Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

BAKALÁŘSKÁ PRÁCE

Plzeň, 2007

Zdeněk Prokop

Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Modelování mraků pro VR aplikace

Plzeň, 2007 Zdeněk Prokop

Abstract

Clouds modeling for VR application

This work concetrates on the modeling of clouds. It´s composed of two main parts. The first part describes real clouds and some existing methods of clouds generating and rendering. The second part describes my own clouds rendering library. This library generates two types of clouds: Cumulus and Stratus. Graphs of library tests results are presented at the end of this document. This document contains two appendices. The first appendix is a set of real clouds and a set of application screenshots. The second appendix is a user's guide.

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne *30.4.2007, Zdeněk Prokop,*

Obsah

1 Úvod

Mraky hrají důležitou roli při simulaci vnějšího prostředí. Realisticky vypadající mraky mohou být jedním z nejpřesvědčivějších grafických prvků venkovních scén zvláště pro aplikace, které se snaží co nejvěrohodněji se přiblížit skutečnosti. V takovýchto případech pak nestačí pouze vytvořit dobře vypadající oblohu z jednoho typu mraku, ale je nutné pokrýt celé spektrum různých druhů mraků, od malých osamocených bílých obláčků, až po černé bouřkové mraky, které se ve skutečném světě vyskytují. Mraky také nejsou statické, ale po obloze se pohybují a to ne vždy stejnou rychlostí, protože ta je ovlivněna povětrnostními podmínkami.

Dále také musíme brát ohled na výpočetní techniku, která nás v tomto směru značně omezuje. Aby námi vytvořená obloha byla použitelná i pro praktické interaktivní aplikace, nesmí být příliš náročná, protože v těchto aplikacích se počítá také spousta dalších věcí, jako například terén okolní krajiny, fyzika, umělá inteligence apod.

1.1 Popis kapitol

Tato práce je rozdělena na dvě hlavní části, teoretickou a praktickou. V teoretické části se zabývám mraky ve skutečné přírodě, tj. tím, co to mrak je, jak vypadá, jak vzniká a jaké druhy mraků existují. Dále zde popisuji některé metody pro modelování mraků a také některé metody pro zobrazování mraků.

V praktické části se zabývám vlastní implementací. Popisuji nejdůležitější metody z vytvořené knihovny a uvádím výsledky z testování i s jejich hodnocením.

1.2 Cíl práce

Cílem práce bylo seznámit se s metodami používanými pro modelování mraků typu cumulus a stratus, poté naimplementovat vlastní grafickou knihovnu pro zobrazování těchto mraků a otestování této knihovny.

2 Teorie

Tato část se zabývá teorií mraků, jejich vznikem a dělením na různé typy. Dále jsou zde uvedeny různé metody pro modelování a zobrazování mraků, které se v praxi používají.

2.1 Mraky ve skutečné přírodě

2.1.1 Definice oblaku

Oblak je viditelná soustava nepatrných částic vody nebo ledu v ovzduší. Tato soustava může obsahovat zároveň i větší částice vody nebo ledu a také jiné částice pocházející např. z průmyslových exhalací, kouře nebo prachu.

2.1.2 Vzhled oblaků

Vzhled oblaku určují rozměry, množství a prostorové rozdělení částic, ze kterých se skládá, a dále také intenzita a barva světla, které na oblak dopadá, na poloze místa pozorování a na poloze světelného zdroje vůči oblaku. Nejdůležitější z těchto veličin jsou jas a barva. Jas oblaku závisí na tom, kolik světla oblakové částice odrážejí, rozptylují a propouštějí. Světlo přichází většinou přímo ze Slunce nebo z oblohy jako rozptýlené světlo. Může ale také přicházet i od zemského povrchu, pokud ho odráží rozsáhlejší ledové nebo sněhové plochy. Barva oblaku je závislá především na barvě světla, jež oblak osvětluje. Barvu však může do jisté míry změnit i zákal mezi oblakem a pozorovatelem. Pokud je Slunce dostatečně vysoko nad obzorem, jsou oblaky osvětlené slunečním světlem bílé nebo šedé a části přijímající světlo od modré oblohy jsou modravě šedé.

Pro vznik mraku je nejpodstatnějším faktorem vzestupný pohyb vlhkého vzduchu. (Naopak klesající pohyb vzdušné masy působí rozpad oblačnosti). Stoupající vzduch se rozpíná a ochlazuje a přitom se část odpařené vody kondenzuje a vytváří mikrokapičky. V průměrně hustém mraku se vyskytuje přibližně 0.5 g / m^3 vody. Z hlediska skupenství vody lze oblaky rozdělit na vodní, ledové a smíšené. Ke kondenzaci vzdušné vlhkosti dochází buď kvůli poklesu teploty k rosnému bodu, nebo kvůli změně tlaku a nebo pokud odpařovaná vlhkost kondenzuje až k bodu nasycení (to znamená, že při daném tlaku a teplotě už žádné

další množství vodních par v plynném skupenství není atmosférickým vzduchem absorbováno).

2.1.3 Dělení mraků

Dělení mraků podle tvaru zavedl Angličan Howard v roce 1803. Howardova stupnice se někdy označuje jako morfologická. Hlavními mraky jsou:

- *Cirrus (řasa)*
  Cirrus je vysoký, lehký, bílý vláknitý oblak tvořený ledovými krystalky. Tyto oblaky mívají hedvábný lesk a nevypadávají z nich žádné srážky. Nacházejí se ve výšce 6-10 km.

Obr. 2.1: Cirrus

- *Cumulus (kupa)*
  Osamocené bílé, husté oblaky, obvykle s ostře ohraničenými obrysy. Vyvíjí se směrem vzhůru ve tvaru kup nebo věží. Jejich horní část má často podobu květáku. Části oblaku ozářené Sluncem bývají nejčastěji zářivě bílé, základna oblaku bývá poměrně tmavá a téměř vodorovná. Tyto oblaky se skládají hlavně z vodních kapiček. Ledové krystaly se mohou tvořit v těch částech oblaku, kde je teplota značně pod bodem mrazu.

Obr. 2.2: Cumulus

- *Stratus (sloha)*
  Stratus je stejnoměrná vrstva oblaků podobných mlze, která zpravidla pokrývá celou oblohu. Prosvítá-li vrstvou stratu Slunce, jsou jeho obrysy zřetelně patrné. Vyskytuje se poměrně blízko povrchu, nejvýše do tří kilometrů.

Obr. 2.3: Stratus

Dále se rozlišuje sedm typů mraků, které tvoří hranici mezi základními typy:

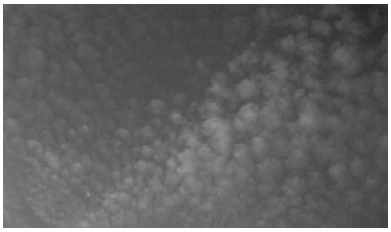
- *Altocumulus (vysoká kupa)*
  Menší nebo větší skupiny nebo vrstvy oblaků. Mají bílou nebo šedou barvu a vrhají vlastní stíny. Skládají se z malých oblačných částí v podobě vln. Mnohdy mají částečně vláknitý nebo rozplývavý vzhled. Bývají tvořeny převážně kapkami vody.

Obr. 2.4: Altocumulus

- *Altostratus (vysoká sloha)*
  Šedavá nebo modravá oblačná vrstva, která částečně nebo úplně pokrývá oblohu. Mívá buď vláknitou nebo žebrovitou strukturu, nebo je bez patrné struktury. Je tak tenká, že místy mohou být patrné obrysy Slunce.

Obr. 2.5: Altostratus

- *Cirrocumulus (řasová kupa)*
  Tenké menší nebo větší skupiny nebo vrstvy bílých oblaků, složené z velmi malých částí v podobě zrnek nebo vlnek. Jednotlivé části mohou být buď navzájem oddělené, nebo mohou spolu souviset. Tyto mraky nevrhají vlastní stín a lidově se nazývají „beránky“.



Obr. 2.6: Cirrocumulus

- *Cirrostratus (řasová sloha)*
  Průsvitný bělavý závoj oblaků, vláknitého nebo hladkého vzhledu. Často pokrývá celou oblohu. Bývají v něm patrné halové jevy.

Obr. 2.7: Cirostratus

- *Cumulonimbus (bouřkový mrak)*

  Obr. 2.8: Cumulonimbus

  Mohutný a hustý oblak velkého vertikálního rozsahu, který narůstá až do výšky několika kilometrů. Alespoň část jeho vrcholu je obvykle hladká nebo vláknitá či žebrovitá a téměř vždy zploštělá. Tato část se rozšiřuje do podoby kovadliny nebo širokého chocholu. Pod velmi tmavou základnou oblaku se často vyskytují nízké roztrhané oblaky. Cumulonimbus se většinou vytváří ze silně vyvinutých oblaků Cumulus.

- *Nimbostratus (dešťová sloha)*

  Obr. 2.9: Nimbostratus

  Šedá, často tmavě šedá jednolitá vrstva, ze které často vypadávají intenzivní srážky. Vrstva je tak hustá, že přes ní není patrná poloha Slunce a pod touto vrstvou se často vyskytují nízké roztrhané oblaky.

- *Stratocumulus (slohová kupa)*

  Obr. 2.10: Stratocumulus

  Šedé až bělavé skupiny nebo vrstvy oblaků, které mají téměř vždy tmavá místa a bývají tvořeny převážně vodními kapičkami. Oblak se skládá z částí podobných oblázkům a valounům a nemívá vláknitý vzhled. Jednotlivé části oblaku spolu buď souvisí nebo mohou být oddělené.

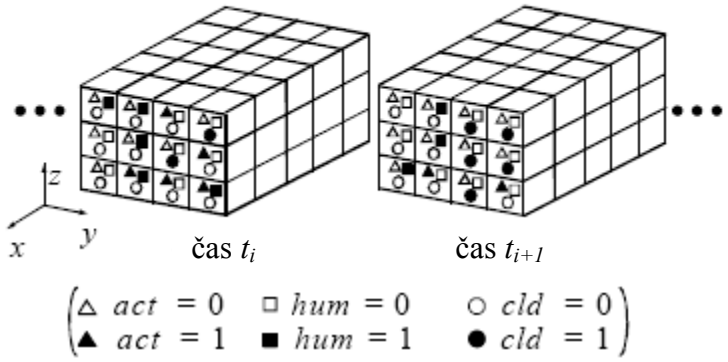
Všechny ilustrační obrázky mraků uvedené výše v této kapitole pocházejí z [5] a jsou také uvedeny barevně v příloze A. Více informací o jednotlivých typech mraků naleznete v [5], [6], [7].

