STUDIA INFORMATICA 2007
Volume 28 Number 1 (70)

Jakub GRUDZIŃSKI, Adrian DĘBOWSKI, Tomasz GRUDZIŃSKI
Politechnika Śląska, Instytut Informatyki

ZASTOSOWANIE SYSTEMÓW CZĄSTECZKOWYCH W SYMULACJI I WIZUALIZACJI ZJAWISK ATMOSFERYCZNYCH W CZASIE RZECZYWISTYM W ŚRODOWISKU FRS

Streszczenie. Niniejsza publikacja jest kontynuacją tematu publikacji „Symulacja pogody w czasie rzeczywistym w środowisku FRS" [3]. Omówiono w niej system uproszczonej symulacji oraz wizualizacji zjawisk atmosferycznych, pracujący w czasie rzeczywistym, ze szczególnym naciskiem na zjawiska, takie jak: opady czy tornada.

Słowa kluczowe: atmosfera, chmury, wirtualna rzeczywistość, efekty cząsteczkowe

THE USE OF PARTICLE SYSTEMS IN REAL-TIME ATMOSPHERIC PHENOMENA SIMULATION AND RENDERING IN FRS ENGINE

Summary. The paper is the continuation of the subject of the "Real-Time Atmospheric Phenomena Simulation in FRS Engine" [3] publication. A real-time simplified atmospheric phenomena simulation and rendering system is presented with emphasis put on such phenomena as tornadoes and precipitation.

Keywords: atmosphere, clouds, virtual reality, particle effects

1. Środowisko FRS

FRS (ang. *Flexible Reality Simulation*) jest silnikiem programistycznym, pozwalającym na łatwe pisanie aplikacji opartych na wirtualnej rzeczywistości. Silnik jest obecnie intensywnie rozwijany i nie wszystkie założenia zostały spełnione, jednak osiągnięto już pewną

funkcjonalność pozwalającą na zaprezentowanie prac szerszej publiczności i wyciągnięcie pewnych wniosków[1].

Twórcy FRS postawili sobie za główny cel stworzenie silnika uniwersalnego i łatwego w użyciu, który poza standardowymi elementami podobnych projektów zawierałby komponenty bardziej wyspecjalizowane, możliwe do użycia, lecz nieobowiązkowe dla piszącego aplikację bazującą na silniku. W ten sposób powstaje silnik, który poza możliwością wyświetlania trójwymiarowej grafiki, interakcją między obiektami i sprawną obsługą dźwięku, umożliwi także symulację pogody, programowe generowanie modeli roślinności, symulację mimiki ludzkiej twarzy, użycie zaawansowanych algorytmów sztucznej inteligencji oraz sterowanie całością poprzez oferujący dużą funkcjonalność język skryptowy.

Jednym z komponentów silnika jest komponent realizujący matematyczny model służący do symulacji zjawisk atmosferycznych. Został on opisany w publikacji „Symulacja pogody w czasie rzeczywistym w środowisku FRS" [3]. W dalszej części publikacji skupiono się na zaprezentowaniu kolejnych prac nad komponentem oraz omówieniu pewnych zagadnień związanych z renderingiem chmur i innych efektów atmosferycznych.

