Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Нижегородский государственный университет им. Н.И. Лобачевского»

Институт информационных технологий, математики и механики

Отчет

по лабораторной работе

«Линейная фильтрация изображений (горизонтальное разбиение). Ядро Гаусса 3x3»

Выполнил:

студент группы 1606-1

Пономарев А.С.

Проверил:

к.т.н Сысоев А.В.

Нижний Новгород

2018

Оглавление

Введение

Обычно изображения, сформированные различными информационными системами, искажаются действием помех. Это затрудняет как их визуальный анализ человеком-оператором, так и автоматическую обработку в ЭВМ. При решении некоторых задач обработки изображений в роли помех могут выступать и те или иные компоненты самого изображения. Например, при анализе космического снимка земной поверхности может стоять задача определения границ между её отдельными участками - лесом и полем, водой и сушей и т.п. С точки зрения этой задачи отдельные детали изображения внутри разделяемых областей являются помехой.

Для того, чтобы убрать различного рода помехи используют фильтрацию. Существует множество методов для решения данной задачи, но в данной работе автор останавливается на одной группе, а именно линейные матричные фильтры, и на одном её члене, а именно линейном фильтре Гаусса.

Описание алгоритма

Линейные фильтры представляют собой семейство фильтров, имеющих очень простое математическое описание. Вместе с тем они позволяют добиться самых разнообразных эффектов. Будем считать, что задано исходное полутоновое изображение A, и обозначим интенсивности его пикселей A(x, y). Линейный фильтр определяется веществennозначной функцией F, заданной на растре. Данная функция называется ядром фильтра, а сама фильтрация производится при помощи операции дискретной свертки (взвешенного суммирования)

$$B(x,y) = \sum_i \sum_j F(i,j) \cdot A(x+i, y+j). \qquad (1)$$

Результатом служит изображение B. В определении (1) мы опустили пределы суммирования. Обычно ядро фильтра отлично от нуля только в некоторой окрестности N точки (0, 0). За пределами этой окрестности F(i, j) или в точности равно нулю, или очень близко к нему, так что можно им пренебречь. Суммирование в (1) производится по $(i,j) \in N$, и значение каждого пикселя B(x, y) определяется пикселями изображения A, которые лежат в окне N, центрированном в точке (x, y) (мы будем обозначать это множество N(x, y)). Ядро фильтра, заданное на прямоугольной окрестности N, может рассматриваться как матрица m на n, где длины сторон являются нечетными числами. При задании ядра матрицей M_{kl}, ее следует центрировать:

$$F(i,j) = M_{i+\frac{m-1}{2} j+\frac{n-1}{2}} \qquad (2)$$

Также нуждается в дополнительном прояснении ситуация, когда пиксель (x, y) находится в окрестности краев изображения. В этом случае A(x + i, y + j) в определении (1) может соответствовать пикселю A, лежащему за пределами изображения A. Данную проблему можно разрешить несколькими способами.

- Не проводить фильтрацию для таких пикселей, обрезав изображение B по краям или закрасив их, к примеру, черным цветом.
- Не включать соответствующий пиксель в суммирование, распределив его вес F(i, j) равномерно среди других пикселей окрестности N(x, y).
- Доопределить значения пикселей за границами изображения при помощи экстраполяции. Например, считать постоянным значение интенсивности вблизи границы (для пикселя (-2, 5) имеем A(-2, 5) = A(0, 5)) или считать постоянным градиент интенсивности вблизи границы (A(-2, 5) = A(0, 5) + 2(A(0, 5) - A(1, 5))).
- Доопределить значения пикселей за границами изображения, при помощи зеркального отражения (A(-2, 5) = A(2, 5)).

В данной работе выбор пал на первый вариант.

