Високопродуктивна реставрація образів на платформі GPGPU-CPU

Корольов В'ячеслав Юрійович

Центр таймерних обчислювальних систем Інституту кібернетики НАНУ
dshv937@meta.ua

Abstract

Visualization and image processing tools integration is a tendency of recent years. Image restoration is a topical research trend for methods and tools development of degradations and noise compensations that are added to the recorded data. The effective evaluation of image restoration algorithms needs state-of-the-art computer platforms. It is shown that using NVIDIA graphic accelerator for general purpose problem (GPGPU) computing with central processing unit (CPU) speeds up image restoration evaluation in 4 times in comparison with only CPU realization. The common recommendations that are based on finding and results of different image processing realizations on GPGPU-CPU platform for computations accelerations are given in the end of the article.

Вступ

Створення програмних засобів для графічного відображення наукових, технічних, медичних даних і процесів є однією з областей використання комп'ютерної графіки. Дослідникам, аналітикам і багатьом іншим фахівцям доводиться мати справу з великими об'ємами інформації або вивчати перебіг процесів великої складності. При чисельному моделюванні, наприклад, можуть видаватись файли, що містять тисячі або навіть мільйони значень. Технічні реєструючі пристрої також часто накопичують великі об'єми даних швидше ніж оператор може їх фізично обробити. Аналіз цих великих масивів цифр з метою передбачення подальшої поведінки відповідних процесів і виявлення закономірностей в багатьох випадках не дозволяє ефективно використати людський потенціал. Проте, якщо форму цих даних перетворити за допомогою методів сучасної візуалізації, то в багатьох випадках стають очевидними тенденції процесів і кореляція величин. При цьому інтеграція в єдиній системі машинної графіки окремих комп'ютерів, мереж, систем машинного бачення і відображення інформації створює нові організаційні можливості для взаємодії людини і комп'ютерної техніки.

Процедури обробки зображень у широкому розумінні — це внесення змін або інтерпретація вже існуючих графічних даних, таких як фотографія, відеоплівки і т. п. Хоча методи, якими користуються у комп'ютерній графіці і при обробці зображень, часто є однаковими, проте ці дві області пов'язані з фундаментально різними підходами. У комп'ютерній графіці за допомогою комп'ютера формується образ. Методи обробки зображень використовуються для покращення якості відеоданих, їх аналізу або розпізнавання тих чи інших візуальних характеристик. В той же час методи обробки зображень часто застосовуються у комп'ютерній графіці, а методи сучасної візуалізації можуть бути використані для представлення реальних зображень.

Для підвищення продуктивності обчислювальних процесів при розв'язанні задач комп'ютерної графіки і візуалізації можна використати графічні прискорювачі. Наприклад, спеціальна алгоритмізація дозволяє отримати суттєве зменшення часу розв'язання таких задач як візуальне моделювання динаміки фізичних процесів різної природи, цифрова обробка зображень, розв'язання систем лінійних рівнянь, сортування, матрична алгебра, розв'язання систем диференційних рівнянь в часткових похідних, обробка інформації в базах даних і т. п.

Враховуючи вказану вище тенденцію інтеграції засобів комп'ютерної графіки і цифрової обробки зображень, суттєвий науковий інтерес представляє порівняння результатів чисельних експериментів, отриманих з одночасним використанням графічного (GPU) і центрального процесора (CPU), а також виключно CPU для задачі реставрації образів.

Реставрація образів є актуальним науковим напрямком по розробці методів [1-4] і засобів компенсації спотворень, що вносяться в зображення в процесі його формування різними системами і джерелами. Цифрова модель спотворення зображення при його формуванні систе-

мою реєстрації з неточковою апаратною функцією має вигляд [1-4]:

$$\mathbf{G} = \mathbf{HF} + \boldsymbol{\nu}, \qquad (1)$$

де **G** — матриця спотвореного образа, **H** — матриця апаратної функції, **ν** — невідома реалізація шуму.

