Vizualizarea interactivă la distanță a unor scene de obiecte 3D

Claudiu Mera

Catedra Calculatoare, Universitatea Tehnică Cluj-Napoca

Str. Bariţiu nr. 28, 400027, Cluj-Napoca

claude_mera@yahoo.com

Dorian Gorgan

Catedra Calculatoare, Universitatea Tehnică Cluj-Napoca

Str. Bariţiu nr. 28, 400027, Cluj-Napoca

dorian.gorgan@cs.utcluj.ro

REZUMAT

În acest articol vom descrie realizarea unei aplicaţii de vizualizare interactivă la distanţă a unor scene de obiecte 3D. În acest scop, se prezintă o aplicaţie interactivă care permite atât utilizatorilor unor sisteme cu capabilităţi grafice reduse, cât şi celor având sisteme performante, să vizualizeze scene ale căror complexitate grafică implică un efort de calcul semnificativ. Sunt introduse aspecte teoretice legate de activitatea curentă în acest domeniu. În finalul lucrării, se prezintă realizările obţinute prin implementarea aplicaţiei.

Cuvinte cheie

Serviciu Web, CUDA, zgomot Perlin, GPU

INTRODUCERE

De-a lungul anilor, au fost dezvoltate o serie de sisteme de vizualizare la distanţă a scenelor 3D. Un exemplu este ScanView [1], produs de către cei de la Universitatea Stanford, care prezintă principalul dezavantaj de a limita utilizatorul la o configuraţie grid existentă.

Odată cu apariţia arhitecturii Nvidia CUDA (*Compute Unified Device Architecture*) [2], posibilitatea de a reda scene grafice complexe a fost simplificată prin introducerea unui model revoluţionar de programare paralelă. Astfel, operaţiile computaţionale efectuate de către placa video sunt distribuite în mod egal către sub-unităţile arhitecturale ale acesteia (blocuri de procesare). În prezent, acest model este utilizat cu succes în diverse domenii precum cercetarea medicală, transcodarea video sau modelarea şi vizualizarea fenomenelor geografice.

Principalele obiective ale acestei lucrări sunt crearea unei aplicaţiei care să permită unui client specificarea parametrilor de vizualizare şi simulare ale unei scene 3D. Scena respectivă este redată de un server cu ajutorul tehnologiei CUDA, şi apoi transpusă în conţinut video trimis clientului pentru vizualizare.

ARHITECTURA SISTEMULUI DE VIZUALIZARE

Sistemul realizat este compus din două componente (Fig.1) :

a) Aplicaţia client - aplicaţie desktop care permite vizualizarea la distanţă a unor scene 3D

b) Aplicaţia server - serviciu Web folosit pentru generarea şi transmiterea de conţinut video

Aplicaţia client

Aplicaţia client este o aplicaţie desktop care comunică cu un server prin intermediul unui serviciu Web.

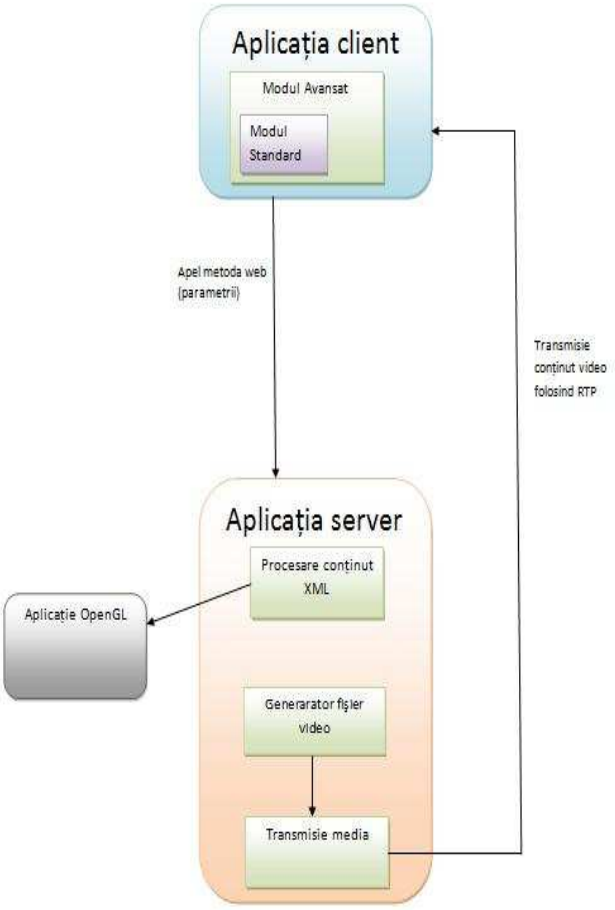

Fig. 1. Arhitectura sistemului

În funcţie de resursele grafice ale sistemului client, aplicaţia poate fi configurată în două moduri distincte:

Modul standard – recomandat utilizatorilor care nu deţin un sistem echipat cu o placă grafică (GPU) bazată pe arhitectura CUDA. Acest mod permite operaţii elementare precum selectarea unui scenariu şi specificarea parametrilor de vizualizare şi de simulare prin intermediul

unei interfeţe grafice. Astfel, utilizatorul trebuie să precizeze rezoluţia filmului ce urmează a fi generat, numărul de cadre pe secundă, precum şi alţi parametrii specifici scenariului ales. Parametrii specificaţi sunt transmişi în format XML către server prin apelul unei metode Web. Utilizatorul poate vizualiza ulterior conţinutul video generat de server pe baza parametrilor primiţi. O altă opţiune disponibilă în acest mod este aceea de a salva configuraţia existentă (constând din parametri specificaţi) într-un fişier XML, care poate fi încărcat ulterior când se doreşte solicitarea unei noi recepţii video.

Modul avansat - recomandat utilizatorilor care deţin un sistem echipat cu un GPU bazat pe arhitectura CUDA. Spre deosebire de modul standard, acest mod include opţiuni de lansare în execuţie a scenariilor grafice direct de pe sistemul client. Specificarea parametrilor se realizează în acelaşi mod ca şi în cazul solicitării recepţiei de conţinut video. Diferenţa in acest caz constă în faptul că se lansează în mod dinamic executabilul cu parametrii specificaţi de utlizator, acesta având posibilitatea de a interacţiona direct cu scena. Scopul pentru care acest mod este introdus este acela de a nu îl constrange pe un utilizator al unui sistem cu resurse grafice performante în intenţia sa de a interacţiona într-un mod cât mai dinamic cu scena.

Aplicaţia server

Aplicaţia server este concepută ca un serviciu Web. Aceasta este realizată folosind tehnologia JAX – WS (*Java API for XML Web Services)* [3].

Principalele funcţionalităţi ale aplicaţiei server sunt generarea de conţinut video în format AVI şi transmiterea conţinutului generat către client.

Acest serviciu conţine metode Web care au ca date de intrare specificaţii XML ale parametrilor selectaţi de către utilizator. Aceşti parametrii conţin detalii referitoare la rezoluţia filmului, numărul de cadre pe secundă, precum şi informaţii specifice scenariului ales. Toate aceste informaţii sunt salvate în conţinut XML, care este transmis către server. Odată preluat parametrul de intrare, acesta este trimis unei clase responsabile cu prelucrarea conţinutului XML. În cadrul acestei clase, se extrage parametrul care indică scenariul ales pentru vizualizare, iar apoi se preiau parametrii specifici (rezoluţia filmului, numărul de cadre pe secundă, modul de trasare grafica - prin puncte, linii sau triunghiuri etc.). După acest pas, urmează lansarea în execuţie a scenei grafice având ca parametrii valorile obţinute din interpretarea conţinutului XML.

Generare conţinut video

Imediat după lansarea în execuţie a trasării grafice a scenei 3D, se captează conţinutul imaginii la intervale de timp regulate (indicate prin numărul de cadre pe secundă ales de utilizator). Fiecare imagine captată astfel este scrisă într-un cadru AVI folosind clasa AVIOutputStream [4]. Această clasă aparţine unui API *open source* [5] şi oferă o serie de metode utile care permit descrierea conţinutului video. Astfel, se poate seta numărul de cadre pe secundă, dimensiunea filmului, calitatea compresiei. După încheierea timpului alocat generării filmului, se închide executabilul, iar fişierul video este stocat pe server.

Transmiterea înregistrării video

De îndată ce filmul este complet generat, se iniţează transmisia video către client. Pentru partea de transmisie - recepţie media se foloseşte Java Media Framework [6]. Pentru a începe acest proces trebuie precizate locaţia de pe server a fişierului video ce urmează a fi transmis, precum şi adresa de IP şi portul utilizat. Transmisia propriu-zisă se realizează prin protocolul RTP *(Real time Transport Protocol)* care se află la baza Java Media Framework.