2. Symulacja i rendering zjawisk pogodowych a systemy cząsteczkowe

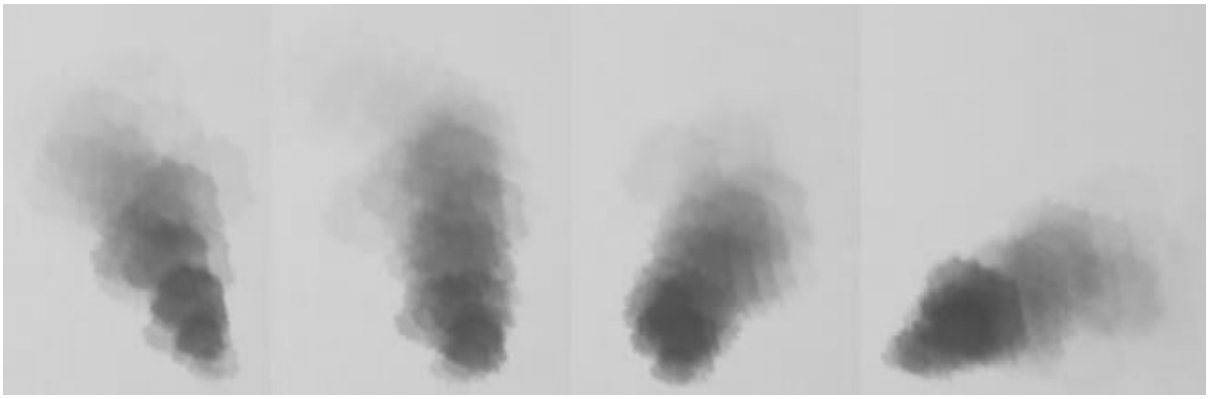
Omawiany system symulacji pogody pracuje opierając się na cząsteczkach [3]. Zastosowanie cząsteczek jest najbardziej naturalną metodą generowania danych dla wizualizacji wszelkich trójwymiarowych obiektów, których kształt jest zmienny w czasie. Dotyczy to przede wszystkim gazów i cieczy (typowe zastosowania to dym, chmury, eksplozje, strumień wody etc.). Zastosowanie reprezentacji siatkowej [6] jest w przypadku takich obiektów znacznie trudniejsze.

Rozróżnia się dwa podstawowe rodzaje systemów cząsteczkowych, w zależności od tego, czy cząsteczki są zależne od swoich sąsiadów czy też nie [7]. W pierwszym przypadku przestrzeń dzielona jest na trójwymiarową[2] siatkę, gdzie każda cząsteczka odpowiada jednej komórce. Tego typu systemy stosuje się do precyzyjnego modelowania zjawisk fizycznych, w tym również powstawania i zaniku chmur [8]. Duża dokładność pociąga za sobą jednak dużą złożoność obliczeniową. Druga metoda nie uwzględnia wzajemnych zależności cząsteczek – każda z nich posiada pewne parametry (szybkość, czas życia czy trajektoria ruchu). Przestrzeń nie jest reprezentowana przez siatkę, lecz jest ciągła. Umożliwia to generowanie

[1] Różne zagadnienia związane z silnikiem FRS opisano w [1, 2, 3, 4, 5].

[2] W niektórych zastosowaniach również dwu- (modelowanie ruchu ubrań) lub jednowymiarową (sprężyny etc.).

niemalże dowolnych efektów cząsteczkowych, opartych na znacznie uproszczonym aparacie matematycznym. Symulacja nie jest więc dokładna, ale cechuje się znacznie mniejszą złożonością obliczeniową. Przykładowy efekt osiągnięty tą metodą przedstawiono na rys. 1.



Rys. 1. Efekt dymu zrealizowany z użyciem systemu cząsteczkowego. Płynne zmiany kierunku ruchu snopa dymu zrealizowane są za pomocą prostych szumów
Fig. 1. The effect of smoke implemented with the use of a particle system. Fluent changes in the direction of the fume are realized by means of a simple noise

Istotą obu rodzajów systemów cząsteczkowych jest oddzielenie warstwy logicznej od warstwy renderingu. Niezależnie od obliczeń, jakie wykonywane są „wewnątrz" systemu, dane dla silnika wyświetlającego są identyczne, dzięki czemu oddzielny komponent może wyświetlać w taki sam sposób różne efekty. Zastosowany sposób wyświetlania ma przy tym bardzo duży wpływ na wydajność – nie wszystkie algorytmy pozwalają na rendering w czasie rzeczywistym, podczas gdy inne nie dają realistycznych rezultatów.

Większość modeli matematycznych, symulujących powstawanie, rozwój i zanik chmur, oparta jest na systemach z podziałem przestrzeni, przy czym zależności między sąsiednimi komórkami realizowane są albo na podstawie równania *Naviera-Stokesa* [8], albo prostych funkcji logicznych [9]. Omawiany w niniejszej publikacji oraz w [3] model bazuje natomiast na drugim rodzaju systemów cząsteczkowych, czyli ciągłej przestrzeni i cząsteczkach niezależnych od siebie. Szczegółowy opis modelu można znaleźć w [3] oraz [5], natomiast w dalszych rozdziałach przedstawiono dodatkową funkcjonalność systemu, wprowadzoną w ciągu ostatniego roku, oraz wybrane zagadnienia związane z renderingiem chmur i efektów pogodowych.

