

## Оценка зависимостей времени работы алгоритма для восстановления расфокусированных изображений, выполняемого на CPU и GPU

*А.Ю. Кулакович, Е.Ю. Баранов*

*Донской государственной технической университет, Ростов-на-Дону*

**Аннотация:** Предметом исследования является проблема выбора наиболее эффективной аппаратной архитектуры, реализующей алгоритм деконволюции (восстановления) искаженных изображений. В качестве рассматриваемого алгоритма деконволюции взят фильтр Винера, ввиду своей эффективности как с точки зрения качества восстановления изображений, так и по причине приемлемой временной сложности. Объект исследования – процесс определения временной сложности рассматриваемого алгоритма восстановления поврежденных изображений, при его выполнении на центральном процессоре (central processing unit, CPU) и графическом процессоре (graphics processing unit, GPU). Рассмотрены основные функции смаза и дефокусировки изображений: размытие по Гауссу, эффект Боке, Motion blur или размытие в движении. Описаны результаты вычислений, проведенных с целью сравнения временной зависимости работы фильтра Винера, выполняемого на центральном процессоре и графическом процессоре, от размера восстанавливаемого изображения. Установлено, что при использовании изображений, размеры которых не превышают размеры 1920\*1080 фильтр Винера целесообразнее реализовывать на CPU, а при восстановлении изображений, размеры которых превышают 1920\*1080 - на GPU. Данный результат актуален для изображений, искаженных при помощи всех трех рассмотренных функций смаза и дефокусировки. Полученные оценки характерны для компьютера ASUS N56 с характеристиками Intel® Core™ i7-3630QM, CPU @2.40 GHz, ОЗУ 8 ГБ, 64 – разрядная ОС, под управлением ОС Windows 8.

**Ключевые слова:** деконволюция изображений, фильтр Винера, распараллеливание, восстановление расфокусированных изображений, CPU и GPU, оптимизация алгоритма.

### Введение

Деконволюция, или восстановление расфокусированных изображений является актуальной научной-практической проблемой, по причине широкого применения графических данных при построении математических и информационных моделей. К основным факторам, обуславливающим проблему дефокусировки изображений, можно отнести: неправильную наводку фокуса на фотоаппарате, движение камеры или объекта перед камерой, искусственное размытие изображений и др. [1,2].

В настоящее время усложняются процессы управления доступными вычислительными ресурсами. Одним из новых подходов к распределению вычислительных ресурсов является применение GPU для решения

оптимизационных задач. Активное использование ресурсов GPU для решения классических оптимизационных задач требует не только исследования особенностей использования алгоритмов, ориентированных на GPU, но и сравнение эффективности использования алгоритмов, разработанных для GPU и CPU [1,2].

### **Постановка задачи**

На скорость деконволюции влияют не только выбранный алгоритм и размер искаженного изображения, но и аппаратная архитектура, на которой осуществляется процесс восстановления. Фильтр Винера в настоящее время является наиболее известным и часто используемым средством для восстановления изображений, по этой причине в данной работе, вкуче со своими достоинствами и недостатками, он рассматривается как некоторое эталонное средство для деконволюции [1,2]. Таким образом, актуальной задачей является получение экспериментальных оценок временной сложности алгоритма восстановления расфокусированных изображений на основе фильтра Винера. В рассматриваемом случае оценка временной сложности подразумевает определение зависимостей времени исполнения алгоритма на CPU и GPU от размерностей восстанавливаемых изображений. На основе полученных экспериментальных данных необходимо, по возможности, сделать вывод о наиболее подходящем процессоре – CPU или GPU, обеспечивающем скорейшее выполнение алгоритма деконволюции изображения указанной размерности. В представленной работе минимальная размерность восстанавливаемого изображения  $512 \times 512$ , а максимальная -  $1920 \times 1080$ . Данный диапазон размерностей был выбран по причине наиболее частого использования.

### **Краткий обзор проблемы потери качества изображения**

Построение модели искажения состоит в применении функции искажения к исходному изображению, называется convolution (русс. свёртка).

---

Определенная область первичного, неискаженного изображения сворачивается в один пиксель искаженного изображения по определенному закону [1-3]:

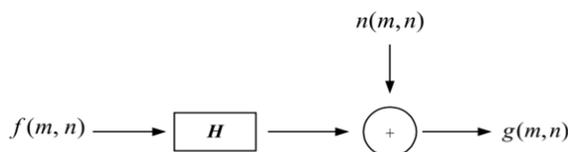


Рис.1 - Процесс искажения изображения

$f(m, n)$  – исходное, неискаженное изображение,  $H$  – искажающая функция,  $n(m, n)$  - аддитивный шум,  $g(m, n)$  - искаженное изображением.

