ISBN 978-5-901795-13-2 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2008

М. Д. Недев

О приближенном вычислении свертки[*)]

Научный руководитель: д.т.н. проф. В. М. Хачумов

Аннотация. Работа посвящена реализации и сравнению двух схем приближенного вычисления свертки применительно к дискретным функциям, заданным матрицами в ограниченной области. Для ускорения счета проведен эксперимент с использованием суперкомпьютера семейства «СКИФ».

1. Введение

Свертка — один из важнейших процессов в цифровой обработке сигналов. Она получила широкое распространение в физике, в системах наведения. Поэтому важно уметь эффективно ее вычислять. Прямое вычисление свертки требует $N \times M$ умножений, где N — длина исходного сигнала, а M — длина ядра свертки. Для ее реализации используются различные методы.

Основное назначение быстрых вычислений — обработка в режиме реального времени радиолокационной, телевизионной и тепловизионной информации, выделение и обработка целей. Быстрые вычисления достигаются либо использованием математических преобразований [1], либо с помощью аппаратных и программно-аппаратных средств. Современная вычислительная техника располагает несколькими альтернативными подходами к ускорению вычислений свертки — это использование спецпроцессоров [2], многоядерных, многопроцессорных и кластерных вычислительных устройств (КВУ).

2. Постановка задачи

Математически задача вычисления свертки сводится к расчету двойного интеграла:

$$Z(\delta x, \delta y, k, \gamma) = \int\int_S F_1(x, y) \cdot F_2(x, y) ds$$

[*)] Представлено по тематике: *Математические основы программирования, Программное обеспечение для суперЭВМ.*

В литературе известны методы быстрого вычисления свертки, когда функции $F_1(x, y)$ и $F_2(x, y)$ представлены, например, в виде полиномов [1]. В нашем же случае речь идет об обработке цифровых изображений, доставленных системами технического зрения. Здесь $F_2(x, y)$ — полутоновое изображение, а $F_1(x, y)$ — преобразующая матрица (маска, фильтр). Пусть $F_1(x, y)$ задана дискретно на равномерной сетке (M_1, M_2) с размером ячейки l. $F_2(x, y)$ также задана дискретно на равномерной сетке (N_1, N_2) с размером ячейки $L = k \cdot l$, смещенной относительно первой сетки по осям X и Y на Δx и Δy соответственно и повернутой на угол γ. Графическое представление задачи показано на рис. 1.

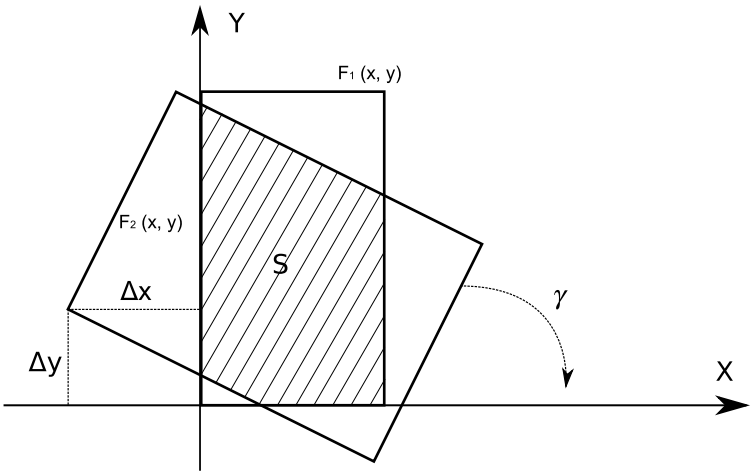

Рис. 1. Графическая интерпретация задачи

3. Методы решения

Рассмотрим два алгоритма вычисления свертки. Они оба работают непосредственно со значениями функций $F_1(x, y)$ и $F_2(x, y)$, не используя переход к частотам (чего можно достигнуть с помощью преобразований Фурье).

3.1. Сеточный алгоритм (СА)

Сеточный алгоритм заключается в прямом перемножении соответствующих значений функций $F_1(x, y)$ и $F_2(x, y)$. Последовательность действий можно описать следующим образом:

(1) для каждой ячейки x маски, такой, что x принадлежит области S, выясняется, находятся ли под ней ячейки сдвинутой и повернутой области, задающей $F_2(x, y)$;
(2) если да, то выбирается ближайшая к x ячейка $y \in F_2(x, y)$, их значения перемножаются и произведение добавляется в общую сумму; иначе — происходит переход к следующей ячейке $F_1(x, y)$;
(3) значением свертки будет являться значение общей суммы, деленной на число слагаемых в ней.