3. Opady i wyładowania atmosferyczne

Oczywiste jest, że intensywność opadów atmosferycznych zależy od grubości pokrywy chmur. Zależność ta nie jest jednak liniowa, co wynika z wielu innych uwarunkowań i zależności, z których najważniejszy jest kierunek ruchu powietrza. Często zdarza się, że mimo

ciemnych, burzowych chmur, opady są bardzo niewielkie bądź nie ma ich wcale. Jest to związane z silnymi prądami wstępującymi, tworzącymi się wewnątrz przednich części chmur burzowych [22], które wynoszą krople wody i kryształki lodu na bardzo dużą wysokość. Bardzo silne i gwałtowne opady występują natomiast w środkowej i tylnej części chmury, gdzie dominują prądy zstępujące.

W omawianym modelu łatwo określić zarówno grubość pokrywy chmur, jak i lokalizację prądów wstępujących. Ponieważ każda cząsteczka w danej chwili znajduje się na określonej pozycji w przestrzeni, więc grubość pokrywy chmur estymowana jest przez rzut cząsteczek na płaszczyznę poziomą[3]. Prądy wstępujące występują natomiast tam, gdzie funkcja generacji chmur [3, 5] ma bardzo dużą wartość. Od otrzymanego w omówiony sposób pola skalarnego odejmuje się więc wartość funkcji (z odpowiednim współczynnikiem), otrzymując w ten sposób kolejną macierzową funkcję. Opady są wprost proporcjonalne do jej wartości, z wyjątkiem obszarów, gdzie przyjmuje ona wartości ujemne (w tych rejonach opady nie występują).

Wyładowania atmosferyczne również powstają tam, gdzie grubość pokrywy chmur jest największa. Podobnie jak w przypadku opadów, nie jest to jednak jedyny czynnik wpływający na częstotliwość ich występowania. Wyładowania pojawiają się niemalże tylko i wyłącznie w przypadku chmur burzowych[4]. Jak pokazano w [3, 5], w omawianym modelu rodzaje chmur pojawiających się na niebie zależą od współczynnika równowagi w atmosferze (w meteorologii rozróżnia się równowagę stałą oraz chwiejną, *vide* [22]). Symulacja wyładowań atmosferycznych jest więc równie prosta, jak opadów – błyskawice generowane są losowo z rozkładem zależnym od grubości pokrywy chmur i wartości współczynnika równowagi. Mechanizm ten zapewnia pojawianie się wyładowań niemalże tylko w przypadku chmur burzowych, co daje realistyczne efekty przy znikomym obciążeniu obliczeniowym.

Osobnym zagadnieniem jest wizualizacja wyładowań i opadów. W przypadku opadów możliwe rozwiązania przedstawiono w [4] oraz [19]. Wyładowania mogą być wyświetlane jako niezależne efekty cząsteczkowe, np. w postaci pojedynczej cząsteczki z teksturą wybraną losowo z zestawu przygotowanych wcześniej tekstur. Aby efekt był realistyczny, cząsteczce musi towarzyszyć bardzo silne źródło światła, natomiast cały system renderingu powinien uwzględniać algorytm HDR (*High Dynamic Range rendering* lub *High Dynamic Range lighting*, *vide* [20]).

[3] W aktualnej implementacji, przy rozmiarze pojedynczej komórki uzyskanego w ten sposób pola skalarnego rzędu 200 m, grubość pokrywy chmur w danej komórce szacowana jest poprzez średnią ważoną wartości tej komórki oraz komórek sąsiednich, ponieważ pole wykazuje zdecydowanie zbyt duży rozrzut.

[4] Mechanizm powstawania wyładowań atmosferycznych jest dość złożony, natomiast z punktu widzenia uproszczonego modelu na potrzeby symulacji w czasie rzeczywistym jest nieistotny, toteż w niniejszej publikacji nie został uwzględniony.

