Symulacja rozprzestrzeniania zanieczyszczeń
Symulacje Dyskretne Systemów Złożonych

Wstęp teoretyczny

Krzysztof Żywiecki Szymon Duda Piotr Krześniak

Semestr letni 2020

1 Cel projektu

Celem projektu było stworzenie programu do przeprowadzenia wizualizacji rozprzestrzeniania zanieczyszczeń w środowisku. Za cel zespół przyjął stworzenie programu który symulowałby proste zjawiska fizyczne takie jak wiatr i dyfuzja w otoczeniu, oraz wykazywał się dużą szybkością.

2 Działanie algorytmu

Program działa na bazie automatu komórkowego. Każda komórka przechowuje swoje zanieczyszczenia (komórki mają wielkość jednego metra kwadratowego), które następnie ulegają rozprzestrzenieniu do innych komórek.

Dane wejściowe są na stałe zapisane w programie. Niestety aktualizowanie danych w trakcie działania programu okazało się zbyt dużym wyzwaniem przy użytej technologii. Program do działania wymaga zaprogramowania

- Dwuwymiarowej tablicy o wielkości równej ilości komórek z liczbami zmiennoprzecinkowymi reprezentującymi ilość wpływających zanieczyszczeń w arbitralnych jednostkach ilości na metr kwadratowy w czasie jednego pełnego kroku symulacji.
- Tablica dowolnej wielkości z wektorami o długości maksymalnie 1 przechowującymi prędkość wiatru.
- Ilość próbek określających jakość symulowania dyfuzji, oraz mnożnik prędkości wiatru.

2.1 Dyfuzja

Do symulowania dyfuzji zanieczyszczeń w otoczeniu program liczy średnią ważoną wszystkich sąsiadów komórki, a następnie miesza ją z obecnym stanem komórki na podstawie ilości próbek symulacji.

Listing 1: Pseudokod przedstawiający liczenie dyfuzji w komórce

```
sum = 0
for neighbor in neighbors:
    if neighbor is diagonal:
        sum = sum + neighbor * 0.5
    else
        sum = sum + neighbor
sum = sum + state    //uwzglednij stan badanej komorki w sredniej
average = sum / (4.0 + 4.0 * 0.5 + 1.0) // sum / 7.0
```

W kodzie zmienna state oznacza stan aktualnie badanej komórki.

2.2 Wiatr

Do liczenia wiatru pobierany jest wektor wiatru w badanej komórce, a następnie po kolei badany jest iloczyn wektorowy wiatru oraz wektorów łączących sąsiadów z badaną komórką. Następnie, program odejmuje od stanu zanieczyszczenia proporcjonalnie do prędkości wiatru, i dodaje zanieczyszczenia komórki najbardziej po przeciwnej stronie wektora wiatru, określonej na podstawie wyliczonego iloczynu wektorowego. Ta metoda jest lepsza niż poprzednio użyta, gdyż nie generuje dodatkowych zanieczyszczeń. Powoduje ona jednak że w symulacji wiatr działa na komórki tylko jeśli jest skierowany zgodnie z osiami x/y, albo jest pod kątem 45^o od nich.