Для чего могут применяться сглаживающие фильтры? Одним из их возможных применений является шумоподавление, т.е. задача восстановления исходного изображения, к пикселям которого добавлен случайный шум. Естественным

предположением об исходном незашумленном изображении будет схожесть значений интенсивности пикселей, находящихся рядом. Причем чем меньше расстояние между пикселями, тем больше вероятность их похожести. Это и отличает исходное незашумленное изображение от шумовой компоненты, для которой схожесть пикселей никак не зависит от расстояния между ними. Эффективное шумоподавление можно, таким образом, осуществить, если влияние пикселей друг на друга будет уменьшаться с расстоянием. Этим свойством обладает **гауссовский** фильтр с ядром:

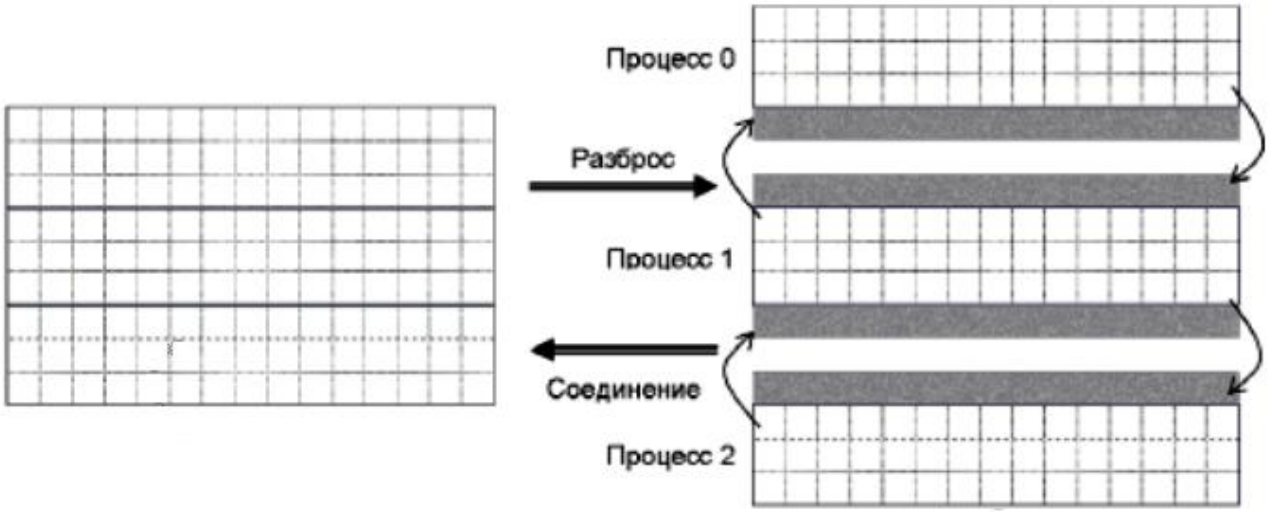
$$F_{gauss}(i,j) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{i^2+j^2}{2\sigma^2}\right). \qquad (8.3)$$

Гауссовский фильтр имеет ненулевое ядро бесконечного размера. Однако ядро фильтра очень быстро убывает к нулю при удалении от точки (0, 0), и потому на практике можно ограничиться сверткой с окном небольшого размера вокруг (0, 0) (например, взяв радиус окна равным 3σ).

Гауссовская фильтрация также является сглаживающей. Однако, в отличие от прямоугольного фильтра, образом точки при гауссовой фильтрации будет симметричное размытое пятно, с убыванием яркости от середины к краям, что гораздо ближе к реальному размытию от расфокусированных линз. Как и следовало ожидать, гауссовская фильтрация более эффективна при шумоподавлении: влияние пикселей друг на друга при гауссовой фильтрации обратно пропорционально квадрату расстояния между ними. Как видно из (3), коэффициент пропорциональности, а следовательно, и степень размытия, определяются параметром σ.

Метод распараллеливания

По условию задачи необходимо выполнить горизонтальное разбиение на блоки. Это самое естественное разбиение, поскольку данные в массиве обычно хранятся по строкам. Серые зоны содержат информацию о границах. Этой информацией обмениваются соседние проце



Особенности программной реализации

Последовательная версия

Программная реализация алгоритма расположена в функции *Pixel** seqFilter(Pixel** genImage, int width, int height, double** kernel, int kerRadius),* на вход которой приходит исходное изображение и его размеры, а также ядро Гаусса и его радиус. На выходе - преобразованная копия исходного изображения. Под преобразованием понимается применённая к каждому пикселю операция свёртки с ядром Гаусса.

OpenMP

Параллельная реализации расположена в функции *parFilter* , на вход которой поступают те же параметры, и дополнительно целочисленный параметр - количество потоков.