2.2 Modelování mraků

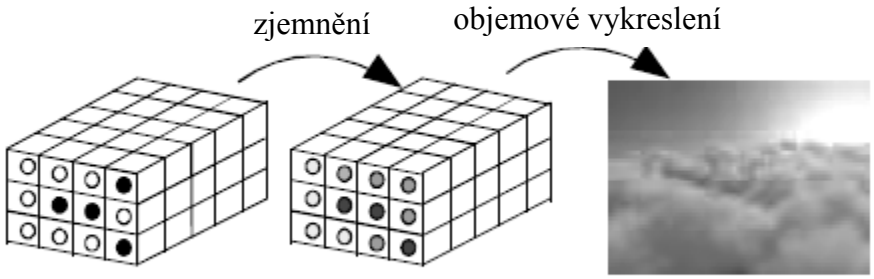
Metody pro modelování mraků se dají rozdělit na dvě skupiny. Jednou skupinou jsou metody, které vytvářejí mraky napodobováním jejich skutečného vzniku, druhou skupinou jsou metody, které se snaží co nejjednodušším způsobem vytvořit mraky, které se co nejvíce podobají těm skutečným.

Metody, které napodobují skutečný vznik mraku mají nevýhodu v tom, že jsou většinou výpočetně velmi náročné a proto jsou vhodné pouze pro statické scény. Naopak jejich velkou výhodou je, že mraky jimi vygenerované vypadají opravdu realisticky. Metoda, která mezi ně patří a není výpočetně příliš náročná, je metoda modelování využívajícího buněčné automaty (viz [1]), která pracuje níže popsaným způsobem.

Simulační prostor je rozdělen na voxely, každý voxel odpovídá jedné buňce buněčného automatu. V každé buňce jsou tři logické proměnné: *vlhkost* (*hum*, humidity), *mraky* (*cld*, clouds) a *fázová přeměna*, nebo *aktivace* (*act*, phase transition or activation factor). Každá z těchto proměnných může nabývat hodnot 0 nebo 1. Pokud je 0 u *vlhkosti*, znamená to, že zde není dostatečná vlhkost pro vznik mraku, pokud je 1, vlhkost je dostatečná. Proměnná *mraky* určuje, jestli se v daném voxelu vyskytuje mrak, či nikoliv. Pokud se zde vyskytuje, je její hodnota 1, jinak 0. *Fázová přeměna* udává, jestli může proběhnout fázová přeměna. Fázová přeměna (fázový přechod) je fyzikální pojem označující skokovou změnu makroskopických vlastností termodynamického systému (fáze) při změně nějaké termodynamické proměnné (např. teploty). Pokud tato přeměna proběhnout může, je hodnota proměnné *fázová přeměna* 1, pokud ne, je její hodnota 0.



(a) Proces simulování



(b) Proces vykreslování

Obrázek 2.11: Buněčné automaty, obrázek pochází z [1]

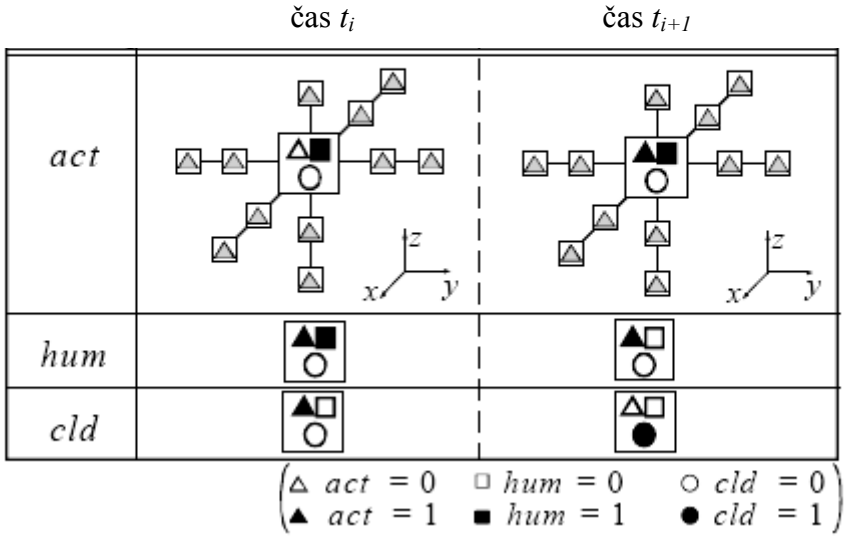
Proces simulace je znázorněn na obrázku 2.11 (a). Celá simulace se pak řídí pravidly (2.1) naznačenými níže:

$$\begin{aligned} hum(i,j,k,t_{i+1}) &= hum(i,j,k,t_i) \wedge \neg act(i,j,k,t_i), \\ cld(i,j,k,t_{i+1}) &= cld(i,j,k,t_i) \vee act(i,j,k,t_i), \\ act(i,j,k,t_{i+1}) &= \neg act(i,j,k,t_i) \wedge hum(i,j,k,t_i) \wedge f_{act}(i,j,k), \end{aligned} \tag{2.1}$$

$f_{act}(i,j,k)$ je booleovská funkce, jejíž hodnota se vypočítá podle vztahu (2.2):

$$\begin{aligned} f_{act}(i,j,k) &= act(i+1,j,k,t_i) \vee act(i,j+1,k,t_i) \vee act(i,j,k+1,t_i) \\ &\vee act(i-1,j,k,t_i) \vee act(i,j-1,k,t_i) \vee act(i,j,k-1,t_i) \vee act(i-2,j,k,t_i) \\ &\vee act(i+1,j,k,t_i) \vee act(i,j-2,k,t_i) \vee act(i,j+2,k,t_i) \vee act(i,j,k-2,t_i), \end{aligned} \tag{2.2}$$

kde *i*, *j*, *k* jsou indexy buňky, t_i je aktuální čas a t_{i+1} je následující čas. Příklad je znázorněn na obrázku 2.12. Na začátku simulace se proměnná mraky nastaví na nulu a ostatní dvě proměnné náhodně. Před samotným vykreslením mraku je ještě nutné zjemnit jednobitovou informaci v proměnné mraky. To se provádí vypočtením spojitého rozložení hustoty v každé buňce (obrázek 2.11 (b)).

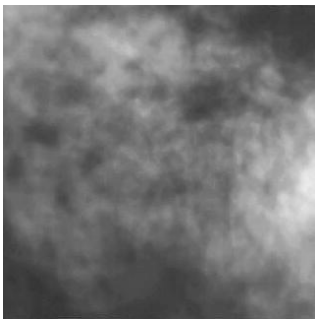


Obrázek 2.12: Základní přechodová pravidla, obrázek pochází z [1]

Další metoda, která se využívá pro generování mraků je CML (Coupled Map Lattice). Mraky jsou modelovány simulací vypařování vody. Tato metoda je rozšířením buněčných automatů. Prostor simulace je rozdělen do mřížky, v každém prvku mřížky je několik stavových proměnných a každá tato proměnná je závislá na proměnných sousedících prvků mřížky. Hlavní rozdíl mezi CML a buněčnými automaty je, že CML používá reálné hodnoty proměnných, zatímco v buněčných automatech se používají diskrétní hodnoty proměnných. Výhody této metody jsou, že je výpočetně málo náročná, je vhodná pro paralelní zpracování a už při hrubé mřížce je možné získat poměrně kvalitní simulaci mraku. Více informací o této metodě je možné získat v [9].

Mezi druhou skupinu metod, které se snaží mrak vizuálně napodobit, se může například zařadit metoda využívající fraktálních charakteristik fBm plochy (fBm, fractional Brownian motion = zlomkový Brownův pohyb), která se dá využít pro generování dvourozměrných textur mraků. Stačí, pokud se tato plocha zobrazí

v kolmém průmětu tak, že výška bude reprezentována jako barva, například odstíny šedi. Dostaneme tak poměrně věrohodný obraz mraku (viz [4]).



Obrázek 2.13: Textura mraku vytvořená průmětem fBm plochy do roviny xy, obrázek pochází z [4]

2.3 Zobrazování mraků

Existuje několik různých způsobů, jak se dají ve VR aplikacích zobrazovat mraky. Asi nejjednodušším a dříve velmi často používaným způsobem je tzv. „skybox“, neboli krychle, na jejíž stěny se vykreslí textura oblohy. Tento přístup má velkou výhodu v tom, že je díky své jednoduchosti i velmi nenáročný, ale na druhou stranu má i značné nevýhody. Například mraky nevypadají trojrozměrně a nelze se k nim příliš blízko přibližovat a už vůbec ne do nich vletět. Další nevýhodou je, že obloha je pouze statická.

Následující dvě metody pro zobrazování mraků používají techniku billboardů, nebo spritů. Proto si nyní tyto dva pojmy objasníme (viz [4]). Billboardy jsou často využívány v aplikacích virtuální reality. Billboard slouží k reprezentaci objektu, je tvořen jedním polygonem, který je pevně umístěn do jednoho místa ve scéně a při zobrazování se vždy natočí tak, aby rovina polygonu byla kolmá na pohled uživatele. Tím se zamezí tomu, aby uživatel spatřil nosný polygon z boku, se silně zkreslenou texturou. Obraz na billboardu je statický a předem připravený. Sprite je podobně jako billboard tvořen jedním nosným polygonem s obrazem jednoho objektu, avšak narozdíl od billboardu se jeho orientace při změně polohy pozorovatele nemění. Může se proto stát, že uživatel při procházení scénou spatří sprite z boku, promítnutý do úsečky. Proto se často používá rozšířená varianta, která obsahuje kombinaci několika navzájem se protínajících nosných polygonů. Polygony jsou různě natočeny na každém z nich je nanesen obraz téhož prostorového objektu

pořízený z odpovídajícího pohledu. Sprite a Billboard jsou pro svoji jednoduchost velmi oblíbené náhradní reprezentace komplexních útvarů.

