka)
$\lim_{h' \to 0} W(r, h') = \delta(r)$	(Diracova-delta podmínka)
$W(r, h) \geq 0$	(Podmínka pozitivity)
$W(r, h) = W(-r, h)$	(Podmínka symetrie)
$W(r, h) = 0 pro \|\|r\|\| > h$	(Podmínka kompaktní podpory)

Normalizační podmínka zaručuje, že obsah plochy pod grafem funkce bude vždy roven 1. Diracova-delta podmínka zaručuje pro jádro s nulovým podpůrným poloměrem rovnost s Diracovou-delta funkcí. Následná podmínka říká, že funkční hodnota jádra musí být vždy nezáporná. Podmínka symetrie zaručuje radiální symetrii pro jakoukoliv délku r. Podmínka kompaktní hodnoty říká, že funkční hodnota pro vektor mezi dvěmi částicemi, jehož velikost je větší než podpůrný poloměr, je rovna 0.

Nicméně, typickou volbou funkce pro vyhlazovací jádro je kubická spline křivka. [1]

$$W(r,h) = \mathrm{o}_d \begin{cases} 6(q^3+q^2)+1 & pro\ 0 \leq q \leq \frac{1}{2} \\ 2(1-q)^3 & pro\ \frac{1}{2} < q \leq 1 \\ 0 & jinde \end{cases}$$

Kde $q = \frac{1}{h}||r||$ a normalizační faktor je pro počet dimenzí $d = 1, 2, 3$ v následujícím pořadí $o_1 = \frac{4}{3h}, o_2 = \frac{40}{7h^2}$ a $o_3 = \frac{8}{\pi h^3}$.

Pro vyčíslení Navier-Stokesových rovnic potřebujeme kromě diskretizace pole veličiny, také diskretizaci jeho prostorových derivátů. Na základě rovnice (3) může být aproximace gradientu zapsána následovně. [1]

$$\nabla A \approx \sum_j A_j \frac{m_j}{\rho_j} \nabla W_{ij}$$

Kde W_{ij} je $W(x_i - x_j, h)$. Bohužel tato "přímá" derivace není dostatečně přesná. V práci [5] je pomocí Lagrangeovského odhadu hustoty odvozen tzv. symetrický vzorec. Symetrický vzorec je vhodnější, protože konzervuje lineární a úhlovou hybnost [1].

$$\nabla A_i \approx \rho_i \sum_j \left(\frac{A_i}{\rho_i^2} + \frac{A_j}{\rho_j^2} \right) \nabla W_{ij} \tag{4}$$

Gradient vyhlazovacího jádra vypočteme vynásobením první derivace kubické spline křivky s gradientem q.

$$\nabla q = \frac{||r||}{r}$$

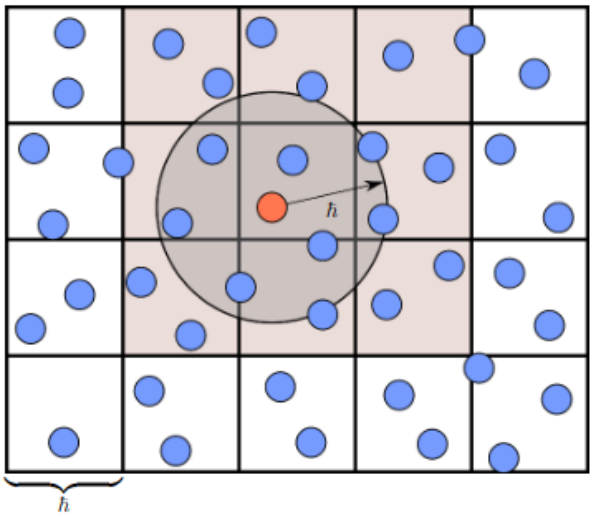
$$\nabla W = \mathrm{o}_d \nabla q \begin{cases} 6(q^2+q) & pro\ 0 \leq q \leq \frac{1}{2} \\ -6(1-q)^2 & pro\ \frac{1}{2}\ q \leq 1 \\ 0 & jinde \end{cases}$$

2 Výpočet působících sil

Je zřejmé ze zápisu Navier-Stokesových, že na každou částici musí působit hned několik sil. V následující kapitole rozebereme jak a proč síla působí, ale nejdříve musíme předvést algoritmus pro hledání sousedních částic (sousedství).

