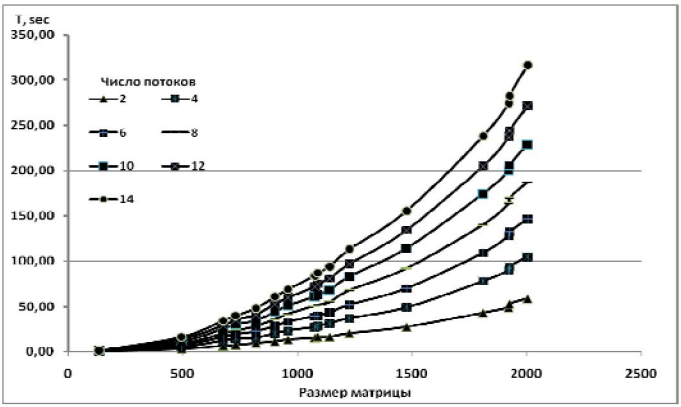

Рис. 3.

Однако, применение хэш-функций для представления хранения разреженных матриц дает положительный эффект в плане ускорения вычислений за счет эффективности доступа к элементам матриц.

1. *К. Ю. Богачев* Основы параллельного программирования - М.:БИНОМ. Лаборатория знаний, 2003. - 342с.
2. *С. Писсанецки* Технология разреженных матриц - М.: Мир, 1988. – 410 с.

Поступила 23.01.2009р.

УДК 004.81

І.В. Хіміченко, аспірант,
Ю. Р. Валькман, д.т.н., зав. отд. МННЦІТтаС, Київ

МЕТОДИ ПІДВИЩЕННЯ ЧАСОВОЇ ЕФЕКТИВНОСТІ ФРАКТАЛЬНОГО АЛГОРИТМУ

Вступ

Об'єктом дослідження алгоритми стиснення статичних растрових зображень.

Предмет дослідження - оптимальність алгоритмів стиснення статичних растрових зображень з використанням фрактальної графіки.

Мета дослідження - побудова методу фрактального стиснення для растрових статичних зображень з використанням класичного алгоритму фрактального стиснення статичних зображень.

Очікувані результати - ресурсно-ефективний комбінований алгоритм

фрактального стиснення статичних зображень, розроблений на основі запропонованих методів.

В даний час існує багато різноманітних алгоритмів стиснення зображень. Гідне місце серед них займають фрактальні алгоритми стиснення. Існує велика кількість варіацій фрактальних алгоритмів, але всі вони базуються на алгоритмі фрактального стиснення, запропонованим ще Джеквіном. Розглянемо основні етапи даного алгоритму.

Сегментація зображення. Зображення розбивається на блоки фіксованого розміру, що називають ранговими блоками. Кожний ранговий блок шляхом рядкового сканування перетвориться у вектор $R_m = (r_1, r_2, ..., r_m)$ розмірністю n, де n - кількість пікселів, що складають ранговий блок.

Створення пулу доменів. З початкового зображення з кроком l пікселів вирізаються доменні блоки, розмір кожного з яких в два рази перевищує розмір рангового блоку. Шляхом усереднення значень інтенсивності чотирьох сусідніх пікселів кожний доменний блок стискається так, щоб його розміри відповідали розмірам рангових блоків, і, шляхом рядкового сканування, перетвориться у вектор $D_k = (d_1, d_2, ..., d_k)$. Доменні блоки можуть перекривати (і, як правило, перекривають) один одного. Множина всіх отриманих доменних блоків складає пул доменів (кодову книгу).

Пошук. Для кожного рангового блоку R_m обчислюється оптимальна апроксимація $R_m \approx s \cdot D_k + o \cdot 1$, де $1 = (1,1,...,1)^T \in \Re^n$. Для цього виконуються наступні кроки:

• Для кожного блоку D_j з кодової книги обчислюється оптимальне наближення $R_m \approx s \cdot D_j + o \cdot 1$:

- Проводиться оптимізація за методом найменших квадратів по формулах (1) і (2), внаслідок чого отримують дійсні коефіцієнти:

$$s = \frac{n \cdot (\sum_{i=1}^{n} d_i \cdot r_i) - (\sum_{i=1}^{n} d_i) \cdot (\sum_{i=1}^{n} r_i)}{n \cdot (\sum_{i=1}^{n} d_i^2) - (\sum_{i=1}^{n} d_i)^2} \quad (1)$$

$$o = \frac{\sum_{i=1}^{n} r_i - s \cdot (\sum_{i=1}^{n} d_i)}{n} \quad (2)$$

де:

$R_m = (r_1, r_2, ..., r_n)$ - вектор, що представляє ранговий блок;

$D_i = (d_1, d_2, ..., d_n)$ - вектор, що представляє доменний блок;

n - розмірність вектора рангового (доменного) блоку;

- Виконується квантування коефіцієнтів s та o.