Распараллеливание происходит с помощью директивы препроцессора.

```
#pragma omp parallel for schedule(dynamic)
```

Также перед этим устанавливается количество потоков

```
omp_set_num_threads(threadsNumber)
```

TBB

Обработка происходит в функции с такой же сигнатурой и именем. Но вызов происходит с помощью функции из библиотеки tbb:

```
tbb::parallel_for(tbb::blocked_range<int>(0, height, grainsize), filter)
```

Параметр filter - это реализованный в отдельном классе функтор, с имплементацией метода `void operator() (const tbb::blocked_range<int>& range) const`

Результаты работы

Полные результаты работы предоставлены ниже(см. Приложения).

Измерения производились на ПК со следующими характеристиками:

CPU - Intel Core i7-3770, 3.40 GHz, 4 Cores, 8 Logical processors, Caches(256KB, 1.0MB, 8.0MB)

RAM - 8 GB, 1600 MHz

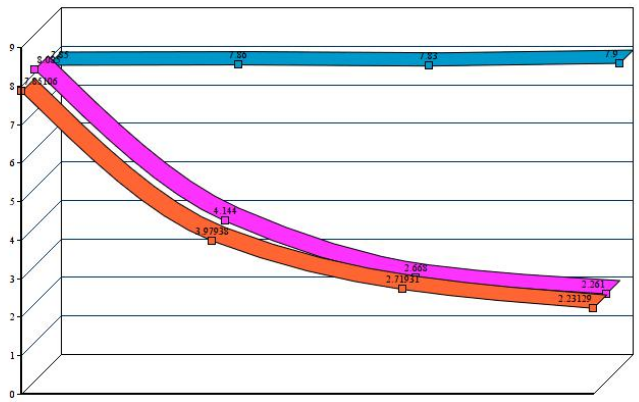
OS - Windows 10 x64

Одновременно были включены некоторые посторонние приложения(поэтому погрешность может возрасти):

Word, Chrome, Visual Studio.

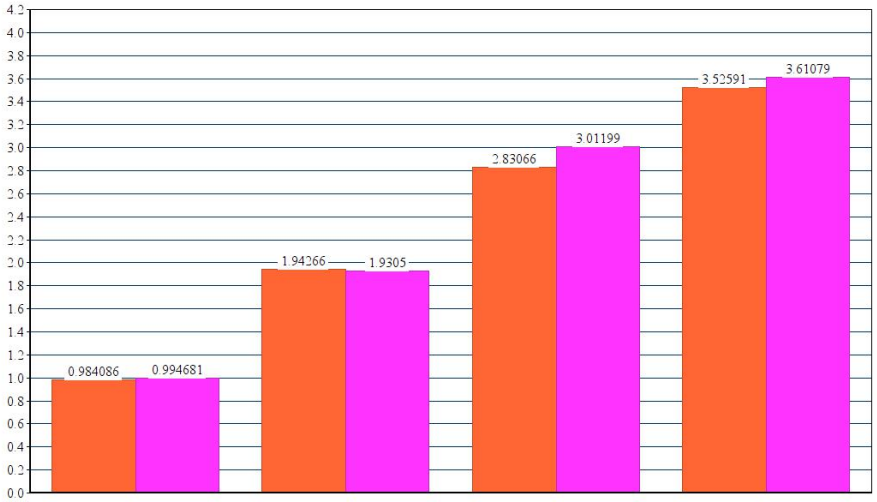
Сравнение производится на автоматически сгенерированном изображении размера 6400 x 3600.

Количество потоков	Время OMP	Время TBB	Время последовательного выполнения
1	7.85106	8.085	7.72612; 8.042
2	3.97938	4.144	7.7306; 8
4	2.71931	2.668	7.69743; 8.036
8	2.23129	2.261	7.86733; 8.164