2.1 Hledaní sousedních častíc

Hledání sousedních částic v SPH simulaci je zásadní pro správné výpočty. Hledaní sousedních částic je velice často nejpomalejším krokem všech modelů pro simulaci tekutin. Existují různé algoritmy, jak najít sousedy částice. Například iterací všech částic, nebo mřížovou metodou. Iterace všech částic je snadno implementovatelná, ale kvůli její výpočetní komplexitě $O(n^2)$ není vhodná pro větší množství částic. Mřížová metoda je lepším kandidátem pro početnější simulace a je velice populární, právě protože je její výpočetní komplexita konstantní a to $O(3^d - 1)$.



Obrázek 2: Hledání sousedních částic v mříži [1]

Výpočetně nejnáročnějším krokem mřížového algoritmu je aktualizace a tvorba mříže. Mříž s buněčnou velikostí h, má unikátní vlastnost. To právě, že hledané sousední částice se nacházejí pouze v přímo sousedících buňkách. Využití podpůrného poloměru je tedy kritické pro zaručení rychlého vyhledávaní.

Populárním algoritmem pro vytváření mříže je hašování pozice buňky [6]. Tato práce využívá limitaci domény pro tvorbu mříže.

2.2 Tlak

Tlaková síla vyrovnává tlaky mezi různými oblastmi tekutiny. Tlaková síla působí z oblasti s vyšším počtem částic směrem do oblasti s nižším počtem. Zde je jemně naznačeno, že částice musí mít dynamickou hustotu. Hustota pro jakoukoliv částici je přímo úměrná počtu okolních částic a dosazením do SPH aproximace v rovnici (3) dostaneme následující vztah.

$$\rho_i = \sum_j m W_{ij}$$

Kde ρ_i je hustota v částici i, m je počáteční hmotnost částice. Jedním způsobem pro výpočet tlaku je použití zákonu ideálního plynu. Zákon vyjadřuje vztah mezi tlakem, teplotou a objemem plynu. Nicméně, tato metoda není vždy nejspolehlivější volbou, jelikož předpokládá, že se kapalina chová jako ideální plyn, což nemusí vždy platit. Dalším způsobem je využití stavové rovnice, která byla předvedena v [7] a vyjadřuje vztah mezi tlakem, objemem a vnitřní energií kapaliny. Tento přístup je obecnější a může být použit pro širší spektrum tekutin.

$$p_i = k(\frac{\rho_i}{\rho_0} - 1)$$

Kde p_i je tlak v částici i, k je uživatelem nastavený faktor a ρ_0 je klidová hustota. Takto vypočtený tlak můžeme dále využít pro výpočet tlakových sil v každé částici. Tlakovou sílu můžeme podle Navier-Stokesových rovnic vyjádřit jako záporný gradient tlaku. To znamená, že síla bude směřovat z oblastí s vysokým tlakem do tlakových nížin.

Výpočet tlakového gradientu může být hardwarově náročný, obzvláště pro velké množství částic. Pro zrychlení kalkulací, použijeme symetrický vzorec z rovnice (4). Dosazením do symetrického vzorce můžeme gradient tlaku vyčíslit.

$$\nabla p \approx \rho_i \sum_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) \nabla W_{ij}$$

Důležité je zmínit že tato metoda není perfektní a nezaručuje nulovou divergenci rychlostního pole a vyžaduje velice malé časové kroky pro přesné výsledky.

Celkově, kalkulace tlakových sil je důležitá součást simulace kapaliny. Protože určuje způsob, jakým se kapalina pohybuje a interaguje s okolním prostředím. Použitím přesné a

efektivní metody pro výpočet tlakových sil můžeme vytvořit realistické a vizuálně příjemné simulace.

2.3 Viskozita

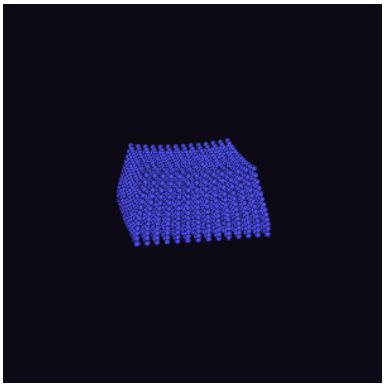
Viskózní síla vzniká třením mezi částicemi uvnitř tekutiny. Z Navier-Stokesových rovnic můžeme výpočet viskozity odvodit v následné podobě.

