# APLICAȚII PENTRU PROCESAREA DE IMAGINI PE PLATFORMA CUDA – STUDIU DE CAZ

**Ramona Din** **Andrei Grumăzescu** **Daniela Saru**

din.ramona@yahoo.com andrei.grum@yahoo.com saru@aii.pub.ro

**Ştefan Mocanu** **Radu Dobrescu**

smocanu@rdslink.ro radud@isis.pub.ro

Universitatea "Politehnica" Bucureşti

**Rezumat:** Procesoarele grafice (GPUs – *Graphical Processing Units*) au cunoscut în ultimii ani o evoluție spectaculoasă, apropiindu-se din ce în ce mai mult de statutul de co-procesor al unității centrale de prelucrare. Pe lângă funcțiile specializate de prelucrare/redare a imaginilor, plăcile video oferă în prezent facilități performante pentru realizarea calculelor matematice complexe şi rularea unor aplicații ce necesită putere ridicată de procesare. Exploatarea platformelor asociate plăcilor video a devenit accesibilă tuturor programatorilor odată cu lansarea arhitecturilor dedicate GPGPU. Această lucrare îşi propune evidențierea performanțelor pe care le oferă una dintre cele mai competitive arhitecturi ale momentului, CUDA, pentru elaborarea unor algoritmi optimizați de prelucrare a imaginilor. Studiul de caz prezintă rezultatele obținute în urma implementării unui algoritm de calcul paralel al histogramei unei imagini pe 256 niveluri de gri şi o analiză comparativă a performanțelor obținute în urma prelucrării pe CPU, respectiv pe GPU.

**Cuvinte cheie:** procesor grafic (GPU), CUDA, procesare paralelă, GPGPU, histogramă.

**Abstract:** Graphic processors (GPUs), originally designed exclusively for game industry and visual effects, have turned into complex multi-threaded architectures which are now closer than ever to gain the co-processor statute. GPUs can be used to process data and to deal with computationally intensive applications really fast. Programmers' access to these platforms has been enhanced along with the official launch of the GPGPU dedicated architectures. This paper focuses on the study of the performances offered by one of the most competitive architecture, CUDA, used to elaborate optimized image processing algorithms. This case-study presents the results gathered running a parallel algorithm designed to compute a 256 gray level image histogram on CUDA platform and it points out the performance differences between the results collected when the algorithm was run on the CPU vs. on the GPU platform.

**Key words:** Graphical processing unit (GPU), CUDA, parallel computing, GPGPU.

## 1. Introducere

În contextul cerințelor pieței pentru o putere de calcul din ce în ce mai mare, dar şi ca urmare a evoluției din punct de vedere tehnologic, prelucrarea de tip secvențial a informației a fost înlocuită cu algoritmi proiectați pentru rularea în paralel a mai multor fire de execuție, în special în cazul volumelor mari de date ce necesită prelucrări complexe. Dacă în urmă cu câțiva ani exploatarea puterii de calcul a unui sistem se rezuma la paralelizarea algoritmilor pentru nucleele procesoarelor unității centrale, în ultimul timp s-a urmărit proiectarea unor arhitecturi hardware care să permită procesare în paralel şi în afara acestora. Astfel, a apărut conceptul *General-Purpose computation on Graphics Processing Units* (*GPGPU*) [4]. Spre deosebire de arhitectura procesorului central, procesorul grafic (GPU) a fost proiectat pentru execuția foarte rapidă a unui set redus de instrucțiuni, având la bază mai multe nuclee de procesare pe care pot rula în paralel diferite taskuri. În timp ce în cazul CPU se acordă o atenție deosebită accesului la memorie sau transferului de date, proiectarea GPU a pus accentul pe optimizarea operațiilor ce implică procesarea efectivă a datelor [1]. Mai mult, trebuie menționat faptul că memoria video GDDR3 (accesată si utilizată de GPU pentru stocarea informațiilor) este mult mai rapidă decât memoria unității centrale, DDR2 sau DDR3.

