КОМП'ЮТЕРНІ ЗАСОБИ, МЕРЕЖІ ТА СИСТЕМИ

УДК 004.932

П.Ю. САБЕЛЬНІКОВ

ОБЧИСЛЕННЯ МОМЕНТІВ ІНЕРЦІЇ КОНТУРІВ ОБ'ЄКТІВ У ЗОБРАЖЕННЯХ З ВИКОРИСТАННЯМ ВЕКТОРНИХ ОПЕРАЦІЙ

Мета роботи – це дослідження, що пов'язані з модифікацією алгоритмів обробки відеоданих та їх реалізацією за допомогою векторних операцій, що може значно прискорити обчислювальний процес при розв'язанні подібних задач на сучасних обчислювальних пристроях, або надасть поштовх для створення нових високоефективних паралельних процесорів. Це обумовлено наявністю на ринку процесорів, в яких реалізовані підсистеми команд з апаратною підтримкою векторних операцій, з фіксованою і плаваючою точкою (наприклад, процесори фірми ARM з системою команд ARM v 8), а також наявністю високоінтегрованих ПЛІС (наприклад, FPGA фірми Xilinx, Virtex-4, 5, 6, 7), на яких можна реалізувати векторні процесори для паралельної обробки декількох тисяч даних.

Раніше автором у роботі [1] для задач порівняння об'єктів у зображеннях за їх формою за умов афінних перетворень та часткового спотворення був запропонований метод і алгоритм геометричного порівняння контурів з використанням моментів інерції окремих ділянок контурних ліній, що дозволяє порівнювати з еталоном як замкнені контури об'єктів, так і окремі їх розірвані відрізки.

Метод передбачає перед безпосереднім накладанням і співставленням контурів об'єкта і еталона здійснювати пошук їх тотожних ділянок за моментними функціями [2] інваріантними до зсуву, повороту і масштабу. Для обчислення таких функцій використовуються моменти інерції різних поряд-

P.Y. Sabelnikov

THE CALCULATION OF MOMENTS OF INERTIA OF THE CONTOURS OF OBJECTS IN IMAGES USING VECTOR OPERATIONS

Algorithms of calculation of moments inertia of the contours of objects in images using vector operations are proposed.
Key words: contour, moments of inertia, vector operations.

Предложены алгоритмы вычисления моментов инерции контуров объектов на изображениях с использованием векторных операций.
Ключевые слова: контур, моменты инерции, векторные операции.

Запропоновані алгоритми обчислення моментів інерції контурів об'єктів у зображеннях з використанням векторних операцій.
Ключові слова: контур, моменти інерції, векторні операції.

ків цих ділянок, які розраховуються як сума моментів ліній, що з'єднують сусідні пікселі по горизонталі, вертикалі та діагоналі.

Формула обчислення моментів контурних ліній до *к* -ого порядку:

$$M^{j,k-j} = \sum_{i=1}^{n-1} b_s \cdot x_i^j y_i^{k-j}, \; j=(0, k),$$

де $M^{j,k-j}$ – моменти контурних ліній;

b_i – значення, які дорівнюють 1 для міжпіксельних ліній по горизонталі і вертикалі та $\sqrt{2}$ для міжпіксельних ліній по діагоналі;

x_i, y_i – координати середин міжпіксельних ліній;

n – кількість пікселів контурної лінії (кількість міжпіксельних ліній на одиницю менше).

Розрахунок моментів відрізків контурних ліній буде простим і мало затратним за часом, якщо при обході і перетворенні в векторну форму контурних ліній для кожної з них будуть обчислені вектори моментів *VM* розміром *N-1*, наприклад, VM^{00}, VM^{01}, VM^{10}, VM^{02}, VM^{20}, VM^{11} для k=2. Кожний компонент вектора *VM* дорівнює моменту відрізка від початку до відповідної точки контурної лінії.

Тоді момент *M*(*Б–С*) від точки Б до точки С буде дорівнювати (рис. 1):

$$M(Б–С) = VM(А–С) – VM(А–Б),$$

де А – початок контурної лінії, *VM*(*А–С*) – момент відрізка від А до С, *VM*(*А–Б*) – момент відрізка від А до Б.