4. Ruch wirowy chmur

Cyrkulacja mas powietrza w skali makro związana jest z układami barycznymi (wyżami i niżami [22]). Przemieszczające się z wyżu do niżu powietrze zostaje wprawione w ruch wirowy, wynikający z działania siły Coriolisa. Niemniej jednak w mniejszej skali również może wystąpić ruch wirowy – dotyczy to największych chmur burzowych, występujących w ziemskiej atmosferze, tzn. **superkomórek** [22]. Ruch ten stosunkowo łatwo dostrzec, ponieważ chmura przyjmuje dzięki niemu dość regularny kształt.

W omawianym modelu ruch wirowy w przypadku dużych chmur zrealizowany jest przez spiralne skręcenie trajektorii ruchu cząsteczek (zagadnienie ruchu cząsteczek zostało szczegółowo omówione w [3] oraz [5]). Pozornie takie rozwiązanie może spowodować, że wszystkie chmury będą wirować (co byłoby efektem niepożądanym), nie jest to jednak prawdą, ponieważ w przypadku mniejszych chmur trajektorie cząsteczek są niemalże pionowe, przez co ruch wirowy jest pomijalnie mały. W przypadku wysokich chmur, jak *cumulonimbus* [22], w końcowej fazie „życia" cząsteczki poruszają się poziomo – skręcenie trajektorii ich ruchu względem osi pionowej powoduje więc wyraźny ruch wirowy w górnej części chmury.

Zastosowane rozwiązanie jest bardzo proste i nie wymaga skomplikowanych obliczeń, dzięki czemu nie wpływa znacząco na złożoność obliczeniową całego modelu. Niemniej jednak nie jest doskonałe, ponieważ zapewnia ruch wirowy tylko w górnych partiach chmur burzowych, podczas, gdy w dolnych jest on niezauważalny (w rzeczywistości ruch wirowy występuje na każdej wysokości).

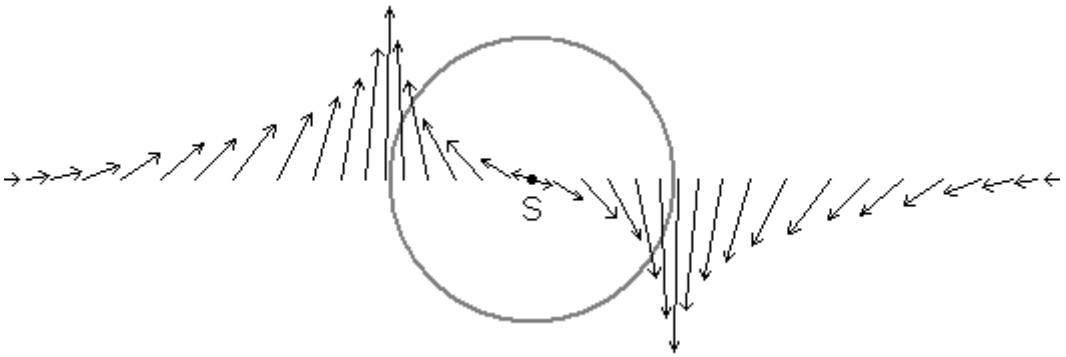
5. Tornada

Tornada nie są zjawiskiem powszechnym, występują sporadycznie w przypadku wyjątkowo gwałtownych burz. Niemniej jednak meteorolodzy poświęcają wiele czasu i środków na ich badania, ponieważ powodują wyjątkowo duże straty materialne, a niejednokrotnie doprowadzają do wypadków śmiertelnych. Uproszczona symulacja tornad, np. na potrzeby programów prezentacyjnych, może więc być bardzo przydatna.

W omawianym modelu, w każdej klatce symulacji cząsteczki zmieniają położenie w związku z ruchem wynikającym z ich trajektorii i prędkości [3, 5] oraz zostają przesunięte o odpowiednie wektory wynikające z wiatru (wiatr jest przedstawiony w postaci dwu-wymiarowego pola wektorowego, *vide* [3, 5]). Składowa wynikająca z trajektorii odpowiada głównie za przemieszczanie się cząsteczek w pionie, natomiast składowa wynikająca z wiatru – w płaszczyźnie poziomej. Prędkość cząsteczek w pionie jest wprost proporcjonalna do

wartości **funkcji generacji chmur** [3, 5], która wyznacza częstotliwość ich pojawiania się, a tym samym grubość pokrywy chmur[5]. Cząsteczka powstająca w obrębie tornada porusza się więc z dużą prędkością (tornada powstają tylko w przypadku bardzo dużych chmur) do góry, co jest zgodne z rzeczywistością. Aby jednak tornado powstało, należy również uwzględnić intensywny ruch wirowy w płaszczyźnie poziomej.