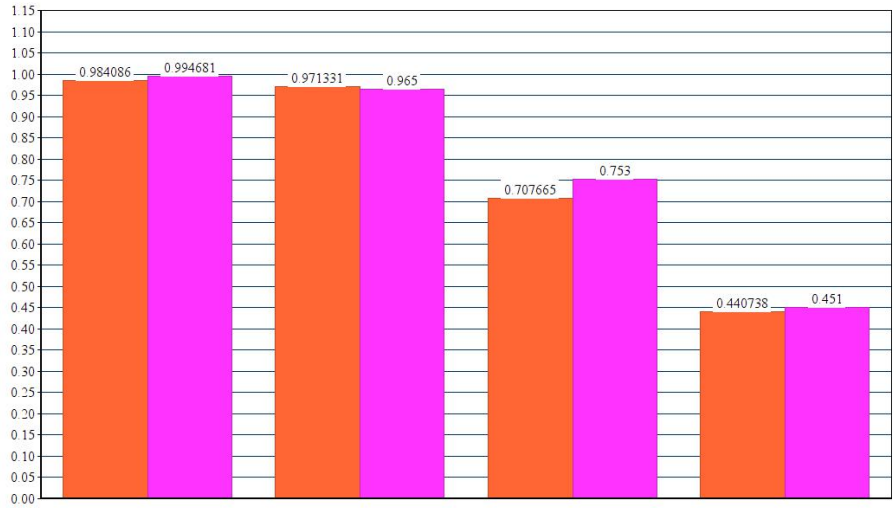


На этом графике визуально отображены данные из таблицы временных результатов. Синим цветом обозначен график времени последовательного выполнения, розовым - TBB, оранжевым - OMP. На нижней оси - количество потоков. Точки отмечены при количестве потоков = 1, 2, 4, 8.

Количество потоков	Boost OMP	Effectivity OMP	Boost TBB	Effectivity TBB
1	0.984086	0.984086	0.994681	0.994681
2	1.94266	0.971331	1.9305	0.965
4	2.83066	0.707665	3.01199	0.753
8	3.52591	0.440738	3.61079	0.451



На этой диаграмме визуально отображено сравнение ускорений. Оранжевый - OMP, Розовый - TBB. На горизонтальной оси - количество потоков 1, 2, 4, 8.



Данная диаграмма показывает различие в эффективности. Цвета соотносятся также как и на предыдущем изображении.

Как можно заметить по данным результатам, различия в скорости работы, ускорениях и эффективности незначительны. Причинами неполного совпадения результатов могут быть, помимо некоторых отличий в архитектуре данных систем, также влияние сторонних приложений и программ в ОС, как системных, так и пользовательских, и не одинаковое выделение времени процессором.

Заключение

В течение выполнения данной работы, автор познакомился с технологиями параллельного программирования с использованием библиотек TBB и OpenMP, реализовал алгоритм линейной фильтрации ядром Гаусса, и выполнил сравнение производительности средств вышеуказанных библиотек для решения данной задачи.

Литература

1. Матричные фильтры обработки изображений https://habr.com/post/142818
2. Лекция по матричным фильтрам (Компьютерная графика) http://www.apmath.spbu.ru/ru/staff/pogozhev/lection07.pdf
3. Введение в OpenMP https://software.intel.com/ru-ru/blogs/2011/11/21/openmp-c
4. TBB parallel for https://software.intel.com/en-us/node/506153
5. Лекции на ИНТУИТ https://www.intuit.ru/studies/courses/993/163/lecture/4505.
6. Фильтрация изображения. Неизвестный автор. http://dsp-book.narod.ru/dspimage/chapter3.pdf

Приложения

Исходный код программы

Репозиторий с исходным кодом программы находится по адресу:
https://github.com/panzer2344/parallel_programming_course

Полные результаты работы

OMP

ImWidth = 800 ImHeitht = 450 ProcCount = 1
Elapsed time for seq version = 0.125879
Elapsed time for par version = 0.140915
Boost = 0.8933
Effectivity = 0.8933

ImWidth = 1600 ImHeitht = 900 ProcCount = 1
Elapsed time for seq version = 0.505865
Elapsed time for par version = 0.495499
Boost = 1.02092
Effectivity = 1.02092

ImWidth = 3200 ImHeitht = 1800 ProcCount = 1
Elapsed time for seq version = 1.95263
Elapsed time for par version = 1.98145
Boost = 0.985456
Effectivity = 0.985456

ImWidth = 6400 ImHeitht = 3600 ProcCount = 1
Elapsed time for seq version = 7.72612
Elapsed time for par version = 7.85106