Odată cu dezvoltarea puterii de calcul din punct de vedere tehnologic, producătorii de pe piața componentelor hardware au înțeles că este necesară crearea unor platforme care să permită exploatarea puterii GPU. Soluția celor de la NVIDIA pentru GPGPU este *CUDA* (*Compute Unified Device Architecture*). Arhitectură dedicată procesării paralele, CUDA oferă programatorilor pasionați atât suportul hardware cât şi uneltele software necesare implementării algortimilor, dar şi exemple şi documentație, ce pot fi descărcate gratuit de pe site-ul producătorului [5].

Propunându-şi a fi o continuare a abordării propuse în [6], această lucrare prezintă un studiu comparativ al performanţelor obţinute în urma implementării unui algoritm de prelucrare a imaginilor în vederea execuţiei pe platforma CPU, respectiv pe platforma unui GPU din clasa GTX260, cu suport CUDA. Din volumul mare de rezultate obţinute a fost selectat un eşantion considerat reprezentativ pentru ilustrarea situaţiilor în care decizia de a paraleliza algoritmul şi a-l rula pe platforma plăcii video conduce la performanţe deosebite, dar şi a celor în care o astfel de variantă nu este adecvată.

## 2. Studiu de caz

Pentru a testa performanţele noii arhitecturi a fost proiectat un sistem software pentru calculul histogramei unei imagini pe 256 niveluri de gri. Acest tip de aplicaţie este o modalitate bună de test, având în vedere faptul că o imagine digitală este reprezentată ca o matrice de pixeli. Mai mult, numărul elementelor acestei matrice variază în funcţie de dimensiunea imaginii, astfel încât pentru o imagine cu o rezoluţie mare este necesar un volum de calcul substanţial. Histograma unei imagini, o unealtă foarte utilă în domeniul fotografiei digitale, are aspectul unui grafic ce indică poziţia în care se găsesc toate nivelurile de luminozitate. Tonurile sunt dispuse pe orizontală, de la cele mai întunecate (în stânga) până la cele mai luminoase (în dreapta), iar cantitatea luminozităţii este reprezentată pe axa verticală. În total, histograma cuprinde 256 niveluri de luminozitate, de la 0 la 255 (de la negru pur la alb pur), care pot fi grupate în mai multe zone, în funcţie de intensitate. Majoritatea aparatelor foto digitale, dar şi mediile de editare precum Adobe Photoshop sau GIMP, pot afişa histograma unei imagini. Aplicaţia realizată cuprinde o bibliotecă statică pentru citirea secvenţială (de pe disc) a imaginii în format BMP necomprimat, construirea unei matrice de pixeli şi procesarea acesteia în vederea obţinerii histogramei. Pentru o prezentare comparativă, algoritmul de construcţie a histogramei a fost implementat atât în mod clasic, secvenţial (şi rulat pe CPU), cât şi respectând considerentele de performanţă specifice arhitecturii CUDA (şi rulat pe placa video).

În ceea ce priveşte abordarea tradiţională, în urma citirii pixelilor din imagine, au fost calculate valorile de intensitate luminoasă corespunzătoare fiecărui pixel, iar rezultatele au fost stocate în memoria centrală a sistemului. Pentru calculul histogramei, algoritmul a fost compilat şi executat pe unitatea centrală de prelucrare (CPU). Pentru cea de-a doua abordare, respectiv paralelizarea algoritmului şi rularea acestuia pe platforma procesorului grafic, s-a urmărit distribuirea datelor de intrare şi a calculelor pe mai multe fire de execuţie. Astfel, fiecare sub-diviziune, procesată de un *warp*, generează o sub-histogramă. În final, aceste componente sunt reunite pentru determinarea rezultatului final. La adaptarea metodei pe arhitectura procesorului grafic, trebuie avute în vedere o serie de constrângeri. Deşi s-a urmărit o paralelizare a algoritmului cu o granularitate cât mai mare, trebuie precizat faptul că anumite operaţii (de exemplu, accesul la matricea de date iniţială) rămân secvenţiale. Astfel, execuţia unui program tipic CUDA începe cu execuţia codului specific CPU, iar atunci când acesta invocă un *kernel*, procesul de execuţie este transferat procesorului grafic [3]. Conform recomandărilor prezentate în [6], pentru obţinerea unor performanţe bune, s-a urmărit reducerea volumului de date stocate în memoria comună. Prin procesarea la nivel de *warp*, presiunea asupra memoriei comune a scăzut, utilizând din 16KB disponibili, numai 6KB (6 *warps* $\times$ 256 locaţii ale histogramei $\times$ 4 octeţi/locaţie). Mai mult, grupând cele 192 fire de execuţie (6 *warps* $\times$32 fire de execuţie/*warp*) într-un bloc, au fost rulate două blocuri pe un multiprocesor, atingând astfel un grad ridicat de încărcare a GPU.