W [3, 5] przedstawiono mechanizm wpływu tworzenia się cząsteczek na pole wektorowe wiatru. Poruszające się do góry cząsteczki odpowiadają prądom wstępującym [22], które powodują „zasysanie" powietrza pod chmurą, skierowując wiatr w jej stronę. Przedstawione rozwiązanie nie zapewnia jednak możliwości zaistnienia ruchu wirowego w przypadku dużych chmur, w aktualnej implementacji wprowadzono więc odpowiednią modyfikację opisanej metody. W każdej klatce symulacji dla każdej wygenerowanej cząsteczki do pola wiatru w kilku losowych miejscach zostają dodane wektory, których wartości, kierunki i zwroty są zgodne ze schematem przedstawionym na rys. 2.



Rys. 2. Schemat tworzenia tornada (opis w tekście publikacji)
Fig. 2. Tornado generation scheme (description in the text of the publication)

Cząsteczka powstaje w pewnym punkcie S. Wektory wiatru wyznaczone w losowych miejscach są skierowane w stronę punktu S, jeśli znajdują się daleko od tego punktu, natomiast jeśli są blisko – przeciwnie. W jednym i drugim przypadku zostaje na nie nałożona dodatkowa składowa, która wprowadza odchylenie od wyznaczonego kierunku. Wartość tej składowej rośnie gwałtownie, gdy wektor jest obliczany w pewnej granicznej odległości od punktu S, zależnej od wartości funkcji generacji chmur w tym punkcie (na rysunku odległość ta jest równa promieniowi szarego okręgu). Ponieważ w trakcie symulacji generowana jest bardzo duża liczba cząsteczek, przedstawiony algorytm może doprowadzić do powstania tornada (na rysunku oznaczonego szarym okręgiem). Dzieje się tak jednak tylko wtedy, gdy wartość funkcji generacji chmur jest bardzo duża na niewielkim obszarze. Gdy jej wartości są podobne na dużym obszarze, to wektory wiatru obliczane dla powstających w różnych miejscach cząsteczek znoszą się, dzięki czemu wiatr jest stosunkowo słaby i jednorodny.

[5] Zagadnienie to zostało szczegółowo opisane w [3, 5].

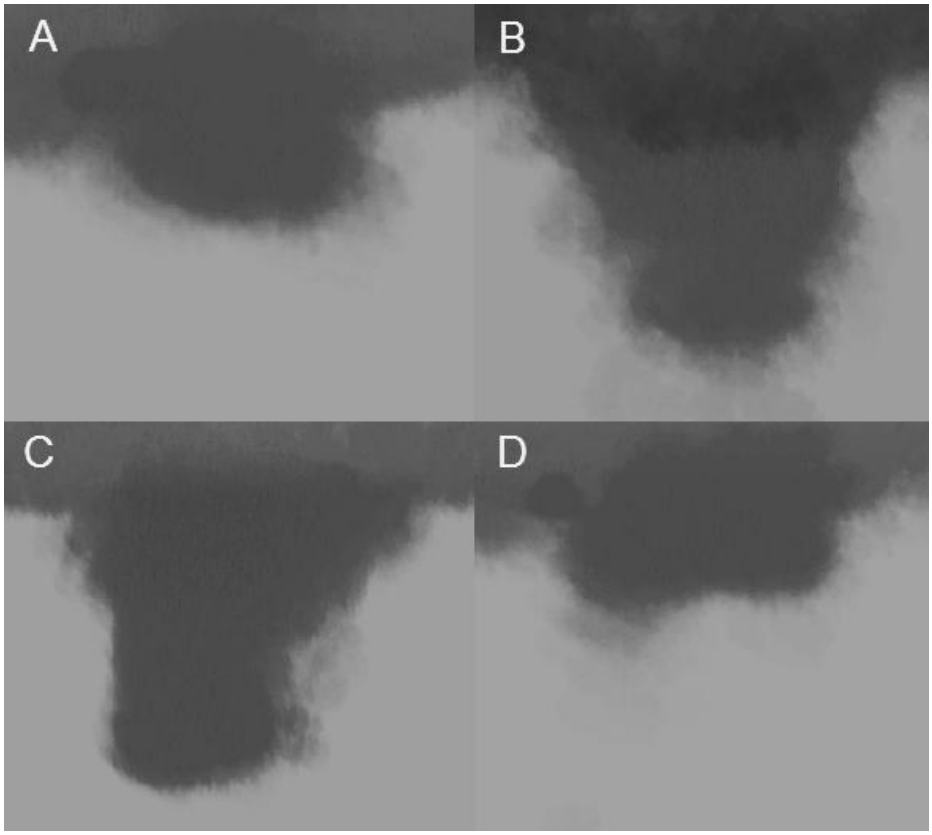
Dzięki temu tornado nigdy nie powstanie w przypadku nawet bardzo grubej pokrywy chmur warstwowych *nimbostratus*, natomiast może wystąpić w trakcie silnej burzy.