В результате работы алгоритма происходит ровно $M_1 \cdot M_2$ сравнений и не более $M_1 \cdot M_2$ умножений. Алгоритм не имеет регулировок и, согласно предварительным оценкам, работает практически с одной и той же точностью. Время работы алгоритма почти не зависит от того, как много ячеек функции $F_1(x, y)$ имеют хоть одну соответствующую ячейку функции $F_2(x, y)$ (то есть от размера области S, см. рис. 1).

Данные оценки должны быть подтверждены или опровергнуты последующими практическими экспериментами.

Возможна модификация алгоритма (СА-М), заключающаяся в том, что каждая ячейка в маске делится на n частей. Это позволяет достичь большей точности вычислений, так как одной ячейке маски может соответствовать сразу несколько ячеек $F_2(x, y)$. Очевидно, что с ростом n увеличивается точность алгоритма, но также растет и время его работы.

3.2. Метод Монте-Карло (МК)

Альтернативный алгоритм «МК» имеет следующую схему.

(1) случайным образом выбирается ячейка x, принадлежащая маске, такая, что x принадлежит области S; выясняется, находятся ли под ней ячейки сдвинутой и повернутой области, задающей $F_2(x, y)$;

(2) если да, то выбирается ближайшая к ячейке x ячейка $y \in F_2(x, y)$, их значения перемножаются и произведение добавляется в общую сумму; иначе — выбирается другая ячейка $x \in F_1(x, y)$;
(3) значением свертки будет являться значение общей суммы, деленной на число слагаемых в ней.

Работа алгоритма завершается при накоплении требуемого количества слагаемых m. Введение параметра m дает возможность управления качеством результата: чем больше m, тем больше время работы, но выше точность, и наоборот.

4. Проведение экспериментов

С целью оценки возможного ускорения работы алгоритмов были проведены эксперименты на однопроцессорной ЭВМ и КВУ.

4.1. Вычисления на однопроцессорной ЭВМ

Для экспериментов выбраны следующие данные:

- функция $F_1(x, y)$, заданная матрицей размером 64×128 либо 128×256;
- функция $F_2(x, y)$, заданная матрицей размером 2048×2048;
- параметры Δx, Δy и γ, которые подбирались таким образом, чтобы обеспечить необходимый размер области S (см. рис. 1);
- истинное значение свертки во всех случаях равно 0.5.

Исследовались следующие закономерности:

- влияние на время работы размера матрицы $F_1(x, y)$ (маски);
- влияние на время работы числа выбранных ячеек в $F_1(x, y)$, но не попавших в S (назовем это «площадью покрытия», C);
- влияние числа итераций на точность вычислений (для метода «МК»).

В каждой таблице приведена графа «Максимальное Отклонение», показывающая, как сильно ошибся каждый алгоритм при расчете свертки. Пусть R и T — соответственно истинное и полученное значения свертки, тогда отклонение вычисляется по формуле: $\frac{R-T}{R} \cdot 100$. Для каждого алгоритма из всех значений выбирается максимальное.

Эксперименты проводились на вычислительной системе с процессором AMD Athlon MP 1800+, управляемой ОС Linux (один процессор фронтенда skif.botik.ru). Замерялось «чистое» время свертки, из него исключалось время на инициализацию матриц.

4.1.1. *Сеточные методы*

В этом эксперименте в «СА-М» каждая ячейка делилась на две части.

Из таблиц 1 и 2 видно, что время работы линейно зависит от размера маски. Уменьшение «площади покрытия» снижает время работы, так как умножение не производится для ячеек маски, не имеющих соответствующих ячеек $F_2(x, y)$.

Условия	Время (сек)	Ответ
2048×2048, 64×128, C = 100%	0.0044	0.5017
2048×2048, 64×128, C = 50%	0.0036	0.4953
2048×2048, 64×128, C = 25%	0.0033	0.4990
2048×2048, 128×256, C = 100%	0.0172	0.4961
2048×2048, 128×256, C = 50%	0.0160	0.5023
2048×2048, 128×256, C = 25%	0.0147	0.5014
Максимальное Отклонение (%)	0.98	

Таблица 1. Результаты для «СА»

Условия	Время (сек)	Ответ
2048×2048, 64×128, C = 100%	0.0079	0.4997
2048×2048, 64×128, C = 50%	0.0070	0.5002
2048×2048, 64×128, C = 25%	0.0065	0.4998
2048×2048, 128×256, C = 100%	0.0323	0.5000
2048×2048, 128×256, C = 50%	0.0301	0.5001
2048×2048, 128×256, C = 25%	0.0285	0.4999
Максимальное Отклонение (%)	0.06	

Таблица 2. Результаты для «СА-М»

4.1.2. *Метод Монте-Карло*

Как показано в таблицах 3 и 4, с увеличением числа итераций точность вычислений растет вместе со временем работы. Также видно, что время растет почти линейно с уменьшением «площади покрытия». Влияние размера маски на скорость работы можно считать незначительным или вовсе отсутствующим.