RTP este considerat principala soluţie atunci când vine vorba de transmisie de conţinut media în cadrul unei reţele. Principalul său avantaj este acela că poate fi utilizat combinat cu alte protocoale precum H.263 şi RTSP, precum şi detecţia pachetelor care nu ajung la destinaţie în ordinea prestabilită (lucru care se întâmplă frecvent într-o reţea).

Imediat ce procesul de transmisie este iniţiat, în cadrul aplicaţiei client se lansează un player video care redă conţinutul media recepţionat (Fig.2).

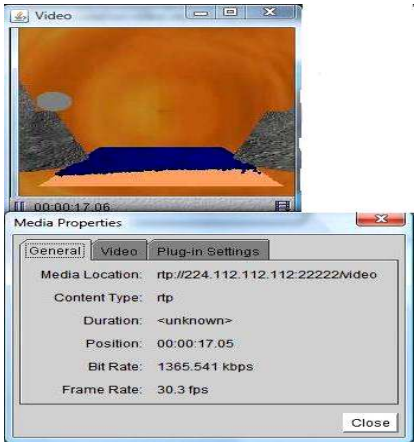

Figura 2. Recepţia conţinutului video de către client

SCENARII DE VIZUALIZARE ŞI SIMULARE

Aplicaţia server conţine, pe lângă serviciul Web prezentat anterior, şi executabile ale unor aplicaţii OpenGL realizate cu ajutorul tehnologiei Nvidia CUDA.

Din punct de vedere al implementării, fiecare scenariu este compus din două părţi: partea centrală *(host)* care conţine codul executat de CPU, respectiv partea dispozitiv (*device)* care conţine codul executat de către placa video [7].

Principiul de bază al programării în CUDA constă în utilizarea funcţiilor speciale denumite nucleu (*kernel)*, care se execută în mod concurent pe fiecare bloc arhitectural al procesorului grafic. În acest mod se asigură paralelismul aplicaţiei, având drept efect îmbunataţiri remarcabile ale performanţei în comparaţie cu varianta secvenţială.

S-au implementat două categorii de scenarii de vizualizare:

Generare de relief geografic, care cuprinde o varietate de forme începând de la nivelul oceanului până la nivelul munţilor (Fig. 3.a, 3.b). O proprietate comună a tuturor formelor de relief este aceea că fiecare categorie prezintă acelaşi tip de variaţii ale înălţimii (câmpiile prezintă variaţii mici, în timp ce munţii prezintă variaţii mari).

Pentru o reproducere cât mai fidelă a acestor forme se utilizează funcţia Perlin [8]. Aceasta însumează valorile mai multor funcţii zgomot. Funcţia Perlin se caracterizează prin următorii parametrii : amplitudine, număr de octave, respectiv lacunaritate.

Tabelul 1. Descrierea parametrilor funcţiei Perlin

Parametru	Descriere
amplitudine	Diferenţa dintre extremele funcţiei; determină altitudinea reliefului
număr de octave	Numărul de funcţii zgomot care sunt însumate; determină netezimea suprafeţelor
lacunaritate (persistenţă)	Specificarea amplitudinii în funcţie de frecvenţă, determină densitatea reliefului

Aceşti parametrii sunt folosiţi de către nucleul CUDA pentru a calcula valoarea funcţiei Perlin pentru fiecare punct definit prin coordonatele x şi y. În cadrul nucleului se realizează calculul valorii pentru un singur punct, însă apelul concurent garantează calculul valorilor pentru întreaga gamă de puncte.

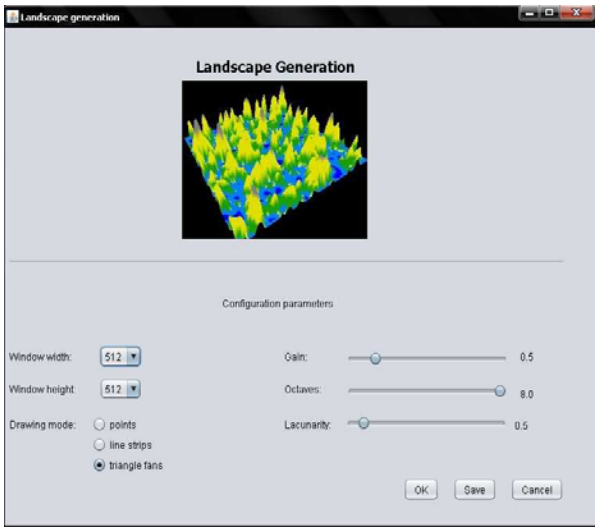

Fig. 3.a) Specificarea parametrilor pentru generarea de relief utilizând interfaţa grafică a aplicaţiei client

