STUDIA INFORMATICA 2006
Volume 27 Number 1 (66)

Jakub GRUDZIŃSKI, Adrian DĘBOWSKI
Politechnika Śląska, Instytut Informatyki

SYMULACJA POGODY W CZASIE RZECZYWISTYM W ŚRODOWISKU FRS

Streszczenie. W publikacji omówiono system uproszczonej symulacji zjawisk atmosferycznych, pracujący w czasie rzeczywistym. Symulacja oparta jest na generacji cząsteczek, będących elementarnymi fragmentami chmur i uwzględnia lokalne efekty pogodowe związane ze zjawiskami zachodzącymi w skali makro, takimi jak fronty atmosferyczne.

Słowa kluczowe: atmosfera, chmury, wirtualna rzeczywistość

REAL-TIME WEATHER SIMULATION IN FRS ENGINE

Summary. The paper presents a simplified weather simulation system, which works in real time. Simulation is based on generation of particles, which are elementary fragments of clouds and includes local weather effects caused by phenomena which take place in macro scale, such as atmospheric fronts.

Keywords: atmosphere, clouds, virtual reality

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 są już spełnione, jednak osiągnięto już pewną funkcjonalność, pozwalającą na zaprezentowanie prac szerszej publiczności i wyciągnięcie pewnych wniosków.

Twórcy FRS postawili sobie za główny cel stworzenie silnika uniwersalnego i łatwego w życiu, 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. W dalszej części publikacji skupiono się na zaprezentowaniu wniosków i tez wynikłych z pracy nad tym komponentem.

2. Wprowadzenie do zagadnienia symulacji pogody w systemach wirtualnej rzeczywistości

Prognostyczne systemy symulacji pogody wykorzystują skomplikowane modele matematyczne, oparte na równaniach różniczkowych [1, 2]. Ich podstawowym celem jest osiągnięcie możliwie dużej dokładności, natomiast złożoność obliczeniowa jest tak wielka, że wymaga wieloprocesorowych superkomputerów, przy czym i tak dalekie są od pracy w czasie rzeczywistym.

W przypadku symulacji na potrzeby systemów wirtualnej rzeczywistości wymagania dotyczące szybkości działania są jeszcze większe – należy bowiem pamiętać, że kompleksowy system symulacji wirtualnej rzeczywistości zajmuje się również wieloma innymi zagadnieniami, które realizowane są współbieżnie w czasie rzeczywistym. Oczywisty jest fakt, że ilość czasu procesora dostępnego na potrzeby symulacji pogody będzie zależała od konkretnych wymagań stawianych całemu systemowi wirtualnej rzeczywistości, jednak zawsze będzie to tylko część, z reguły niewielka, rzędu kilku do kilkunastu procent.

Wobec tak silnych ograniczeń czasowych narzuconych odgórnie niemożliwe jest zastosowanie pełnego modelu matematycznego. Symulacja musi więc być w możliwie dużym stopniu uproszczona i ukierunkowana jedynie na generowanie danych, które w sensie wizualnym będą realistyczne. Okazuje się jednak, że opierając symulację na łatwych do zaobserwowania, podstawowych zjawiskach i zależnościach występujących w skali mikro, można zaprojektować model, który w czasie rzeczywistym będzie symulował nie tylko podstawowe zasady powstawania chmur, ale również wiele innych zjawisk, również tych zachodzących w skali makro. Te ostatnie, z punktu widzenia obserwatora znajdującego się na powierzchni Ziemi, powodują ogólne zmiany pogody na dużym obszarze. Najważniejszymi z nich są fronty atmosferyczne wraz z występującymi po ich przejściu strefami równowagi (*vide* punkt 3.7). Pełna symulacja musi uwzględniać wpływ takich zjawisk na lokalną pogodę, co wiąże się nie tylko z powstawaniem różnych chmur na różnych wysokościach, ale także ze

zmianami podstawowych wielkości fizycznych, takich jak temperatura czy wiatr. W punkcie 3. niniejszej publikacji omówiono powyższe zagadnienia w sposób bardziej szczegółowy.

3. Możliwe sposoby realizacji

Mniej lub bardziej uproszczoną symulację atmosfery wprowadzono do silników symulacji wirtualnej rzeczywistości stosunkowo niedawno. Większość autorów w swoich publikacjach poświęconych atmosferze w grafice komputerowej koncentrowała się na samym zagadnieniu efektywnego wyświetlania chmur [3, 4, 5]. Całkowita rezygnacja z symulacji i przedstawienie chmur w postaci trójwymiarowych modeli[1] [6] jest – oczywiście – najbardziej wydajne czasowo, ale również najmniej elastyczne.

Szybki rozwój mikroprocesorów i związane z nim znaczące zwiększenie ich mocy obliczeniowej umożliwiły w ostatnich latach rozpoczęcie prac nad modelami matematycznymi, mogącymi pracować w czasie rzeczywistym. Mimo znaczącego obciążenia czasowego narzucanego przez modele oparte na równaniach różniczkowych, podejmowano próby zastosowania ich w systemach wizualizacji chmur poprzez ograniczenie modelu jedynie do symulacji dynamiki chmur [7]. Rozwiązania tego typu mają jednak ograniczone możliwości – przedstawiony w [7] model Marka Jasona Harrisa pozwala jedynie na symulację powstawania chmur kłębiastych.

Rozwiązaniem kompromisowym między rozwiązaniami całkowicie pomijającymi symulację a zaawansowanymi modelami matematycznymi jest **automat komórkowy** oparty na prostych funkcjach logicznych [8]. Ogólna zasada działania automatów komórkowych sprowadza się do podziału przestrzeni na komórki (w przypadku symulacji powstawania chmur trójwymiarowych), które mają pewne parametry. Parametry te zmieniają się w kolejnych momentach (czas jest tu zdyskretyzowany) zależnie od swojego otoczenia, czyli parametrów sąsiednich komórek. Rozwiązanie to jest niezwykle proste ideowo i było już niejednokrotnie stosowane. W swojej publikacji [8] japońscy naukowcy omówili automat, w którym każdej komórce odpowiadają trzy zmienne binarne: *cld* – czy w danej komórce jest chmura, *hum* – wilgotność oraz *act* – aktywacja. Ich stan w chwili t wyznaczany jest przez proste funkcje logiczne, których argumentami są zmienne sąsiednich komórek i danej komórki w chwili t-1. Idea działania jest więc niezwykle prosta, natomiast pewnym problemem jest ustalenie postaci wspomnianych powyżej funkcji logicznych – to one regulują zachowanie automatu. Również problem szybkości działania jest tu częściowo rozwiązany z uwagi na możliwość

[1] Takie rozwiązanie zastosowano między innymi w programie Microsoft Flight Simulator 2004.

zastosowania pól bitowych, a tym samym równoległego obliczania stanu wielu komórek jednocześnie. Okazuje się jednak, że to nie wystarcza do osiągnięcia symulacji w czasie rzeczywistym na wystarczająco dużym, z punktu widzenia systemu symulacji wirtualnej rzeczywistości, obszarze. Ponadto, automat ma bardzo ograniczony zasięg działania – symuluje tylko pewne rodzaje chmur i całkowicie pomija pozostałe zjawiska atmosferyczne, które również powinny być symulowane (jak choćby wpływ podstawowych wielkości fizycznych, czyli ciśnienia, wilgotności czy wiatru na chmury), pozostając przy tym niemalże całkowicie nieparametryzowalnym.