Рівняння (1) є некоректною зворотною задачею, тому для її розв'язку може бути використано метод узагальненої вінерівської фільтрації з тихонівською регуляризацією. В області просторових частот розв'язок задачі (1) має вигляд [1-4]:

$$\dot{F}_0(\omega_x, \omega_y) = \frac{\dot{G}(\omega_x, \omega_y) H^*(\omega_x, \omega_y)}{\left|\dot{H}(\omega_x, \omega_y)\right|^2 + \alpha \cdot (\omega_x^2 + \omega_y^2)}, \qquad (2)$$

де $\dot{G}(\omega_x, \omega_y)$ — результат перетворення Фур'є від спотвореного образу, $\dot{H}(\omega_x, \omega_y)$ — частотний спектр апаратної функції, $(\omega_x^2 + \omega_y^2)$ — тихонівський регуляризуючий оператор, α — параметр регуляризації, $\dot{F}_0(\omega_x, \omega_y)$ — оцінка початкового образу.

Сучасні стандарти відображення відеоданих передбачають виведення кольорових чотирьохкомпонентних зображень формату 1000x1000 пікселів і більше. Очевидно також, що цифрова обробка зображень повинна виконуватись у режимі, близькому до реального масштабу часу. Тому реалізація процесу розв'язку задачі реставрації образів передбачає наявність високопродуктивної обчислювальної платформи як для виконання операцій над дійсними числами, так і для відображення відеоданих.

В табл. 1 наведено характеристики найбільш продуктивних обчислювальних компонент сучасного комп'ютера. За оцінками спеціалістів при переході на двоядерну архітектуру і внесенні відповідних змін у програмне забезпечення можливий приріст продуктивності обчислень досягає 65-70%. В інших випадках продуктивність зростає за рахунок збільшення частоти шини процесору. Як видно з табл. 1, підвищення швидкості операцій над дійсними числами можна отримати при використанні графічного прискорювача для розрахунків широкого класу.

Таблиця 1. Порівняння обчислювальних характеристик Pentium 4 і GeForce 6600.

Обчислювальний пристрій	Продуктивність, Гфлопс	Смуга пропускання пам'яті, Гб/с	Кількість транзисторів, млн.
GeForce 6600	60	11,2	220
Pentium 4 3 ГГц	12	6,4	55

Напрям GPGPU (General Purpose Graphic Processor Unit) Computing — універсальні обчислення з використанням графічного прискорювача [5,6] інтенсивно розвивається в останні 4-5 років після появи високо продуктивних відеокарт з можливістю виконання операцій над дійсними числами.

Суттєва перевага серійних графічних прискорювачів над масовими центральними процесорами середньої вартості за показниками, які наведено у табл. 1, пояснюється тим, що модернізація відеокарти полягає в її заміні, а не в оновленні її частин (процесора, мікросхем пам'яті, материнської плати та ін.). Тому у виробників графічних прискорювачів менше проблем з сумісністю компонентів і, відповідно, більше свободи у підвищенні потужності пристроїв. Внаслідок цього виробники відеоприскорювачів випускають нові продукти раз у півроку, а виробники центральних процесорів раз у два-три роки. Зазначимо також, що відносне збільшення кількості операцій з плаваючою точкою в секунду від старого покоління до нового для центральних процесорів складає близько 6 гигафлопс у рік, а для графічних прискорювачів в середньому 25 гігафлопс за рік.

Сумісне використання CPU і GPU вимагає залучення спеціальних бібліотек. Прикладом такої бібліотеки є OpenGL (Open Graphic Library), яка представляє собою процедурно-орієнтовану бібліотеку для програмування графіки в реальному масштабі часу з використанням GPU і CPU. Узагальнена структурна схема [8,9], що пояснює функціонування прикладення з використанням OpenGL API (Application Program Interface), представлена на рис. 1.

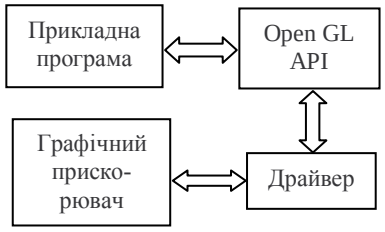

Рис. 1

Бібліотека OpenGL API забезпечує ефективне використання всіх ресурсів сучасних GPU і є посередником між прикладенням і графічним прискорювачем, тобто надає програмісту інтерфейс, абстрагований від конкретного графічного прискорювача. Специфічні особливості реального прискорювача звичайно інкапсулюються у відповідному драйвері.

