Symulacja rozprzestrzeniania się dymu w pomieszczeniu

Akademia Górniczo-Hutnicza w Krakowie
Wydział Informatyki, Elektroniki i Telekomunikacji

Projekt wykonany na przedmiot Modelowanie Systemów Dyskretnych

Martyna Paliwoda Martyna Olszewska Jakub Domogała Jolanta Śliwa

2021/2022

1. Wstęp

Podjęliśmy się zadania zaprojektowania oraz zamodelowania zjawiska rozprzestrzeniania się dymu w zamkniętym pomieszczeniu. Nasz model będzie uwzględniać czynniki wpływające na to zjawisko fizyczne, między innymi temperaturę czy kierunki ruchu powietrza.
Głównym zadaniem było zaimplementowanie modelu oraz przedstawienie jego wyników w symulacji 3D.

2. Harmonogram prac

Data	**Cel**
4.05.22r.	**Wybranie środowiska pracy oraz opracowanie teorii**
11.05.22r.	**Wstępna implementacja mechaniki symulacji**
18.05.22r.	**Dopracowanie implementacji**
25.05.22r.	**Wizualizacja**
1.06.22r.	**Porównanie danych z publikacjami**
8.06.22r.	**Końcowe korekty i prezentacja**

3. Literatura

1. Numerical Simulation of Fire Smoke Spread in a Super High-Rise Building for Different Fire Scenarios
2. Visual Simulation of Smoke
3. **Real-time Simulation of Smoke Using Graphics Hardware** - Marinus Rørbech
4. Numerical Analysis of Smoke Spreading in a Medium-High Building under Different Ventilation Conditions - Zdzislaw Salamonowicz, Malgorzata Majder-Lopatka, Anna Dmochowska, Aleksandra Piechota-Polanczyk and Andrzej Polanczyk
5. Numerical simulations of outdoor wind effects on smoke spreading along a corridor: Physical and sensitivity analysis A.J. Wang, B. Manescau, Quentin Serra, K. Chetehouna, Éric Florentin, C. de Izarra

4. Teoria

Main source, mathematical and physical formulas from: ***Real-time Simulation of Smoke Using Graphics Hardware*** *-* ***Marinus Rørbech***

Nomenclature

d	Density field
$\mathbf{f}$	Force field
h	Spatial discretization
$\mathbf{N}$	Vorticity location field
p	Pressure field
T	Temperature field
T_0	Mean temperature
t	Time
$\mathbf{u}$	Velocity field
β	Thermal expansion coefficient
ε	Vorticity confinement control factor
η	Gradient field of vorticity magnitude
ω	Vorticity
∇	Gradient operator $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z})$
ν	Kinematic viscosity
ρ	Density

The motion of a non-turbulent, viscous fluid is modeled by the incompressible Navier-Stokes equations

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

$$\nabla\cdot\mathbf{u} = 0 \quad (2)$$

where (1) describe the change in the fluid velocity, and (2) enforces conservation of mass.
To make up for some of the small-scale detail thus missing, a vorticity confinement force:

$$\mathbf{f}_{vc} = h\,\varepsilon(\mathbf{N}\times\omega)$$

,where

$$\omega = \nabla\times\mathbf{u} \qquad \mathbf{N} = \frac{\eta}{|\eta|}, \quad \eta = \nabla|\omega|$$

Derived from a simple thermal buoyancy force, given by:

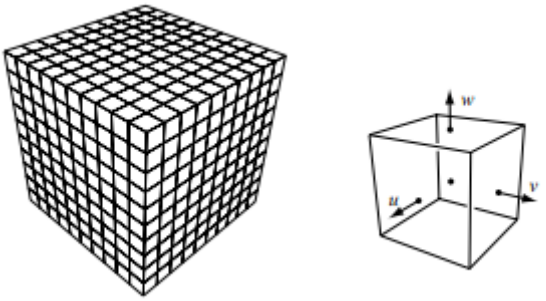
$$\mathbf{f}_T = \beta\,(T - T_0)\cdot(0,1,0)^T$$

Density movement:

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

Extended theory consistent with the previous source:
https://web.stanford.edu/class/cs237d/smoke.pdf

- a more detailed, explanation of the above formulas,
- their derivation,
- expected results.