Korzystając z pewnych założeń systemu opartego na automacie komórkowym, można jednak przedstawić inne, bardziej złożone rozwiązanie. W automatach komórkowych zastosowano ideę przedstawienia chmur w postaci cząsteczek (jedna komórka to jedna cząsteczka), będących elementarną objętością skroplonej pary wodnej, czyli chmury, natomiast symulacja sprowadza się do bardzo małej skali (pojedynczej komórki), przy odgórnym założeniu, że prawidłowe przedstawienie zjawisk o tak niewielkim zasięgu automatycznie da realistyczne rezultaty w większej skali. Zastosowanie cząsteczek jest w naturalny sposób sensowne, ponieważ zjawiska zachodzące w chmurach są bardzo podobne, niezależnie od rodzaju chmury, natomiast drugie założenie jest konieczne z uwagi na ograniczenia związane z szybkością działania systemu.

Przyjmując reprezentację chmur w postaci cząsteczek oraz założenie dotyczące symulacji zjawisk jedynie w bardzo małej skali, oraz rezygnując z przedstawienia przestrzeni w postaci trójwymiarowej macierzy (a więc *de facto* rezygnując z podstawowej idei automatu komórkowego), można opracować inny system symulacji. System ten, z uwagi na oparcie jego działania na losowości, można określić jako **model probabilistyczny**.

Obserwacja ruchu powietrza w chmurach (a tym samym ruch skroplonej pary wodnej, czyli cząsteczek) prowadzi do wniosku, że ruch ten ma charakter przede wszystkim konwekcyjny, a tym samym pionowy. Jeśli więc założy się, że cząsteczki będą pewnymi abstrakcyjnymi obiektami w symulowanej rzeczywistości, można nadać im odpowiednią prędkość oraz trajektorię ruchu, przy czym zarówno prędkość, jak i trajektoria muszą zależeć od aktualnych warunków atmosferycznych. Dodatkowo, z uwagi na stopniowe zanikanie chmur, cząsteczki muszą mieć określony „czas życia”.

Zakładając, że da się w jakiś sposób uzależnić generację cząsteczek od warunków atmosferycznych, można już przyjąć, że są to podstawy całkowicie nowego podejścia do problemu.

4. Model probabilistyczny - szerzej

Założenie przedstawione w ostatnim akapicie poprzedniego punktu jest oczywiście nie tylko wyjątkowo słabo sprecyzowane, ale również dotyczy niezwykle szerokiego problemu, gdyż to właśnie postać systemu regulującego generację cząsteczek będzie odpowiedzialna za niemalże całą symulację. Jednak już na tym etapie rozważań można definitywnie stwierdzić, że taki model, dzięki oddzieleniu cząsteczek od samego systemu symulującego, jest lepiej parametryzowalny od modelu opartego na automacie komórkowym. Dalsza analiza prowadzi do wniosku, że zmieniając proste, liczbowe parametry można nie tylko zmieniać ilość tworzonych chmur, ale również ich kształt i ogólny charakter.

4.1. Funkcja generacji chmur

Narzucającym się sposobem regulacji tworzenia cząsteczek jest wprowadzenie pewnej powierzchniowej funkcji, której wartości będą proporcjonalne do liczby generowanych cząsteczek. Zakładając, że obszar symulacji jest ograniczony w sensie płaszczyzny poziomej (co jest oczywiste), można wprowadzić jego podział na podobszary. Przyjmując takie założenie, można tę funkcję przedstawić w postaci dwuwymiarowej macierzy, której wartości są jednakowe w obrębie konkretnego podobszaru.

Tak skonstruowana funkcja stanowić będzie uogólnioną powierzchniową funkcję gęstości prawdopodobieństwa tworzenia cząsteczek, a w dalszych rozważaniach nazywana będzie umownie **funkcją generacji chmur**. Uogólnienie jest tu związane z niespełnieniem podstawowej własności funkcji gęstości prawdopodobieństwa – objętość pod omawianą funkcją wcale nie musi być równa 1 – w danym kwancie czasu generowana jest pewna liczba cząsteczek wprost proporcjonalna do tej objętości (a więc również do średniej wartości funkcji), a postać funkcji reguluje ich rozkład w obszarze symulacji.

Takie rozwiązanie pozornie ogranicza rozdzielczość pojawiania się cząsteczek do podobszarów, jednak w ramach danego podobszaru można wyznaczyć jej dokładne początkowe położenie poprzez wygenerowanie dwóch liczb pseudolosowych o rozkładzie równomiernym.

4.2. Zdarzenie generacji cząsteczki

Zastosowanie funkcji generacji chmur jest tylko technicznym rozwiązaniem problemu regulacji tworzenia cząsteczek – w celu uzyskania symulacji dającej wyniki zbliżone do rzeczywistości należy jeszcze uwzględnić faktyczne procesy odpowiedzialne za tworzenie chmur, definiując w określony sposób ich wpływ na tę funkcję.

Tworzenie się chmur, jak już wcześniej wspomniano, opiera się przede wszystkim na ruchach powietrza o charakterze konwekcyjnym. Powietrze unosi się i w miarę rozprężania obniża się jego temperatura, co skutkuje skraplaniem się pary wodnej. Powstały w ten sposób „komin" działa jak odkurzacz, zasysając powietrze ze swojego otoczenia. Występuje tu więc **dodatnie sprzężenie zwrotne**, które prowadzi do rozbudowywania się chmury. W pewnych warunkach zasysanie staje się tak intensywne, że tworzy się lokalny niż baryczny i powstaje burza, a w rejonach równikowych i podrównikowych nawet huragan.

Dysponując opisaną powyżej funkcją generacji chmur w postaci macierzy, można takie sprzężenie uwzględnić w prosty sposób, zwiększając wartość funkcji w miejscu, w którym w danym momencie została wygenerowana cząsteczka, bądź w najbliższym jej otoczeniu.

Nietrudno jednak spostrzec, że w krótkim czasie doprowadzi to do wzrostu wartości funkcji, a tym samym tworzonych chmur, do nieskończoności. W rzeczywistości, dodatnie sprzężenia zwrotne zostają na skutek różnych czynników zahamowane, przez co po pewnym czasie rozpraszanie chmury zaczyna dominować nad jej rozbudowywaniem się, a w końcu konwekcja ustaje i chmura zanika. Z punktu widzenia uproszczonej symulacji nieistotna jest dokładna analiza owych czynników (całe zjawisko jest w swojej naturze dość skomplikowane), lecz ich efekt. Zanikanie chmur można więc symulować, wyrównując co pewien czas funkcję generacji chmur i kontrolując jej średnią wartość. Z uwagi na losowy charakter tworzenia cząsteczek, w dowolnej chwili jest możliwość zintensyfikowania jej wartości na innym obszarze, czego wynikiem jest powstanie kolejnej chmury.