$$f_{visc} = \mu \nabla^2 v$$

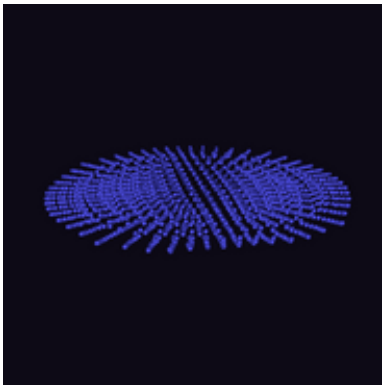
Kde Laplacian můžeme vypočítat diskretizací SPH, jenže to zavádí dva důležité problémy. První spočívá v náchylnosti tohoto výpočtu k částicovému zmatku a byl hlouběji analyzován v [8]. Druhým problémem je změna znaménka druhé derivace často volených vyhlazovacích jader uvnitř podpůrného poloměru. V práci [9] byla vytvořena nová metoda využívající derivace konečného rozdílu.

$$\nabla^2 v = 2(d+2) \sum_j \frac{m_j}{\rho_j} \frac{v_{ij} \cdot x_{ij}}{||x_{ij}||^2 + 0.01h^2} \nabla W_{ij}$$

Kde $x_i j = x_i - x_j$ a $v_i j = v_i - v_j$. Výraz $0.01h^2$ zabraňuje tvorbě singularit. Tato metoda pro výpočet Laplacianu pole rychlosti má navíc jednu velice pozitivní vlastnost a to, že pro částice pohybující se ve stejném směru se Laplacian přibližuje nule.



Obrázek 3: Simulace s vysokou viskozitou



Obrázek 4: Simulace s nízkou viskozitou

2.4 Povrchové napětí

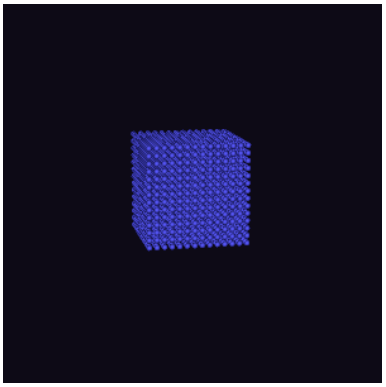
Síla povrchového napětí je zodpovědná za minimalizaci povrchu vůči objemu kapaliny. Vliv této síly můžeme pozorovat u jakékoliv kapaliny. Nejzřetelnější efekt mají tyto síly při pozorovaní úzkých kapilár, nebo kapek. Kapky vody se bez působení externích sil uskupují do tvaru koule. Pro výpočet povrchového napětí lze použít jednu ze dvou obecných metod.

Makroskopická metoda, představena v [10], spočívá v hledaní hraničních částic kapaliny a výpočtu povrchových normálů. Tento způsob skoro vždy využívá ve výpočtu Laplacianu, ovšem to je častá příčina numerické nestability.

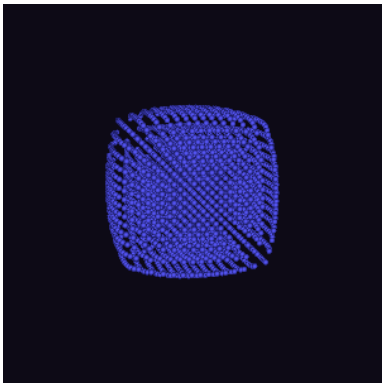
Upravená mikroskopická metoda navržená v [11] vychází z molekulární analýzy povrchového napětí. Povrchové napětí vzniká absencí kapalinných částic mimo tekutinu. Částice v kapalině působí slabou přitažlivou silou na ostatní částice. Tyto síly se uvnitř kapaliny vyruší, protože ze všech stran působí přibližně stejně velké síly. Na hranicích tekutiny zvenčí nepůsobí žádné síly, které by vyrušily síly působící na částici zevnitř okolní tekutiny. Metoda je vhodná pro vysoce dynamické povrchy kapalin.

$$\frac{dv_i}{dt} = -\frac{\kappa}{m_i} \sum_j m_j W_{ij}$$

Kde κ je uživatelem zadaný parametr, který škáluje působící sílu.