• Значення квантованих коефіцієнтів s та o підставляються у формулу (3) та обчислюється значення функції помилки $E(R_m, D_j)^2$

$$E(R_m, D_j)^2 = \frac{1}{n} \cdot \left[\sum_{i=1}^{n} r_i^2 + s \cdot \left(s \cdot \sum_{i=1}^{n} d_i^2 - 2 \cdot \sum_{i=1}^{n} (d_i \cdot r_i) + 2 \cdot o \cdot \sum_{i=1}^{n} d_i \right) + o \cdot \left(o \cdot n - 2 \cdot \sum_{i=1}^{n} r_i \right) \right] \quad (3)$$

• Серед всіх блоків D_i з кодової книги знаходять блок D_k з найменшим значенням помилки $E(R_m, D_k)^2 = \arg\min_j E(R_m, D_j)^2$.

• Отримані коефіцієнти s, o та індекс (координати) оптимального кодового блоку D_k зберігаються як вихідні дані.

Проведемо пооперацiйний аналіз даного алгоритму. Підрахуємо кількість операцій, що задаються алгоритмом при стисненні півтонового зображення розміром $N \times N$ пікселів. Передбачимо, зображення розбивається на рангові блоки розміром $n_r \times n_r$ пікселів, $2 \cdot n_r$ ділить N. Всього отримуємо N^2 / n_r^2 блоків.

Якщо доменні блоки вибрані розміром, що в два рази перевищує розмір рангових блоків - $2n_r \times 2n_r$, і ми допускаємо їх розташування з кроком в 1 піксель, тобто, фактично не накладаємо ніяких обмежень на розмір доменного пулу, то виходить $(N - 2 \cdot n_r + 1)^2$ доменних блоків. Для кожного рангового блоку R_m ми повинні пройтися по всіх доменних блоках D_j та знайти оптимальні коефіцієнти s та o, отримати їх значення після квантування/деквантування і обчислити результуюче значення помилки $E(R_m, D_j)^2$ з цими коефіцієнтами.

Обчислення коефіцієнта s за формулою (1) при $n = 16$ вимагає 36 операцій множення і 72 операції складання, а обчислення коефіцієнта o за формулою (2) - 32 операції множення і 30 операцій складання. Обчислення ж значення помилки за формулою (3) займає 84 операції множення і 80 операцій складання. Таким чином, отримуємо загальну кількість операцій множення і складання при $n = 16$:

$N_{множ} = N^2 / n_r^2 \cdot (N - 2 \cdot n_r + 1)^2 \cdot (36 + 84)$;

$N_{склад} = N^2 / n_r^2 \cdot (N - 2 \cdot n_r + 1)^2 \cdot (32 + 80)$.

Покладемо розмір рангових блоків рівним 4×4. Відповідно, розмір доменних блоків дорівнюватиме 8×8. Підставляючи в наведені вище формули $n_r = 4$, отримаємо:

$$N_{множ} = 7.5 \cdot N^2 \cdot (N - 7)^2 \quad (4)$$

$$N_{склад} = 7 \cdot N^2 \cdot (N - 7)^2 \quad (5)$$

Очевидно, що асимптотична складність алгоритму складає $O(N^4)$ 0(N*), Дана оцінка є неприйнятно великою навіть для сучасних високопродуктивних процесорів. Не дивлячись на те, що ефективність алгоритму можна трохи збільшити, оптимізувавши формули обчислення s, o та $E(R_m, D_j)^2$, це не дає прийнятного часу архівації. Тому всі сучасні дослідження в області фрактального стиснення направлені на пошуки шляхів зменшення об'єму обчислень, що виконуються при кодуванні, із збереженням прийнятної якості зображення. Розглянемо різні методи підвищення часової ефективності стандартного фрактального алгоритму стиснення зображень.