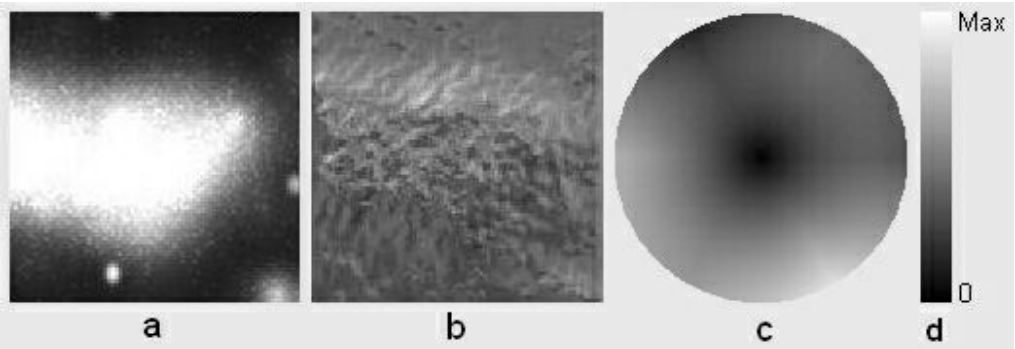
4.3. Parametry globalne i pola

Przyjmując założenie, że system dysponuje pewnymi **globalnymi parametrami** ogólnie charakteryzującymi pogodę, nietrudno uzależnić od nich intensywność wyrównywania funkcji generacji chmur. Daje to możliwość odwzorowania różnych rodzajów chmur, co nie było możliwe w przypadku automatu komórkowego. W przypadku bardzo intensywnego wyrównywania chmury będą tworzone równomiernie na całej powierzchni, co odpowiada warstwowym chmurom typu *stratus*, natomiast przy niewielkim stopniu wyrównywania cząsteczki będą wykazywać tendencję do gromadzenia się w pewnych obszarach, co odpowiada powstawaniu chmur kłębiastych (*cumulus*).

Oparcie symulacji na zdarzeniu generacji cząsteczki wsparte pewną podstawową wiedzą teoretyczną pozwala również na znacznie szerszą symulację, niż ograniczającą się jedynie do tworzenia chmur. Jak już napisano w poprzednim podpunkcie, tworząca się chmura opiera się na ruchach konwekcyjnych, które powodują zasysanie powietrza z otoczenia oraz tworzenie się lokalnego niżu barycznego (z reguły jest on zupełnie pomijalny, ale w przypadku superkomórek burzowych może już mieć znaczenie, a w przypadku huraganów obniża ciśnienie nawet o kilkadziesiąt hPa). Widać tu więc wyraźnie interakcję między chmurami a dwo-

ma wielkościami fizycznymi, jakimi są wiatr i ciśnienie. Nieco głębsze sięgnięcie do teorii pozwala na ustalenie podobnych zależności, uwzględniających inne wielkości fizyczne, takie jak wilgotność, temperatura czy opady. Jeśli przedstawi się je w postaci **dwuwymiarowych pól** (wektorowe dla wiatru oraz skalarne dla pozostałych), których reprezentacją będzie, podobnie jak w przypadku funkcji generacji chmur, macierz, można będzie symulować je na podobnej zasadzie, jak to miało miejsce w przypadku funkcji generacji chmur: zdarzenie pojawienia się cząsteczki wyzwala pewne zmiany w swoim otoczeniu, np. zmniejsza nieco ciśnienie oraz powoduje skierowanie wiatru z sąsiadujących podobszarów do siebie.

Taki sposób symulacji jest zgodny z przyjętym na wstępie założeniem, że symulowane są tylko stosunkowo proste zjawiska, mające miejsce w bardzo niewielkiej skali. Przy założeniu że są symulowane prawidłowo, efekty w większej skali również powinny być zgodne z rzeczywistością. Rysysnek 1 przedstawia przykładowy efekt wypiętrzania się chmury osiągnięty za pomocą dodatniego sprzężenia zwrotnego w funkcji generacji chmur oraz zasysania powietrza z najbliższego otoczenia cząsteczki.



Rys. 1. Dodatnie sprzężenie zwrotne oraz zasysanie powietrza pod chmurę: a) funkcja generacji chmur, b) wiatr, c) „róża wiatrów”, d) wzorcowa skala szarości dla funkcji generacji chmur

Fig. 1. Positive feedback and air being sucked in under the cloud: a) cloud generation function, b) wind, c) “wind rose”, d) model grey scale for cloud generation function

Jasność obrazu funkcji generacji chmur jest wprost proporcjonalna do jej wartości. W wyniku pojawiania się cząsteczek pod chmurą i w jej bliskich okolicach wartość funkcji rośnie tam, gdzie grubość chmur jest największa. Cząsteczki generowane są zgodnie z rozkładem zadanym przez funkcję, w związku z czym w rozpatrywanym przypadku w środku obszaru symulacji rozbudowuje się duża chmura. Wiatr natomiast jest polem wektorowym; dla oznaczenia jego kierunku wykorzystano barwę[1]. Zamieszczona obok „róża wiatrów” umożliwia odczytanie jego kierunku; każdy punkt odpowiada wektorowi od jej środka do tego punktu. Jak widać, wiatr układa się tak wyraźnie, że na obrzeżach powstałej chmury

[1] Z uwagi na druk w skali szarości w *Studia Informatica*, w obrazach nie uwzględniono barwy. Niniejszą publikację w kolorze można pobrać ze strony www.artifexmundi.com.

kieruje się ku jej centrum, w środku jest silnie zmienny, natomiast poza jej obszarem jest słabszy i bardziej jednorodny. Efekt powstały w dużej skali jest więc zgodny z oczekiwaniami, co oznacza, że przyjęte założenie symulacji tylko zjawisk mających miejsce w bardzo niewielkiej skali jest prawidłowe.

4.4. Algorytm symulacji

W poprzednim podpunkcie został omówiony przykładowy wpływ parametrów globalnych na wyrównywanie funkcji generacji chmur. Wpływ takich parametrów można jednak znacznie rozszerzyć, wykonując pewne, zależne od nich, operacje również na wszystkich bądź niektórych polach wielkości fizycznych. Wszystkie one muszą być również co pewien czas wyrównywane, ponieważ nie ma żadnej przeciwwagi dla wpływu generacji cząsteczek – przykładowo, na mapie ciśnienia pozostanie ślad po chmurze jeszcze na długo po jej zaniku.

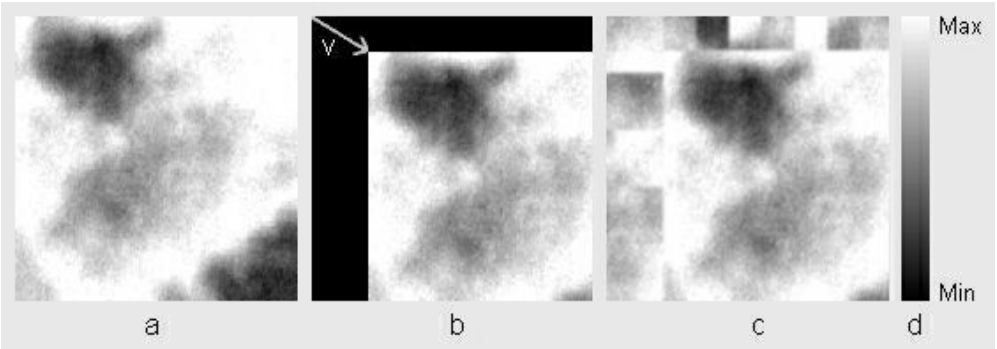
Parametry globalne można jednak rozszerzyć również na regulowanie średnich wartości symulowanych wielkości fizycznych. Dzięki temu, przyjmując, że istnieje jakiś moduł symulacji wielkoskalowej, który co jakiś czas dostarcza owe parametry, teoretycznie można nawet zaprojektować system symulujący pogodę na całej planecie. W najprostszym przypadku będą to jednak parametry losowe bądź z góry ustalone (moduł pełnej symulacji w skali makro na długo pozostanie w sferze domysłów, gdyż jest on znacznie trudniejszy niż ten omawiany w niniejszej publikacji). Przykładowo, na potrzeby wizualizacji zmian warunków atmosferycznych (a konkretnie intensywności tworzenia się chmur) można co pewien czas sukcesywnie zmniejszać bądź zwiększać średnią wartość funkcji generacji chmur. Analogicznie można postępować z wszystkimi symulowanymi wielkościami fizycznymi, przy czym należy zwrócić uwagę na to, że w modelu opartym na automacie komórkowym nie tylko nie można było nimi w żaden sposób sterować, ale – przede wszystkim – w ogóle żadna ich symulacja nie miała miejsca.