Boost = 0.984086

Effectivity = 0.984086

ImWidth = 12800 ImHeitht = 7200 ProcCount = 1

Elapsed time for seq version = 30.9905

Elapsed time for par version = 31.0182

Boost = 0.999107

Effectivity = 0.999107

ImWidth = 800 ImHeitht = 450 ProcCount = 2

Elapsed time for seq version = 0.129901

Elapsed time for par version = 0.0628303

Boost = 2.06749

Effectivity = 1.03375

ImWidth = 1600 ImHeitht = 900 ProcCount = 2

Elapsed time for seq version = 0.475562

Elapsed time for par version = 0.245165

Boost = 1.93976

Effectivity = 0.96988

ImWidth = 3200 ImHeitht = 1800 ProcCount = 2

Elapsed time for seq version = 1.92351

Elapsed time for par version = 0.995579

Boost = 1.93205

Effectivity = 0.966027

ImWidth = 6400 ImHeitht = 3600 ProcCount = 2

Elapsed time for seq version = 7.7306

Elapsed time for par version = 3.97938

Boost = 1.94266

Effectivity = 0.971331

ImWidth = 12800 ImHeitht = 7200 ProcCount = 2

Elapsed time for seq version = 30.8647

Elapsed time for par version = 16.8799

Boost = 1.82849

Effectivity = 0.914244

ImWidth = 800 ImHeitht = 450 ProcCount = 3

Elapsed time for seq version = 0.124939

Elapsed time for par version = 0.0727392

Boost = 1.71763

Effectivity = 0.572543

ImWidth = 1600 ImHeitht = 900 ProcCount = 3

Elapsed time for seq version = 0.476251

Elapsed time for par version = 0.191702

Boost = 2.48433

Effectivity = 0.828109

ImWidth = 3200 ImHeitht = 1800 ProcCount = 3

Elapsed time for seq version = 1.95145

Elapsed time for par version = 0.819619

Boost = 2.38093

Effectivity = 0.793643

ImWidth = 6400 ImHeitht = 3600 ProcCount = 3

Elapsed time for seq version = 7.72296

Elapsed time for par version = 2.97221

Boost = 2.59839

Effectivity = 0.866129

ImWidth = 12800 ImHeitht = 7200 ProcCount = 3

Elapsed time for seq version = 31.2525

Elapsed time for par version = 12.749

Boost = 2.45136

Effectivity = 0.81712

ImWidth = 800 ImHeitht = 450 ProcCount = 4
Elapsed time for seq version = 0.161102
Elapsed time for par version = 0.0530957
Boost = 3.03418
Effectivity = 0.758545

ImWidth = 1600 ImHeitht = 900 ProcCount = 4
Elapsed time for seq version = 0.496083
Elapsed time for par version = 0.186914
Boost = 2.65408
Effectivity = 0.663519

ImWidth = 3200 ImHeitht = 1800 ProcCount = 4
Elapsed time for seq version = 1.9436
Elapsed time for par version = 0.684461
Boost = 2.83961
Effectivity = 0.709901

ImWidth = 6400 ImHeitht = 3600 ProcCount = 4
Elapsed time for seq version = 7.69743
Elapsed time for par version = 2.71931
Boost = 2.83066
Effectivity = 0.707665

ImWidth = 12800 ImHeitht = 7200 ProcCount = 4
Elapsed time for seq version = 30.9909
Elapsed time for par version = 12.2028
Boost = 2.53966
Effectivity = 0.634915