Под ядром искажения понимают функцию искажения  $h(x, y)$ . Аддитивный шум - определенный тип мешающего воздействия в канале передачи информации.

**Gaussian blur** или **размытие по Гауссу** описывается формулой [1-4]:

$$y(m, n) = \frac{1}{2\pi r^2} \exp\left(-\frac{u^2 + v^2}{r^2}\right) x(m + u, n + v) \quad (2)$$

где  $y(m, n)$  – искаженное изображение,  $r$  – параметр размытия Гаусса,  $n$  – размеры исходного и неискаженного изображения.

**Out-of-focus blur** или **эффект Боке** описывается следующей формулой [1-4]:

$$h(m, n) = \begin{cases} \frac{1}{\pi R^2}, & \text{при } \sqrt{m^2 + n^2} \leq R \\ 0, & \text{в другом случае} \end{cases} \quad (3)$$

где  $h(m, n)$  – размытое изображение,  $R$  – радиус размытия,  $m$  и  $n$  – размеры исходного и неискаженного изображения.

**Motion blur** описывается следующей формулой [1-4]:

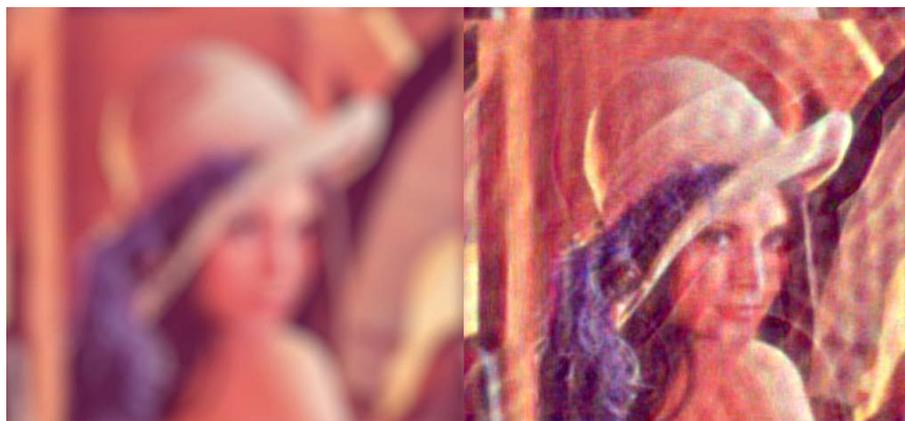
$$h(m) = \begin{cases} \frac{1}{L}, & \text{при } -\frac{L}{2} \leq m \leq \frac{L}{2} \\ 0, & \text{в другом случае} \end{cases} \quad (4)$$

где  $h(m)$  – смазанное изображение,  $L$  – длина следа размытия,  $m$  – длина смазанного изображения.

Формула фильтра Винера приведена ниже [2-5]:

$$\hat{F}(u, v) = \left( \frac{1}{H(u, v) \left[ |H(u, v)|^2 + \frac{S_n(u, v)}{S_f(u, v)} \right]} \right) G(u, v) \quad (5)$$

где  $\hat{F}(u, v)$  – восстановленное изображение,  $H(u, v)$  –искажающая функция,  $G(u, v)$  –искаженное изображение,  $S_n / S_f$ –заменяется константой  $k = 0,00000000001$ .



а) Смазанное изображение      б) Восстановленное изображение

Рис. 2 - Результаты работы программного обеспечения для реального изображения на основе фильтрации Винера, out of focus blur

### **Определение наиболее подходящей платформы для восстановления расфокусированных изображений.**

Определим зависимости времени работы алгоритма, реализованных для CPU и GPU, от размерности восстанавливаемого изображения.

Для реализации алгоритма фильтра Винера на GPU использовалась библиотека «Cudafy».

Библиотека «Cudafy» была выбрана по причине открытости кода, простоты в использовании, поддержка нескольких графических процессоров. Принцип работы библиотеки «Cudafy» состоит в следующем: вначале она преобразует код ядра C# в cuda во время выполнения (с возможностью

сериализации и кеширования), на следующем этапе она может легко запустить один и тот же код ядра на CPU (в основном для отладки).