Manipulowanie parametrami globalnymi wiąże się z pewnymi problemami. Nienaturalne np. byłoby gwałtowne, skokowe zmniejszenie jakiejś symulowanej wielkości fizycznej. Automatyczne przeskalowanie całego pola nie wchodzi więc w grę. Przykładowym rozwiązaniem może być stopniowe dostosowywanie, np. poprzez średnią ważoną; nowa wartość średnia będzie średnią ważoną poprzedniej wartości średniej oraz tej podanej przez parametr globalny. Naturalnym efektem takiego postępowania będzie asymptotyczne zbliżanie symulowanej wielkości fizycznej do odpowiadającego jej parametru globalnego, przy czym szybkość będzie zależeć od odstępu między kolejnymi krokami oraz wartości wag.

Całym sterowaniem polami, cząsteczkami oraz parametrami globalnymi musi się zajmować pewien „nadzorca”, wywołujący w odpowiednich momentach odpowiednie funkcje i przeliczający wszystkie parametry. Będzie to **algorytm symulacji**, który, jak

pokazano w dalszych rozważaniach, będzie miał znacznie więcej zadań niż tylko te opisane powyżej.

Symulacja wiatru w lokalnej skali, związana z ruchami konwekcyjnymi tworzącymi chmury, nie jest – oczywiście – wystarczająca. W rzeczywistości, w atmosferze tworzą się strumienie wiatru, przekraczające często kilka tysięcy kilometrów długości, związane z niżami i wyżami, a których efektem jest przesuwanie się po niebie chmur na ogromnym obszarze z mniej więcej stałą prędkością. Przyjmując, że obszar symulacji to nie więcej niż kwadrat o boku 15 km (jak wynika z testów, większy byłby zbyt obciążający bądź zbyt niedokładny), można przyjąć, że za takie prądy odpowiadają globalne wartości wiatru przekazywane przez moduł symulacji wielkoskalowej. W efekcie, średni wiatr po pewnym czasie ustabilizuje się na poziomie globalnego. Żeby jednak chmury się przesuwały, należy zgodnie ze średnią wartością wiatru przesuwać funkcję generacji chmur. To z pozoru proste zagadnienie stanowi *de facto* niebanalny problem natury czysto algorytmicznej oraz ma znaczący negatywny wpływ na wydajność czasową. Problem związany z wydajnością czasową staje się bardziej wiarygodny, gdy weźmie się pod uwagę, że analogicznie należy postępować z wszystkimi pozostałymi polami. Problem natury algorytmicznej związany jest natomiast z automatycznym „wypełnianiem” powstałego „pustego” obszaru (*vide* rys. 2).



Rys. 2. „Pusty” obszar spowodowany przesuwaniem pól: a) pole przed przesunięciem, b) pusty obszar powstały w wyniku przesunięcia o wektor v, c) pole po przesunięciu, d) wzorcowa skala szarości

Fig. 2. „Empty” area caused by field translation: a) field before translation, b) empty area caused by translation of the field by vector v, c) field after translation, d) model gray scale

W wyniku przemieszczenia pola o pewien wektor wprost proporcjonalny do wartości średniej wiatru, część pola zostaje przemieszczona poza obszar symulacji i jest ignorowana – z drugiej strony natomiast powstaje „pusty” obszar. Żeby symulacja była spójna, ogólny charakter każdego pola powinien zostać zachowany – nie można więc wypełnić go równomiernie wartością średnią danego pola. Zdefiniowanie „charakteru” jest jednak problemem czysto abstrakcyjnym, którego nie można określić prostymi statystykami jak wartość średnia czy wariancja. Stosunkowo prostym rozwiązaniem jest kopiowanie losowych fragmentów tego

samego pola i wyrównywanie ich krawędzi celem wyeliminowania gwałtownych skoków (*vide* rys. 2c). Nie jest ono jednak dobre w przypadku pól o charakterze silnie niejednorodnym, np. funkcji generacji chmur tworzącej duże chmury kłębiaste, natomiast nie wymaga skomplikowanych obliczeń, w związku z czym jest szybkie, co, z uwagi na wymogi, jest istotne.

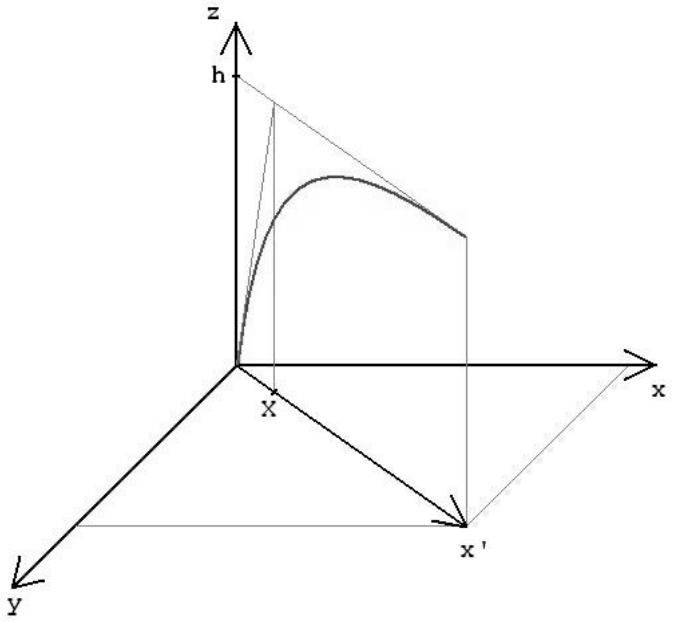
Przesuwanie pól może być również związane z ruchem obserwatora, przy czym zakłada się, że znajduje się on w środku obszaru symulacji. Z technicznego punktu widzenia jest to jednak problem identyczny z przesuwaniem w wyniku działania wiatru i nie wymaga dalszej analizy. Samo przesuwanie – w obu przypadkach – stanowi jednak kolejne stosunkowo obszerne zadanie, które musi być kontrolowane przez algorytm symulacji. W przypadku systemu działającego na potrzeby szerszego silnika wirtualnej rzeczywistości, poza dodatkowym ograniczeniem dostępnego czasu procesora, duży nacisk pada na równe rozłożenie obciążenia w czasie. Naturalnie, stan systemu musi być co pewien czas odświeżany, natomiast algorytm symulacji powinien rozłożyć obliczenia równo między kolejne kroki, niezależnie bądź częściowo niezależnie od kwantu czasu odświeżania całego systemu. Można tu zastosować wiele różnych metod, choćby tak oczywistych, jak np. przesuwanie tylko jednego pola w każdym kolejnym kroku. Analogicznie można postępować z wyrównywaniem pól, jednak jest to nieefektywne, ponieważ niektóre należy wyrównywać częściej niż inne – stąd rozsądnym rozwiązaniem jest wprowadzenie niezależnych kwantów czasu odświeżania dla każdego pola, przy czym niejako z założenia powinny być one znacznie większe niż kwant odświeżania całego systemu, a ich wartości tak dobrane, żeby jak najrzadziej dochodziło do sytuacji, gdy w jednym kroku odświeżane są dwa pola, a w następnym żadne.