Obrázek 5: Simulace bez gravitace a povrchového napětí



Obrázek 6: Simulace bez gravitace s povrchovým napětím

3 Implementace

Začněme specifikací žádaného výsledku. Tedy hlavním cílem výsledné simulace je uplatnění nabytých znalostí a demonstrace implementace relativně jednoduché simulace kapaliny. Výsledná simulace by měla být stabilní, ale nemusíme klást důraz na přesnost simulace. Přirozeně cílíme na nejlepší dosažitelný výkon, ale nepočítáme s masivním rozšířením programu, takže se spokojíme s délkami výpočtu snímku v řádech sekund.

Ve výsledné simulaci využíváme vlastního programu pro vykreslovaní částic na obrazovku. Protože výše bylo zmíněno, že hlavním cílem je demonstrace znalostí. Takže využíváme vlastní program, ovšem soubory výsledných simulací mohou být přestrukturované pro další použití například v komerčně dostupných programech, jako například Blender. Novější verze programu pro některé výpočty využívají upravených funkcí z [12] implementace.

3.1 Způsob přechovávaní částic

V přiloženém programu byly implementovány dva způsoby ukládaní částic. Ukládaní částic je pravděpodobně nejlepší v poli. Ovšem chtěli bychom rozebrat způsob jakým jsou tato pole strukturovaná.

Ve starších verzích programu byly částice ukládány jako objekty v poli. Mřížový objekt následně přechovával pole částic a hašovací mapu, ve které klíč reprezentoval pozici v tabulce a hodnota k němu přiřazená reprezentovala surový ukazatel na objekt částice v poli.

Novější verze, inspirované implementací [12] ukládají pouze veličiny částic do příslušných polí. Každá veličina (pozice, hustota, tlak, atd.) má příslušné pole. Tato metoda sice trpí podobným problémem s mutabilitou. Ale je vhodnější pro časté přeskupovaní pole. Ukládaní částic tímto způsobem navíc zjednoduší přístup k jednotlivým veličinám. Protože funkce nepotřebuje mít přístup k celé částici, pouze jejímu indexu. Funkce tak může vybrat pouze hodnoty veličin, které potřebuje.

3.2 Konstrukce mříže

Konstrukce a následující aktualizace mříže je pravděpodobně nejzajímavější součástí programu. Aktualizaci můžeme rozdělit do dvou částí. V první části aktualizace potřebujeme spolehlivý způsob jak převést pozici částice na index v poli, nebo hašovací mapě.

Starší verze přiloženého programu využívaly jednoduchého vzorce $g_i = \frac{x_i}{h}$. Kde g_i je pozice buňky, ovšem tato implementace povolovala záporné hodnoty pro g_i. Záporné hodnoty se později ukazují jako problém při převodu na index pole. Starší verze byli vytvářené s úmyslem prakticky nekonečně velké mříže. Novější verze, opět inspirované [12] využívají malé modifikace tohoto vzorce $g_i = \frac{x_i}{h} - g_s$. Kde g_s je počátek tabulky. Touto modifikaci se zbavme nežádoucích záporných hodnot. Novější verze limituje velikost mříže. Navíc funkce pro konverzi pozice na index pole obsahuje kontrolu platnosti požadované pozice.

Dále potřebujeme převést pozici buňky na index vyhledatelný v poli. Zde se opět nabízí dvě možnosti. Nekonečně velká doména a hašovací mapa [6], nebo pokud máme pevně stanovenou velikost domény. Můžeme s mříží pracovat jako s multidimenzionálním polem, takže můžeme jednoduchým převodem $i = g_i.x * max.y * max.z + g_i.y * max.z + g_i.z$; převést vektor na číslo uvnitř tabulky.

Opakováním těchto dvou kroků pro všechny částice můžeme vytvořit pole, ve kterém každý index odpovídá indexu částice a hodnota odpovídá indexu buňky ve které se částice nachází. Při generaci toho pole, můžeme také vygenerovat podobné pole, ze kterého můžeme získat zpět počet částic v jakékoliv buňce.

Ve třetí nepovinné, znovu [12] inspirované části aktualizace můžeme seřadit částice ve stejných buňkách za sebe, pro rychlejší vyhledáváni sousedství. Algoritmus pro třídění částic implementovaný v přiloženém programu využívá předem připraveného seznamu z předchozího kroku. Pro každou buňku vypočítáme její vzdálenost začátku buňky od začátku buňky. Pro každou částici pak její index vypočítáme přičtením pozice v buňce ke vzdálenosti buňky od začátku pole.

Implementace také využívá modulárního systému pro aktualizaci veličin v sousedství částice, který byl navržen autory [12]

4 Příručka

Přiložený program buď spustíte pomocí předem kompilovaných .exe souborů. Nebo pokud pro vaši architekturu .exe soubor neexistuje můžete si program zkompilovat.