Wzajemne znoszenie się wektorów „wynikających" z różnych cząsteczek uniemożliwia również narastanie wartości wiatru do nieskończoności, mimo że w każdej klatce symulacji nowe wektory są dodawane do pola wiatru.

Pomimo tego, że omawiany algorytm zapewnia powstawanie tornad tylko w pewnych określonych warunkach (bardzo zróżnicowane wartości funkcji generacji chmur), z obserwacji wielu przebiegów symulacji wynika, że tornada pojawiają się zbyt regularnie. Wprowadzono więc dodatkową modyfikację algorytmu. Wektory dodawane do pola wiatru w wyniku powstania cząsteczki są zaburzane losowo, zależnie od wartości dodatkowego **współczynnika regularności wiatru**. Współczynnik ten jest reprezentowany przez pole skalarne, a więc dla danej cząsteczki przyjmuje on więc wartość pola w punkcie, w którym powstaje cząsteczka. Pole natomiast jest modyfikowane cyklicznie za pomocą **szumów Perlina** [21]. Dzięki temu tornado może powstać tylko w przypadku, gdy na pewnym, niewielkim obszarze zarówno wartości funkcji generacji chmur, jak i współczynnika regularności wiatru są bardzo duże. W przypadku gdy wartość funkcji jest duża, a współczynnika mała, to wiatr jest gwałtowny, porywisty i bardzo niejednorodny, co jest typowe dla większości gwałtownych burz.

Ostatnim zagadnieniem dotyczącym symulacji tornad jest wysokość, na jakiej generowane są cząsteczki. Aby tornado mogło sięgnąć ziemi, muszą one – w przeciwieństwie do cząsteczek tworzących chmury – powstawać tuż nad jej powierzchnią. Problem ten został rozwiązany przez uzależnienie początkowego pułapu cząsteczek od siły wiatru. Dla bardzo dużych wartości wektora wiatru wysokość, na której generowana jest cząsteczka, gwałtownie maleje. Dzięki temu pojawiający się wir powoduje obniżanie się chmury w jego obrębie, tworząc trąbę powietrzną, która sięga ziemi, jeśli przekroczona zostanie pewna graniczna wartość wiatru.

Przykładowe tornado powstałe w wyniku symulacji opisaną metodą przedstawiono na rys. 3.



Rys. 3. Cykl życia tornada: A – chmura stropowa, B – trąba powietrzna, C – stadium dojrzałe, D – zanikanie
Fig. 3. Tornado life cycle: A – wall cloud, B – funnel cloud, C – mature stage, D – dissipating

6. Teksturowanie i cieniowanie

Jak napisano w rozdziale 2., warstwa renderingu jest niezależna od warstwy odpowiadającej za logikę symulacji. Dane generowane przez komponent odpowiedzialny za symulację mogą więc być wyświetlane przez różne komponenty odpowiedzialne za rendering.