4.5. Trajektoria ruchu cząsteczki i rodzaje chmur

W punkcie drugim niniejszej publikacji poruszony został temat uzależnienia parametrów generowanej cząsteczki od warunków atmosferycznych. Jednym z kluczowych parametrów jest trajektoria jej ruchu - przedstawienie w prosty, a zarazem zgodny z rzeczywistością sposób ruchu powietrza w chmurach jest jednym z najważniejszych elementów symulacji.

Ponieważ ruch ten ma przede wszystkim charakter konwekcyjny (o czym była już mowa wcześniej), naturalnym rozwiązaniem jest przemieszczanie cząsteczek w górę. W większości przypadków ruch ten jest jednak dość powolny (np. w chmurach warstwowych), choć w chmurach burzowych może osiągnąć nawet kilkadziesiąt m/s – prędkość cząsteczki musi więc również być uzależniona od warunków. W przypadku tak silnych, lokalnych „kominów” prąd wstępujący po osiągnięciu pewnej wysokości wytraca jednak prędkość i chmura rozpływa się na boki, tworząc charakterystyczny dla chmur burzowych kształt kowadła. Skierowanie cząsteczek tylko w górę nie umożliwia osiągnięcia takich kształtów. Znacznie bardziej zgodna z rzeczywistością jest eksponenta, w przypadku której można w prosty

sposób wprowadzić parametr regulujący maksymalną wysokość trajektorii (*vide* rys. 3). Przyjmując, że będzie ona uzależniać wysokość (współrzędna Z) od położenia na pewnej umownej osi X' znajdującej się na płaszczyźnie poziomej i ustalając losowo kierunek tej osi, można osiągnąć kształt chmur burzowych.



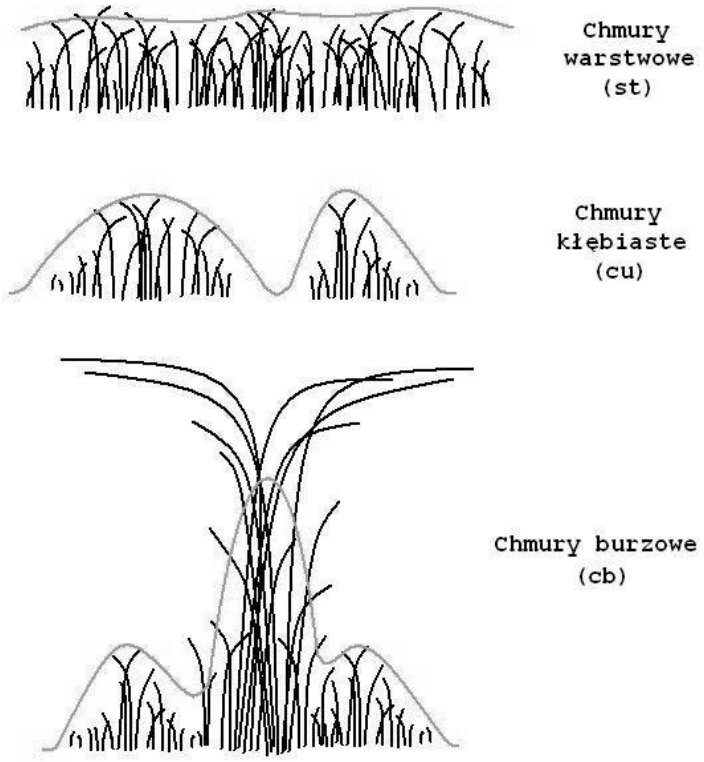
Rys. 3. Trajektoria ruchu cząsteczki
Fig. 3. Trajectory of particle's movement

Początkowe odchylenie trajektorii od pionu można regulować za pomocą zmiennej *X*, natomiast wysokość to *h*. Ponadto, uzależniając „czas życia" cząsteczki od warunków atmosferycznych, można regulować jej faktyczne odchylenie od pionu – jeśli szybko zostanie usunięta, nie zdąży się odchylić.

Najbardziej narzucającym się sposobem wyznaczania parametrów cząsteczek jest uzależnienie ich od funkcji generacji chmur. Jej wartość jest wprost proporcjonalna do liczby tworzonych cząsteczek, a więc również ilości chmur. Wysokość trajektorii i prędkość cząsteczki może więc być wprost proporcjonalna do wartości funkcji. Czas życia również powinien od niej zależeć, natomiast odchylenie trajektorii można przyjąć za stałe. Przy takich założeniach można osiągnąć rozmaite, zgodne z rzeczywistością, kształty chmur, przy czym *de facto* regulować je będzie funkcja generacji chmur.

Na rys. 4. przedstawiono zależność między funkcją generacji chmur (szare linie) a trajektoriami cząsteczek (czarne linie) w przypadku różnych chmur, przy założeniu że wpływ funkcji generacji chmur na parametry cząsteczek jest taki, jak opisano w poprzednim akapicie. Jak widać, w przypadku „płaskiej" funkcji powstają chmury warstwowe, ponieważ parametry wszystkich cząsteczek są do siebie zbliżone. W przypadku gdy funkcja przyjmuje bardziej zmienną postać, parametry cząsteczek przyjmują bardziej zróżnicowane wartości, co

prowadzi do powstawania chmur warstwowo-kłębiastych, kłębiastych, a nawet burzowych. Osiągnięcie takiego efektu nie będzie – oczywiście – możliwe bez wprowadzenia pewnej losowości. Ogólnie przyjmuje się, że w całym modelu niemalże wszystko jest tworzone losowo, przy zastosowaniu głównie generatora o rozkładzie normalnym. Wartość funkcji generacji chmur w miejscu, w którym powstała cząsteczka, jest więc bezpośrednio przekładana na parametry rozkładu (wartość średnia i odchylenie standardowe), natomiast parametry cząsteczki pobierane są z generatora przy zadanych parametrach.



Rys. 4. Zależność między funkcją generacji chmur a trajektoriami cząsteczek (pionowy przekrój przez chmury)
Fig. 4. Relationship between cloud generation function and particles' trajectories (vertical section of clouds)

Z trajektorią cząsteczki wiąże się jeszcze trudność natury algorytmicznej: trajektoria uzależnia wysokość od położenia na pewnej osi X', natomiast zmienną niezależną w symulacji jest czas. Aby cząsteczka z zadaną prędkością poruszała się po wyznaczonej trajektorii, należy obliczać długość łuku przy kolejnym kroku bądź przyjąć pewne przybliżenie. Wprowadza to pewną niedokładność, jednak jej znaczenie jest znikome. Ważne jest natomiast uwzględnienie wiatru – to jednak jest wyjątkowo proste, ponieważ wystarczy w każdym kroku dodatkowo przesuwać cząsteczki o wektor jego prędkości przemnożony przez kwant czasu odświeżania, czego efektem będzie przechylenie trajektorii zgodnie z kierunkiem wiatru.