Условия		Число итераций			
		500	5 K	50 K	1 M
2048×2048, 64×128,C = 100%	Время	0.0006	0.0049	0.0471	0.9312
	Ответ	0.4860	0.4926	0.4993	0.5003
2048×2048, 64×128, C = 50%		0.0009	0.0084	0.0784	1.5753
		0.5040	0.4940	0.4999	0.5000
2048×2048, 64×128, C = 25%		0.0016	0.0150	0.1458	2.9018
		0.5080	0.4934	0.4966	0.5001
Макс. Отклон. (%)		2.8	1.48	0.88	0.06

Таблица 3. Результаты для «МК» (1/2)

Условия		Число итераций			
		500	5 K	50 K	1 M
2048×2048, 128×256,C = 100%	Время	0.0007	0.0047	0.0504	0.9682
	Ответ	0.5060	0.4950	0.5029	0.5000
2048×2048, 128×256, C = 50%		0.0010	0.0089	0.0814	1.5973
		0.5260	0.4932	0.4970	0.5011
2048×2048, 128×256, C = 25%		0.0016	0.0155	0.1505	2.9188
		0.4900	0.4920	0.4971	0.5005
Макс. Отклон. (%)		5.2	1.6	0.6	0.22

Таблица 4. Результаты для «МК» (2/2)

4.1.3. *Сравнение алгоритмов*

Условия, в которых проводилось тестирование:

- функция $F_1(x, y)$, заданная матрицей размером 128×256;
- функция $F_2(x, y)$, заданная матрицей размером 2048×2048;

- параметры Δx, Δy и γ каждый раз выбирались произвольным образом (в некоторых разумных пределах);
- значения элементов матриц функций $F_1(x, y)$ и $F_2(x, y)$ были подобраны таким образом, чтобы значение свертки равнялось 4.5.

В алгоритме «СА-М» каждая ячейка делилась на две части. У методов «МК» в скобках указано число итераций.

Положение		Статистика			
		СА	СА-М	МК (15 К)	МК (1 М)
I	Время	0.0119	0.0210	0.0131	0.8657
	Ответ	4.3531	4.4512	4.4523	4.4951
II		0.0188	0.0354	0.0145	1.9607
		4.5000	4.4999	4.4826	4.5004
III		0.0166	0.0310	0.0185	1.2000
		4.4698	4.4707	4.4922	4.5030
Макс. Отклон. (%)		3.47	1.08	1.06	0.11

Таблица 5. Сравнение

Заметим, обе схемы «СА-М» и «МК» являются регулируемыми по скорости и точности работы. Однако «МК» отличается большей гибкостью, т.к. допускает более «плавную» настройку.

Рекомендации по выбору схем:

- если требуется высокая скорость, то предпочтение отдается «СА» или «МК» с малым числом итераций;
- если требуется точность, то выбирается «СА-М» с мелким разбиением ячеек, или «МК» с достаточно большим числом итераций.

Компромисс между скоростью и точностью достигается правильным подбором параметров в методах «СА-М» или «МК».

4.2. Вычисление на КВУ

Для реализации последовательных алгоритмов был использован язык C++, поэтому параллельная версия была создана на языке T++, который наилучшим образом подходит для наших целей [3].

4.2.1. *Общий подход к распараллеливанию*

В случае необходимости подсчета многих сверток, например, с разными значениями параметров Δx и Δy, достаточно эффективным способом достижения параллелизма будет разделение самих процессов свертки между узлами (параллелизм по данным).

График зависимости времени работы от числа процессоров показан на рис. 2. Таблица 6 отражает данные в числовом виде.