Performanţele aplicaţiei sunt puternic influenţate de configuraţia sistemului pe care rulează. Testul pentru acest studiu de caz a fost realizat pe un sistem dotat cu procesor AMD Athlon64 X2 la 2.2 GHz, 2 module de memorie RAM a câte 1 GB DDR400 care lucrează în dual-channel pe 128 biţi (cu performanţe de scriere şi de citire de 6300 MB/s la o latenţă de 54.2 ns), placă video Nvidia GTX260 ce oferă suport pentru arhitectura CUDA, cu 24 multiprocesoare (192 nuclee de procesare) şi o frecvenţă GPU de 620 MHz. Prin intermediul unei interfeţe pe 448 biţi, procesorul grafic comunică cu memoria video GDDR3 (896 MB la o frecvenţă de 2,2 GHz, cu o lăţime de bandă de 112 GB/s). Se poate observa diferenţa de performanţă între cele două

cupluri de componente, CPU – memorie sistem şi GPGPU - memorie video, de aproape 17,8 ori. Conexiunea între acestea este asigurată printr-o magistrală PCI Express 16x, iar lățimea de bandă dintre ele (conform unui test real al plăcii video) este 1300 MB/s, în comparație cu 4000 MB/s, valoare menționată în specificațiile producătorului.

În tabelul 1.1 este prezentat un eşantion de date reprezentativ, obținut în urma testelor efectuate. Determinarea histogramei caracteristice unei imagini pe 256 tonuri de gri poate fi o operație costisitoare din punct de vedere al timpului de execuție atunci când algoritmul este conceput pentru a rula pe CPU, motiv pentru care am urmărit variația timpului de procesare ($T_p$) şi lățimea de bandă ($B$) pentru dimensiuni diferite ale aceleaşi imagini, pentru fiecare unitate de prelucrare (procesor central, respectiv cele trei multiprocesoare de pe placa video). Se poate observa relativ uşor faptul că, pentru imagini mici, sub $100 \times 100$ pixeli, unitatea centrală de procesare este mai rapidă decât procesorul grafic. În jurul acestei valori performanțele celor două procesoare sunt aproximativ identice, iar pentru dimensiuni mai mari se justifică paralelizarea algoritmului.

**Tabel 3.1. Rezultatele obținute pentru dimensiuni diferite ale imaginii**

| | CPU | | GPU1 | | GPU2 | | GPU3 | |
|---|---|---|---|---|---|---|---|---|
| **Dimensiune imagine (pixeli)** | $T_p$ (ms) | $B$ ( MB/s) | $T_p$ (ms) | $B$ ( MB/s) | $T_p$ (ms) | $B$ ( MB/s) | $T_p$ (ms) | $B$ ( MB/s) |
| 20 x 20 | 0,006 | 56,895 | 0,074 | 5,090 | 0,072 | 5,249 | 0,064 | 5,892 |
| 100 x 100 | 0,065 | 144,649 | 0,064 | 146,925 | 0,068 | 139,691 | 0,063 | 149,971 |
| 600 x 600 | 2,405 | 142,717 | 0,152 | 2254,805 | 0,136 | 2515,901 | 0,153 | 2232,025 |
| 2000 x 2000 | 25,4 | 150,184 | 1,559 | 2445,454 | 1,462 | 2608,382 | 1,256 | 3036,442 |
| 3000 x 3000 | 56,887 | 150,609 | 4,072 | 2107,800 | 3,804 | 2255,750 | 3,099 | 2769,310 |
| 7000 x 7000 | 310,575 | 150,462 | 30,223 | 1546,158 | 28,989 | 1611,979 | 20,231 | 2309,738 |
| 9000 x 9000 | 513,216 | 150,516 | 55,639 | 1388,354 | 53,469 | 1444,717 | 34,684 | 2227,158 |
| 10000 x 10000 | 634,374 | 150,333 | 72,305 | 1318,957 | 63,318 | 1375,795 | 44,113 | 2161,855 |