Initial values needed to use the above formulas.: ***Numerical Analysis of Smoke Spreading in a Medium-High Building under Different Ventilation Conditions*** *- Zdzislaw Salamonowicz, Malgorzata Majder-Lopatka, Anna Dmochowska, Aleksandra Piechota-Polanczyk and Andrzej Polanczy*

5. Realizacja

5.1. Założenia projektu

Naszym założeniem było skupienie się bardziej na części implementacyjnej modelu, więc z tego powodu stwierdziliśmy, że wizualizacja symulacji nie będzie jej najważniejszą częścią, głównie ze względu na czas jaki możemy poświęcić na wykonanie całego projektu.
Na początku mieliśmy kilka pomysłów na jego realizację, jednym z nich było zrealizowanie naszej symulacji za pomocą blendera. Jednak odeszliśmy od tego pomysłu, ze względu na trudności jakie mogłyby się pojawić czy łączeniu naszej implementacji z wizualizacją w tym programie. Po przedyskutowaniu jeszcze kilku innych pomysłów, postanowiliśmy zostać przy zaimplementowaniu silnika symulacji w Javie oraz wizualizacji z pomocą biblioteki Swing. Wzorowaliśmy się w pewnym stopniu na wizualizacjach programów z laboratoriów, które wykonywaliśmy przez cały semestr.

5.2. Implementacja

5.2.1. Solver

W klasie solver mamy cały kod, odpowiedzialny za implementację funkcjonalności modelu. Klasa ta posiada pola, które przechowują informacje między innymi o źródłach dymu, pomieszczeniu oraz przeszkodach w jego wnętrzu. Posiada pole, którym jest siatka 3D czyli podstawa do obliczeń.

Mamy także metody odpowiedzialne za ustawienie źródeł dymu (które można ustawić na wygaśnięcie po ustalonej liczbie iteracji), stworzenie przeszkód. Metoda, która uruchamia kolejno metodę diffuse oraz wind.
W metodzie wind zawieramy również mechanikę grawitacji ze względu na podobieństwo ich oddziaływań dla naszych założeń.

W klasie są zawarte również metody, które odpowiadają za część związaną z wizualizacją i resetowaniem naszej siatki.

Ostatnią rzeczą jaką dodaliśmy do klasy było umożliwienie obliczania podkroków (substeps) dla zwiększenia precyzji obliczeń oraz aby zapobiec defektom symulacji dla np większej siły wiatru lub większej rozszerzalności dymu.

5.2.2. Point

Klasa reprezentuje pojedynczy punkt w przestrzeni. Obiekty posiadają pola, które przechowują jego sąsiadów, informacje czy punkt jest barierą, źródłem lub żadnym z nich oraz wartość gęstości z tej i z poprzedniej iteracji.

5.2.3. Grid3D

Obiekty tej klasy przechowują naszą siatkę oraz jej parametry. Mamy metodę inicjalizującą, która w każdej komórce umieszcza obiekt - Point oraz ustawia wartości początkowe. Każdemu punktowi przypisujemy też jego sąsiadów. Znajdują się tam też metody, odpowiedzialne za rysowanie siatki oraz aktualizowanie.

5.2.4. GUI

Klasa ta odpowiada głównie za stworzenie gui, czyli za funkcjonalność guzików, wyświetlanie siatki symulacji, zmienianie poziomu, który aktualnie widzimy, zatrzymywanie i restartowanie.

5.3. Wizualizacja

Część z wizualizacją nie była naszym priorytetem, zatem zdecydowaliśmy się na pokazanie obrazu przecięcia siatki 3D z dwóch perspektyw. Na lewej siatce mamy widok od góry na prawej zaś od boku. Możemy zmieniać poziomy na jakich chcemy oglądać symulacje, a także ją zatrzymywać. Symulacje można realizować na 6 różnych sposobów ze względu na możliwość wyłączania oddzielnie każdego z wymienionych części algorytmu (diffuse, wind, gravity).
Można również przełączyć solver na poprzednią wersję naszej implementacji jednak najdokładniejsze efekty udało nam się otrzymać używając wersji zaindeksowanej jako 2.

Zdjęcie 1. Okienko programu do wizualizacji zaimplementowanej symulacji

6. Testowanie

W ramach testów uruchomiliśmy symulację dla różnej liczby źródeł, przeszkód, sił wiatru oraz różnej rozszerzalności dymu, testowaliśmy czy wiatr gromadzi się w rogach pomieszczenia tak jak przewiduje to nasz model oraz jak zachowuje się napotykając przeszkody na swojej drodzę. Dla niskich wartości rozszerzalności dymu oraz niskiej siły wiatru wszystko działa poprawnie, przy większych wartościach potrzebne jest jednak ustawienie większej ilości kroków pomiędzy każdą iteracją dla zachowania płynności symulacji. Zbyt duża ilość kroków między iteracjami nie jest jednak wskazana ze względu na dużej zwiększenie kompleksowości obliczeniowej.

7. Wnioski

W trakcie pisania modelu przekonaliśmy się jak ciężko jest napisać silnik w taki sposób aby nie tworzył nieprzewidywalnych interakcji oraz zachował sumaryczną ilość dymu w trakcie kolejnych kroków. Napotykanie barier również sprawiło wiele problemów jednak ostatecznie

udało nam się zaimplementować działający model poprawnie wizualizujący następujące zmiany dla założonych warunków.