Архітектурно OpenGL представляє собою відкритий процедурний інтерфейс до GPU, що дозволяє розробникам апаратно-програмних засобів графічного прискорювача оперативно додавати нові можливості шляхом реалізації нових функцій у вигляді розширень (extensions) вихідної бібліотеки OpenGL від виробників. Такий підхід дозволяє прикладним програмістам ефективно і оперативно використовувати всі можливості сучасних графічних прискорювачів. Для того щоб підвищити швидкість відображення сцен, компанія NVIDIA використовує центральний процесор для обробки частини вершин полігональної моделі зображення (vertex processing) за допомогою технології багатопоточних обчислень.

В якості середовища розробки в цій роботі було використано Visual Studio 2005, Visual C++.NET, C#.NET. NET Framework — це кероване середовище для розробки і виконання прикладень, що забезпечує контроль типів. Зазначене середовище управляє виконанням програми: виділяє пам'ять під дані і команди, відповідає за виділення обчислювальних ресурсів і їх звільнення. NET Framework складається з двох основних компонентів: загальномовного виконуючого середовища (Common Language Runtime — CLR) і бібліотеки класів .NET.Framework.

Середовище CLR реалізоване як класичний COM-сервер, код якого зберігається у стандартній Windows DLL (Dynamic Link Library) в режимі користувача. Фактично всі компоненти .NET Framework реалізовані як стандартні Windows DLL в режимі користувача (жоден з компонентів .NET Framework не працює в режимі ядра), що займають рівень над некерованими функціями Windows API.

Спрощено реставрацію образу за методом умовної деконволюції можна представити у вигляді наступної послідовності кроків:

1. Зчитування компонент кольорового зображення у масиви і їх нормування.
2. Обчислення швидкого перетворення Фур'є (ШПФ) від компонент і апаратної функції заданого формату.
3. Виконання узагальненої вінерівської фільтрації.
4. Обернене ШПФ від оброблених компонент.
5. Денормування значень масивів, що відповідають R-,G-, B- компонентам, і виведення результату на екран.

В табл. 2 представлено час виконання задачі реставрації образів для описаних раніше мов програмування і бібліотек комп'ютерної графіки. Розв'язок задачі реставрації виконувався за ідентичними алгоритмами без оптимізації для конкретного процесора. Єдиною відмінністю між програмами було використання різних математичних бібліотек і типів даних.

Таблиця 2. Час виконання задачі реставрації образів для різних форматів зображень

Засіб розробки	Формат зображення, NxN			
	128	256	512	1024
OpenGL	1 с	4 с	14 сек	1 хв 18 с
C++.NET	5 с	15 с	1 хв 5 с	4 хв 49 с
C#.NET	7 с	25 с	1 хв 45 с	8 хв 2 с

Таблиця 3. Час виконання ШПФ для різних форматів зображень

Засіб розробки	Формат зображення, NxN			
	128	256	512	1024
OpenGL	<0.5 с	0.5 с	1 с	3 с
C++.NET	<0.5 с	1 с	3 с	5 с
C#.NET	2 с	3 с	6 с	25 с

Аналіз отриманих даних доводить, що поточний (stream) процесор NVIDIA є у чотири рази ефективнішим для розв'язання задачі реставрації образів ніж послідовний (serial) процесор Intel. Це пояснюється більшою швидкістю виконання операцій з дійсними числами і шириною смуги пропускання пам'яті графічного прискорювача порівняно з центральним процесором і його оперативною пам'яттю. З іншого боку при виконанні ШПФ абсолютні показники не дуже сильно відрізняються на користь використання GPU. Зазначимо, що алгоритм ШПФ було розроблено для зменшення кількості складних операцій множень з дійсними числами. Втім ШПФ характеризується суттєвою кількістю операцій управління послідовністю виконання програми і звернення до віддалених одна від іншої комірок пам'яті. Крім того, при проектуванні універсальних процесорів сімейства Pentium обов'язковими вимогами є ефективність передбачення розгалужень програми і низька латентність пам'яті, яка виконується багаторівневою високочастотною кеш-пам'яттю. Сукупність вказаних факторів обумовила високу продукти-

вність CPU при виконанні ШПФ. В той же час суттєво низькі показники для C#.NET в усіх тестах пояснюються тим, що ця мова була розроблена в першу чергу для контрольованого виконання програми і як RAD-система (Rapid Application Development — швидка розробка програмного забезпечення).