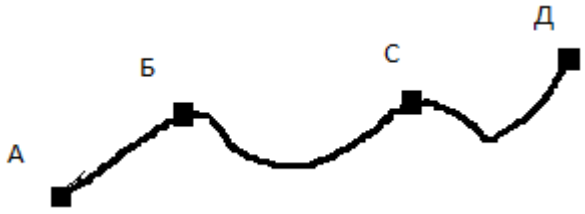

РИС. 1

Алгоритм обчислення моментів контурних ліній

Пояснимо правила обчислення векторів моментів контурних ліній на прикладі тестового бінарного зображення на рис. 2.

Отже, маємо зображення, де по горизонталі пронумеровані координати *x*, а по вертикалі координати *y*. Визначаємо контур, як замкнену ломану лінію, яка з'єднує центри граничних пікселів зовнішнього чорного або внутрішнього білого об'єкта, що примикають один до одного по прямій або діагоналі і належать цьому об'єкту. Щоб не виникало колізій з перетинанням контурів встановимо, що точка між пікселями при переходах по діагоналі належить чорним об'єктам і є перепоною для білих об'єктів.

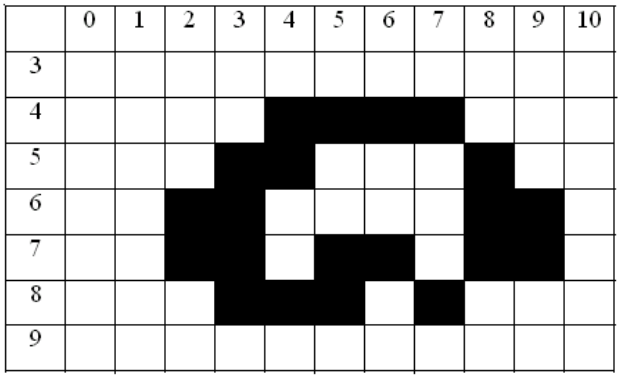

РИС. 2

За початок контурної лінії будемо брати лівий верхній піксел, а порядок слідкування буде проти годинникової стрілки. Тоді після перетворення контуру зовнішнього чорного об'єкта у векторну форму представимо його сукупністю двох векторів однакового розміру X і Y, компонентами яких є координати граничних пікселів відповідно по горизонталі і вертикалі. Оскільки контур замкнений його початком і кінцем є один і той же піксел.

Тому розміри цих векторів будуть на одиницю більше ніж кількість пікселів у контурі.

$$X = \{4; 3; 2; 2; 3; 4; 5; 6; 7; 8; 9; 9; 8; 7; 6; 5; 4\},$$
$$Y = \{4; 5; 6; 7; 8; 8; 8; 7; 8; 7; 7; 6; 5; 4; 4; 4; 4\}.$$

Надалі будемо застосовувати скалярні векторні операції, що означають арифметичні і логічні операції над компонентами векторів.

Вектор B з компонентами довжин міжпіксельних ліній, розмір якого буде дорівнювати кількості пікселів у контурі, обчислюється наступним чином:

$$XD = \left|X - (X << 1)\right| = \{1; 1; 0; 1; 1; 1; 1; 1; 1; 1; 0; 1; 1; 1; 1; 1\},$$
$$YD = \left|Y - (Y << 1)\right| = \{1; 1; 1; 1; 0; 0; 1; 1; 1; 0; 1; 1; 1; 0; 0; 0\},$$
$$B = (C1 - XD \times YD) + (C2 \times XD \times YD),$$
$$B = \{\sqrt{2}, \sqrt{2}, 1, \sqrt{2}, 1, 1, \sqrt{2}, \sqrt{2}, \sqrt{2}, 1, 1, \sqrt{2}, \sqrt{2}, 1, 1, 1\},$$

де XD, YD – вектори міжпіксельних відстаней відповідно по горизонталі і вертикалі;

$C1$, $C2$ – вектори, компонентами яких є константи, відповідно 1 і $\sqrt{2}$;

<<1 – операція зсуву компонентів вектора ліворуч на одну позицію;

×, /, –, + – покомпонентні арифметичні операції, відповідно множення, ділення, віднімання і додавання.

Зауважимо, що для множення векторів *XD* і *YD* замість арифметичного множення можна використовувати операцію логічного множення.