4.6. Chmury teksturalne i powierzchniowe

Uzależnienie parametrów cząsteczki od wartości funkcji generacji chmur pozwala osiągnąć rozmaite kształty chmur, jednak dotyczy to tylko chmur piętra niskiego oraz tych, których wysokość jest tak duża, że przechodzą przez więcej niż jedną warstwę (*cumulonimbus* i *nimbostratus*). W ziemskiej atmosferze chmury występują w trzech piętrach: niskim, średnim i wysokim (więcej na ten temat można znaleźć w [10]). Obserwacja nieba pozwala jednak na wyciągnięcie wniosku, że chmury, występujące w dwóch wyższych piętrach z punktu widzenia obserwatora znajdującego się na powierzchni Ziemi, są niemalże dwuwymiarowe. Wynika to nie tylko z samego faktu, że są wyżej, ale również z tego, że są z reguły albo małe (*vide* rys. 5), albo stosunkowo płaskie.

Rys. 5. *Altocumulus* – przykład chmur piętra średniego
Fig. 5. *Altocumulus* – an example of middle clouds

Biorąc pod uwagę fakt, że generacja i przeliczanie położenia cząsteczek jest jednym z najbardziej obciążających zadań algorytmu symulacji, można przedstawić chmury piętra średniego i wysokiego w postaci dwuwymiarowej. Ponieważ niektóre z nich przesłaniają często całe niebo i są niemalże zupełnie jednolite (*cirrostratus* i *altostratus*), najwygodniejszym sposobem ich symulacji będzie przedstawienie ich w postaci dodatkowych pól skalarnych – korzystający z symulacji system wyświetlania może takie pole przedstawić w postaci generowanej automatycznie, dynamicznej tekstury ustawionej poziomo na pewnej wysokości. Tekstura taka przesłania całe niebo, natomiast poziom przezroczystości powinien być odwrotnie proporcjonalny do wartości pola w danym podobszarze. Dzięki takiemu

rozwiązaniu, w miejscach w których pole ma wartość zerową (czyli nie ma chmury), nie będzie ona widoczna.

Niektóre chmury wyższych pięter (np. przedstawiony na rys. 5. *altocumulus*) przyjmują jednak często bardzo charakterystyczne i bardzo regularne kształty i zastosowanie w ich przypadku pól przekładanych bezpośrednio na teksturę jest raczej niemożliwe. Jedynym prostym rozwiązaniem jest przygotowanie zestawu tekstur odpowiadających konkretnym chmurom bądź ich grupom i losowe tworzenie chmur jako niezależnych obiektów.

Przedstawienie chmur piętra średniego i wysokiego w postaci dwuwymiarowej wpływa negatywnie na efekt wizualny w przypadku obserwatora poruszającego się w przestrzeni na różnych wysokościach, np. w przypadku symulatorów lotu. Niemniej jednak korzyść związana z wydajnością, a wynikająca ze znacznego uproszczenia jest znacząca. Należy poza tym pamiętać, że chmury piętra niskiego, a więc te najbardziej dynamiczne i najbardziej atrakcyjne wizualnie, symulowane są za pomocą cząsteczek, a więc są w pełni trójwymiarowe i mogą być obserwowane zarówno z powierzchni ziemi, jak i z „lotu ptaka".

4.7. Fronty atmosferyczne, strefy równowagi i rozszerzanie systemu

Wprowadzenie symulowanych w bardzo uproszczony sposób chmur piętra średniego i wysokiego jest tylko pewnym rozszerzeniem systemu. Pokazuje jednak, że ogólna jego struktura pozwala w prosty sposób rozszerzać jego możliwości, co w przypadku automatu komórkowego byłoby znacznie trudniejsze. Jednym z możliwych rozszerzeń jest uwzględnienie w symulacji frontów atmosferycznych, które odpowiadają za znaczną część zjawisk atmosferycznych, a w szerokościach geograficznych odpowiadających strefie klimatu umiarkowanego za znakomitą ich większość (więcej na ten temat można znaleźć w [10]).

Fronty atmosferyczne są jednak zjawiskami niewspółmiernie większymi od obszaru symulacji (mogą osiągać nawet 10000 km długości), w związku z czym ich symulacja może być przeprowadzana jedynie na poziomie modułu symulacji wielkoskalowej. Można jednak symulować lokalne zjawiska wywoływane przez fronty, przy założeniu że sam front jest przekazywany jako pewien zbiór parametrów z tego modułu.

Front powstaje na styku dwóch lub trzech mas powietrza o różnej temperaturze. Zależnie od tego, która masa napiera na inną, wyróżnia się kilka rodzajów frontów, których rezultatem jest powstawanie różnych chmur na różnych wysokościach, zależnie od rodzaju frontu; np. w przypadku frontu ciepłego najpierw na niebie pojawiają się przypominające pióra chmury pierzaste, potem warstwowe piętra wysokiego, warstwowe piętra średniego, aż w końcu powstające w piętrze niskim, ale sięgające aż do średniego warstwowe chmury deszczowe *nimbostratus*. Front należy więc przedstawić w postaci przemieszczających się stref, przy czym z uwagi na niewielkie rozmiary obszaru symulacji, ich granicami mogą być linie proste prostopadłe do kierunku ruchu frontu. Dokładne symulowanie frontów jest dość skompliko-

wanym zagadnieniem natury *stricte* matematycznej, a ponadto można tu zastosować różne rozwiązania, więc w niniejszej publikacji nie będą one omawiane. Najistotniejszy jest sam fakt, że system jest na tyle elastyczny, że pozwala na rozszerzanie symulacji nawet o tak obszerne zagadnienia.

Analogicznym rozszerzeniem mogą być strefy równowagi, czyli obszary o specyficznej pogodzie znajdujące się poza bezpośrednim wpływem frontów. Rozróżniane są dwa ich rodzaje: **równowaga stała** oraz **równowaga chwiejna**. Następują one po przejściu frontów i charakteryzują się skrajnie różnym zachowaniem atmosfery (dlatego są istotne z punktu widzenia symulacji!). W przypadku równowagi stałej niemalże nie występują prądy konwekcyjne, wiatr jest słaby, a widoczność – z uwagi na przepełnienie powietrza różnymi zawiesinami – nienajlepsza. Równowaga chwiejna natomiast charakteryzuje się silną konwekcją, a co za tym idzie, silniejszym i bardziej zmiennym wiatrem, lepszą widocznością i powstawaniem dużych chmur kłębiastych, powodujących przelotne opady.

Uwzględnienie równowagi w symulacji wiąże się z przyjęciem pewnego współczynnika, od którego można bezpośrednio uzależnić wszystkie ważniejsze elementy symulacji, jak intensywność wyrównywania map czy tworzenie cząsteczek, przy czym problemem nie jest tu samo uzależnienie, lecz obliczanie aktualnej wartości współczynnika. Przyjmując, że każdy front „pozostawia po sobie" pewien współczynnik, którego wpływ na pogodę zanika z czasem, można obliczać globalny współczynnik jako średnią ważoną wszystkich aktywnych współczynników, przy czym ich wagi powinny zależeć od czasu, który minął od odejścia odpowiadających im frontów. W porównaniu do samych frontów jest to więc zagadnienie znacznie prostsze, a sprowadzenie go do postaci jednego współczynnika jest najlepszym dowodem na łatwość rozszerzania systemu.