Методи класифікації

Найбільш простим і інтуїтивно зрозумілим способом підвищення ефективності фрактального алгоритму є різні схеми класифікації.

Метод Джеквіна. У оригінальному методі Джеквіна доменні блоки розбиваються на класи згідно геометричним характеристикам, при цьому виділяється тільки три основні характеристики блоків: тіньові блоки, блоки різкості і серединні блоки (клас, що представляє щось середнє між першими двома). У тіньових блоках значення інтенсивності варіюється трохи, тоді як в блоках різкості присутні великі перепади яскравості пікселів. Таке відбувається, наприклад, на кордоні деякого об'єкту, що відображується. Клас блоків різкості ділиться ще на два підкласи - прості і змішані блоки. Серединні блоки мають великі перепади інтенсивності в порівнянні з тіньовими блоками, проте градієнт перепадів у них не настільки яскраво виражений як біля блоків різкості. Як правило, ці блоки містять елементи текстури. Оскільки рангові блоки, які класифікуються як тіньові блоки, добре апроксимуються вектором 1 з постійною яскравістю, помноженому на визначений коефіцієнт o, для них немає необхідності в пошуку відповідного домена. Фактично для таких блоків коефіцієнт s встановлюється в 0. Таким чином, в даній схемі присутні тільки два класи доменних блоків, в одній з яких проводиться пошук для кожного не тіньового рангового блоку.

Класифікація за варіацією і інтенсивністю. У класифікації Фішера, Якобса і Боса доменний блок розбивається на 4 підблоки (верхній лівий, верхній правий, нижній лівий, нижній правий) і в кожному з них обчислюється середня інтенсивність A_i і варіація V_i пікселів ($i = 1, 2, 3, 4$). Далі, доменні блоки впорядковуються в залежності від того, як розташовуються інтенсивності їх підблоків:

основний клас 1: $A_1 \geq A_2 \geq A_3 \geq A_4$

основний клас 2: $A_1 \geq A_2 \geq A_4 \geq A_3$

основний клас 3: $A_1 \geq A_4 \geq A_2 \geq A_3$

Як тільки орієнтація по інтенсивності зафіксована, існує ще 24 різних варіанти розташування значень варіації, що визначає 24 підкласи для кожного основного класу. Якщо коефіцієнт масштабування s в апроксимації

$sD_k + o1$ для рангового блоку від'ємний, то порядок виразів в класах необхідно відповідно змінити. Таким чином, для кожного даного рангового блоку необхідно виконати пошук в 2 з 72 підкласах для того, щоб врахувати як додатні, так і від'ємні значення коефіцієнта масштабування.

Даний підхід не є задовільним в тому плані, що поняття сусідніх класів чітко не визначене. Так, якщо пошук в одному з підкласів не дає прийнятної відповідності для домена і рангового блоку, немає можливості розширити пошук на сусідні підкласи. Вирішення даної проблеми дане в роботі Касо, Обредора і Куо, де недостатньо гнучкий порядок значень варіацій пікселів був заміщений на вектори варіацій. Ці вектори квантуються, що дає набір класів, де біля кожного класу є сусідні класи, в яких у разі потреби може бути проведений пошук. Інше рішення представлене в методах кластеризації, які розглянемо нижче.

Класифікація архетипів. У роботі Якобса і Боса представлений метод, який визначає класи апріорно в результаті деякого емпіричного дослідження навчальних зображень. Архетипом набору доменних блоків є доменний блок, який щонайкраще покриває всі інші доменні блоки з погляду середньоквадратичної погрішності. Для множини блоків D_i це є блок D_k такий, що:

$$D_k = \arg\min \sum_{i \neq k} \min_{\substack{D_k \\ a,b}} \left\| D_i - (s \cdot D_k + o \cdot B) \right\|,$$

де B - множина фіксованих (базисних) блоків, незалежних від вхідного зображення.