Różne metody renderingu chmur przedstawiono w [8-18]. Większość zaawansowanych technik nie nadaje się jednak do renderingu w czasie rzeczywistym, jeśli wyświetlane jest więcej chmur niż pojedynczy *cumulus*. Istotną metodę optymalizacji przedstawiono w [11] oraz [4]. Polega ona na rysowaniu w każdej klatce jedynie chmur znajdujących się blisko obserwatora, podczas gdy odległe chmury wyświetlane są przy użyciu pojedynczych, dużych cząsteczek[6], których tekstura aktualizowana jest ze znacznie mniejszą częstotliwością.

[6] Cząsteczka w kontekście renderingu oznacza dwuwymiarowy obiekt pokryty teksturą, zwrócony w stronę obserwatora przeważnie tą samą stroną. Nie należy mylić tego pojęcia z cząsteczką w kontekście symulacji, gdzie jest ona jedynie pewną abstrakcyjną wielkością. Niemniej jednak w najprostszych algorytmach wyświetlania chmur (np. w opisywanym

Znaczne przyspieszenie uzyskać można również przez stosowanie uproszczonych algorytmów cieniowania chmur. W aktualnej implementacji w środowisku FRS jasność każdej cząsteczki zależy od wysokości na jakiej się znajduje oraz od grubości pokrywy chmur na danym obszarze. Metoda ta nie uwzględnia pozycji słońca, daje jednak stosunkowo dobre rezultaty przy niewielkiej złożoności obliczeniowej (przykładowe chmury wyświetlane tą metodą przedstawiono w [3]). Nie sprawdza się natomiast w przypadku chmur silnie rozbudowujących się w pionie, np. *cumulonimbus*. Pewnym udoskonaleniem renderingu takich chmur może być natomiast stosowanie różnych tekstur, w zależności od wysokości na jakiej znajduje się cząsteczka, oraz uzależnienie wielkości cząsteczki od wysokości na jakiej się znajduje. Obserwacja chmur rozbudowujących się w pionie pozwala bowiem stwierdzić, że w swoich górnych partiach są one znacznie bardziej „rozmyte" niż w dolnych. Dla górnych partii należałoby więc użyć większych cząsteczek o bardziej „rozmytej" teksturze.

7. Uwagi końcowe

Zarówno komponent odpowiedzialny za symulację pogody, jak i komponent odpowiedzialny za jej wizualizację są wciąż rozwijane. W wizualizacji brakuje omówionej w [11] techniki optymalizacyjnej, która jest konieczna w przypadku wyświetlania chmur na dużym obszarze, testowane są jednak różne techniki ich cieniowania. W komponencie odpowiedzialnym za symulację brakuje natomiast symulacji powstawania i zaniku mgły oraz rozróżnienia między rodzajami opadów atmosferycznych (oczywiste jest, że w temperaturze dodatniej najczęściej pada deszcz, a w ujemnej najczęściej śnieg – większym problemem są natomiast opady typu mżawka, grad czy krupy śnieżne). Zagadnienia te zostaną w najbliższym czasie uwzględnione w obu komponentach.

LITERATURA

1. Grudziński T., Mysłek T., Ross J.: Wykrywanie kolizji obiektów trójwymiarowych w środowisku FRS. Studia Informatica, Vol. 26, No. 4(65), Wyd. Pol. Śl., Gliwice 2006.
2. Grudziński T., Mysłek T., Ross J.: Zaawansowane techniki animacji trójwymiarowych modeli szkieletowych w środowisku FRS. Studia Informatica, Vol. 27, No. 1(66), Pol. Śl., Gliwice 2006.

w publikacji algorytmie zastosowanym w środowisku FRS) pojedyncza cząsteczka w warstwie renderingu odpowiada pojedynczej cząsteczce w warstwie symulacji.

