# SIMULATION OF WATER WAVES IN REAL-TIME

**Martin Pilch**
Master Degree Programme (2), FIT BUT
E-mail: xpilch00@stud.fit.vutbr.cz

Supervised by: Adam Herout
E-mail: herout@fit.vutbr.cz

**Abstract**: The goal of this work is creation of real-time simulation of the water waves. It is implemented on Mac OS X platform using OpenGL. This work is based on height map surface. Heigh map is computed by summing of sinusoids with complex, time-based amplitudes. Fast Fourier Transform, Phillips spectrum and gauss random generator are used to solve this problem. The program was also ported to iOS mobile platform. Built-in functions and shaders are use to compute FFT, normals and light model.

**Keywords**: water, waves, Phillips spectrum, FFT, OpenGL, GLSL ES

## 1 ÚVOD

Tématem práce je simulace vlnění vody v reálném čase. Cílem bylo vytvořit aplikaci zobrazující vizuálně přesvědčivé vlnění vody, která je schopna běžet v reálném čase jak na běžných počítačích, tak i na mobilních zařízeních.

V následujících kapitolách krátce rozeberu teoretickou část práce, především statistický model simulace vlnění vody založený na Phillipsově spektru a zmíním několik důležitých aspektů implementace, konkrétně implementaci rychlé Fourierovy transformace, rozložení zátěže mezi procesor a grafickou kartu a vývoj pro mobilní zařízení s operačním systémem iOS.

## 2 SIMULACE VLNĚNÍ VODY

Během vývoje jsem vyzkoušel několik metod simulace vlnění vody. Začal jsem jednoduchou metodou, v níž se vlnění šířilo z jednoho bodu všemi směry. Druhou pak byly Gertsnerovy vlny [2]. Tyto metody však byly příliš jednoduché nebo neposkytovaly přesvědčivé výsledky. Poté jsem přešel ke složitějším metodám, jako jsou NSE, Navier-Stokes rovnice. Jedná se o rovnice popisující proudění nestlačitelné Newtonské kapaliny [4]. Jsou ale příliš náročné pro výpočet v reálném čase, protože se pokoušejí o přesnou fyzikální simulaci chování kapalin.

Nakonec čerpám z práce J. Tessendorfa [1]. Ta je založena na výpočtu Phillipsova spektra [3]. Phillipsovo spektrum vychází ze statistické analýzy fotografií, radarových snímků a měření na otevřeném moři. Nejedná se tedy o metodu založenou na přesných fyzikálních výpočtech, které jsou příliš náročné pro zobrazení v reálném čase, přesto je vizuálně přesvědčivá.

Reprezentací výškového pole vlny je pak výška vlny $h = (x, t)$, kde $x$ je bod v prostoru $(x, z)$, jako suma sinusoid s komplexními, časově závislými amplitudami:

$$h(\mathbf{x},t) = \sum_k \tilde{h}(\mathbf{k},t) exp(i\mathbf{k}\cdot\mathbf{x}) \tag{1}$$

Kde $t$ je čas a $\mathbf{k}$ jsou vektory (používané i v následujících vztazích) představující vlny.

Tato suma je vypočtena za pomoci rychlé Fourierovy transformace. Sinusoidy jsou vypočítány z počátečních hodnot vygenerovaných pomocí Phillipsova spektra [3]: $P_h(k) = A\frac{exp(-1/(kL)^2)}{k^4}\left|\hat{\mathbf{k}}\cdot\hat{w}\right|^2$

V tomto vzorci $L = V^2/g$ představuje největší možnou vlnu, kterou může vítr o síle $V$ vytvořit. $g$ je gravitační konstanta, $\hat{w}$ pak směr větru. $A$ je numerická konstanta a faktor kosinu $\left|\hat{\mathbf{k}}\cdot\hat{w}\right|^2$ eliminuje vlny, které se pohybují kolmo ke směru větru.

Phillipsovo spektrum se používá ve vzorci pro generování počátečních hodnot komplexních amplitud $\tilde{h_0}(\mathbf{k}) = \frac{1}{\sqrt{2}}(\xi_r + i\xi_i)\sqrt{P_h(k)}$ kde $\xi_r, \xi_i$ jsou hodnoty z gaussovského generátoru náhodných čísel.

Do vzorce 1 se dosazují časově závislé amplitudy z tohoto vztahu:
$\tilde{h}(\mathbf{k},t) = \tilde{h_0}(\mathbf{k})exp\{i\omega(k)t\} + \tilde{h_0}^*(-\mathbf{k})exp\{-i\omega(k)t\}$

Pro vektory vln $\mathbf{k}_i$ pak platí, že $\omega^2(\mathbf{k}) = g\mathbf{k}$.

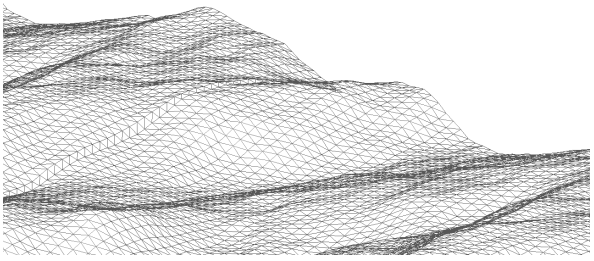
## 3 IMPLEMENTACE

Protože je práce vyvíjena v operačním systému Mac OS X, je snadno přenositelná na mobilní zařízení s operačním systémem iOS, která podporují rozhraní OpenGL ES, jazyk GLSL ES a knihovny pro výpočet rychlé Fourierovy transformace. Tato mobilní zařízení jsou zároveň dostatečně výkonná pro výpočet v reálném čase. Výpočet rychlé Fourierovy transformace a normál vrcholů probíhá na procesoru, vykreslování a výpočet osvětlení na grafické kartě.