Обчислення векторів координат середин міжпіксельних ліній *XL* і *YL* також достатньо просте.

$$XL = (X + (X << 1)) / 2 =$$
$$=\{3.5;\ 2.5;\ 2;\ 2.5;\ 3.5;\ 4.5;\ 5.5;\ 6.5;\ 7.5;\ 8.5;\ 9;\ 8.5;\ 7.5;\ 6.5;\ 5.5;\ 4.5\}$$
$$YL = (Y + (Y << 1)) / 2 =$$
$$=\{4.5;\ 5.5;\ 6.5;\ 7.5;\ 8;\ 8;\ 7.5;\ 7.5;\ 7.5;\ 7;\ 6.5;\ 5.5;\ 4.5;\ 4;\ 4;\ 4\}.$$

За рахунок векторних операцій множення обчислюємо вектори з компонентами *XL* і *YL* в необхідних ступенях та помножуючи їх на вектор *B* отримуємо вектори, компонентами яких є моменти міжпіксельних ліній необхідного порядку. Наприклад, вектором моментів нульового порядку є безпосередньо сам вектор *B*, оскільки координати межпіксельних ліній у нульовій степені дорівнюють одиниці.

Отримати вектори *VM* можно послідовно підсумовуючи компоненти векторів моментів міжпіксельних ліній. При цьому кількість операцій підсумовування буде дорівнювати кількості пікселів у контурі для кожного з векторів.

Автором пропонується спосіб і алгоритм отримання векторів *VM* за допомогою векторних операцій, що значно пришвидшить обчислювальний процес. Спосіб придатний для отримання такого роду векторів, маючі на вході вектори з довільними значеннями компонентів. Він полягає у наступному.

Нехай вхідний вектор з кількістю компонентів *n*, що дорівнює 10, заноситься в результуючий вектор *T*.

$$T = \{t_0;\ t_1;\ t_2;\ t_3;\ t_4;\ t_5;\ t_6;\ t_7;\ t_8;\ t_9\}.$$

За послідовними кроками результуючий вектор складається з таким же вектором, але зсунутим праворуч на 2^i позицій.

$$T = T + (T >> 2^i),$$

де i – номер кроку, що змінюється в діапазоні від 0 до

$$(Ceil(\log_2 n)) - 1, \qquad (1)$$

функція $Ceil(A)$ повертає найближче більше або рівне A число, $>> 2^i$ – операція зсуву компонентів вектора праворуч на 2^i позицій.

При зсуві додатка звільнені зліва позиції заповнюються нулями.

Доказ коректності роботи алгоритму обчислення векторів *VM* запропонованим способом.

Доказ побудовано за рахунок дослідження крок за кроком номерів компонентів вектора, що підсумовуються в кожній позиції. Достатньо показові результати наведені в табл. 1. В комірках таблиці надані номери компонентів вхідного вектора, сума яких у кожній позиції вектора *T* отримана на попередньому кроці,

а також ті, що додаються до них на поточному кроці. Кількість кроків згідно з виразом (1) для n рівного 10 дорівнює 4. Прочерками в таблиці вказані позиції, які заповнюються нулями після зсуву. З таблиці видно, кінцевий результат у кожній позиції має суми компонентів вхідного вектора від нульової до поточної позиції, тобто те, що потрібно було отримати.

ТАБЛИЦЯ 1

Крок i		Номери компонентів вхідного вектора, сума яких отримана в T на попередньому кроці, а також ті, що додаються на поточному кроці									
0	T	0	1	2	3	4	5	6	7	8	9
	додаток	-	0	1	2	3	4	5	6	7	8
1	T	0	1;0	2;1	3;2	4;3	5;4	6;5	7;6	8;7	9;8
	додаток	-	-	0	1;0	2;1	3;2	4;3	5;4	6;5	7;6
2	T	0	1;0	2;1;0	3;2;1;0	4;3;2;1	5;4;3;2	6;5;4;3	7;6;5;4	8;7;6;5	9;8;7;6
	додаток	-	-	-	-	0	1;0	2;1;0	3;2;1;0	4;3;2;1	5;4;3;2
3	T	0	1;0	2;1;0	3;2;1;0	4;3;2;1;0	5;4;3;2;1;0	6;5;4;3;2;1;0	7;6;5;4;3;2;1,0	8;7;6;5;4;3;2;1	9;8;7;6;5;4;3;2
	додаток	-	-	-	-	-	-	-	-	0	1;0
Кінцевий результат T		0	1;0	2;1;0	3;2;1;0	4;3;2;1;0	5;4;3;2;1;0	6;5;4;3;2;1;0	7;6;5;4;3;2;1;0	8;7;6;5;4;3;2;1;0	9;8;7;6;5;4;3;2;1;0