Innym możliwym rozszerzeniem jest rozwiązanie specyficznego problemu, jakim jest „pusty horyzont". Jeśli obszar symulacji jest ograniczony, np. do kwadratu o boku 10 km, a chmury piętra wysokiego występują na podobnej wysokości (nawet nieco wyżej), to obserwator znajdujący się na powierzchni Ziemi, patrząc pod kątem mniejszym niż 45 stopni, nie będzie widział żadnych chmur – będą się one pojawiać pod takim kątem z jednej strony i zanikać z drugiej. W przypadku niższych pięter kąt ten będzie mniejszy, jednak pewien „pusty" obszar pozostaje. Możliwym częściowym rozwiązaniem tego problemu jest rozszerzanie obszaru symulacji wraz z wysokością – symulacja chmur średnich i wysokich jest znacząco uproszczona i nie stanowi dużego obciążenia pod względem czasowym. Przy założeniu około trzykrotnego rozszerzenia dla piętra średniego i sześciokrotnego dla wysokiego, kąt pustego horyzontu będzie dla wszystkich trzech pięter mniej więcej taki sam i nie powinien przekraczać 20 stopni. Nie rozwiązuje to jednak problemu całkowicie i należałoby się zastanowić nad ewentualnym generowaniem danych, określających bardzo ogólnie sto-

pień zachmurzenia linii horyzontu – jest to jednak zagadnienie bardzo specyficzne, ponieważ w dużej mierze zależy od możliwości systemu wyświetlającego.

4.8. Przekazywanie danych do systemu wyświetlającego

Przekazywanie danych do systemu wyświetlającego jest ogólnie problemem dość specyficznym, zależnym w dużej mierze od architektury całego silnika wirtualnej rzeczywistości, oraz od samej implementacji systemu wyświetlającego. Sama linia horyzontu jest tylko pewnym rozszerzeniem i należy liczyć się z niemożliwością jej wyświetlenia w przypadku systemu o małej elastyczności.

Ogólnie, podstawowym wymogiem stawianym systemowi wyświetlającemu jest możliwość obsługi cząsteczek. Moduł symulacji tworzy ich jednak zbyt dużo i bezpośrednie ich wyświetlanie może stanowić zbyt duże obciążenie dla systemu wyświetlającego. Częściowym rozwiązaniem tego problemu jest wprowadzenie do symulacji pewnych specyficznych optymalizacji; np. tworzenie większych cząsteczek w miarę wzrostu odległości od obserwatora (czyli środka obszaru symulacji) kosztem ich liczby – większa odległość to mniejsze wymagania dotyczące dokładności, a mniej cząsteczek to mniej obliczeń. To jednak nie wystarcza, w związku z czym należy spełnić pewne wymagania stawiane odgórnie przez system wyświetlający. Wymagania te mogą być różne w zależności od konkretnej implementacji, jednak z dużym prawdopodobieństwem można założyć, że będą się sprowadzały do przynajmniej częściowego wyeliminowania bezkontekstowości. Jeśli jedynymi danymi, dotyczącymi cząsteczek, będzie wektor ich pozycji, dane będą całkowicie pozbawione kontekstu, przez co system wyświetlający nie będzie mógł stosować wielu optymalizacji, wynikających z podstawowej wiedzy, które cząsteczki z poprzedniego kroku odpowiadają którym z aktualnego.

Częściowe wyeliminowanie bezkontekstowości gwarantuje przekazywanie w każdym kroku wektora indeksów cząsteczek „nowych”, czyli takich, które pojawiły się od czasu poprzedniego kroku, oraz cząsteczek „martwych”, czyli takich, które zanikły od czasu poprzedniego kroku (należy tu zwrócić uwagę na sposób przechowywania cząsteczek: zanikają one w różnych momentach, w związku z czym nowo wygenerowane nie muszą wcale trafiać na koniec wektora, lecz na pierwsze wolne miejsca – dlatego indeks cząsteczki w wektorze nie daje podstaw do wyciągnięcia wniosku, że jest to ta sama cząsteczka, która w poprzednim kroku była na jej miejscu). Dzięki takiemu rozwiązaniu system wyświetlający może np. grupować cząsteczki znajdujące się blisko siebie, tworząc z nich większe obiekty.

5. Uwagi końcowe i przykładowe wyniki

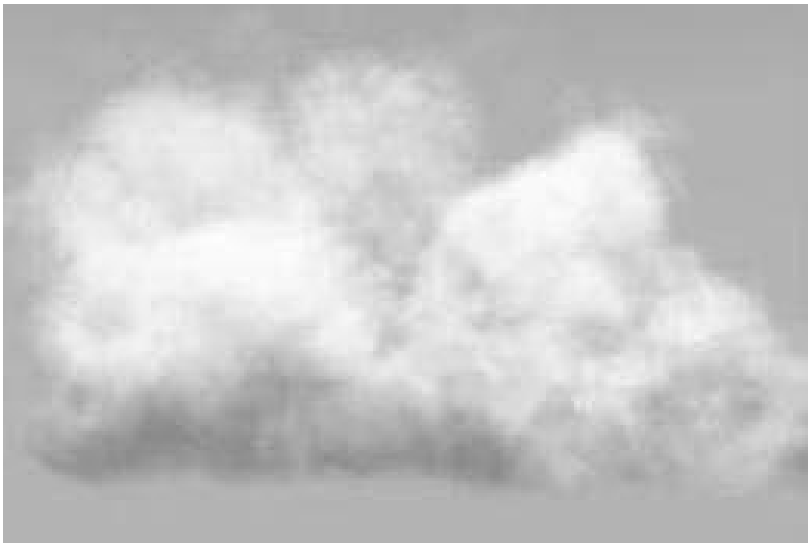
Zaproponowany w niniejszej publikacji model probabilistyczny jest, co wykazano w powyższych rozważaniach, znacznie szerszy, niż stosowane dotychczas konkurencyjne rozwiązania, a zachowuje przy tym w dużej mierze wymaganą szybkość działania. Z testów wynika, że czas procesora przydzielony na potrzeby komponentu odpowiedzialnego za symulację atmosfery to jedynie około 2% czasu przydzielonego na cały silnik FRS, co jest wynikiem zaskakująco dobrym. Podobnie pozytywnie wypadają testy szybkościowe samej symulacji. Przy maksymalnie uproszczonym wyświetlaniu (cząsteczki bez tekstur i z jednakowym kolorem), przy użyciu procesora Intel Core Duo T2500 i karty graficznej ATI Mobility Radeon X1600 symulacja wykonywana jest z kilkusetkrotnym przyspieszeniem względem czasu rzeczywistego (dokładne przyspieszenie zależy od wielu parametrów symulacji[1]). Niemniej jednak okazuje się, że problem z szybkością działania występuje po stronie systemu wyświetlającego, z uwagi na konieczność wyświetlenia bardzo dużej liczby cząsteczek. Wymagane jest tu więc zastosowanie pewnych mechanizmów optymalizacyjnych, np. renderowanie odległych od obserwatora grup cząsteczek na płaską teksturę z częstotliwością znacznie mniejszą od częstotliwości aktualizacji całej sceny (więcej na temat algorytmów związanych z wyświetlaniem można znaleźć w [3 – 9]).