3. Grudziński J., Dębowski A.: Symulacja pogody w czasie rzeczywistym w środowisku FRS. Studia Informatica, Vol. 27, No. 1(66), Wyd. Pol. Śl., Gliwice 2006.
4. Dębowski A., Grudziński J., Grudziński T.: Problemy i metody wizualizacji zjawisk atmosferycznych w czasie rzeczywistym w środowisku FRS. Studia Informatica, Vol. 28, No. 1(70), Wyd. Pol. Sl., Gliwice 2007.
5. Grudziński J.: Algorytmy symulacji zjawisk atmosferycznych realizowane w czasie rzeczywistym. Praca dyplomowa magisterska, Pol. Śl., Instytut Informatyki, Gliwice 2005.
6. Grudziński T.: Rendering animowanych obiektów trójwymiarowych realizowany w czasie rzeczywistym. Praca dyplomowa magisterska, Pol. Śl., Instytut Informatyki, Gliwice 2003.
7. Błaż J.: Systemy cząsteczkowe i ich wykorzystanie w grafice komputerowej do modelowania wybranych zjawisk fizycznych. Praca dyplomowa magisterska, Pol. Śl., Gliwice 2004.
8. Harris M. J.: Real-Time Cloud Simulation and Rendering. Technical Report, Department of Computer Science, University of North Carolina, 2003.
9. Dobashi Y., Keneda K., Yamashita H., Okita T., Nishita T.: A Simple, Efficient Method for Realistic Animation of Clouds. SIGGRAPH, 2000.
10. Wang N.: Realistic and Fast Cloud Rendering in Computer Games. SIGGRAPH, 2003.
11. Wang N.: Realistic and Fast Cloud Rendering. Journal of Graphics Tools, 2004.
12. Harris M. J.: Real-Time Cloud Rendering for Games. Proceedings of Game Developers Conference 2002, march 2002.
13. Harris M., Lastra A.: Real-Time Cloud Rendering. Computer Graphics Forum, Blackwell Publishers, Vol. 20, s. 76÷84, 2001.
14. Elbert D.: Procedural Volumetric Modeling and Animation of Clouds and Other Gaseous Phenomena, SIGGRAPH, 2002.
15. Schpock J., Simons J., Ebert D. S., Hansen C.: A Real-Time Cloud Modeling, Rendering, and Animation System. SIGGRAPH, 2003.
16. Erez E.: Interactive 3D Lighting in Sprites Rendering. Gamasutra, 2005.
17. Heinzlreiter P., Kurka G., Volkert J.: Real-Time Visualization of Clouds. V. Skala -red, Journal of WSCG 2002, Vol. 10(3), 2002.
18. Trembilski A., Broßler A.: Surface-Based Efficient Cloud Visualisation for Animation Applications. V. Skala, editor, Journal of WSCG, Vol. 10, 2002.
19. Wang N., Wade B.: Rendering Falling Rain and Snow. SIGGRAPH, 2004.
20. Unger J., Wrenninge M., Wanstrom F., Ollila M.: Real-Time Image Based Lighting in Software Using HDR. Computer graphics and interactive techniques in Australasia and South East Asia, 2003.

21. Elias H.: Perlin Noise. http://freespace.virgin.net/hugo.elias/models/m_perlin.htm.
22. Environmental Science Published for Everybody Round the Earth. http://www.espere.net/.

Recenzent: Dr hab. inż. Maria Pietruszka, Prof. Pol. Łódzkiej

Wpłynęło do Redakcji 16 listopada 2006 r.

Abstract

The paper is the continuation of the subject of the "Real-Time Atmospheric Phenomena Simulation in FRS Engine" (Studia Informatica, Vol. 27 Number 1 (66)) publication. General information about particle systems are given (with exemplary particle effect in fig. 1) and a real-time simplified weather simulation and rendering particle system is presented. In the referenced publication main ideas of the system are described and in the present publication additional issues are discussed. These include the simulation of precipitation, atmospheric discharges, rotation of supercells and tornadoes in chapters 3, 4 and 5 (with tornado generation scheme in fig. 2 and exemplary tornado evolution presented in fig. 3). Rendering issues are mentioned in chapter 6, including cloud shading and the use of multiple textures. Additional remarks are given in chapter 7.

Adresy

Jakub GRUDZIŃSKI: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, jakub.grudzinski@polsl.pl.
Adrian DĘBOWSKI: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, adrian.debowski@polsl.pl.
Tomasz GRUDZIŃSKI: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, tomasz.grudzinski@polsl.pl.