Перевіримо роботу алгоритму на конкретному прикладі. Припустимо, що після запису вхідного вектора в вектор T, останній буде дорівнювати такій послідовності чисел:

$$T = \{6; 3; 1; 5; 3; 2; 7; 5; 1; 3\}.$$

Послідовно підсумовуючи компоненти вектора, отримуємо наступний кінцевий результат:

$$T = \{6; 9; 10; 15; 18; 20; 27; 32; 33; 36\}.$$

А тепер продемонструємо роботу запропонованого способу і алгоритму для наведеного прикладу за допомогою табл. 2, схожій на табл. 1. В комірках цієї таблиці надані значення компонентів вектора *T*, які отримані на попередньому кроці, а також значення, що додаються на поточному кроці.

Кінцеві результати співпадають, що підтверджує правоту наданих доказів. При цьому кількість операцій підсумовування всіх векторів порівняно з кількістю підсумовувань при послідовному способі накопичення сум у $n/(Ceil(\log_2 n))$ разів менше.

ТАБЛИЦЯ 2

Крок i		Значення компонентів вектора *T*, які отримані на попередньому кроці, а також значення, що додаються на поточному кроці									
0	*T*	6	3	1	5	3	2	7	5	1	3
	додаток	0	6	3	1	5	3	2	7	5	1
1	*T*	6	9	4	6	8	5	9	12	6	4
	додаток	0	0	6	9	4	6	8	5	9	12
2	*T*	6	9	10	15	12	11	17	17	15	16
	додаток	0	0	0	0	6	9	10	15	12	11
3	*T*	6	9	10	15	18	20	27	32	27	27
	додаток	0	0	0	0	0	0	0	0	6	9
Кінцевий результат *T*		6	9	10	15	18	20	27	32	33	36

При можливості процесора одночасно обробляти тільки *k* даних, з урахуванням того, що в додатку (див. табл. 2) крок від кроку збільшується кількість нулів, які не потрібно додавати, виграш у кількості підсумовувань *Q* буде дорівнювати:

$$Q = \frac{n}{\sum_{i=0}^{Ceil(\log_2 n)-1} Ceil((n-2^i)/k)}.$$

Для простоти сприйняття оцінки розглянемо такі два випадки.

1), у цьому випадку, як було раніше вказано $Q = n/(Ceil(\log_2 n))$,

2) $k < n$, *k* і *n* дорівнюють ступеню двійки, в цьому випадку

$$Q = n/(\frac{n}{k}\times(\log_2 n) - (2^{\log_2 n - \log_2 k} - 1)) = n/(\frac{n}{k}\times(\log_2 n) - \frac{n}{k} + 1) =$$

$$= n/(\frac{n}{k}\times((\log_2 n) - 1) + 1).$$

Якщо не враховувати у виразі останню одиницю $Q = k/((\log_2 n)-1)$, тобто виграш у кількості підсумовувань буде більше одиниці тільки тоді, коли $k > ((\log_2 n)-1)$.

Висновки. Запропоновані алгоритми суттєво прискорять обчислювальний процес за умови їх реалізації на процесорах або спеціалізованих пристроях, що апаратно підтримують векторні операції. Ефективність залежить від функціональних можливостей процесора (повноти підсистеми векторних команд), кількості даних, які одночасно процесор може обробити та часу, потрібного для виконання однієї векторної команди. Як правило, за належною апаратною підтримкою час операції над двома векторами дорівнює часу операції над двома окремими даними. Таким чином, при обчисленні векторів моментів міжпіксельних інтервалів виграш буде в стільки разів скільки даних векторний процесор може паралельно обробити. При подальшому обчисленні векторів *VM* виграш буде значним, коли *k* буде значно більше ніж $(Ceil(\log_2 n)-1)$. Така ситуація цілком реальна при створенні спеціалізованих паралельних процесорів на сучасній елементній базі.

1. *Sabelnikov P.Y.* Algorithm geometric comparison of contour images of objects. *Journal of Qafqaz University. Mathematics and Computer Science.* (Baku). 2014. Vol. 2. N 2. С. 166–175.
2. *Глумов Н.И.* Построение и применение моментных инвариантов для обработки изображений в скользящем окне. *Компьютерная оптика.* 1995. № 14. С. 46–54.

Одержано 11.10.2016