Починаючи з будь-якої класифікації (наприклад, даною Фішером) блоків, взятих з навчальних зображень, можна обчислити архетип для кожного класу. Потім блоки знову класифікуються згідно архетипу, яким вони покриваються щонайкраще. В результаті виходить нова класифікація, і процес обчислення архетипа і нової класифікації повторюється доти, поки в процесі ітерації з'являються зміни. Фінальний набір архетипів стає часткою алгоритму кодування. Для даного зображення, що необхідно стискувати, алгоритм кодування визначає доменний пул, і класифікує всі доменні блоки, тобто для кожного доменного блоку шукається архетип, який щонайкраще покриває доменні блоки, що залишилися. Таким чином, можна вважати, що даний ранговий блок може бути добре апроксимований доменним блоком з відповідного класу. Фактично, це підтверджується результатами, отриманими в роботі. Для того, щоб досягти певної якості декодованого зображення, необхідно виконати пошук по меншому числу класів, що підвищує часову ефективність алгоритму. З іншого боку, процес класифікації складніший в порівнянні з іншими схемами. В результаті, звичайні схеми класифікації є швидшими при кодуванні зображень з низькою якістю, тоді як якнайкраща якість декодованого зображення може бути досягнута набагато швидше при використанні класифікації архетипів.

Методи кластеризації. У методах кластеризації доменні блоки і рангові блоки групуються навколо кластерних центрів, які обчислюються або адаптивно з даного зображення, або з набору навчальних зображень. Класи залежать від вибраного алгоритму кластеризації і від функції-критерію, що використовується для опису якості кластеризації. Перша спроба застосувати адаптивну кластеризацію з використанням нейронних мереж Кохонена для фрактального стиснення зображень представлена в роботі Богдана і Мідовса [50]. Реалізація з використанням частотно-контекстних нейронних мереж дана в роботі Уолла і Кинснера. Ефективний метод кластеризації, заснований на алгоритмі LBG, запропонований в Йенаб Лепсой, Рамстеда.

У цих роботах також вперше представлена техніка децимації блоків, яка дозволяє виконувати кластеризацію на нижчому рівні здатності (якості). У [71] підхід на основі нейронних мереж скомбінований з класифікацією блоків за інтенсивністю по Фішеру, і з характеристичними векторами по Саупу і отримана схема класифікації за відстанню.

Розглянемо детальніше один з підходів реалізації схеми кластеризації. Нехай $\{\pm\phi(D_1),...,\pm\phi(D_{N_D})\}$ є множиною спроектованих доменних блоків. Необхідно розділити дану множину на кінцеве число непересічних підмножин (кластерів), визначених кластерними центрами так, що вектори в одному кластері ближче один до одного, ніж вектори в різних кластерах. Якість кластеризації вимірюється функцією-критерієм, яку і необхідно оптимізувати. Наприклад, можна вибрати і сконструювати кластерні центри так, що сума квадратів евклідових відстаней

$$J = \sum_{j=1}^{N_D} \left\| \phi(D_j) - m(\phi(D_j)) \right\|^2 + \left\| -\phi(D_j) - m(\phi(D_j)) \right\|^2$$

буде мінімальною. Тут $m(\pm\phi(D_j))$ означає центр кластера, що є найближчим до спроектованого доменного блоку $\pm\phi(D_j)$. Центр кластера m формується групуванням біля m всіх спроектованих доменних блоків, у яких m є найближчим сусіднім блоком. Після того, так кластерні центри були вибрані, множина спроектованих доменних блоків $\{\pm\phi(D_1),...,\pm\phi(D_{N_D})\}$ кластеризується відображенням кожного вектора $\pm\phi(D_j)$ на його найближчий кластер. Ранговий блок R_m кодується в два етапи. Спочатку його характеристичний вектор $\phi(R_m)$ відображується на найближчий кластерний центр $m(\phi(R_m))$. Потім, ранговий блок порівнюється тільки з доменними блоками, характеристичні вектори яких належать кластерним центрам $m(\phi(R_m))$. Можливо також проводити пошук в найближчих сусідніх кластерах, у випадку якщо не вдалося апроксимувати ранговий блок із заданою точністю. Це дозволяє виконати акуратніше кодування зображення за рахунок збільшення часу роботи алгоритму.