Разработанное ПО позволяет загружать фотографии формата jpeg, png, jpg и размерности до 1920\*1080. Вычислительный эксперимент производился на компьютере ASUSN56(Inte® Core™ i7-3630QM, CPU @2.40 GHz, ОЗУ 8 ГБ, 64 – разрядная ОС), под управлением ОС Windows 8.

Так же было решено произвести временные измерения для каждой функции смаза: out of focus blur, motion blur и функция размытия по Гауссу. Временные измерения проводились на GPU и соответственно CPU.

Таблица 1 - Оценка времени выполнения алгоритма на CPU и GPU. Out of focus blur

	Время выполнения алгоритма (сек)					
Размер изображения	512*512	800*600	1024*768	1280*1024	1366*768	1920*1080
CPU	2,3	5,7	9,51	18,8	30,23	70,27
GPU	4,8	8,9	12,5	23,7	30,32	62,4

Графическая форма данных из таблицы 1 приведена на рисунке 4.

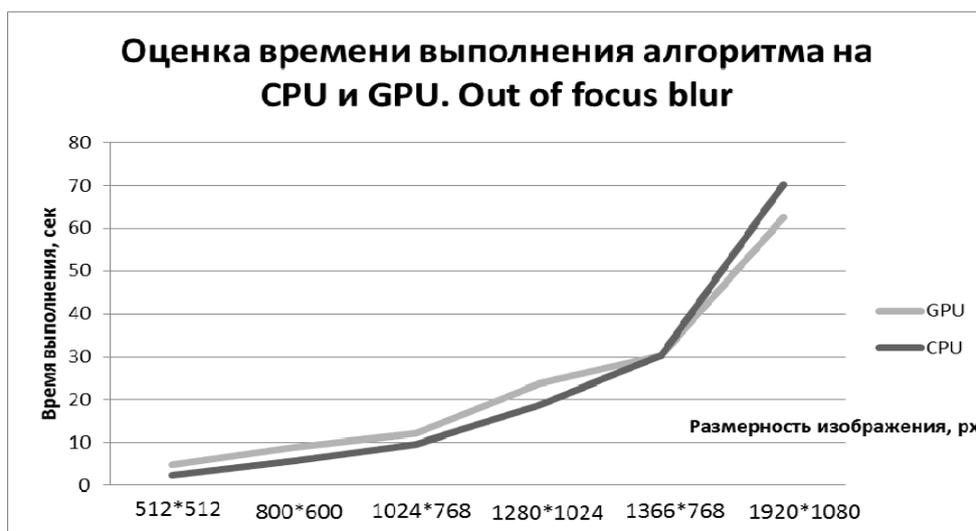


Рис. 4 Графики усредненных зависимостей общего времени выполнения алгоритмов с функцией смаза out of focus blur на GPU и CPU

В таблице 2 приведены данные, отражающие зависимость времени выполнения алгоритма от размерности восстановленного изображения на CPU и GPU с функцией смаза motion blur.

Таблица 2 - Оценка времени выполнения алгоритма на CPU и GPU.

Motion blur

	Время выполнения алгоритма (сек)					
Размер изображения	512*512	800*600	1024*768	1280*1024	1366*768	1920*1080
CPU	2,3	5,72	9,53	18,83	30,25	70,29
GPU	4,8	8,91	12,2	23,71	30,33	62,42

Графическая форма данных из таблицы 2 приведена на рисунке 5.