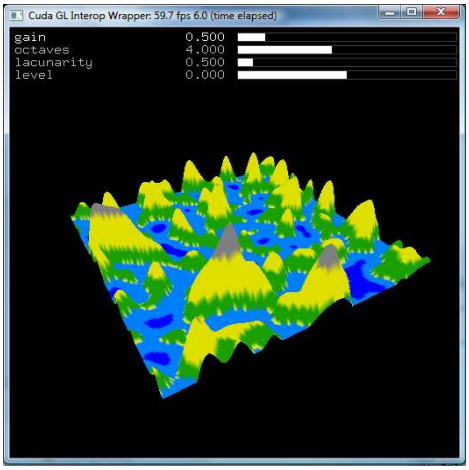

Fig. 3.b) Rularea aplicaţiei cu parametrii specificaţi de utilizator

Simularea procesului de flux-reflux al mării

Pentru realizarea acestui scenariu s-a folosit modelarea fizică prin particule permisă de sistemul Nvidia CUDA SDK [9]. Avantajele folosirii particulelor pentru modelarea fluidelor sunt uşurinţa de paralelizare a aplicaţiei, efectuarea calculelor doar acolo unde este necesar şi conservarea implicită a masei. Un dezavantaj este dat de faptul că este nevoie de un număr mare de particule pentru obţinerea unor rezultate realiste [10].

Fiecare particulă este descrisă prin poziţie şi viteză. Asupra fiecărei particule din sistem acţionează mai multe forţe precum forţa de frecare, forţa de elasticitate sau forţa de gravitaţie. Între particulele vecine au loc coliziuni care le modifică proprietăţile acestora. Astfel, se impune găsirea tuturor particulelor vecine pentru procesarea interacţiunilor, lucru care necesită efectuarea a n^2 comparaţii. O soluţie mai eficientă este utilizarea subdiviziunii spaţiale, care implică împărţirea spaţiului 3D în celule (cuburi) identice având dimensiunea egală cu cea a particulei. După această diviziune, se plasează particulele în cuburile corespunzătoare. Drept urmare, e nevoie să comparăm doar particulele aflate în celulele învecinate. În cazul CUDA, acest proces este simplificat prin aplicarea următorului algoritm :

1. Se calculează celula corespunzătoare fiecărei particule.
2. Se calculează indexul celulei.
3. Se sortează particulele în funcţie de indexul celulei.
4. Se procesează coliziunea cu particulele din cele 26 de celule vecine (3x3x3-1 celule).

Un exemplu de scenariu care utilizează modelul particulelor este cel de modelare al procesului de flux-reflux al mării (Fig. 4.a). Acest fenomen se produce datorită forţei de atracţie a Lunii. Se observă ca forţa de gravitaţie, aplicată în direcţie orizontală, produce acelaşi efect ca şi forţa de atracţie descrisă anterior. Aşadar, prin modificarea la intervale de timp regulate ale forţei gravitaţionale specifice particulelor, se crează acest efect: o valoare scăzută a forţei gravitaţionale induce efectul de

flux al mării (Fig. 4.b), în timp ce o valoare ridicată induce efectul de reflux (Fig. 4.c).

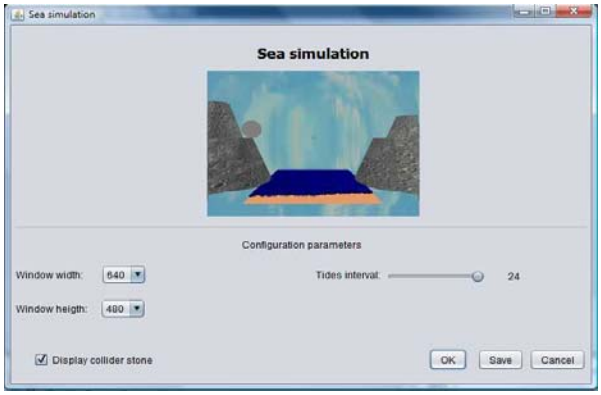

Figura 4. a) Specificarea intervalului dintre flux şi reflux prin interfaţa aplicaţiei client