4.1 Kompilace

Abyste mohli program zkompilovat budete potřebovat programovací jazyk Rust. Rust nejjednodušeji stáhnete pomocí oficiálního návodu a nástroje rustup. Oficiální instalační návod naleznete na

```
https://www.rust-lang.org/learn/get-started
```

Po úspěšné instalaci Rustu se navigujte do složky programu. A spusťte příkaz .

```
cargo build --release
```

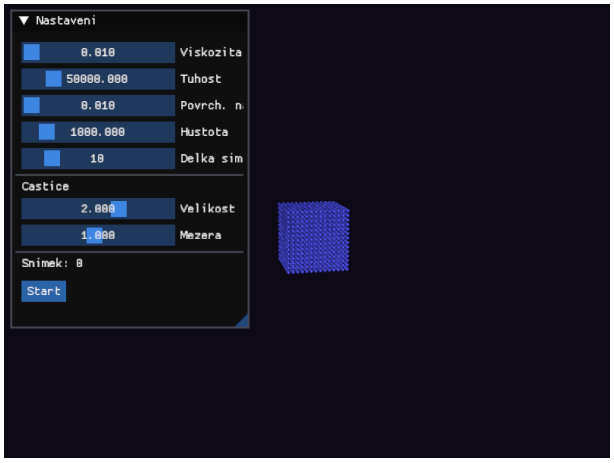
Zkompilované soubory naleznete ve složce target/release.

4.2 Výpočet simulace

Program zatím musí být spuštěn pokaždé z hlavní složky projektu. Přiložený program má dva módy. Základní mód slouží pro simulaci tekutin a druhý mód pro přehrávání simulace tekutin. Každá simulace je prováděna s počáteční krychlí o rozměrech $14x14x14$ částic.

Základní mód nabídne uživateli parametry pro simulaci tekutiny.

- **Viskozita**: ovlivňuje tekutost kapaliny (nedoporučuje se překračovat 1,5)
- **Tuhost**: určuje odpor kapaliny vůči změně stavu
- **Povrchové napětí**: škáluje vypočítanou velikost povrchového napětí
- **Hustota**: počáteční hustota
- **Délka simulace**: délka simulace v sekundách
- **Velikost**: velikost jedné simulační částice
- **Mezera**: počáteční velikost mezery mezi částicemi



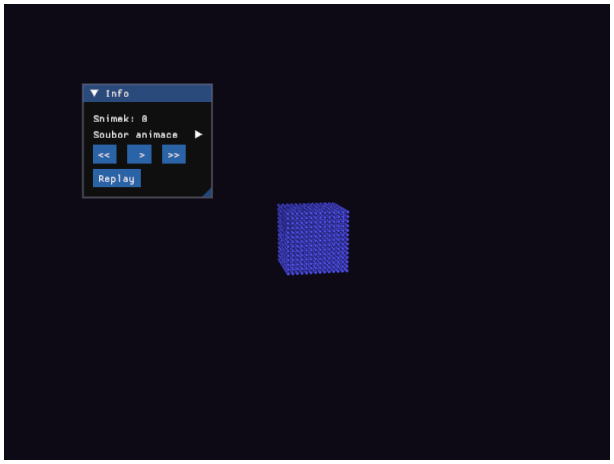
Obrázek 7: Uživatelské rozhraní základního módu

Dále uživatelské rozhraní zobrazuje aktuální snímek a tlačítko start. Po zapnutí simulace se provede simulace po dobu zadanou uživatelem a pozice částic v čase je zaznamenána do souboru. Po skončení simulace je uživateli nabídnuto simulaci restartovat.

4.3 Přehrání simulace

Druhý mód nahraje simulace umístěné v současné složce a zobrazí uživateli nabídku dostupných simulací. Pokud uživatel simulaci spustí a žádný soubor nevybral je spuštěna simulace v souboru s názvem simulation.nk. Uživatelské rozhraní zobrazuje aktuální snímek, nabídku simulací a ovládání simulace.