Інваріантні моменти. Новак присвоює кожному доменному блоку 4-

елементний характеристичний вектор. Компонентами характеристичного вектора є інваріантні моменти, визначені згідно розподілу градацій сірого в межах блоку. Корисна властивість інваріантних моментів полягає в тому, що вони інваріантні щодо геометричних трансформацій, тобто можна використовувати всього один характеристичний вектор на один домен. Ізометрично трансформовані версії доменного блоку мають один і той самий інваріантний момент. Проте, моменти не є інваріантними для афінних трансформацій щодо яскравості. Щоб обійти цю проблему, Новак запропонував процедуру нормалізації. У даному підході існує три проблеми: блоки з від'ємною яскравістю відкидаються, і якість декодованого зображення, таким чином, знижується. Значення інваріантних моментів варіюються в межах декількох порядків, так що доводиться використовувати логарифмічне перемасштабування перш ніж виконувати пошук найближчого сусіднього блоку. І, найголовніше, метод є емпіричним в тому сенсі що не існує теорії, на якій було б засновано твердження, що близькість векторів в характеристичному просторі означає гарну апроксимацію по методу найменших квадратів.

Функціональні методи

У [6] Бедфорд, Деккінс і Книг пропонують критерій, який дозволяє стверджувати, що даний доменний блок не може дати гарну апроксимацію для рангового блоку. Ідея полягає в тому, щоб порівнювати не рангові і доменні блоки один з одним, а рангові і доменні блоки з незалежно вибраним одиничним вектором. І лише у тому випадку, коли результати цих порівнянь збігаються для доменного і рангового блоку, то вони порівнюються між собою. Це дозволяє усунути велику кількість доменно-рангових порівнянь і винести основну частину обчислень на етап передобчислень.

Теорема 1.3.2 (Необхідна умова):

Нехай $\delta > 0$ та U - одиничний вектор в $\Re^n$. Нехай $R_m, D_k \in \Re^n$ та $\langle R_k, \phi(R_k) \rangle \geq \delta$. Якщо $E = \min_{s, o_j \in R} \left\| R_m - (sD_k + \sum_{j=1}^{p} o_j B_j) \right\| \leq \delta$, то

$$\left\| \left| \langle \phi(R_m), U \rangle \right| - \left| \langle \phi(D_k), U \rangle \right| \right\| \leq \sqrt{2 - 2\sqrt{1 - \frac{\delta^2}{\langle R_m, \phi(R_m) \rangle^2}}} \qquad (2.1)$$

Доведення. Використовуючи нерівність Коші-Шварца, обчислимо:

$$\left(\left| \phi(R_m) - \phi(D_k), U \right| \right)^2 \leq \left(\langle \phi(R_m), U \rangle - \langle \phi(D_k), U \rangle \right)^2 =$$

$$= \left| \langle \phi(R_m) - \phi(D_k), U \rangle \right|^2 \leq \left\| \phi(R_m) - \phi(D_k) \right\|^2 \cdot \left\| U \right\|^2 =$$

$$= \left| \phi(R_m) - \phi(D_k) \right|^2 = 2 - 2\langle \phi(R_m), \phi(D_k) \rangle$$

Якщо через ρ позначити $\langle \phi(R_m), \phi(D_k) \rangle$, то квадрат помилки

$E^2 = \langle R_m, \phi(R_m) \rangle^2 (1-\rho^2)$. Таким чином, з припущення $E \le \delta$ виходить, що

$$\langle R_m, \phi(R_m) \rangle^2 (1-\rho^2) \le \delta^2$$

Можна покласти, що $\rho \ge 0$ (інакше замінити D_k на $-D_k$), тоді

$$\rho \ge \sqrt{1 - \frac{\delta^2}{\langle R_m, \phi(R_m) \rangle^2}}.$$

Підставляючи даний результат в нерівність вище доводимо початкове твердження теореми. Легко перевірити, що в разі $\rho = 1$ коли $\Im$ представлено фіксованим блоком постійної яскравості, $B = (1,1,...,1)/\sqrt{n}$, для будь-якого блоку $Z \in \Re^n$ отримуємо

$$\phi(Z) = \frac{1}{\sqrt{V(Z)}}(z_1 - \bar{z}, ..., z_n - \bar{z})$$

де $\bar{z} = (z_1 + z_2 + ... + z_n)/n$ усереднена яскравість і $V(Z) = \sum_{k=1}^{n} (z_k - \bar{z})^2$ дисперсія.