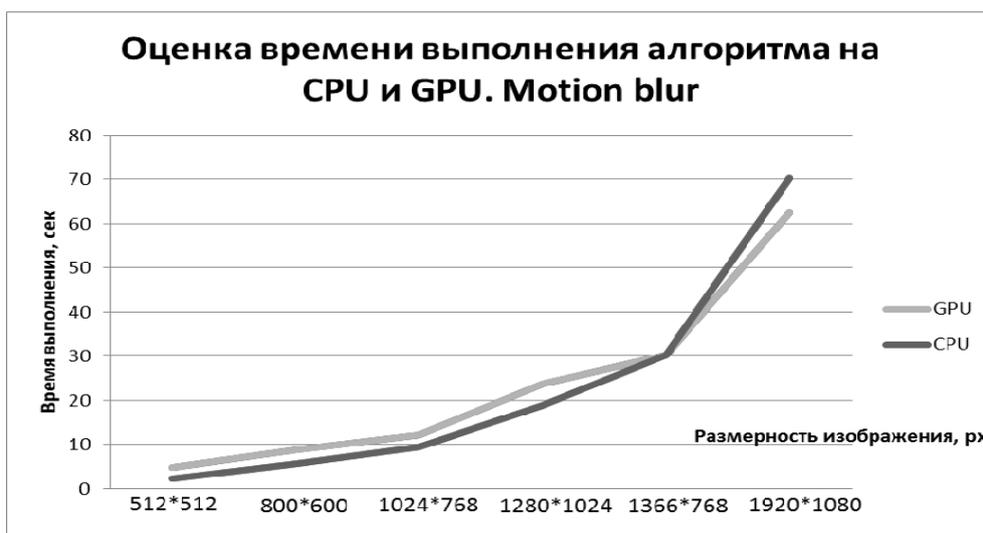


Рис. 5 Графики усредненных зависимостей общего времени выполнения алгоритмов с функцией смаза out of focus blur на GPU и CPU

В таблице 3 приведены данные, отражающие зависимость времени выполнения алгоритма от размерности восстановленного изображения на CPU и GPU с функцией смаза Гаусса.

Таблица 3 - Оценка времени выполнения алгоритма на CPU и GPU.  
Gaussian blur.

Размер изображения	Время выполнения алгоритма (сек)					
	512*512	800*600	1024*768	1280*1024	1366*768	1920*1080
CPU	2,31	5,73	9,57	18,83	30,24	70,28
GPU	4,8	8,91	12,1	23,71	30,31	62,41

Графическая форма данных из таблицы 3 приведена на рисунке 6.

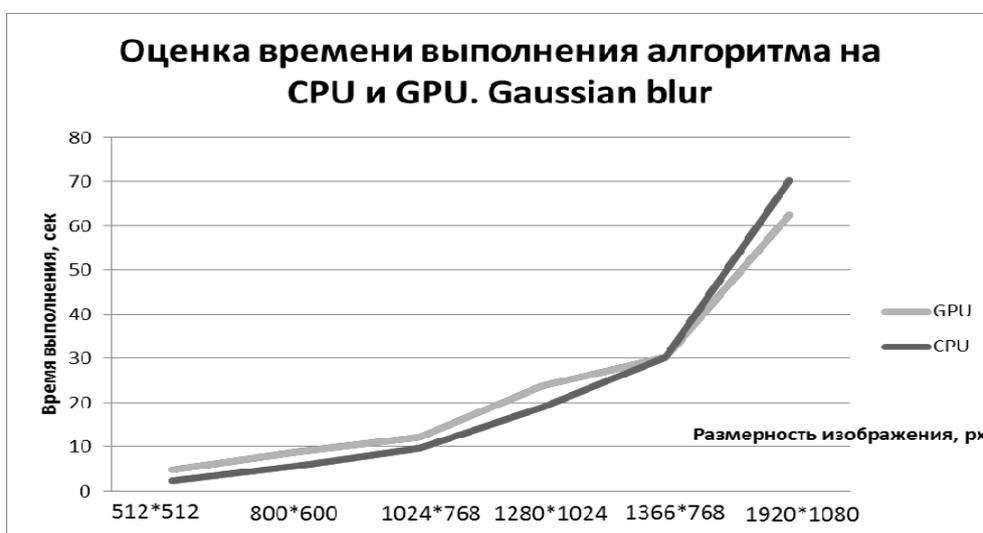


Рис. 6 Графики усредненных зависимостей общего времени выполнения алгоритмов с функцией смаза out of focus blur на GPU и CPU

Из представленных на рисунке данных следует, что отличие способов смаза не существенно влияет на временную сложность работы алгоритма как на графическом процессоре, так и на центральном процессоре. Так же было определено, что изображения при увеличении размерности изображения до 1920\*1080 алгоритм на графическом процессоре начинает опережать алгоритм на центральном процессоре. Определить точную границу предпочтительности каждого алгоритма невозможно по причине разницы в системных требованиях среди ЭВМ [6-8].

Архитектуре графических процессоров GPU необходимо определенное время на этап инициализации, в процессе которого подготавливается процессор, производится выгрузка библиотеки, и производятся другие операции, которые мы, как разработчики, контролировать не можем [9,10]. Распределение времени требуемого для работы алгоритмов, на CPU и GPU приведено в таблице 1.