ImWidth = 800 ImHeitht = 450 ProcCount = 5

Elapsed time for seq version = 0.118967

Elapsed time for par version = 0.556925

Boost = 0.213614

Effectivity = 0.0427228

ImWidth = 1600 ImHeitht = 900 ProcCount = 5

Elapsed time for seq version = 0.483088

Elapsed time for par version = 0.163267

Boost = 2.95888

Effectivity = 0.591777

ImWidth = 3200 ImHeitht = 1800 ProcCount = 5

Elapsed time for seq version = 1.97599

Elapsed time for par version = 0.645662

Boost = 3.06041

Effectivity = 0.612081

ImWidth = 6400 ImHeitht = 3600 ProcCount = 5

Elapsed time for seq version = 7.74494

Elapsed time for par version = 2.47596

Boost = 3.12806

Effectivity = 0.625612

ImWidth = 12800 ImHeitht = 7200 ProcCount = 5

Elapsed time for seq version = 31.0225

Elapsed time for par version = 11.0814

Boost = 2.79951

Effectivity = 0.559902

ImWidth = 800 ImHeitht = 450 ProcCount = 6

Elapsed time for seq version = 0.118175

Elapsed time for par version = 0.0677673

Boost = 1.74384

Effectivity = 0.29064

ImWidth = 1600 ImHeitht = 900 ProcCount = 6

Elapsed time for seq version = 0.478204

Elapsed time for par version = 0.166597

Boost = 2.87042

Effectivity = 0.478403

ImWidth = 3200 ImHeitht = 1800 ProcCount = 6

Elapsed time for seq version = 1.95832

Elapsed time for par version = 0.599502

Boost = 3.26658

Effectivity = 0.544429

ImWidth = 6400 ImHeitht = 3600 ProcCount = 6

Elapsed time for seq version = 7.80743

Elapsed time for par version = 2.24762

Boost = 3.47365

Effectivity = 0.578941

ImWidth = 12800 ImHeitht = 7200 ProcCount = 6

Elapsed time for seq version = 31.0667

Elapsed time for par version = 10.8641

Boost = 2.85956

Effectivity = 0.476594

ImWidth = 800 ImHeitht = 450 ProcCount = 7

Elapsed time for seq version = 0.116873

Elapsed time for par version = 0.237093

Boost = 0.492943

Effectivity = 0.0704205

ImWidth = 1600 ImHeitht = 900 ProcCount = 7

Elapsed time for seq version = 0.480787

Elapsed time for par version = 0.140339

Boost = 3.42591

Effectivity = 0.489415

ImWidth = 3200 ImHeitht = 1800 ProcCount = 7

Elapsed time for seq version = 1.97333

Elapsed time for par version = 0.618629

Boost = 3.18984

Effectivity = 0.455691

ImWidth = 6400 ImHeitht = 3600 ProcCount = 7

Elapsed time for seq version = 7.86445

Elapsed time for par version = 2.18257

Boost = 3.60331

Effectivity = 0.514758

ImWidth = 12800 ImHeitht = 7200 ProcCount = 7

Elapsed time for seq version = 31.082

Elapsed time for par version = 11.5386

Boost = 2.69374

Effectivity = 0.38482

ImWidth = 800 ImHeitht = 450 ProcCount = 8

Elapsed time for seq version = 0.137604

Elapsed time for par version = 0.43205

Boost = 0.318492

Effectivity = 0.0398115

ImWidth = 1600 ImHeitht = 900 ProcCount = 8

Elapsed time for seq version = 0.484463

Elapsed time for par version = 0.13491

Boost = 3.59102

Effectivity = 0.448877

ImWidth = 3200 ImHeitht = 1800 ProcCount = 8

Elapsed time for seq version = 2.03051

Elapsed time for par version = 0.5793

Boost = 3.5051

Effectivity = 0.438138

ImWidth = 6400 ImHeitht = 3600 ProcCount = 8

Elapsed time for seq version = 7.86733

Elapsed time for par version = 2.23129

Boost = 3.52591

Effectivity = 0.440738

ImWidth = 12800 ImHeitht = 7200 ProcCount = 8

Elapsed time for seq version = 31.1483

Elapsed time for par version = 8.89769

Boost = 3.50072

Effectivity = 0.43759

TBB

ImWidth = 800 ImHeight = 450 procCount = 1

Elapsed time for seq version = 0.13

Grainsize: 450

Elapsed time for parallel version = 0.14

Boost = 0.928571

Effectivity = 0.116071

ImWidth = 1600 ImHeight = 900 procCount = 1