Po skończeniu wyliczeń dyfuzji i wiatru, ostateczny stan komórki liczy się jako

$$r = (state \cdot (1 - (1/totalSteps)))+$$

$$(diffusion + inflow) \cdot (1/totalSteps) + balance/diffusionStepCount$$

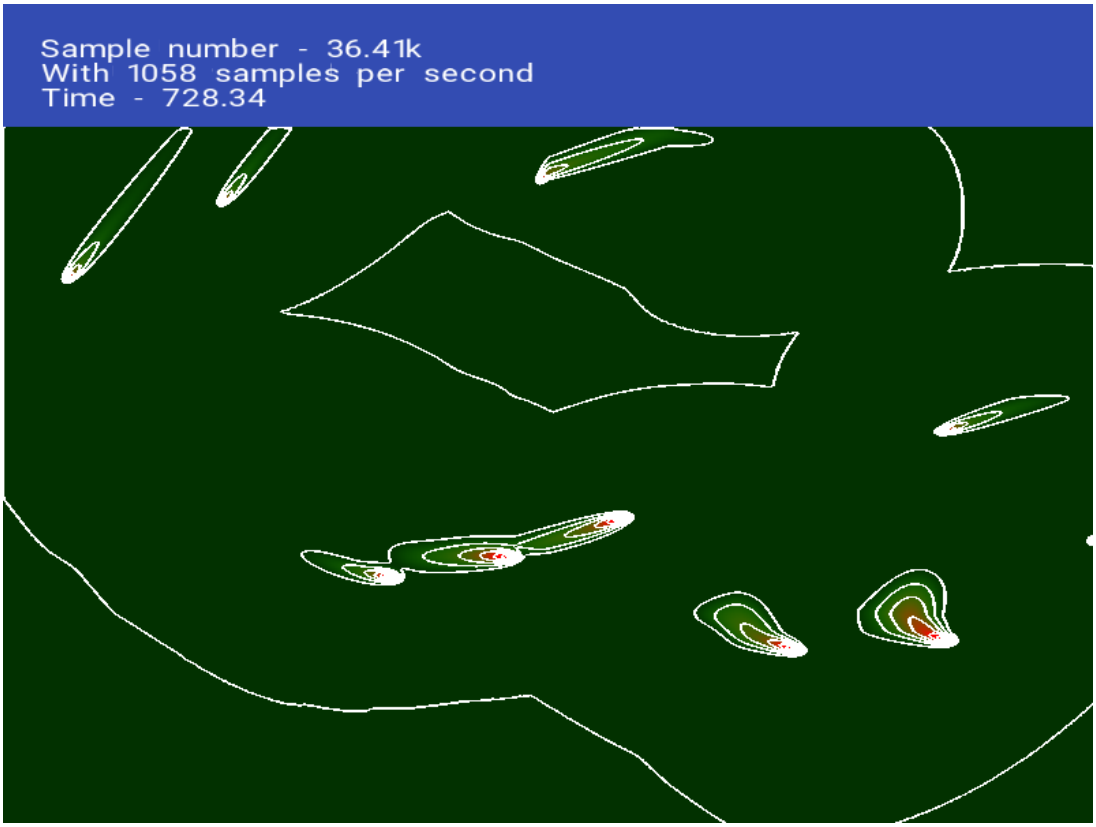
Gdzie $totalSteps$ oznacza liczbę ilość próbek symulacji dyfuzji przemnożoną przez mnożnik prędkości wiatru.

Aplikacja wyświetla dodatkowo bardzo podstawowe informacje: ilość iteracji, czas, czyli ilość iteracji podzieloną przez ilość próbek, oraz ilość iteracji na sekundę.

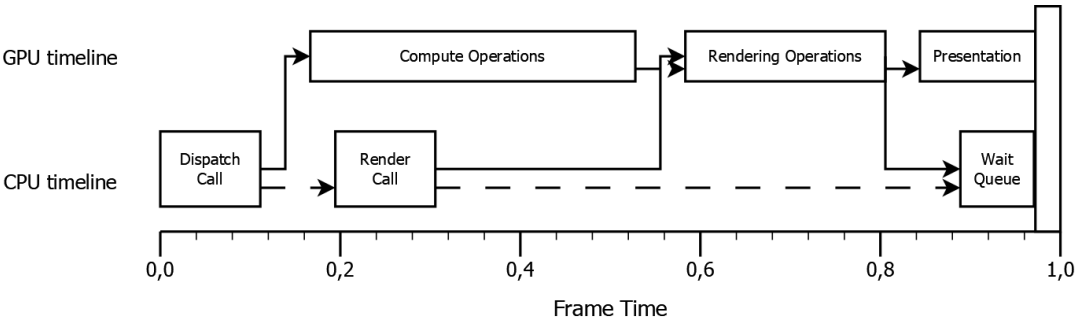
3 Wydajność

Aplikacja do działania wykorzystuje standard Vulkan. Użycie go pozwala na wykorzystanie w liczeniu kartę graficzną, która jest doskonałym wyborem dla symulowania automatu komórkowego. Zrzut ekranu aplikacji wykonano na platformie z kartą graficzną Nvidia GTX 1050ti, która pozwoliła na średnio 1000 iteracji na sekundę na mapie o wielkości 1024 na 1024 komórki.

Wydajność osiągnięta w testach jest jednak w pewien sposób wynikiem kompromisów, takich jak statyczność danych wejściowych. Dynamiczna zmiana zawartości pamięci jest możliwa, ale jej implementacja wykracza poza zdolności programistyczne zespołu.



Rysunek 1: Zrzut ekranu przedstawiający rezultat działania aplikacji po około minucie. Ciemnozielone obszary oznaczają brak zanieczyszczeń, a jasnoczerwone oznaczają stężenie > 1. Białe linie rysowane są co 0.2 poziomy.



Rysunek 2: Uogólniony rozkład czasu operacji wykonywanych przez procesor oraz kartę graficzną. Compute Operations reprezentuje czas niezbędny na zaktualizowanie mapy, a Rendering Operations obejmuje wszystkie operacje graficzne, w tym rysowanie mapy i wyświetlanie tekstu

Literatura

[1] http://courses.washington.edu/cewa567/Plumes.PDF

[2] https://personalpages.manchester.ac.uk/staff/paul.connolly/teaching/practicals/gaussian_plume_modelling.html