Reprezentarea grafică a rezultatelor obținute ca urmare a testelor efectuate evidențiază şi mai clar performanțele specifice celor două abordări. Întrucât plaja de valori a rezultatelor este destul de cuprinzătoare, sunt prezentate separat cazul imaginilor de dimensiuni mici, respectiv al celor de dimensiuni mari.

Analizând informațiile reprezentate în figura 3.1 se poate observa că lățimea de bandă pentru imagini de dimensiuni mari este influențată în mod direct de dimensiunea imaginii. De asemenea, se constată că reducerea acesteia este mai mare în cazul variantei fără operații atomice la nivel de memorie.

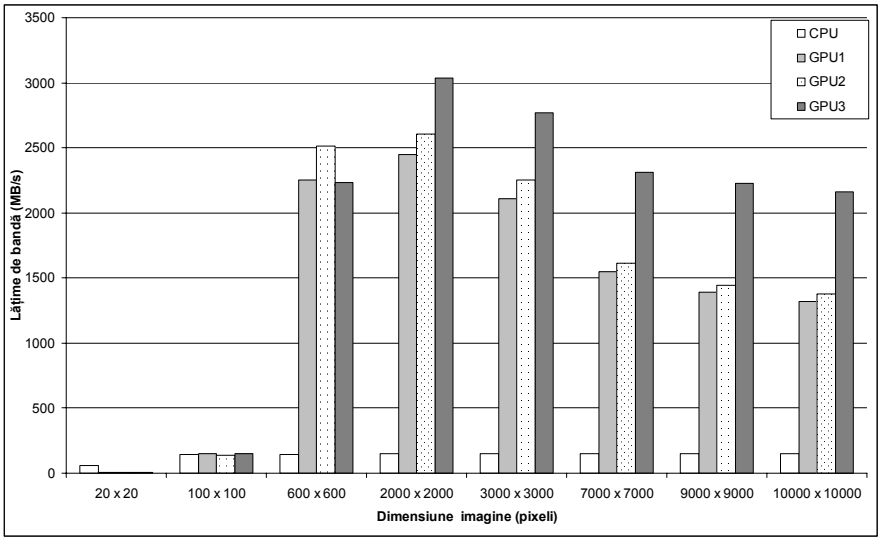


**Figura 3.1. Variația lățimii de bandă în funcție de dimensiunea imaginii**

Ilustrată grafic în figura 3.2, variația lățimii de bandă apărută la nivelul procesorului central, respectiv la nivelul procesorului grafic subliniază faptul că, pentru imagini relativ mici, unitatea centrală de procesare este totuşi mai rapidă decât procesorul grafic. Cu toate acestea, începând cu imagini de dimensiuni mai mari de $100 \times 100$ pixeli, performanța obținută în urma procesării pe platforma grafică este semnificativ mai bună, atingând o valoare maximă pentru imagini de $2000 \times 2000$ pixeli.