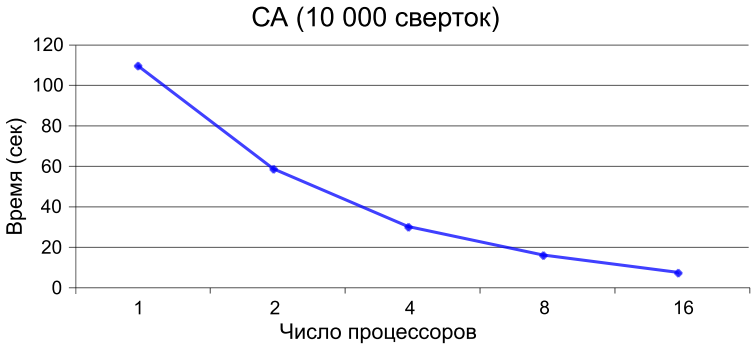

Рис. 2. Параллелизм при общем подходе. Результаты

Число проц. (N)	Время (сек) $t(N)$	Ускорение $c(N) = t(1)/t(N)$	КПД $c(N)/N$
1	109.6954	1	1
2	59.7163	1.836942342	0.918471171
4	31.5985	3.471538206	0.867884551
8	15.8577	6.917484881	0.86468561
16	8.8423	12.40575416	0.775359635

Таблица 6. Подробное представление времени работы

4.2.2. *Распараллеливание алгоритма Монте-Карло*

Для ускорения вычислений можно использовать параллелизм самого алгоритма «МК», но лишь при некоторых ограничениях. Требуется задание весьма большого числа итераций для обеспечения достаточного «веса» гранулы параллелизма.

Пусть P — заданное число итераций, Q — число узлов. Тогда описать схему можно так:

- каждому узлу поручается выполнить P/Q итераций, посчитав для них сумму произведений ячеек $F_1(x, y)$ и $F_2(x, y)$;
- значения всех сумм собираются со всех узлов, и высчитывается окончательный результат.

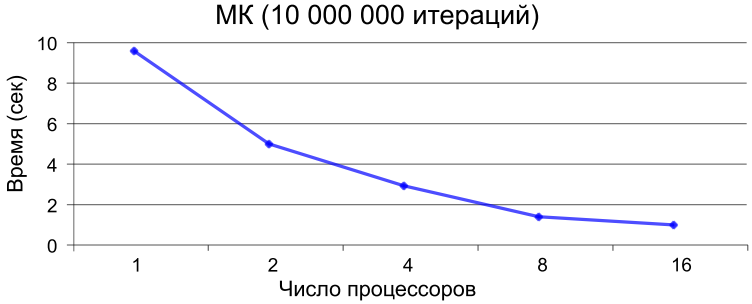

Рис. 3. Параллелизм «МК». Результаты

Число проц. (N)	Время (сек) $t(N)$	Ускорение $c(N) = t(1)/t(N)$	КПД $c(N)/N$
1	9.6165	1	1
2	4.9101	1.958514083	0.979257042
4	2.8641	3.357599246	0.839399811
8	1.5103	6.367278024	0.795909753
16	0.9317	10.3214554	0.645090963

Таблица 7. Подробное представление времени работы

На рис. 3 показана зависимость времени счета от числа узлов. В таблице 7 приведены подробные результаты по времени работы.

5. Заключение

Предложены, реализованы и изучены две схемы приближенного вычисления свертки. Показано, что схема на основе Монте-Карло позволяет регулировать точность и время за счет выбора числа итераций. Использование КВУ позволяет ускорить вычисления, причем

имеет место масштабирование времени с ростом числа процессоров. Результаты данной работы переданы во ФГУП «КБ Машиностроения» (г. Коломна) для проведения дальнейших испытаний на стендах.

Список литературы

[1] Нуссбаумер Г. Быстрое преобразование Фурье и алгоритмы вычисления сверток. — М.: Радио и связь, 1985. — 248 с.

[2] Миронов С., Дударев В., Богатов А. *Цифровая обработка радиолокационных сигналов на основе процессора Л1879ВМ1* // ЭЛЕКТРОНИКА: Наука, Технология, Бизнес. — **2003, № 3**, с. 66–72.

[3] *Т-Система* // http://wiki.botik.ru/OPENTS/.

M. D. Nedev. *About Approximate Convolution Calculation* // Proceedings of Prograanm Systems institute scientific-practical conference “Program systems: Theory and applications”, devoted to the 15^{th} anniversary of Pereslavl University named A. K. Ailamazyan. — Pereslavl-Zalesskij, 2008. — p. 195 — 204. — ISBN 978-5-901795-13-2 *(in Russian).*

Abstract. The goal of this paper is to implement and compare two methods for calculating convolution of discrete functions. Author also considers parallel implementations of a given algorithms. Keywords: approximate convolution, parallel algorithm.

Перевод проверен: д.т.н. проф. В. М. Хачумов