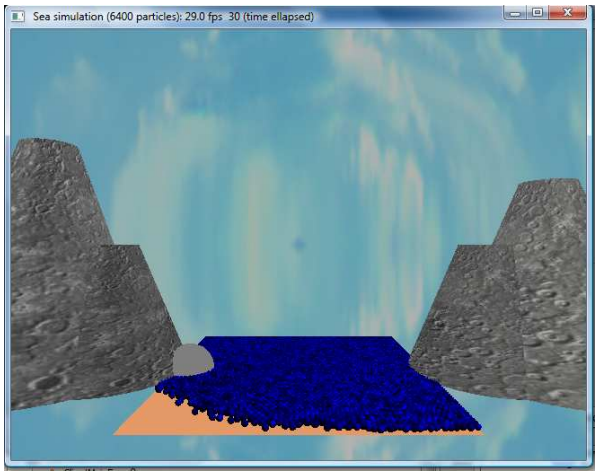

Figura 4. b) Simularea procesului de flux al mării folosind 6400 de particule

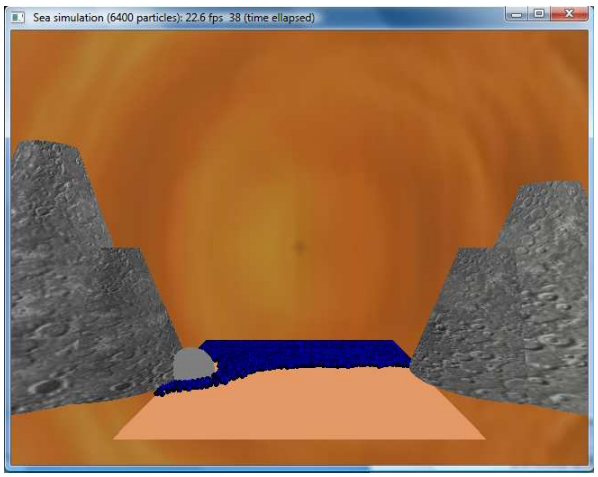

Figura 4. c) Simularea procesului de reflux al mării folosind 6400 de particule

CONCLUZII

S-a prezentat o modalitate de realizare a unei aplicaţii de vizualizare interactivă la distanţă a unor scene de obiecte 3D. Scenele grafice au fost realizate folosind CUDA, ceea ce implică o creştere considerabilă a paralelismului aplicaţiei. Folosirea serviciilor Web şi a protocolului RTP pentru transmiterea conţinutului video fac posibilă vizualizarea scenariilor specificate dinamic de către utilizatori, folosind terminale cu resurse grafice modeste. Există o serie de direcţii în vederea unor dezvoltări ulterioare; de exemplu integrarea aplicaţiei pe un cluster grafic ar conferi scalabilitate sistemului. De asemenea, integrarea unui modul care să conţină configuraţiile tuturor aplicaţiilor 3D s-ar dovedi foarte utilă în cazul adăugării de scenarii noi, sporind astfel flexibilitatea sistemului.

REFERINŢE

1. ScanView – un sistem pentru vizualizarea la distanţă a modelelor 3D, http://graphics.stanford.edu/software/scanview/ (2009)
2. Nvidia CUDA http://www.nvidia.com/object/cuda_home_new.html
3. Salomie I.; Cioara T.; Anghel I. “Distributed Computing and Systems”, Editura Albastra, 2009, 247-293
4. Documentation of class AVIOutputStream http://www.randelshofer.ch/cubetwister/javadoc/ch/randelrandel/media/avi/AVIOutputStream.html
5. Writing AVI Videos in Pure Java http://www.randelshofer.ch/blog/2008/08/writing-avi-videos-in-pure-java/
6. Java Media Framework API Guide, http://ditra.cs.umu.se/
7. CUDA C Programming Guide, http://developer.download.nvidia.com/compute/cuda/3_0/tttttt/docs/
8. Perlin Noise Algorithm, http://freespace.virgin.net/hugo.elias/models/m_perlin.htm
9. Nvidia CUDA SDK , http://developer.nvidia.com/cuda-downloads
10. Muller M.; Charypar D.;Gross M. “Particle based fluid simulation”, 2009, http://mathiasmuller.info/publications/sca03.pdf