Elapsed time for seq version = 0.548

Grainsize: 900

Elapsed time for parallel version = 0.503

Boost = 1.08946

Effectivity = 0.136183

ImWidth = 3200 ImHeight = 1800 procCount = 1

Elapsed time for seq version = 2.027

Grainsize: 1800

Elapsed time for parallel version = 2.014

Boost = 1.00645

Effectivity = 0.125807

ImWidth = 6400 ImHeight = 3600 procCount = 1

Elapsed time for seq version = 8.042

Grainsize: 3600

Elapsed time for parallel version = 8.085

Boost = 0.994681

Effectivity = 0.124335

ImWidth = 12800 ImHeight = 7200 procCount = 1

Elapsed time for seq version = 32.301

Grainsize: 7200

Elapsed time for parallel version = 32.326

Boost = 0.999227

Effectivity = 0.124903

ImWidth = 800 ImHeight = 450 procCount = 2

Elapsed time for seq version = 0.125

Grainsize: 225

Elapsed time for parallel version = 0.127

Boost = 0.984252

Effectivity = 0.123031

ImWidth = 1600 ImHeight = 900 procCount = 2

Elapsed time for seq version = 0.503

Grainsize: 450

Elapsed time for parallel version = 0.268

Boost = 1.87687

Effectivity = 0.234608

ImWidth = 3200 ImHeight = 1800 procCount = 2

Elapsed time for seq version = 2.022

Grainsize: 900

Elapsed time for parallel version = 1.028

Boost = 1.96693

Effectivity = 0.245866

ImWidth = 6400 ImHeight = 3600 procCount = 2

Elapsed time for seq version = 8

Grainsize: 1800

Elapsed time for parallel version = 4.144

Boost = 1.9305

Effectivity = 0.241313

ImWidth = 12800 ImHeight = 7200 procCount = 2

Elapsed time for seq version = 32.286

Grainsize: 3600

Elapsed time for parallel version = 17.538

Boost = 1.84092

Effectivity = 0.230115

ImWidth = 800 ImHeight = 450 procCount = 3

Elapsed time for seq version = 0.156

Grainsize: 150

Elapsed time for parallel version = 0.043

Boost = 3.62791

Effectivity = 0.453488

ImWidth = 1600 ImHeight = 900 procCount = 3

Elapsed time for seq version = 0.503

Grainsize: 300

Elapsed time for parallel version = 0.172

Boost = 2.92442

Effectivity = 0.365552

ImWidth = 3200 ImHeight = 1800 procCount = 3

Elapsed time for seq version = 2.009

Grainsize: 600

Elapsed time for parallel version = 0.708

Boost = 2.83757

Effectivity = 0.354696

ImWidth = 6400 ImHeight = 3600 procCount = 3

Elapsed time for seq version = 7.971

Grainsize: 1200

Elapsed time for parallel version = 2.626

Boost = 3.03542

Effectivity = 0.379427

ImWidth = 12800 ImHeight = 7200 procCount = 3

Elapsed time for seq version = 32.454

Grainsize: 2400

Elapsed time for parallel version = 12.353

Boost = 2.62722

Effectivity = 0.328402

ImWidth = 800 ImHeight = 450 procCount = 4

Elapsed time for seq version = 0.124

Grainsize: 112

Elapsed time for parallel version = 0.063

Boost = 1.96825

Effectivity = 0.246032

ImWidth = 1600 ImHeight = 900 procCount = 4

Elapsed time for seq version = 0.497

Grainsize: 225

Elapsed time for parallel version = 0.161

Boost = 3.08696

Effectivity = 0.38587

ImWidth = 3200 ImHeight = 1800 procCount = 4

Elapsed time for seq version = 2.02

Grainsize: 450

Elapsed time for parallel version = 0.69

Boost = 2.92754

Effectivity = 0.365942

ImWidth = 6400 ImHeight = 3600 procCount = 4

Elapsed time for seq version = 8.036

Grainsize: 900

Elapsed time for parallel version = 2.668

Boost = 3.01199

Effectivity = 0.376499

ImWidth = 12800 ImHeight = 7200 procCount = 4

Elapsed time for seq version = 32.319

Grainsize: 1800

Elapsed time for parallel version = 12.36

Boost = 2.61481

Effectivity = 0.326851

ImWidth = 800 ImHeight = 450 procCount = 5

Elapsed time for seq version = 0.177

Grainsize: 90