Další metoda, která se používá, je vykreslení celého jednoho mraku na billboard, nebo sprite. Tento mrak potom vypadá velice dobře a realisticky z nehybné kamery, ale pokud se kamera začne točit kolem něj vypadá velmi nepřirozeně, proto pro aplikace virtuální reality není příliš vhodný.

Mezi poměrně nové přístupy patří vytváření mraku jako shluku menších textur (viz [2]). Takto vytvořený mrak vypadá dobře jak z nehybné kamery, tak i z kamery pohybující se kolem něj. Poměrně přirozeně působí i pohyb skrze tento mrak, což u předchozích dvou metod nebylo vůbec možné. Někdy se používají pro každý mrak jedinečné textury, ale s rostoucím počtem mraků velmi stoupá i náročnost na paměť grafické karty. V jiných systémech se zase používá sada předem vygenerovaných nebo nakreslených textur, ze kterých se všechny mraky modelují. Použitím pouze malého počtu textur se velmi sníží náročnost na paměť grafické karty.

3 Realizace

Z metod pro generování a zobrazování mraků popsaných v předchozí kapitole jsem přímo nepoužil ani jednu metodu, pouze jsem se některými inspiroval při vytváření vlastních metod, převážně u metody pro zobrazování použitím billboardů. Obě tyto metody se nyní pokusím podrobně popsat. Dále se zmíním o osvětlení mraků Sluncem a jejich pohybu. Na konci této kapitoly se budu zabývat testováním celé aplikace a dosažené výsledky se pokusím zhodnotit.

3.1 Generování mraků

Prostor oblohy, ve kterém se mohou nacházet mraky, jsem si rozdělil na stejně velké krychle a každá z těchto krychlí reprezentuje jeden prvek pole. V každém políčku se může vyskytovat nejvýše jeden mrak a tento mrak může ležet kdekoli v prostoru vyhrazeném pro toto políčko a může i částečně přesahovat do prostoru sousedících políček. Tento přesah je dovolen z toho důvodu, aby jednotlivé mraky nevypadaly příliš izolovaně a mohly vytvářet shluky.

Generování mraků jsem si rozdělil na tři části. V první části nejprve zjišťuji, jestli pro daný prvek v poli bude mrak existovat či nikoliv, což záleží pouze na zadané pravděpodobnosti vzniku mraku. Takto získané pole by se už dalo použít pro vykreslení oblohy, ale mraky jsou pouze náhodně rozmístěné s větší či menší hustotou a neshlukují se do větších oblak.

Proto v dalším kroku projde pole několika iteracemi, ve kterých se ke každému prvku pole zjišťují jeho sousedící prvky a podle toho, v kolika z nich existuje mrak a jak velkou plochou spolu sousedí, dostane každý tento prvek přidělený počet bodů. Pokud spolu sousedí celou jednou stranou, přičtou se mu tři body, pokud sousedí jednou hranou, přičtou se mu dva body a pokud spolu sousedí pouze jedním vrcholem, přičte se mu jeden bod. Samozřejmě, že pokud spolu nesousedí, neboli v sousedním políčku mrak neexistuje, nepřičte se mu bod žádný. Pokud v políčku, pro které zjišťujeme sousedy mrak existuje, dostane navíc čtyři body. V takto ohodnocených políčkách, pokud mají malý počet bodů, může mrak zaniknout a pokud naopak dostala velký počet bodů a mrak v nich neexistuje, mají velkou pravděpodobnost, že v nich mrak vznikne. Tímto způsobem dochází postupně k tomu, že se mraky shlukují do skupin. Čím více iterací provádíme, tím více se mraky shlukují a vytvářejí se velká oblaka a malé osamocené mráčky se ztrácejí.

V těchto dvou krocích se pouze zjišťovalo, v kterých prvcích pole bude mrak existovat. Nyní přichází třetí fáze, ve které se pro každý prvek s mrakem vygeneruje jeho pozice a rozměr tak, aby se celý mrak nacházel v místě určeném pro tento prvek, nebo v povoleném přesahu k sousedům. Dále se mu také přiřadí konkrétní textura, kterou přiděluji pouze náhodně. Poté se ještě spočítá počet mraků, které se nachází bezprostředně nad tímto mrakem, kvůli pozdějšímu určování odstínu mraku.

Tímto máme vygenerované pole mraků, kde každý mrak má určenou svojí pozici, velikost a texturu. Pracovat s tímto polem by ale bylo neefektivní, protože skoro nikdy by nebyly vyplněné všechny prvky a také by nám práci s ním ztěžovalo to, že je trojrozměrné. Proto jsem se rozhodl převést toto pole na jednorozměrné a vkládat do něj pouze existující mraky. Na uspořádání mraků v tomto poli již nezáleží, protože svojí pozici už přidělenou mají. Tuto pozici ale ještě částečně upravíme, protože doposud prezentované mraky tvořily rovinnou oblohu, což by nevypadalo příliš věrohodně a proto jí musíme zakulatit. Na to nám stačí pouze upravit skutečnou výšku mraku na zobrazovanou, která je na okrajích pole nulová a směrem ke středu pole se po sinusové křivce blíží ke skutečné výšce. Na konec ještě spočítáme odstín

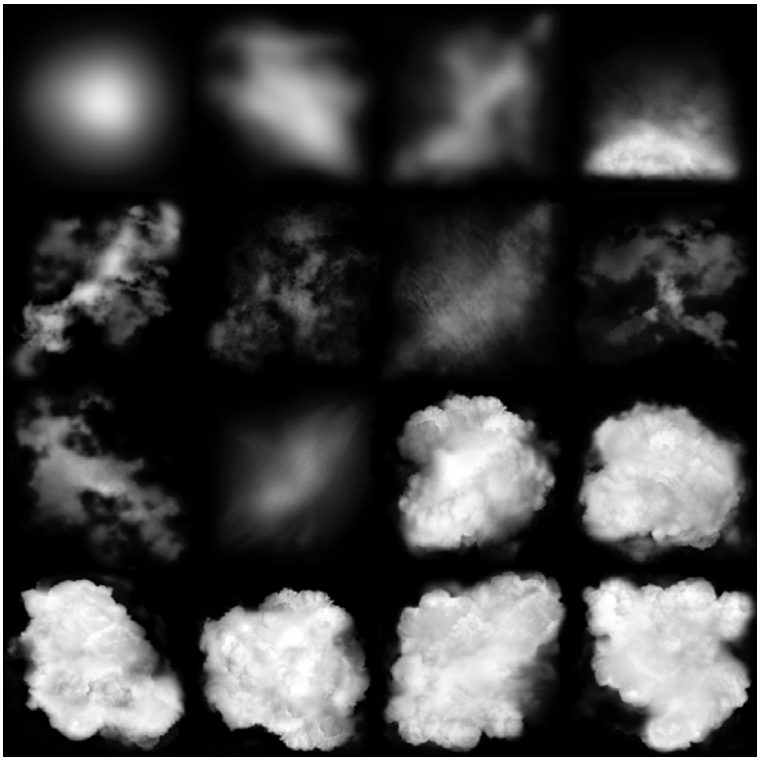
mraku. Protože barvu mraku vyjadřujeme pouze v odstínech šedi, stačí na určení barvy jedno číslo v rozsahu od 0 do 255, které pak při vytváření barvy v systému RGB přiřadím všem třem jeho složkám. Na hodnotě čísla odstínu se podílí různou měrou několik faktorů, například skutečná výška v jaké se mrak nachází, počet mraků, které jsou bezprostředně nad ním, počet sousedících mraků, celkový počet mraků a další. Na odstínu mraku poměrně hodně záleží, aby obloha vypadala co možná nejrealističtěji, proto je výpočet poměrně složitý, ale zdaleka není dokonalý.

Tato metoda poskytovala v některých situacích poměrně hezké výsledky, ale objevily se problémy při práci s proměnným osvětlením od Slunce, protože celá obloha byla reprezentována jednotlivými billboardy a bylo by výpočetně velmi náročné zjišťovat pro každý billboard, zda se nachází v nějakém shluku či nikoliv. Dále také nebylo možné mraky generovat postupně, tak jak se objevují za obzorem, ale musela se vygenerovat celá obloha najednou. Proto jsem se rozhodl navrhnout metodu, která bude generovat pouze jednotlivé oblaky a ty si bude uchovávat v seznamu. Tím je možné přidávat nebo odebírat pouze jednotlivé mraky a také tím, že má každý mrak svou polohu, je možné pracovat s jeho osvětlením od Slunce.

V tomto novém přístupu existuje pro generování každého typu mraku jedna samostatná metoda. Pro mrak typu Cumulus se nejprve vytvoří trojrozměrné pole o velikosti, která je určena velikostí mraku. Toto pole reprezentuje prostor ve tvaru kvádru, v němž se bude generovaný mrak nacházet. Každý prvek tohoto pole bude později nahrazen jedním billboardem, pokud v sobě ponese informaci o tom, že se v daném místě mrak nachází. Dále se vezme políčko uprostřed spodní stěny tohoto kvádru, vygeneruje se číslo z intervalu $\langle 0;1)$ a podle pravděpodobnosti vzniku mraku se rozhodne, zda se na tomto políčku bude mrak vyskytovat, či nikoliv. Pokud se zde mrak vyskytovat bude, vezmou se všechna čtyři políčka, které s tímto políčkem sousedí celou stěnou a zároveň se nacházejí také na spodní stěně kvádru a podobným způsobem se každému určí, zda se v jeho prostoru mrak vyskytuje. Těchto pět políček slouží jako základ pro růst mraku. V iteracích se pak prochází celé pole představující prostor mraku a pro každé políčko se spočítá počet sousedících políček, ve kterých se již mrak vyskytuje, přičemž některá políčka mají větší váhu podle velikosti plochy, se kterou s daným políčkem sousedí. Po spočítání počtu sousedících políček se opět určí, zda se na daném políčku vyskytuje mrak, ale s tím rozdílem, že je pravděpodobnost výskytu mraku ovlivněna právě tímto počtem sousedících políček a také počtem již provedených iterací. S čím větším počtem

políček dané políčko sousedí, tím má větší pravděpodobnost, že se v něm bude mrak vyskytovat. Naopak čím více iterací proběhne, tím se bude pravděpodobnost výskytu mraku snižovat. Po proběhnutí všech iterací je v prostoru mraku určeno, v jakých políčkách se bude vyskytovat. Nyní se pro všechna tato políčka náhodně, přibližně v prostoru tohoto políčka vygeneruje pozice, kde se bude nacházet billboard, také se mu přidělí jedna z textur. Pro modelování mraků se používá 16 poloprůhledných textur, které jsou zobrazeny na obrázku 3.1. a spočítá se poloměr celého mraku tak, že se určí vzdálenost nejvzdálenějšího billboardu od středu. Teď už stačí jen převést pole reprezentující prostor mraku na pole billboardů a mrak je vygenerován. Generování jednotlivých mraků typu Cumulus je také ovlivněno pravděpodobností vzniku mraku tak, že pokud je tato pravděpodobnost menší než 75%, jsou generovány menší mráčky s pravděpodobností 90%, což vytváří lepší vizuální dojem.