Функціональний алгоритм має такий вигляд:

Функціональний алгоритм

1. Вибрати параметр толерантності δ та одиничний вектор U.
2. Передобчислення: для кожного доменного блоку D_k обчислити $\left|\langle \phi(D_k), U \rangle\right|$.
3. Для кожного рангового блоку: обчислити $\langle R_m, \phi(R_m) \rangle$ і верхню границю згідно (2.1).
4. Якщо $\langle R_m, \phi(R_m) \rangle < \delta$, то немає необхідності в пошуку, оскільки $s = 0$ дає середньоквадратичну помилку $E = \langle R_m, \phi(R_m) \rangle \le \delta$ для будь-якого доменного блоку D_k.
5. Якщо $\langle R_m, \phi(R_m) \rangle > \delta$, то обчислити $\left|\langle \phi(R_m), U \rangle\right|$ та відкинути всі доменні блоки D_k, для яких нерівність (2.1) теореми (2) не виконується.

Можна також розширити даний метод, розглянувши декілька незалежних одиничних векторів U і для кожного обчислювати критерій за теоремою (2). Тоді можна чекати, що кількість відкинутих доменних блоків зросте.

Ефективна реалізація функціонального методу не повинна виконувати сканування всього доменного пулу для того, щоб виділити тільки ті домени, які задовольняють умові теореми. Замість цього, як і з будь-яким функціональним методом, краще використовувати алгоритм подібний

наступному:

Узагальнений функціональний алгоритм

Передбачимо, що функція $F:\Re^n \to \Re$ задана таким чином, що $|F(R_m)-F(D_k)| \le \varepsilon_R$ означає, що ранговий блок R_m може бути гарно апроксимована доменом D_k. Позначимо доменний пул як $\{D_1,...,D_{N_D}\}$. На кроці передобчислень виконуємо:

1. Для кожного домена $D_k \in \{D_1,...,D_{N_D}\}$ обчислити $F(D_k)$.
2. Сортувати всі доменні блоки згідно значенню функції в лінійний масив помітити масиви наново так, що

$F(D_1) \le F(D_2) \le ... \le F(D_{N_D})$

Для кожного рангового блоку R виконати:

1. Для кожного рангового блоку: обчислити $\langle R_k, \phi(R_k) \rangle$
2. Обчислити $F(R_k)$ і верхню границю ε_R (тобто, праву частину у виразі (2.1)).
3. Використовуючи метод бінарного пошуку, знайти індекси k_0, k_j такі, що $F(R_m)-F(D_k) \le \varepsilon_R$ якщо, і лише якщо $k_0 \le k \le k_1$
4. Перевірити всі домени D_k з $k_0 \le k \le k_1$.

Даний алгоритм список доменних блоків, які здатні щонайкраще наблизити ранговий блок, складається за час $O(\log N_D)$, тоді як повний пошук з відкиданням доменів, які не минають тест займає час $O(N_D)$.

Методи деревовидного впорядкування

У [5] Касо, Обредора і Куо рекурсивно організовують пул доменних блоків в бінарне дерево. Спочатку з пулу випадковим чином вибираються два батьківські блоки. Потім блоки, що залишилися, поміщаються в одну з двох гілок дерева залежно від того, який з двох батьківських блоків дає якнайкраще наближення даного доменного блоку по методу найменших квадратів. В результаті пул розбивається на дві підмножини. Цей процес повторюється рекурсивно до тих пір, поки не буде досягнутий заданий розмір піддерева.

Тепер, для даного рангового блоку можна знайти в дереві шлях, який в кожній крапці дає якнайкраще наближення по методу найменших квадратів, і вибрати найбільш відповідні коефіцієнти. Такий підхід не обов'язково дає якнайкращі коефіцієнти в глобальному сенсі. Проте, можна розширити пошук і на сусідні з «ідеальною» гілкою елементи. Це дасть якнайкраще (або достатньо гарне до нього наближення) рішення. Для даної мети використовується тест на основі кутового критерію. Процедура близька до пошуку найближчого сусіднього блоку в [6], оскільки критерій найменшої середньоквадратичної погрішності (мінімізація $E(D_k, R_m)$) еквівалентний

критерію відстані (мінімізації $\Delta(\phi(D_k),\phi(R_m))$. Таким чином, бінарне дерево, лежаче в основі методу, може розглядуватися як рандомізована версія kd - дерева.