- «: vrátí simulaci 5 snímků zpět
- ||, >: spustí / zastaví simulaci
- »: posune simulaci 5 snímků vpřed
- **start, replay**: spustí simulaci od začátku



Obrázek 8: Uživatelské rozhraní přehrávače

Závěr

Simulace tekutin je obsáhlé téma. Tekutiny jsou v základu rozdělené podle typu metody použité pro diskretizaci prostředí. Metody jsou Eulerova a Lagrangeova. Z Lagrangeovy metody vychází SPH metoda. SPH nabývá na popularitě, díky pokroku technologií a komerčnímu využití simulací tekutin. SPH totiž nabízí relativně přesné a rychlé simulace. Tyto vlastnosti jsou rozhodující pro rozšíření v zábavním průmyslu.

Simulace založené na metodě SPH jsou vysoce flexibilní a existuje hned několik řešení většiny problémů. Simulace potřebuje spolehlivý způsob jak vyhledávat sousední částice. Dále simulace potřebuje implementovat dostačující způsob pro výpočet tlaku a dodržení typické vlastnosti tekutin, nestlačitelnosti. Simulace také může uplatnit další podmínky a vlastnosti různých kapalin.

Seznam použité literatury

1. KOSCHIER, Dan; BENDER, Jan; SOLENTHALER, Barbara; TESCHNER, Matthias. Smoothed Particle Hydrodynamics Techniques for the Physics Based Simulation of Fluids and Solids. *Eurographics 2019 - Tutorials*. 2019. Available from DOI: 10.2312/EGT.20191035.
2. BRIDSON, Robert; FEDKIW, Ronald; MÜLLER-FISCHER, Matthias. Fluid simulation. *ACM SIGGRAPH 2006 Course notes*. 2006. Available from DOI: 10.1145/1185657.1185730.
3. LUCY, L. B. A numerical approach to the testing of the fission hypothesis. *The Astronomical Journal*. 1977, roč. 82, p. 1013. Available from DOI: 10.1086/112164.
4. GINGOLD, R. A.; MONAGHAN, J. J. Smoothed particle hydrodynamics: Theory and application to non-spherical stars. *Monthly Notices of the Royal Astronomical Society*. 1977, roč. 181, č. 3, pp. 375–389. Available from DOI: 10.1093/mnras/181.3.375.
5. PRICE, Daniel J. Smoothed particle hydrodynamics and magnetohydrodynamics. *Journal of Computational Physics*. 2012, roč. 231, č. 3, pp. 759–794. ISSN 0021-9991. Available from DOI: https://doi.org/10.1016/j.jcp.2010.12.011. Special Issue: Computational Plasma Physics.
6. IHMSEN, Markus; AKINCI, Nadir; BECKER, Markus; TESCHNER, Matthias. A parallel SPH implementation on multi-core cpus. *Computer Graphics Forum*. 2010, roč. 30, č. 1, pp. 99–112. Available from DOI: 10.1111/j.1467-8659.2010.01832.x.
7. DESBRUN, Mathieu; CANI, Marie-Paule. Smoothed Particles: A new paradigm for animating highly deformable bodies. In: BOULIC, Ronan; HEGRON, Gerard (ed.). *Eurographics Workshop on Computer Animation and Simulation (EGCAS)*. Poitiers, France: Springer-Verlag, 1996, pp. 61–76. Available also from: https://hal.inria.fr/inria-00537534. Published under the name Marie-Paule Gascuel.
8. PEER, Andreas; IHMSEN, Markus; CORNELIS, Jens; TESCHNER, Matthias. An Implicit Viscosity Formulation for SPH Fluids. *ACM Trans. Graph.* 2015, roč. 34, č. 4. ISSN 0730-0301. Available from DOI: 10.1145/2766925.
9. MONAGHAN, J J. Smoothed particle hydrodynamics. *Reports on Progress in Physics*. 2005, roč. 68, č. 8, pp. 1738–1744. Available from DOI: 10.1088/0034-4885/68/8/R01.

10. AKINCI, Nadir; AKINCI, Gizem; TESCHNER, Matthias. Versatile Surface Tension and Adhesion for SPH Fluids. *ACM Trans. Graph.* 2013, roč. 32, č. 6. ISSN 0730-0301. Available from DOI: 10.1145/2508363.2508395.

11. BECKER, Markus; TESCHNER, Matthias. Weakly Compressible SPH for Free Surface Flows. In: *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. San Diego, California: Eurographics Association, 2007, pp. 209–217. SCA '07. ISBN 9781595936240.

12. ERIZMR. *Erizmr/Sph_taichi: A high-performance implementation of SPH in Taichi.* 2020. Available also from: https://github.com/erizmr/SPH_Taichi.