Mrak typu stratus se generuje podobným způsobem jako mraky typu Cumulus, jen s tím rozdílem, že existuje pouze jako jeden jediný mrak, který pokrývá celou oblohu. Jeho tvar je stále stejný a tvoří plášť polokoule složený ze tří vrstev billboardů. Nejbližší vrstva k pozorovateli je tvořena průhlednější texturou a naopak nejvzdálenější vrstva je tvořena téměř neprůhlednou texturou. Struktura mraku je tedy pořád stejná, mění se pouze jeho barva.



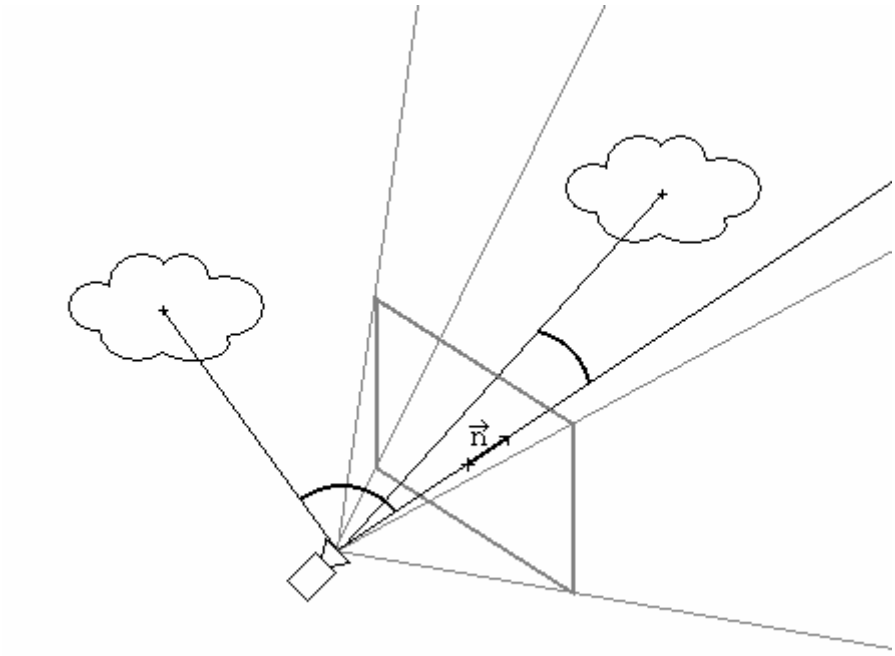
Obrázek 3.1: Textury mraků, obrázek pochází z [2]

3.2 Vykreslování mraků

Metoda pro vykreslování mraků vzhled mraků sice vůbec neovlivní, ale zato může ovlivnit náročnost celé aplikace, proto se je zde nezbytně nutné zjednodušit všechny výpočty, používat co možná nejrychlejší algoritmy a vykreslovat pouze to, co je doopravdy vidět.

Na začátku celé metody se nejprve provádí ořezávání, neboli ze všech vykreslovaných objektů se vyberou ty, které budou na obrazovce doopravdy vidět a dále se pracuje pouze s nimi, čímž se výpočet značně urychlí. První typ ořezávání, který jsem používal, byl poměrně složitý na výpočet a navíc v některých případech hodně neefektivní, ale i tak byl jeho vliv na rychlost vykreslování znatelný. Toto ořezávání fungovalo tak, že se nejprve pro každý mrak zjistilo, jaké by se u pozorovatele naměřili úhly pohledu (tedy *azimuth* a *zenith*), kdyby se díval přesně na daný mrak a potom se porovnávaly s aktuálními úhly pohledu pozorovatele. Pokud se lišily více než byla dovolená odchylka, byl mrak z pole vypuštěn. Problém byl spojit porovnávání obou úhlů tak, aby při vodorovném pohledu (*zenith* = 0°) záviselo vypouštění mraků na obou úhlech pohledu a při svislém pohledu vzhůru (*zenith* = 90°) záviselo vypouštění mraků pouze na úhlu *zenith*. Toto spojení porovnávání obou úhlů pak způsobovalo, že při pohledu s úhlem *zenithu* přibližně 45° se vykreslovalo příliš mnoho mraků, které nebyly viditelné.

Nynější ořezávání je navrženo tak, že se nejdříve prochází všechny objekty reprezentující celé mraky. Podle velikosti úhlu, který svírá normálový vektor promítací roviny s vektorem určeným pozicí pozorovatele a pozicí středu mraku (viz Obrázek 3.2), se rozhodne, zda může být alespoň část mraku viditelná, či nikoliv. Pokud mrak viditelný není, dále se s ním už nepracuje. Pokud nějaká jeho část viditelná je, pokračuje se podobným způsobem se všemi billboardy, které tento mrak tvoří. Opět se prochází jeden billboard po druhém a zkoumá se velikost úhlu, který svírá normálový vektor promítací roviny s vektorem určeným pozicí pozorovatele a pozicí billboardu. Pokud je billboard viditelný, vloží se do pole vykreslovaných billboardů. Po průchodu všech mraků a jejich billboardů je rozhodnuto, které billboardy budou vykreslovány. Tyto billboardy jsou umístěny v poli vykreslovaných billboardů a dále se pracuje pouze s nimi.



Obrázek 3.2: Ořezávání

Protože každý billboard je tvořen poloprůhlednou texturou, je nutné kvůli použití z-bufferu řešení viditelnosti. K tomu je použit malířův algoritmus (viz [4]). Tato metoda je založena na myšlence vykreslování všech ploch postupně odzadu dopředu podobně, jako kdyby malíř nanášel na obraz nejprve barvy pozadí a přes ně kreslil objekty v popředí. Odtud plyne anglický název metody „painter's algorithm", někdy též „priority list". Plochy bližší k pozorovateli jsou vykreslovány později, takže překryjí zadní plochy. Řešení viditelnosti je tedy převedeno na úlohu seřazení ploch podle vzdálenosti od pozorovatele. Předpokládáme, že zpracováváme rovinné plochy, které se navzájem neprotínají (neprosekávaní). V našem případě tedy stačí ke každému billboardu spočítat vzdálenost tohoto billboardu od promítací roviny (tedy roviny, jejíž normálový vektor je směrovým vektorem pohledu pozorovatele), protože všechny billboardy jsou s touto rovinou rovnoběžné. Pole billboardů se pak seřadí podle této vzdálenosti sestupně, tedy tak, že nejvzdálenější mraky budou na začátku pole a nejbližší na konci. Nakonec se už jen všechny billboardy vykreslí.

3.3 Osvětlení mraků Sluncem

Barva každého mraku závisí převážně na barvě světla slunečních paprsků a na jejich směru odkud přichází. Dále se tedy pro zjednodušení předpokládá, že barvu

každého mraku neovlivňuje nic jiného než právě barva a směr slunečních paprsků, tedy že veškeré světlo, které na mraky dopadá pochází pouze přímo od Slunce a že se mraky vzájemně nezastiňují.

Vzhledem k tomu, že barevný tón a sytost barvy budou pro všechny části mraku stejné a bude se měnit pouze jas, není příliš vhodné používat barevný prostor RGB, který rozkládá barvu do těchto tří složek: červená (R, red), zelená (G, green), modrá (B, blue). Mnohem vhodnější je využívat barevný prostor HSV, jehož hlavními parametry jsou barevný tón (H, hue), který označuje převládající spektrální barvu, sytost (S, saturation), která určuje příměs jiných barev a jasová hodnota (V, value), která určuje množství bílého (bezbarvého) světla.

Jak už bylo řečeno, barevný tón a sytost se měnit nebudou, u každého mraku se tedy bude pracovat pouze s jasovou hodnotou. Nejjasnější část mraku bude ta, která je ke Slunci přikloněna a naopak nejtmavší bude ta, která je od Slunce odkloněna. Každý mrak se skládá ze shluku několika billboardů, které tvoří přibližně kouli. Nejjasnější z nich bude ten, který je na přikloněné straně, tedy ten, který je ke Slunci nejblíže. Podobně nejtmavší bude ten, který je od Slunce nejdále. Proto se nejbližšímu billboardu přiřadí jas s maximální hodnotou určenou pro mrak a nejvzdálenějšímu se přiřadí jas s minimální hodnotou určenou pro mrak. Ostatním billboardům se barva určí lineární interpolací podle vzdálenosti od Slunce.

3.4 Pohyb mraků

Protože mrak typu Stratus je jen mlhovina bez viditelné struktury pokrývající celou oblohu, je vytvořen pouze jako jeden jediný mrak vyplňující celou oblohu a je statický. Pohyb mraků se tedy týká pouze mraků typu Cumulus.

Jediným faktorem, který ovlivňuje pohyb mraků, je vítr. Mraky se pohybují tak rychle, jak rychle vítr fouká, a ve směru, ve kterém fouká. Proto jsou jeho nejdůležitějšími parametry rychlost a směr.