Ван-де-Уол працював над вейвлет-представленням фрактального стиснення зображень. По аналогії із звичайним фрактальним стисненням рангові вектора (відповідні піддерева в дереві вейвлет-коефіцієнтів) порівнюються з доменними векторами (також відповідні вузли вейвлет-дерева), які можуть масштабуватися з будь-яким коефіцієнтом масштабування. Для кожного вузла створюється заснований на кутовому критерії між коефіцієнтами векторів в просторі вейвлет-коефіцієнтів характеристичний вектор. Потім ці вектора сортуються в мультирозмірну структуру даних, в якій застосовується алгоритм швидкого пошуку. З погляду відстаней між характеристичними векторами $\pm\phi(D_k)$ можна дати даному методу таку інтерпретацію: визначимо невелику множину базових точок в характеристичному просторі (у позиціях головних компонентів множини всіх характеристичних векторів). Для кожного (спроектованого і нормалізованого) доменного блоку, так само як і для кожного рангового блоку, обчислюються відстані Δ до базових точок. Потім, точка в характеристичному просторі, яка найбільш близька до даного рангового блоку, має обов'язково бути близька до базових точок, котрі недалеко від базових точок рангового блоку. Для того, щоб виконати пошук для даних доменних блоків, блоки організовуються в деревовидну структуру.

Методи на основі кратномасштабного аналізу

У [4] і [5] представлено два підходи до зменшення обчислювальної складності фрактального алгоритму на основі кратномасштабного аналізу. Ідея полягає в тому, щоб використовувати вейвлет-дерево зображення для зменшення кількості обчислень.

Спочатку пошук здійснюється на нижчому рівні здатності. Якщо на даному рівні не знайдено відповідних доменних блоків, то і на більш високому рівні здатності їх не буде. Підвищення ефективності алгоритму досягається тим, що на нижчому рівні здатності потрібна менша кількість операцій для знаходження коефіцієнтів по методу найменших квадратів.

Нехай зображення f представлено у вигляді послідовності зображень $f(0),...,f(r)$, де $f(r)=f$ та

$$f^{(k)}=\frac{1}{4}\sum_{m,i=0}^{1} f^{(k+1)}(2i+m,2j+l).$$

Для $k=0,...,r-1$ та $0\le i,j<2k$. В [4] показано, що, якщо $R_m^{(k)}$ та $D_k^{(k)}$ і є ранговий та доменний блок на рівні здатності k відповідно, то

$$E(D_k^{(k+1)},R_m^{(k+1)})\ge E(D_k^{(k)},R_m^{(k)})$$

Проте, не всі домени на рівні здатності $k+1$ мають відповідні домени на

рівні здатності k. Застосування міркувань, наведених вище, враховує тільки домени на рівні здатності, верхні ліві кути яких розташовані в позиціях $(2i, 2j)$. Для подолання цієї проблеми можна розглянути пірамідальне дерево, де на кожному рівні здатності $k+1$ у домена є чотири зв'язані домени на здатності k в позиціях $(2i, 2j), (2i+1, 2j), (2i, 2j+1), (2i+1, 2j+1)$. Також слід врахувати, що не можна відкинути домен на здатності $k+1$ тільки тому, що у його зв'язаного домена на здатності k стискуючий чинник $|s \cdot k| > 1$. Цілком можливі випадки, коли $|s \cdot k| > 1$, проте $s \cdot k + 1 = 0$.

Методи швидкого пошуку через швидку згортку

Більшість методів, що розглянуті раніше, є методами з втратами в тому сенсі, що вони досягають прискорення за рахунок деякої втрати якості стислого зображення. Метод, представлений тут є, навпаки, методом прискорення фрактального стиснення без втрат. Він заснований на тому, що доменні блоки зазвичай накладаються один на одного. Швидка згортка, заснована на теоремі про швидку згортку в частотному домені, є ідеальним інструментом для дослідження такого роду залежностей. Нехай $<\cdot,\cdot>$ позначає скалярний добуток в просторі Евкліда розмірності n, де n - кількість пікселів в ранговому блоці. Тоді, формули (1) і (2) можна переписати у вигляді:

$$s = \frac{n < D_k, R_m > - < D_k, \mathbf{1} >< R_m, \mathbf{1} >}{n < D_k, D_k > - < D_k, \mathbf{1} >^2},$$

$$o = \frac{1}{n}(< R_m, \mathbf{1} > - s < D_k, \mathbf{1} >)$$

Обчислення зазвичай організовуються в два вкладені цикли:

- Глобальні передобчислення: обчислити $< D_k, D_k >, < D_k, 1 >$ для всіх доменів D_k.
- Для кожного рангового блоку R_m виконати:
  - Локальні передобчислення: обчислити $< R_m, R_m >, < R_m, 1 >$.
  - Для всіх доменних блоків обчислити $< D_k, R_m >$ та $E(D, R_m)$.

Обчислення добутку $< D_k, R_m >$ у внутрішньому циклі вносить основний вклад до обчислювальної складності кодування. Доменні блоки D_k зазвичай є частинами зменшеного в два рази зображення. Будь-який підблок такого зображення, який має таку ж форму як і ранговий блок, може розглядатися як доменний блок. З цієї точки зору, добутки $< D_k, R_m >$ у внутрішньому циклі є ні чим іншим, як кінцевою імпульсною характеристикою зменшеного зображення по відношенню до рангового блоку. Іншими словами, необхідна конволюція (або точніше кроскореляція)

рангового блоку R_m і зменшеного зображення. Ця дискретна дворозмірна згортка може бути обчислена набагато ефективніше в частотному домені, коли розміри рангового блоку не дуже малі (теорема згортки). Дана процедура виносить обчислення скалярного добутку $< D_k, R_m >$ з внутрішнього циклу і розміщує його на крок передобчислень. При цьому використовується швидка згортка перетворення Фур'є.

1. *Kenneth Falconer.* Fractal geometry. Mathematical foundations and applications. – JOHN WILEY & SONS: Chichester, New York, Brisbane, Toronto, Singapore, 1990.
2. *Бондаренко В.А., Дольников В.Л.* Фрактальное сжатие изображений по Барнсли-Слоану. Автоматика и телемеханика. 1994. №5. С. 12-20.
3. *Ваганова Н. А.,* Фрактальное сжатие динамических изображений. Информационные технологии, Новосибирск 1999.
4. *Ватолин Д.С,* Использование ДКП для ускорения фрактального сжатия изображений. Программирование, Номер 3, 1999, стр. 51-57.
5. *Ватолин Д.С.* Алгоритмы сжатия изображений II ISBN 5-89407-041-4 М.: Диалог-МГУ, 1999.
6. *Вишик М.И.* Фрактальная размерность множеств. Соросовский образовательный журнал, № 1. 1998.
7. *Вигтих В.А., Сергеев В.В., Сойфер В.А.* Обработка изображений в автоматизированных системах научных исследований. - М.: Наука, 1982. - 213 с.
8. *Горбачев А.А. Колданов А.П., Потапов А.А., Чигин Е.П.,* Нелинейная радиолокация. Серия "Фракталы. Хаос. Вероятность", М.: Радиотехника, 2005.
9. *Гонсалес Р., Вудс Р.,* Цифровая обработка изображений— М.: Техносфера, 2005. — 1072 с.
10. *Добеши И.* Десять лекций по вычетам II Пер. с анг. Е.В. Мищенко, под ред. А.П.Петухова. М.: Ижевск 2001, 464 стр.
11. *Дьяконов В. П.,* Вейвлеты. От теории к практике. Изд. 2-е, перераб. И доп. - М.: СОЛОН-Прссс, 2004. - 400 с.
12. *Забарянский С.Ф.* Фрактальное сжатие изображений. Компьютеры + программы, № 6(39), 1993.

Поступила 30.01.2009р.

УДК 004.83

Ю. В. Добронравин, аспирант Международного научно-учебного центра информационных технологий и систем, Н.С. Самарин, студент НТУУ «КПИ» ИПСА.

GUIDE BOT – СИСТЕМА НАВИГАЦИИ АНТРОПОМОРФНЫХ АГЕНТОВ ВИРТУАЛЬНОЙ РЕАЛЬНОСТИ

Agents' navigation and movement in 3D space is a high priority task for the developers of virtual reality. Guide Bot is a professional AI pathfinder solution for