Elapsed time for parallel version = 0.108

Boost = 1.63889

Effectivity = 0.204861

ImWidth = 1600 ImHeight = 900 procCount = 5

Elapsed time for seq version = 0.494

Grainsize: 180

Elapsed time for parallel version = 0.134

Boost = 3.68657

Effectivity = 0.460821

ImWidth = 3200 ImHeight = 1800 procCount = 5

Elapsed time for seq version = 2.024

Grainsize: 360

Elapsed time for parallel version = 0.591

Boost = 3.4247

Effectivity = 0.428088

ImWidth = 6400 ImHeight = 3600 procCount = 5

Elapsed time for seq version = 8.196

Grainsize: 720

Elapsed time for parallel version = 2.188

Boost = 3.74589

Effectivity = 0.468236

ImWidth = 12800 ImHeight = 7200 procCount = 5

Elapsed time for seq version = 32.177

Grainsize: 1440

Elapsed time for parallel version = 10.772

Boost = 2.9871

Effectivity = 0.373387

ImWidth = 800 ImHeight = 450 procCount = 6

Elapsed time for seq version = 0.123

Grainsize: 75

Elapsed time for parallel version = 0.046

Boost = 2.67391

Effectivity = 0.334239

ImWidth = 1600 ImHeight = 900 procCount = 6

Elapsed time for seq version = 0.501

Grainsize: 150

Elapsed time for parallel version = 0.659

Boost = 0.760243

Effectivity = 0.0950304

ImWidth = 3200 ImHeight = 1800 procCount = 6

Elapsed time for seq version = 2.036

Grainsize: 300

Elapsed time for parallel version = 0.583

Boost = 3.49228

Effectivity = 0.436535

ImWidth = 6400 ImHeight = 3600 procCount = 6

Elapsed time for seq version = 8.214

Grainsize: 600

Elapsed time for parallel version = 2.163

Boost = 3.7975

Effectivity = 0.474688

ImWidth = 12800 ImHeight = 7200 procCount = 6

Elapsed time for seq version = 32.587

Grainsize: 1200

Elapsed time for parallel version = 9.176

Boost = 3.55133

Effectivity = 0.443916

ImWidth = 800 ImHeight = 450 procCount = 7

Elapsed time for seq version = 0.123

Grainsize: 64

Elapsed time for parallel version = 0.04

Boost = 3.075

Effectivity = 0.384375

ImWidth = 1600 ImHeight = 900 procCount = 7

Elapsed time for seq version = 0.505

Grainsize: 128

Elapsed time for parallel version = 0.142

Boost = 3.55634

Effectivity = 0.444542

ImWidth = 3200 ImHeight = 1800 procCount = 7

Elapsed time for seq version = 2.043

Grainsize: 257

Elapsed time for parallel version = 0.591

Boost = 3.45685

Effectivity = 0.432107

ImWidth = 6400 ImHeight = 3600 procCount = 7

Elapsed time for seq version = 8.115

Grainsize: 514

Elapsed time for parallel version = 2.251

Boost = 3.60506

Effectivity = 0.450633

ImWidth = 12800 ImHeight = 7200 procCount = 7

Elapsed time for seq version = 32.276

Grainsize: 1028

Elapsed time for parallel version = 10.325

Boost = 3.126

Effectivity = 0.390751

ImWidth = 800 ImHeight = 450 procCount = 8

Elapsed time for seq version = 0.123

Grainsize: 56

Elapsed time for parallel version = 0.036

Boost = 3.41667

Effectivity = 0.427083

ImWidth = 1600 ImHeight = 900 procCount = 8

Elapsed time for seq version = 0.511

Grainsize: 112

Elapsed time for parallel version = 0.147

Boost = 3.47619

Effectivity = 0.434524

ImWidth = 3200 ImHeight = 1800 procCount = 8

Elapsed time for seq version = 2.079

Grainsize: 225

Elapsed time for parallel version = 0.6

Boost = 3.465

Effectivity = 0.433125

ImWidth = 6400 ImHeight = 3600 procCount = 8

Elapsed time for seq version = 8.164

Grainsize: 450

Elapsed time for parallel version = 2.261

Boost = 3.61079

Effectivity = 0.451349

ImWidth = 12800 ImHeight = 7200 procCount = 8

Elapsed time for seq version = 32.196

Grainsize: 900

Elapsed time for parallel version = 9.386

Boost = 3.43022

Effectivity = 0.428777