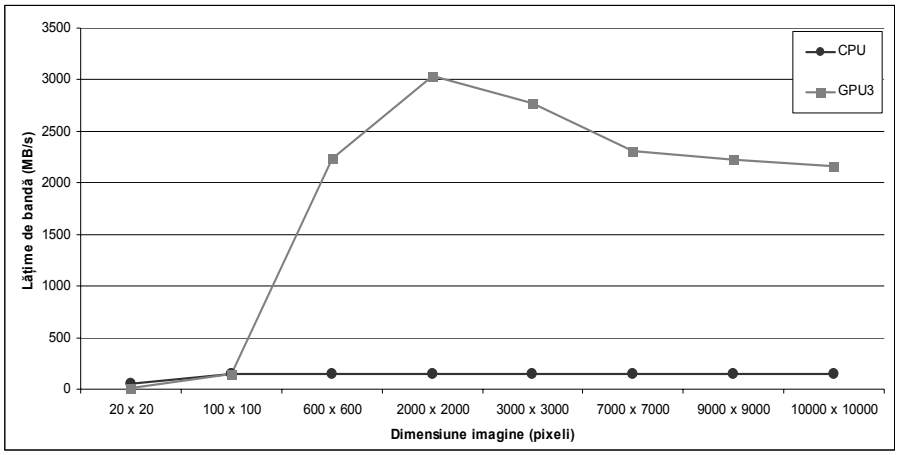


**Figura 3.2. Comparație între variațiile lățimii de bandă pentru CPU şi GPU**

După cum s-a precizat anterior, dimensiunea imaginii reprezintă un factor determinant în variația duratei de execuție a algoritmului de prelucrare. Pe baza rezultatelor obținute în urma testării, în figurile 3.3 şi 3.4 au fost reprezentate variațiile duratei de execuție pentru prelucrarea imaginilor de dimensiuni mici şi respectiv mari.

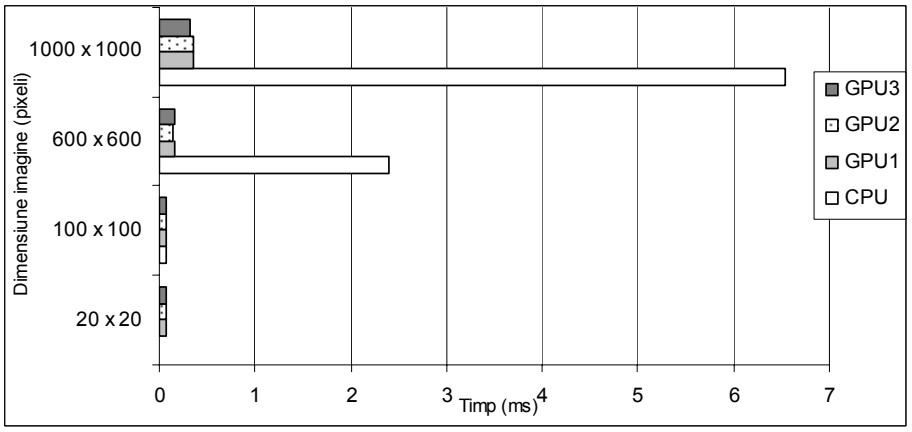


**Figura 3.3. Variația duratei de execuție pentru imagini cu dimensiuni mici**

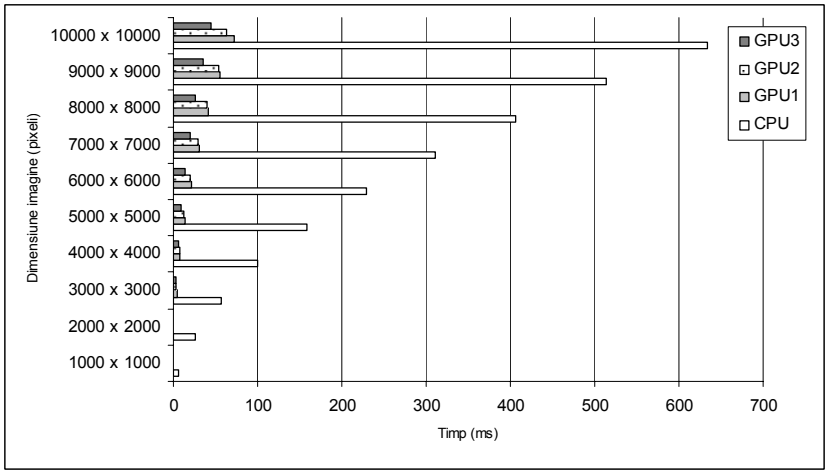


**Figura 3.4. Variația duratei de execuție pentru imagini cu dimensiuni mari**

Utilizând acelaşi eşantion de date prezentat anterior în tabelul 1.1, figura 3.5 ilustrează grafic o analiză comparativă a rezultatelor obținute, cu referire la variația duratelor de execuție în cazul rulării algoritmului pe platforma CPU, respectiv pe platforma GPU.