Metoda pro pohyb mraků nezajišťuje jen jejich pohyb, ale stará se i o jejich generování a zánik, pokud se dostanou příliš daleko. Na začátku metody se nejprve spočítá, o jakou vzdálenost se mají mraky posunout. Tato vzdálenost je závislá na rychlosti větru a na rychlosti vykreslování, tedy počtu snímků za sekundu (FPS, frames per second). Poté se k pozici všech mraků přičte vektor, který má směr

rovnoběžný se směrem větru a velikost rovnou vzdálenosti, kterou mají mraky urazit. Tento vektor se nemusí přičítat k pozici všech billboardů, které mrak tvoří, protože každý billboard má určenou svojí pozici vůči středu mraku, proto stačí, když se změní pouze poloha celého mraku a změní se tím i poloha všech billboardů. Pro každý mrak se ještě upraví jeho výška podle jeho nové polohy.

V předchozím odstavci byl popsán pohyb mraků, nyní bude vysvětlen jejich zánik a v dalším odstavci jejich generování. Podmínka pro zánik je poměrně jednoduchá, pokud se mrak dostane do vzdálenosti větší, než je daná konstanta pro zánik, odstraní se ze seznamu všech mraků. Tato podmínka se testuje u všech mraků ve scéně.

Generování je o něco složitější. Existuje zde jedna proměnná, která si pamatuje, o kolik se mraky posunuly od posledního generování. Pokud se posunuly o více než je daná konstanta, tato konstanta se odečte od vzdálenosti posunutí a vygeneruje se nový pruh mraků, který je kolmý na směr větru a vyskytuje se za obzorem. Generování nových mraků zajišťuje zvláštní vlákno tak, aby byly připraveny dopředu před použitím, aby se aplikace urychlila.

3.5 Testování

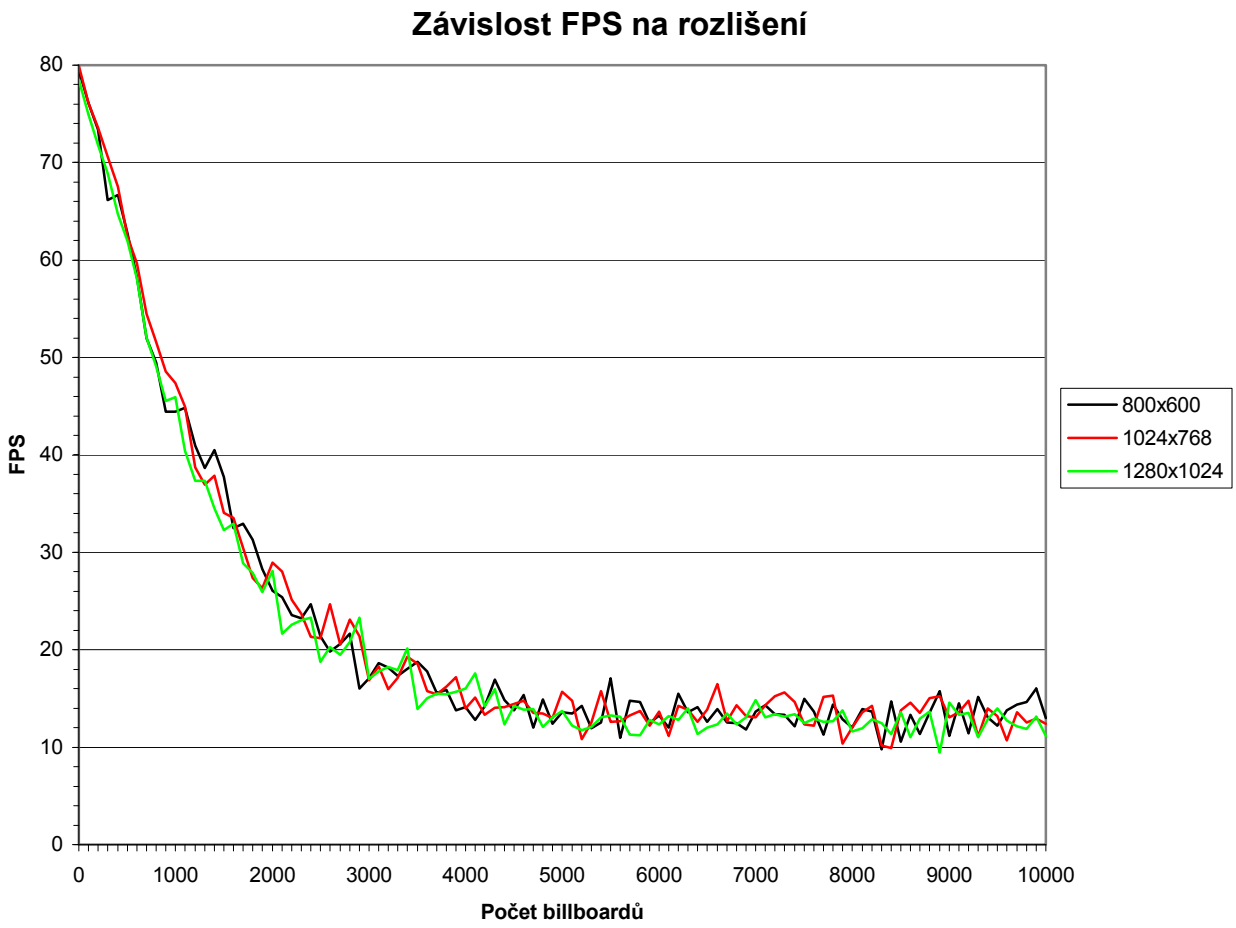
Většina testů byla uskutečněna na počítačové sestavě s touto hardwarovou konfigurací: procesor Intel Celeron D 2,4 GHz, operační paměť 512 MB DDR, grafická karta ATI Radeon X1600 Pro s pamětí 256 MB. Test závislosti FPS na pokrytí oblohy byl navíc proveden i na sestavě se stejným procesorem a operační pamětí, ale s grafickou kartou Geforce MX4000 s pamětí 64 MB.

Testování u prvních třech testů jsem prováděl tak, že jsem spustil aplikaci, nastavil požadované parametry a počet mraků jsem zvýšil na maximum. Potom jsem začal s měřením počtu snímků za sekundu (FPS). Prováděl jsem deset měření pro daný počet mraků a do grafu jsem vynášel jejich průměrnou hodnotu. Po těchto deseti měřeních jsem snížil počet mraků o určitou hodnotu a měření opakoval. Měření skončilo po naměření deseti hodnot bez mraků.

U dalších testů jsem opět po spuštění aplikace nejprve nastavil požadované parametry a poté jsem počkal, až se celá obloha zaplní nově vygenerovanými mraky.

Když byla obloha zaplněna, naměřil jsem opět deset hodnot pro dané parametry a jejich průměr jsem vynášel do grafu.

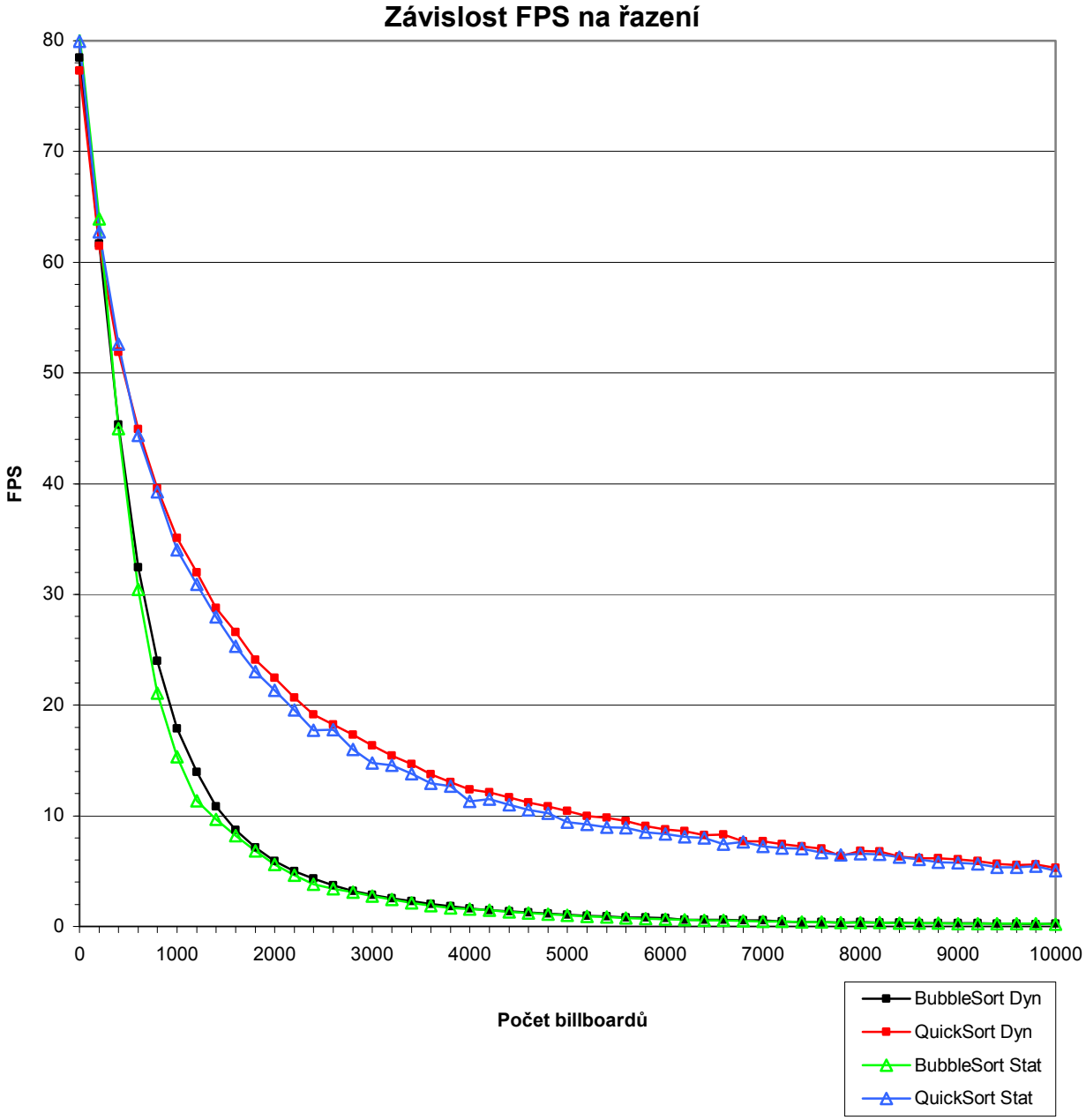
První věc, kterou jsem testoval, byla závislost FPS na rozlišení. Aplikaci jsem testoval pro rozlišení 800 x 600, 1024 x 768 a 1280 x 1024. Jak je patrné z grafu 3.1, rozlišení na počet snímků za sekundu vliv nemá skoro žádný, i když jsem předpokládal, že v rozlišení 800 x 600 bude snímková frekvence nejvyšší a v rozlišení1280 x 1024 nejnižší. Nejspíše to bude tím, že vykreslování nejvíce brzdí ořezávání a řazení billboardů, které je pořád stejné, a samotné vykreslení billboardů je pak pro všechna rozlišení časově přibližně stejně náročné.