### Заключение

1. Проведен краткий обзор функции смаза и дефокусировки изображений.
2. Реализовано программное обеспечение для восстановления расфокусированных изображений в среде Microsoft Visual Studio 2012. Для преобразования Фурье использовалась библиотека «aForge», для реализации алгоритма фильтра Винера на GPU – «CudaFu».
3. Проведено сравнение временной сложности исполняемого алгоритма на CPU и GPU для восстановления изображений, искаженных рассматриваемыми функциями.
4. На основании представленных результатов установлено, что при восстановлении расфокусированных изображений размерами до 1920\*1080 целесообразнее реализовывать на CPU, а при использовании более 1920\*1080 - на GPU.

### Литература

1. Che-Yen Wen, Chien-Hsiung Lee. Point spread functions and their applications to forensic image restoration. Forensic Science Journal. 2002; pp. 15-26.
2. Кулакович А.Ю. Венцов Н.Н. Краткий обзор и программная реализация избранных методов для деконволюции изображений // Инженерный вестник Дона, 2017, №4. URL: ivdon.ru/ru/magazine/archive/n4y2017/4468.



3. Агибалов О.И. Венцов Н.Н. Оценка зависимостей времени работы генетического алгоритма, выполняемого на CPU и GPU // Кибернетика и программирование, № 6, 2017 URL: [e-notabene.ru/kp/article\\_24509.html#3](http://e-notabene.ru/kp/article_24509.html#3).
4. Хуанга, Т. Обработка изображений и цифровая фильтрация / Т. Хуанга. М.: Мир, 1979. – С.320.
5. Андерсон Т. Статистический анализ временных рядов. – М.: Мир, 1976. – с.756.
6. Кулакович А.Ю. Программная реализация однослойной нейронной сети для распознавания цифровых символов // Инженерный вестник Дона, 2018, №3. URL: [ivdon.ru/ru/magazine/archive/n3y2018/5119](http://ivdon.ru/ru/magazine/archive/n3y2018/5119).
7. Сухорослов О.В. Организация вычислений в гетерогенных распределенных средах // Известия ЮФУ. Технические науки. Тематический выпуск: Суперкомпьютерные технологии. 2016. №12 (185). С. 115-130.
8. Richardson W. H. Bayesian-Based Iterative Method of Image Restoration, М.: Journal of the Optical Society of America, 1972. – pp.55-59.
9. Гонсалес Р., Вудс Р. Цифровая обработка изображений. – М.: Техносфера, 2005. 1072 с.
10. Вольфганг Стефан, Общая вариация, Регуляризация для линейных некорректных обратных задач: расширения и приложения, 2008. 151с.

### References

1. Che-Yen Wen, Chien-Hsiung Lee. Forensic Science Journal. 2002; pp. 15-26.
  2. Kulakovich A.Y. Ventsov N.N. Inzhenernyj vestnik Dona (Rus), 2017, №4. URL: [ivdon.ru/ru/magazine/archive/n4y2017/4468](http://ivdon.ru/ru/magazine/archive/n4y2017/4468).
  3. Agibalov O.I. Vencov N.N. Kibernetika i programmirovanie (Rus), №6, 2017. URL: [e-notabene.ru/kp/article\\_24509.html#3](http://e-notabene.ru/kp/article_24509.html#3).
-



4. Khuanga, T., Obrabotka izobrazheniy i tsifrovaya fil'tratsiya [Image processing and digital filtering]. Mir, 1979. 320 p.
5. Anderson T., Statisticheskiy analiz vremennykh ryadov [Statistical analysis of time series]. Mir, 1976. 756 p.
6. Kulakovich A.Y. Inzhenernyj vestnik Dona (Rus), 2018, №3. URL: [ivdon.ru/ru/magazine/archive/n3y2018/5119](http://ivdon.ru/ru/magazine/archive/n3y2018/5119).
7. Sukhoroslov O.V. Izvestiya YuFU. Tekhnicheskie nauki. Tematicheskii vypusk: Superkomp'yuternye tekhnologii. 2016. №12 (185). pp. 115-130.
8. Richardson W. H. Journal of the Optical Society of America. 1972. pp.55-59.
9. Gonsales R., Vuds R., Tsifrovaya obrabotka izobrazheniy [Digital image processing], Tekhnosfera. 2005. 1072 p.
10. Vol'fgang Stefan, Obshchaya variatsiya, Regularizatsiya dlya lineynykh nekorrektnykh obratnykh zadach: rasshireniya i prilozheniya [General variation, Regularization for linear inverse inverse problems: extensions and applications], 2008. 151 p.