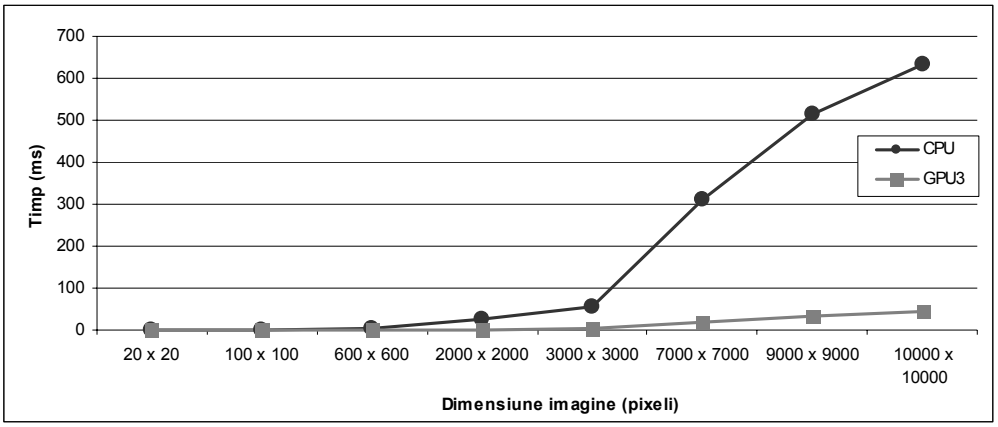


**Figura 3.5. Variația duratei de execuție în raport cu dimensiunea imaginii**

Pentru valorile obținute în cadrul studiului de caz, s-a considerat ca relevantă calcularea unui factor de creştere a performanțelor definit ca raportul dintre durata de execuție a algoritmului pe CPU şi durata de execuție pe GPU, în cazul utilizării memoriei comune. Reprezentarea grafică a acestui factor în funcție de dimensiunea imaginii se regăseşte în figura 3.6.

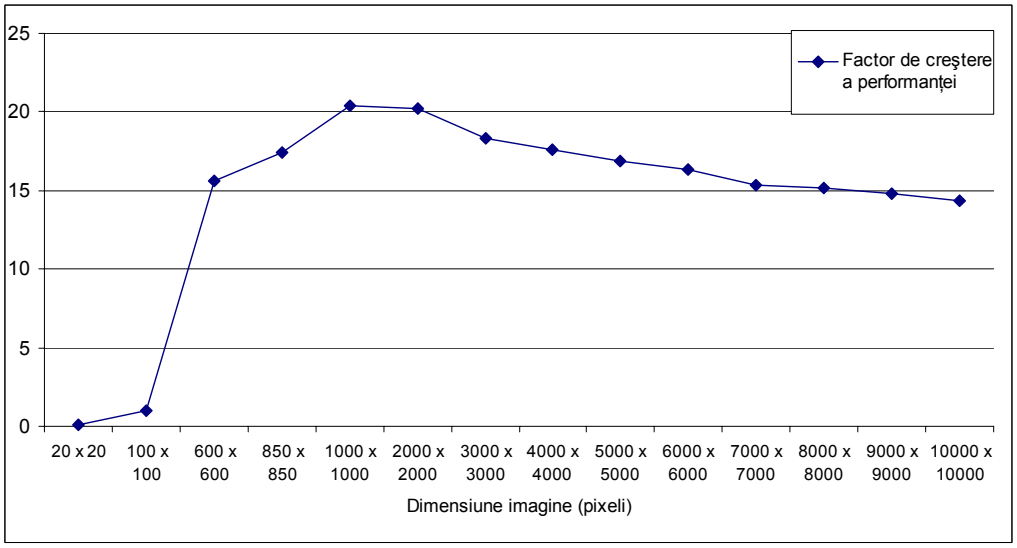


**Figura 3.6. Reprezentarea factorului de creştere a performanței**

Rezultatele studiului de caz conduc la concluzia că performanțele obținute prin utilizarea arhitecturii CUDA sunt semnificativ mai bune în comparație cu cele obținute prin rularea aplicației pe platforma unității centrale de procesare. În cadrul studiului s-a constatat o eficiență mai mare a arhitecturii pentru imagini cu dimensiuni cuprinse între 2 şi 9 megapixeli, interval pentru care a fost obținută şi cea mai mare lățime de bandă, aproximativ 3000 MB/s. Totuşi, se poate observa că pentru imagini de dimensiuni mici efortul de paralelizare nu este justificat, performanțele obținute în urma execuției variantei tradiționale (pe platforma CPU) fiind aproximativ aceleaşi.