Graf 3.1 – Závislost FPS na rozlišení

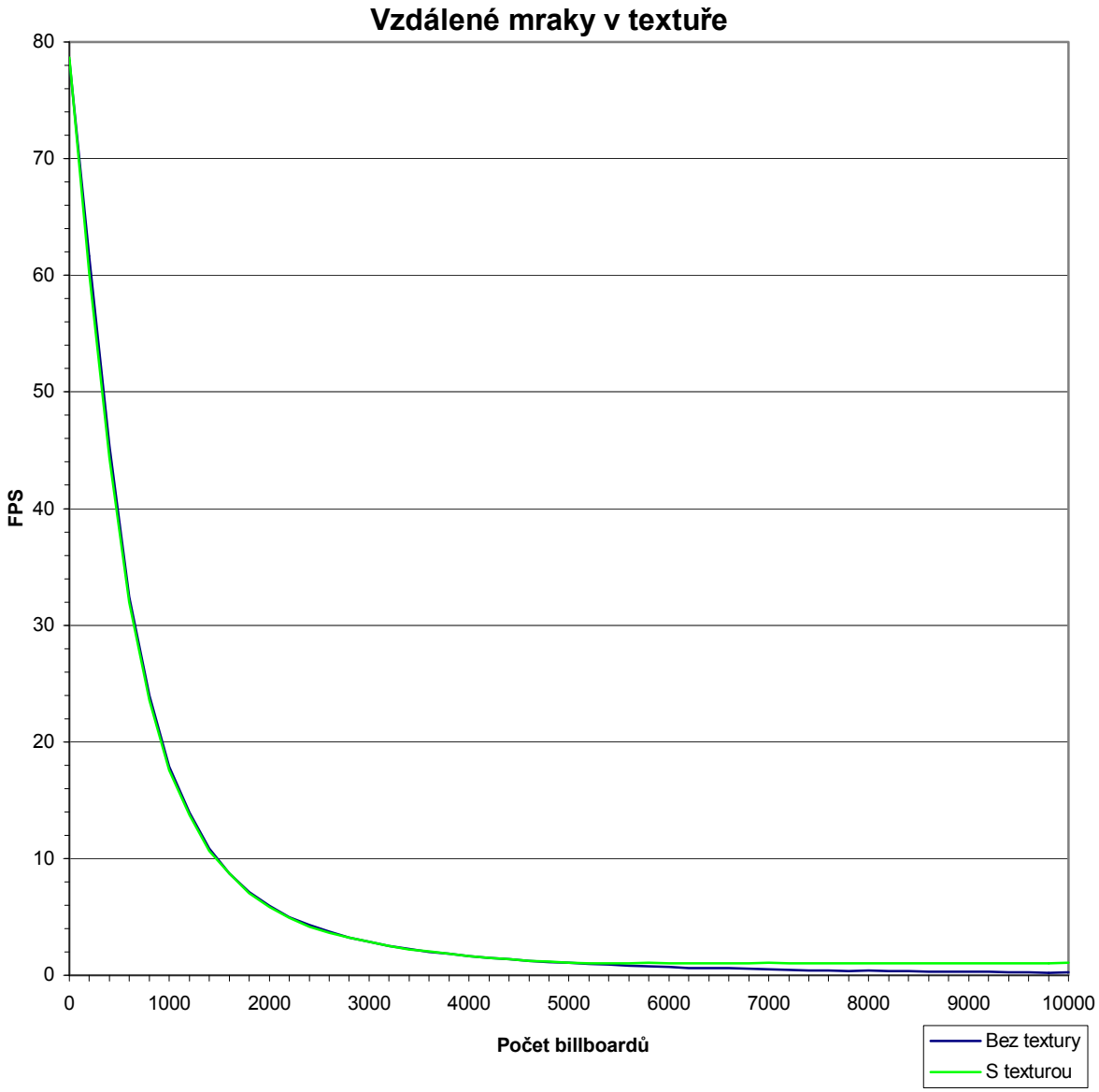
Druhý test, který jsem prováděl, byla závislost FPS na druhu řazení billboardů při vykreslování mraků statické oblohy. Testoval jsem BubbleSort a QuickSort jak pro statickou, tak pro dynamickou kameru, tedy pro kameru, která se nepohybuje a dívá se pořád stejným směrem, a kameru, jejíž směr pohledu se neustále mění. Byl jsem překvapen, jak velký byl mezi nimi rozdíl. QuickSort měl FPS v průměru zhruba

o deset snímků vyšší. Porovnání vidíte na grafu 3.2. Je možné, že pokud by se použilo ještě rychlejší řazení, než je QuickSort, mohlo by to vykreslování mraků ještě více urychlit.



Graf 3.2 – Závislost FPS na řazení

Ve třetím testu jsem zkoumal, jaký vliv na FPS bude mít vykreslování vzdálených mraků do textury. Jak je vidět na grafu 3.3, znatelný rozdíl se projevuje až při vyšším počtu mraků. Samozřejmě také hodně závisí na vzdálenosti, od které se mraky do textury vykreslují, čím kratší je tato vzdálenost, tím znatelnější rozdíl bude vykreslování do textury mít.

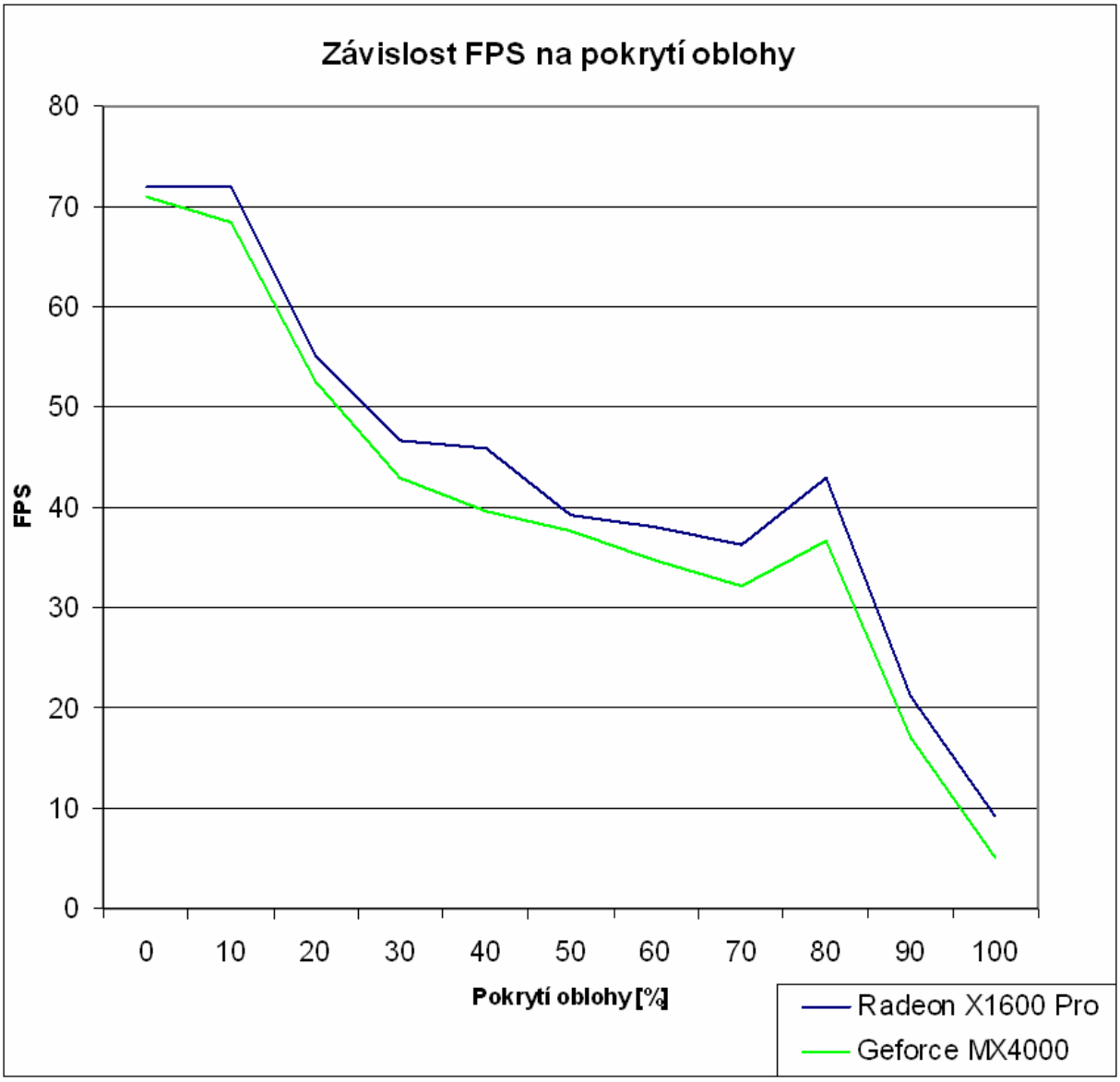


Graf 3.3 – Závislost FPS na renderování vzdálených mraků do textury

Tyto tři testy byly prováděny ještě na starší verzi. V aktuální verzi je již vylepšeno ořezávání, takže naměřené hodnoty by byly o něco vyšší, ale vliv na průběh grafu by neměl být téměř žádný.