Висновки

1. Підвищення якості реставрації образів у сукупності зі зменшенням часу розв'язку задачі є актуальною науковою задачею.
2. За оцінками спеціалістів підвищення продуктивності послідовного алгоритму при переносі його на сучасну двоядерну масову платформу дозволяє підвищити продуктивність на 65-70%.
3. Сучасні графічні прискорювачі середньої вартості мають в 2 рази більшу смугу пропускання пам'яті і у 5 разів більшу швидкість виконання операцій з дійсними числами у порівнянні з масовими центральними процесорами.
4. Загальним напрямом розвитку комерційного сектору інформаційних технологій є прискорення розробки нових програмних продуктів при одночасному зниженні вимог до кваліфікації інженерних кадрів.
5. Створення програмного забезпечення, що дозволяє використовувати всі переваги паралельної архітектури, покладається в основному на її розробника, що вимагає від нього високої кваліфікації у декількох областях обчислювальної техніки.
6. Компілятори та інші засоби, що дозволяють автоматично розпаралелити виконання програми, знаходяться ще в стадії розробки, а готові автоматизовані засоби мають вузько спрямоване застосування.
7. Використання бібліотек OpenGL дозволяє швидко модифікувати готові програми для розв'язання задач з суттєвим математичним навантаженням за допомогою масових графічних прискорювачів і серійних центральних процесорів з мінімальними затратами на настроювання.
8. Наведенні чисельні експерименти підтверджують ефективність вказаного підходу для підвищення продуктивності обчислювальних засобів при розв'язанні прикладних задач цифрової обробки зображень.
9. Таким чином, використання додаткових математичних бібліотек OpenGL для програмування GPU і CPU дозволяє досягти максимальної швидкості при виконанні великої кількості операцій над дійсними числами для алгоритмів без суттєвих розгалужень і звернень до віддалених ділянок пам'яті.

Література

[1] Корольов В. Ю. Реставрація образів у частотній області методом локально-адаптивної умовної деконволюції // Праці 5-ї Міжнародн. конф. "Оброблення сигналів і зображень (УкрОБРАЗ'2000). — Київ: НАН України – Інститут кібернетики. — 2000. — С. 237 – 240.

[2] Бардаченко В. Ф., Корольов В. Ю. Розробка математичного і програмного забезпечення для автоматизованої системи відновлення зображень // Вісті інженерних наук України. — 2003. — № 3(20). — С. 12 – 15.

[3] Королёв В. Ю. Стратегия применения совокупности методов деконволюции гомогенно деградированных образов // Радиоэлектроника. — 2002. — № 7. — С. 50 – 55. (Изв. высш. учеб. заведений).

[4] Королёв В. Ю. Восстановление изображений методом вейвлет-управляемой анизотропной фильтрации // Радиоэлектроника. — 2002. — № 5.— С. 20 – 28. (Изв. высш. учеб. заведений).

[5] GPU Gems. Programming Techniques, Tips, and Tricks for Real-Time Graphics. Edited by Randima Fernando Addison-Wesley, 2004.

[6] GPU Gems 2. Programming Techniques for High-Performance Graphics and General-Purpose Computation. Edited by Matt Pharr. Addison-Wesley, 2005.

[7] Randima Fernando, Mark J. Kilgard. The Cg Tutorial. The Definite Guide to Programmable Real-Time Graphics. Addison-Wesley, 2005.

[8] Боресков А. В. Расширения OpenGL. — СПб.: БХВ-Петербург, 2005. — 688 с.: ил.

[9] OpenGL. Официальный справочник: Пер. с англ./Под ред. Дейва Шрайнера. — СПб: ООО "ДиаСофтЮП", 2002. — 512 с.