# 3. Concluzii

Scopul principal al acestei lucrări a fost cercetarea potențialului de performanță promis de utilizarea arhitecturii CUDA pentru calcul paralel. Rezultatele studiului de caz au confirmat faptul că utilizarea acestei tehnologii pentru rezolvarea problemei de calculare a histogramei

unei imagini pe 256 tonuri de gri poate conduce la obținerea unor performanțe deosebite. Chiar şi în urma optimizării codului sursă, rularea algoritmului pe unitatea centrală de prelucrare s-a dovedit a fi o operație costisitoare din punctul de vedere al duratei de execuție. Utilizarea arhitecturii CUDA a condus la obținerea unui factor de optimizare cu ordin de multiplicare mai mare de 20, pentru imagini cuprinse între 1 şi 4 mega-pixeli. Cu toate acestea, trebuie menționat faptul că, în cazul procesării unui volum mic de date, utilizarea algoritmilor de calcul paralel nu este justificată. Dimensiunea de $100 \times 100$ pixeli, pragul peste care performanțele obținute pentru procesarea pe GPU au fost semnificativ mai bune decât cele pentru procesarea pe CPU, este mică față de dimensiunile imaginilor folosite în mod curent.

Aşadar, în acest caz, tehnologia CUDA şi-a dovedit eficiența, chiar dacă implementarea algoritmului nu a fost concentrată pe optimizarea codului, respectiv pe managementul direct al resurselor puse la dispoziție de NVIDIA. Acest rezultat poate fi susținut şi de faptul că, în ultimul timp, multe companii cu nume sonor şi-au îndreptat atenția către arhitectura paralelă, cu nuclee multiple, a unităților de procesare grafică Nvidia[5]. Domenii precum procesarea audio-video, modelarea şi simularea în timp real a proceselor cu grad ridicat de complexitate, proiectarea electronică automată sau prelucrarea imaginilor din diverse domenii, toate pot avea în comun suportul oferit de tehnologia CUDA pentru dezvoltarea şi utilizarea unor algortimi de o complexitate mult mai mare decât ar fi fost posibil altfel şi o durată de execuție de zeci, chiar sute de ori mai mic. Cu toate acestea, CUDA nu este singura opțiune pentru implementarea algoritmilor de calcul şi exploatarea platformei procesorului grafic. În multe privințe, platforma celor de la RapidMind de exemplu, *Multicore Development Platform* (achiziționată de către Intel în 2009), este superioară platformei CUDA [2]. Totuşi, ca şi în trecut, concurența de pe piața microprocesoarelor nu poate decât să susțină evoluția tehnologică din domeniul IT.

# BIBLIOGRAFIE


1. **CASTANO-DIEZ, D.; MOSER D.; SCHOENEGGER A.; PRUGGNALLER S.; FRANGAKIS A. S.** Performance Evaluation of Image Processing Algorithms on the GPU. Journal of Structural Biology, Vol. 164, 2008, pp. 153 – 160.
2. **HALFHILL, T.** Parallel Processing with CUDA – Nvidia's High – Performance Computing Platform Uses Massive Multithreading, 2008, documentație Internet, http://www.nvidia.com/docs/IO/55972/220401_Reprint.pdf
3. **KIRK, D.; HWU W.** Programming Massively Parallel Processors, Curs Universitatea din Illinois, 2007.
4. *** General-Purpose Computation on Graphics Hardware, documentație Internet, http://gpgpu.org/
5. *** Nvidia Documentation, 2009, documentație Internet, http://www.nvidia.com
6. **MOCANU, Ş.; DOBRESCU R.; SARU D.; DIN R.; GRUMĂZESCU A.** Arhitecturi Complexe folosite în prelucrarea paralelă a imaginilor. Revista Română de Automatică şi Informatică, vol. 20, 2010, pp. 97 – 105.