Následující test, jehož výsledek je znázorněn v grafu 3.4, byl již prováděn na nejnovější verzi knihovny a uvádí závislost snímkové frekvence na procentuálním pokrytí oblohy. Test byl prováděn na dvou různých grafických kartách. Jak se dalo očekávat, platí zde nepřímá úměrnost, čím vyšší je pokrytí oblohy, tím nižší je počet snímků za sekundu. Jak je vidět z grafu, při 80% se FPS mírně zvýší, to je způsobeno tím, že do pravděpodobnosti vzniku 75% jsou mraky vytvářeny trochu

odlišným způsobem a jsou tvořeny větším počtem billboardů, než mraky s pravděpodobností vzniku nad 75%.



Graf 3.4 – Závislost FPS na pokrytí oblohy

Tabulka 3.1 ukazuje průměrnou rychlost vykreslování pouze pro mraky typu Status, pouze pro mraky typu Cumulus a pro oba typy mraků vykreslované najednou. Mraky typu Cumulus v obou případech pokrývají oblohu ze 100 %. Jak je na první pohled zřejmé, vykreslování mraků typu Cumulus je několikrát pomalejší, než vykreslování mraků typu Stratus.

Stratus	29,27
Cumulus	9,51
Cumulus + Stratus	7,49

Tabulka 3.1 – FPS pro různé typy mraků

Poslední tabulka 3.2 slouží k porovnání rychlosti vykreslování bez ořezávání, tedy pokud se vykreslují všechny mraky ve scéně, a rychlosti vykreslování s ořezáváním, kdy se vykreslují pouze viditelné mraky. Jak je na první pohled vidět, ořezávání je velmi účinné.

Bez ořezávání	2,03
S ořezáváním	9,46

Tabulka 3.2 – Srovnání FPS pro vykreslování s ořezáváním a bez ořezávání

Ze závěru měření vyplývá, že nejnáročnější je až samotné vykreslování billboardů, proto je velmi důležité co nejpřesnější ořezávání. Dalším slabým místem může být řazení billboardů kvůli řešení viditelnosti. Ostatní faktory už rychlost ovlivňují pouze zanedbatelně.

4 Závěr

Závěrem bych chtěl zhodnotit, čeho všeho jsem ve své práci dosáhl, jaké se vyskytly komplikace při řešení některých problémů a jak jsem je řešil.

Ve své bakalářské práci jsem se zabýval popisem několika metod pro modelování a zobrazování mraků. Navrhnul jsem funkční knihovnu pro generování a vykreslování mraků typu Cumulus a Stratus. Zaměřoval jsem se hlavně na co nejjednodušší ovládání a co nejmenší náročnost knihovny na hardware, ale i přesto je pro větší počet mraků poměrně náročná.

Jedním z asi největších problémů, na které jsem narazil, bylo řešení viditelnosti u billboardů s poloprůhlednou texturou, když jsem ještě neznal malířův algoritmus. Billboardy jsem sice zkoušel řadit, ale podle vzdálenosti od pozorovatele a nikoliv od celé roviny, což nefungovalo správně. Zkoušel jsem také místo billboardů používat vlastní čtverec s texturou, zde jsem ale narazil na problém s natáčením čtverce přímo proti pozorovateli v některých pozicích. Naštěstí díky malířově algoritmu jsem se mohl vrátit k původnímu řešení zobrazování mraků přes billboardy.

Dalším problémem, kterým jsem se musel zabývat bylo postupné generování mraků a jejich osvětlení Sluncem, z tohoto důvodu jsem se rozhodl od základu předělat metodu pro generování mraků, o čemž píši v kapitole 3.1.

Ostatní problémy jsem vyřešil bez větších komplikací a program by měl fungovat tak, jak byl popsán výše v této dokumentaci.

Výsledkem mé dosavadní práce je tedy program, který umí zobrazovat oblohu s mraky. Tyto mraky jsou tvořeny billboardy a díky tomu vypadají volumetricky a je možné jimi i prolétávat. Jako celek vypadají mraky v některých případech poměrně realisticky, v jiných případech už tolik ne.

Námětem pro další rozšíření práce by mohlo být dodělání metody pro generování mraku typu Cirrus, který se na obloze vyskytuje poměrně často v kombinaci s mraky typu Cumulus. Dále by bylo vhodné rozšířit tuto knihovnu o atmosférické jevy, jako je například déšť a sníh, které často mraky doprovází.

Literatura

[1] DOBASHI, Y. – KANEDA, K. – YAMASHITA, H. – OKITA, T. – NISHITA, T. *A Simple, Efficient Method for Realistic Animation of Clouds*. Proc. SIGGRAPH2000, 2000-7, pp. 19-28.
URL: nis-ei.eng.hokudai.ac.jp/~doba/papers/sig00_cloud.pdf

[2] WANG, Niniane. *Realistic and Fast Cloud Rendering*. 2003.
URL: ofb.net/~niniane/clouds-jgt.pdf

[3] HARTUS, M. J. – LASTRA, A. *Real-Time Cloud Rendering*. EUROGRAPHICS2001, 2001.
URL: www.markmark.net/PDFs/RTClouds_HarrisEG2001.pdf

[4] ŽÁRA, J. – BENEŠ, B. – SOCHOR, J. – FELKEL, P. *Moderní počítačová grafika.* 1. vyd. Brno: Computer Press, 2004. ISBN 80-251-0454-0.

[5] *Atlas oblaků*. URL: mraky.astronomie.cz [cit. 2007-4-18]

[6] *Euromarina – Formování oblačnosti*.
URL: www.euromarina.cz/pocasi/meteorologie/formovani%20oblacnosti.htm
[cit. 2007-4-18]

[7] *Oblačnost a srážky*.
URL: artemis.osu.cz/Gemet/meteo2/obla%C4%8Dnos.htm
[cit. 2007-4-18]

[8] SCHPOK, J. – SIMONS, J. – EBERT, D. S. – HANSEN, C. *A Real-Time Cloud Modeling, Rendering, and Animation System*. Eurographics/SIGGRAPH Symposium on Computer Animation, 2003.
URL: www.ecn.purdue.edu/purpl/level2/papers/scaclouds.pdf

[9] MIYAZAKI, R. – YOSHIDA, S. – DOBASHI, Y. – NISHITA, T. *A Method for Modeling Clouds based on Atmospheric Fluid Dynamics*. EUROGRAPHICS 2002, Short Presentation, pp.405-410, 2002-9.
URL: nis-lab.is.s.u-tokyo.ac.jp/~nis/cdrom/pg/PG2001_ryomiya.pdf

[10] NISHITA, T. – DOBASHI, Y. – NAKAMAE. E. *Display of Clouds Taking into Account Multiple Anisotropic Scattering and Sky Light*. SIGGRAPH, 1996. *URL: delivery.acm.org/10.1145/240000/237277/p379-nishita.pdf?key1=237277&key2=0852308711 &coll=GUIDE&dl=GUIDE&CFID=21408226&CFTOKEN=67048256*

[11] HARRIS, M. J. – BAXTER III, W. V. – SCHEUERMANN, T. – LASTRA, A. *Simulation of Cloud Dynamics on Graphics Hardware*. The Eurographics Association, 2003. *URL: www.markmark.net/cloudsim/harrisGH2003.pdf*

[12] NISHITA, T. – DOBASHI, Y. *Modeling and Rendering of Various Natural Phenomena Consisting of Particles*. Proc. Computer Graphics International 2001, pp. 149-156, 2001. *URL: nis-ei.eng.hokudai.ac.jp/~doba/papers/cgi01_invited.pdf*

[13] MAN, Petr. *Generating and Real-Time Rendering of Clouds*. Czech Technical University in Prague, 2006.
URL: www.cescg.org/CESCG-2006/papers/Prague-Man-Petr.pdf

Příloha A

Ukázky typů mraků v přírodě

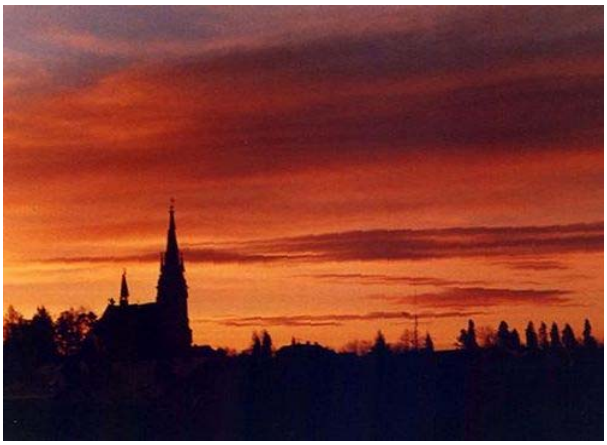
(Obrázky pocházejí z [5].)

Cirrus:

Cumulus:

Stratus:

Altocumulus:



Altostratus:

Cirrocumulus:

Cirrostratus:

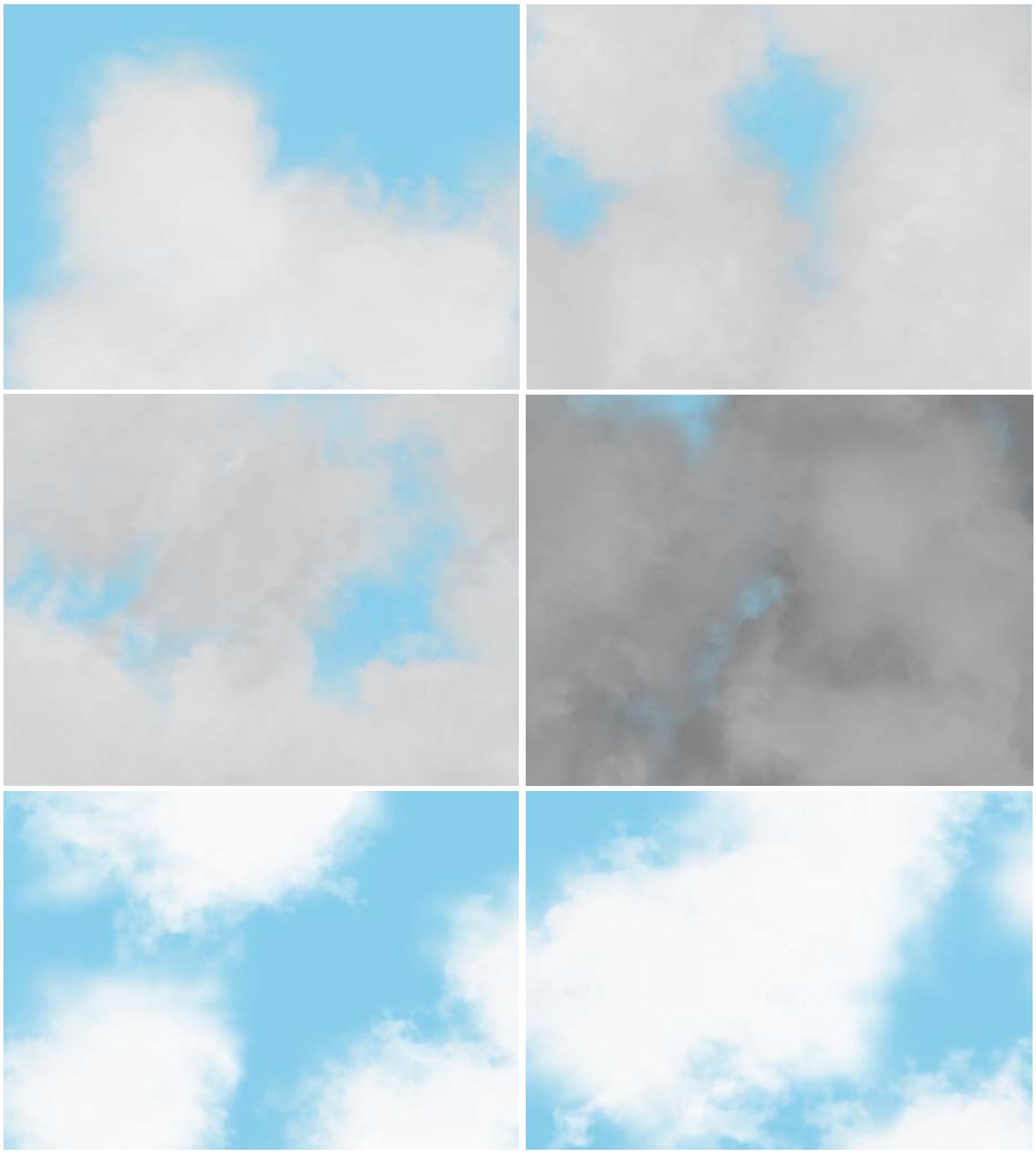
Cumulonimbus:

Nimbostratus:

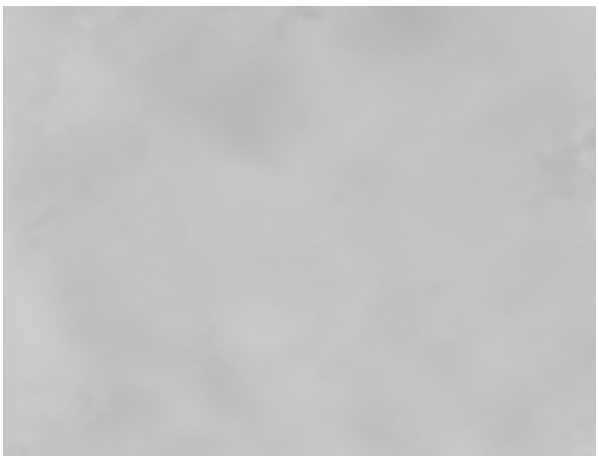
Stratocumulus:

Ukázky výstupu knihovny

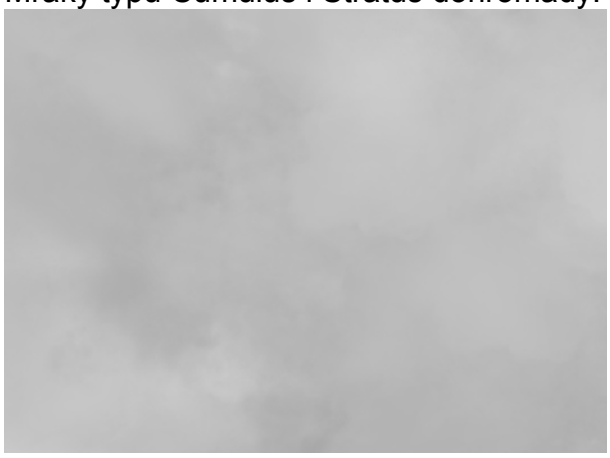
Mraky typu Cumulus:



Mraky typu Stratus:

Mraky typu Cumulus i Stratus dohromady:




Příloha B

Uživatelská příručka

Použití grafické knihovny

Před použitím knihovny je nutné ji nakopírovat i spolu s obrázky m1.tga – m16.tga do složky, ve které se bude naše aplikace spouštět. Knihovna obsahuje dvě třídy *CloudsModeling* a *CloudsException* a jeden výčtový typ *CloudsType*. Výčtový typ *CloudsType* obsahuje všechny typy mraků, které umí knihovna vymodelovat, zatím tedy pouze typy Cumulus a Stratus. Třída *CloudsException* slouží pouze pro vyvolávání výjimky, která je vyhozena při nezadání směru větru.

Nejdůležitější je tedy třída *CloudsModeling*, která slouží ke generování i vykreslování mraků jednoho typu. Obsahuje dva konstruktory, jeden pouze s parametry *type*, *dev* a *centre*, které určují typ mraků, zařízení na kterém se budou vykreslovat a střed kolem kterého budou obíhat. Ostatní parametry jsou nastaveny imlicitně tak, aby obloha vypadala co nejlépe. U druhého konstruktoru je možné zvolit si všechny parametry samostatně. Knihovna ovšem také dovoluje všechny tyto parametry měnit i za běhu programu.

Tyto parametry jsou:

CloudsType type	- typ mraku
float deltaC	- povolený úhel pro ořezávání celých mraků
float deltaB	- povolený úhel pro ořezávání billboardů
float horizon	- vzdálenost horizontu
float underHorizon	- vzdálenost za horizontem, za krerou mraky mizí
int boxes	- počet políček pole ve kterém se generuje
float boxWidth	- šířka políčka pole ve kterém se generuje
float heightMin	- minimální výška ve které se nachazejí mraky
float heightMax	- maximální výška ve které se nachazejí mraky
float cloudSizeMin	- minimální rozměr mraku
float cloudSizeMax	- maximální rozměr mraku
float stratusRadius	- poloměr stratosferických mraků

int stratusNumberOfLevel	- počet úrovní stratosferického mraku
Vector3 centre	- střed kolem kterého obíhají mraky
D3D.Device device	- zařízení na které se vykresluje

U vlastního nastavování parametrů je třeba si uvědomit, že ne pro jakékoliv kombinace musí třída poskytovat přijatelné výstupy. A také při pracování s oběma typy najednou, tzn. při použití dvou instancí této třídy, je potřeba dbát na to, aby mraky z obou tříd nikde nekolidovaly a vykreslovali se nejdříve ty vzdálenější, protože viditelnost je řešena jen v rámci třídy a mohlo by to působit problémy.

Pro samotné vykreslení mraků slouží přetížená metoda *Render*, buď s parametry *mov* a *viewDirection*, která vykreslí scénu s osvětlením Slunce, které bylo zadáno již dříve, nebo s parametry *mov*, *viewDirection*, *sunPosition*, *hue* a *saturation*, která vykreslí scénu s osvětlením Slunce, které je na pozici *sunPosition* a jehož barevný tón a sytost barvy světla jsou *hue* a *saturation*. Tato metoda by měla být volána vždy jako první, aby se nastavily parametry Slunce a pokud se nebude Slunce pohybovat, stačí dále volat jen metodu se dvěma parametry *mov* a *viewDirection*, které určují pozici a směr pohledu pozorovatele. Třída obsahuje také metodu *SetVisible*, která umožňuje nastavit, zda je mrak viditelný, či nikoliv.

Další poměrně důležitou metodou je metoda pro nastavování větru *SetWind*, která má parametry *direction* a *speed*, které slouží pro nastavení směru a rychlosti větru. Jinak by se mraky nepohybovaly.

Poslední užitečnou metodou by mohla být metoda *Clear*, která smaže všechny doposud vygenerované mraky v této třídě.

Ovládání aplikace využívající knihovnu

Ovládání celého programu je poměrně jednoduché. Pro rozhlížení slouží myš a pro chůzi vpřed, vzad, vlevo a vpravo klávesy W, S, A, D, jako v klasických hrách. Navíc je umožněn i pohyb nahoru a dolů pomocí kláves Q a Y. Pohyb je oproti velikosti všech objektů velmi rychlý. Je to z toho důvodu, aby bylo usnadněno prohlížení mraků z různých stran a úhlů.

Pro zobrazení lišty s dvěma teploměry sloužících k nastavení parametrů mraků je určena klávesa N. Pokud je zobrazena lišta pro nastavování nastavení platí pro mrak typu zobrazeného v horní části lišty. Tento typ se přepíná pomocí klávesy C

pro Cumulus a V pro Stratus. Pokud lišta zobrazena není, slouží tyto klávesy k zapínání nebo vypínání zobrazování daného typu mraku (k zapínání a vypínání zobrazování terénu slouží klávesa Z). Při nastavování parametrů pro daný typ mraku, slouží první (levý) teploměr pro nastavování pravděpodobnosti vzniku mraku, kde úplně dole je pravděpodobnost 0% a úplně nahoře 100%. Druhý teploměr slouží pro nastavování nasycení mraku, tedy jeho odstínu. Úplně dole znamená, že je nasycení nejmenší, mrak je bílý, a úplně nahoře znamená, že je nasycení největší a mrak je velmi tmavý.

Další parametr, který lze měnit je rychlost a směr větru. Rychlost se zvyšuje klávesou + na numerické části klávesnice a snižuje se klávesou –. Směr větru lze měnit směrovými šipkami levou a pravou, podle toho na kterou stranu se má vítr otáčet. Aplikace také umožňuje zapínání a vypínání mlhy tlačítkem F.

Posledním ovlivnitelným parametrem za běhu aplikace je rychlost pohybu Slunce, která se zvyšuje stiskem směrové šipky nahoru a snižuje směrovou šipkou dolů.

Program se ukončuje buď klávesou Escape, nebo klasickým stisknutím křížku v pravém horním rohu okna.