Z uwagi na brak analogicznych rozwiązań, przedstawiony system jest projektem o charakterze eksperymentalnym i w chwili obecnej, z powodu na niepełną implementację, nie daje jeszcze wymaganych rezultatów w dowolnych warunkach, w związku z czym definitywne stwierdzenie, że jest lepszy od konkurencyjnych rozwiązań, byłoby nadużyciem. Niemniej jednak stanowi podstawy pewnego nowego podejścia do zagadnienia symulacji atmosfery na potrzeby wirtualnej rzeczywistości oraz już w chwili obecnej pozwala na przeprowadzanie dającej prawidłowe wyniki symulacji w pewnym zakresie warunków atmosferycznych. Przykładowe chmury powstałe w wyniku symulacji przedstawiono na rysunkach 6-10[2]. Powstawaniu większych chmur (np. przedstawiony na rys. 7 *cumulus congestus*) towarzyszą opady i silny wiatr, co jest zgodne z założeniami.

---

[1] Złożoność obliczeniowa algorytmu symulacji jest wprost proporcjonalna do kwadratu długości boku obszaru symulacji oraz do średniej wartości funkcji generacji chmur. Wartość funkcji zależy jednak od aktualnych warunków atmosferycznych, które cechuje znaczna zmienność – stąd trudności w oszacowaniu szybkości działania systemu.

[2] Chmury znacznie lepiej prezentują się w kolorze. Niniejszą publikację w kolorze można pobrać ze strony www.artifexmundi.com. Na stronie tej zostanie również wkrótce umieszczony program prezentacyjny, umożliwiający sterowanie symulacją i obserwację jej efektów (przedstawione zrzuty ekranu pochodzą z programu Ainu, głównego edytora silnika FRS).



Rys. 6. *Cumulus mediocris* (widok z boku)
Fig. 6. *Cumulus mediocris* (side-view)

Rys. 7. *Cumulus congestus* (widok z powierzchni Ziemi)
Fig. 7. *Cumulus congestus* (view from the face of the earth)

Rys. 8. *Cumulus humilis*
Fig. 8. *Cumulus humilis*

Rys. 9. *Cumulus mediocris*
Fig. 9. *Cumulus mediocris*

Rys. 10. *Nimbostratus*
Fig. 10. *Nimbostratus*

Podsumowując, warto zwrócić uwagę na niezwykle bliską interakcję między modułem symulacji a modułem wyświetlającym. Zarówno jeden, jak i drugi musi spełnić pewne wymagania stawiane sobie nawzajem. Mechanizmy odpowiedzialne za komunikację są więc najbardziej zależne od konkretnej implementacji częściami obu systemów, a optymalnym rozwiązaniem byłoby *de facto* połączenie obu komponentów w jeden.

Największą zaletą systemu jest z pewnością łatwość jego rozszerzania. Jednym z głównych założeń ogólnej jego architektury jest oddzielenie cząsteczek od systemu regulującego ich pojawianie się (choć z uwagi na sprzężenia zwrotne nie w pełni) oraz przedstawienie tego ostatniego w postaci prostej macierzowej funkcji modyfikowanej przez wiele różnych czynników. Pozwala to na rozszerzanie systemu poprzez dodawanie własnych podsystemów bezpośrednio ją modyfikujących.

Pozostaje więc mieć nadzieję, że po zakończeniu prac implementacyjnych i prawidłowej parametryzacji systemu uda się, na drodze kompleksowego testowania, doprowadzić go do stanu, w którym symulacja będzie w sposób ciągły dawała prawidłowe rezultaty w dowolnych warunkach. Aktualną implementację, w której uwzględniono moduł symulatora w skali mikro z wszystkimi opisanymi w niniejszej publikacji zagadnieniami, zrealizowano w ramach pracy magisterskiej jednego z autorów [11], gdzie została znacznie szerzej opisana. Przedstawiono tam między innymi w sposób obszerny i wyczerpujący interfejs komponentu.

LITERATURA

1. Finnigan J.: Mathematics in Environmental Science. Mathematical Sciences Symposium at the University of NSW, 23.02.1996, CSIRO Centre for Environmental Mechanics, http://www.maths.mq.edu.au/texdev/MathSymp/Finnigan/Finnigan.html.
2. Palmer T.: A weather eye on unpredictability. New Scientist 11.11.1989, http://members.fortunecity.com/templarser/weather.html.
3. Harris M., Lastra A.: Real-time cloud rendering. Computer Graphics Forum, Blackwell Publishers, vol. 20, 76-84, 09.2001, http://www.markmark.net/PDFs/RTClouds_Harris EG2001.pdf.
4. Harris M. J.: Real-Time Cloud Rendering for Games. Proceedings of Game Developers Conference 2002. March 2002, http://www.markmark.net/PDFs/RTCloudsForGames _HarrisGDC2002.pdf.
5. Elbert D.: Procedural Volumetric Modeling and Animation of Clouds and Other Gaseous Phenomena, SIGGRAPH 2002.
6. Wang N.: Realistic and Fast Cloud Rendering In Computer Games. SIGGRAPH 2003, http://www.ofb.net/~eggplant/clouds/CloudsInGames_NinianeWang.pdf.
7. Harris M. J.: Real-time Cloud Simulation and Rendering. Technical Report, Department of Computer Science, University of North Carolina 2003, ftp://ftp.cs.unc.edu/ pub/publications/techreports/03-040.pdf.
8. Dobashi Y., Keneda K., Yamashita H., Okita T., Nishita T.: A simple, efficient method for realistic animation of clouds. SIGGRAPH 2000, http://www.eml.hiroshima-u.ac.jp/~doba-/papers/sig00_cloud.pdf.
9. Schpock J., Simons J., Ebert D. S., Hansen C.: A Real-Time Cloud Modeling, Rendering, and Animation System. SIGGRAPH 2003.
10. Environmental Science Published for Everybody Round the Earth, http://www. espere.net/
11. Grudziński J.: Algorytmy symulacji zjawisk atmosferycznych realizowane w czasie rzeczywistym. Praca dyplomowa magisterska, Politechnika Śląska, Instytut Informatyki, Gliwice 2005.

Recenzent: Prof. dr hab. Krzysztof Marasek

Wpłynęło do Redakcji 8 lutego 2006 r.

Abstract

The paper introduces a new method for simulation of clouds and general atmospheric phenomena, which is computationally inexpensive and works in real time as a part of a complex virtual reality simulation system. The cloud simulation is based on particles and matrix cloud generation function, which controls their creation and parameters. The system covers such phenomena as atmospheric fronts and simplified simulation of middle and high clouds and generates multiple forms of clouds, depending on global parameters. In chapter 4 main elements of the system are discussed, with particles and cloud generation function in sections 3.1, 3.2 and 3.5 (fig. 3 and 4 illustrate trajectories of particles), global parameters and physical scalar and vector fields in 3.3 (with fig. 1 illustrating the effect that cloud generation has on wind), general simulation algorithm with problems caused by field translation (fig. 2) in 3.4, middle and high clouds in 3.6 (with an example in fig. 5), atmospheric fronts and general ideas of system extension in 3.7 and main ideas of parameter passing for visualisation system in 3.8. At the end, chapter 5 contains conclusions and exemplary findings (clouds in fig. 6, 7, 8, 9 and 10).

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.