Pro rychlý výpočet diskrétní Fourierovy transformace se používá *rychlá Fourierova transformace*, zkráceně *FFT*. Její princip spočívá v rozdělení výpočtu na menší části, které se postupně vypočítávají. Pro výpočet *FFT* jsem použil optimalizované funkce implementované v systému. Všechny potřebné funkce jsou obsaženy ve frameworku `Accelerate.framework`, konkrétně v hlavičkovém souboru
`vecLib/vDSP.h`. Nejdůležitější jsou funkce `vDSP_create_fftsetup` pro vytvoření struktury s nastavením a `vDSP_fft2d_zip` pro provedení *FFT* nad dvourozměrným polem [6].

Aby bylo vykreslování co nejrychlejší, probíhá ve *vertex shaderu*, programu pro práci s jednotlivými vrcholy. Při inicializaci je do paměti grafické karty pomocí `Vertex Buffer` a `Index Buffer` objektů uložena síť vrcholů představujících výškovou mapu s nulovou výškou, jinými slovy klidnou hladinu. Při každém překreslení je pak vertex shaderu předáno pouze pole s výškami, tedy *y-souřadnicemi* jednotlivých vrcholů. Ve vertex shaderu jsou pak tyto souřadnice přičteny k nulové výšce hladiny. Normály jsou předány *fragment shaderu*, programu pro zpracování jednotlivých pixelů, kde se pomocí *Phongova osvětlovacího modelu* vypočítá osvětlení [5].

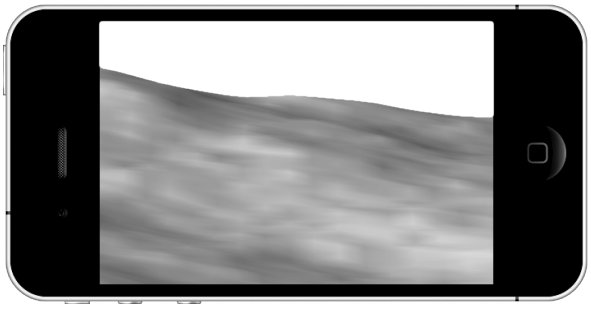
Na obrázku 1 je zobrazena trojúhelníková síť polygonů při výpočtu vlnění.



**Obrázek 1:** Implementace na operačním systému Mac OS X

Pro portaci na operační systém iOS bylo potřeba implementovat vlastní operace s *modelview* a *projection* maticemi pro práci s kamerou. Nejnovější implementace OpenGL ES totiž nepodporuje operace rozhraní OpenGL jako jsou `glRotatef` nebo `glTranslatef` a jiné.

Na obrázku 2 je zobrazen výpočet vlnění na reálném zařízení s operačním systémem iOS.



**Obrázek 2:** Implementace na zařízení s iOS

Na počítači s operačním systémem Mac OS X s integrovanou grafickou kartou nVidia GeForce 9400M probíhal výpočet s rychlostí 200 snímku za sekundu. Na mobilním zařízení iPhone 4 pak okolo 50 snímků za sekundu.

## 4 ZÁVĚR

V této práci se mi podařilo implementovat jednu z metod simulace vlnění vody. Protože se nejedná o náročnou fyzikální simulaci, je možné ji implementovat v reálném čase. Celý proces je optimalizován pro rozložení zátěže mezi procesor a grafickou kartu. Jsou použity vestavěné funkce systému pro výpočet rychlé Fourierovy transformace a výpočet světelného modelu je prováděn ve fragment shaderu. Proto je možné, aby simulace běžela i na mobilních zařízeních s operačním systémem iOS.

Při budoucím vývoji mám v plánu počítat odraz a lom světla pomocí *cube map*. Chci tak dosáhnout co nejrealističtějších výsledků. Práce chci použít jako základ zábavné multimediální aplikace nebo hry.

## REFERENCE


[1] Tessendorf, J.: Simulating Ocean Water, SIGGRAPH 2001

[2] Finch, M.: Chapter 1. Effective Water Simulation from Physical Models, GPU Gems: Programming Techniques, Tips and Tricks for Real-Time Graphics, Addison-Wesley Professional, 2004, s. 131-145, ISBN 0321228324

[3] Phillips, O. M.: On the generation of waves by turbulent wind, Journal of Fluid Mechanics. 2 (5): 417-445, 1957

[4] Acheson, D. J., Elementary Fluid Dynamics, Oxford Applied Mathematics and Computing Science Series, Oxford University Press, 1990, ISBN 0198596790

[5] Phong, B. T., Illumination for computer generated pictures, Communications of ACM 18, no. 6, s. 311-317, 1975

[6] Apple Inc.: vDSP Programming Guide, dokument dostupný na URL `http://developer.apple.com/library/mac/documentation/Performance/Conceptual/vDSP_Programming_Guide/Introduction/Introduction.html` (21.03.2011)