УДК 519.673

**С.В. ХМЕЛЕВОЙ**

***Донецкий национальный технический университет, Украина***

# ИСПОЛЬЗОВАНИЕ GPU ДЛЯ РАСЧЕТОВ СКОРОСТЕЙ ГАЗО-ЖИДКОСТНЫХ СРЕД С ПОМОЩЬЮ МЕТОДА PIV

*Рассматривается реализация метода PIV (particle image velocimetry), который является оптическим методом для получения мгновенных макроскопических параметров потоков газо-жидкостных систем. На настоящее время даже современные CPU не в состоянии вычислять этот метод с достаточной скоростью. Поэтому для расчетов метода подключается GPU, в частности, видеокарты NVIDIA, поддерживающие технологию CUDA, предлагаются различные подходы для уменьшения вычислительной сложности алгоритма. В работе приведены результаты многопоточного запуска приложения. Доказана эффективность переноса расчетов на видеокарту.*

***Ключевые слова:*** *CUDA, PIV, потоки, кросс-корреляция.*

## Введение

Метод PIV основан на обнаружении смещения частиц на последовательностях изображений. Цель метода – определение скоростей различных областей прозрачных потоков жидких (псевдожидких) сред. В непрозрачных средах, таких как кипящий слой, псевдофлуидизацию производят в прозрачном аппарате, ширина которого намного больше, чем его глубина. Получается одномерное поле скоростей на основе последовательности изображений, снятой высокоскоростной видеокамерой. Метод PIV является весьма вычислительно затратным. На данный момент real-time версии алгоритма получить еще не удалось. Поэтому ускорение алгоритма является актуальной научной задачей. В данной работе ускорение производится с помощью переноса расчетов на видеокарту, использования GPU.

## 1. Суть метода PIV

Математически определение скоростей частиц основано на пространственной кросс-корреляции двух последовательных изображений. Изображения разделеняются на несколько так называемых областей интереса (например, 32x32 пикселя). В каждой области интереса выполненяется кросс-корреляция

$$R(x,y)=\frac{1}{N_x N_y}\times \times\sum_{i=1}^{N_x}\sum_{j=1}^{N_y}(I'(i,j)-\bar{I})(I''(i+x,j+y)-\bar{I}), \quad (1)$$

где $R(x,y)$ - величина кросс-корреляции для пикселя (x,y);

$I'(i,j)$ - интенсивность пикселя (i,j) первого региона изображения;

$I''(i+x,j+y)$ - интенсивность пикселя (i+x, j+y) второго региона изображения;

$\bar{I}$ - среднее значение интенсивности пикселей изображения.

По величине кросс-корреляции определяется её максимум для данного региона. По координатам максимума относительно центра региона определяется направление и величина смещения пикселей региона. Процесс нахождения скорости частиц для одного региона может быть представлен так, как на рис. 1.

Как показано в [1], классически метод PIV предлагалось применять в частотной области. И на сегодняшний день это - наиболее быстрый способ найти кросс-корреляцию. Тогда процедура нахождения кросс-корреляции будет выглядеть согласно представленной на рис. 2 [2].

Для двух одинаковых по координатам сегментов двух изображений, снятых через момент времени $\Delta t$, находятся их частотные представления с помощью быстрого преобразования Фурье (fast Fourier transform, FFT). Далее производится комплексное умножение результатов преобразования, после чего результат снова переносится во временную область с помощью обратного преобразования Фурье. Затем во временной области находится максимум кросс-корреляции.

Чтобы избежать случайных всплесков, максимальное значение кросскоррелограммы удаляют, пик находят сглаживанием остальных субмаксимальных значений.

Данный метод является вычислительно затратным. Обработка одного изображения занимает десятки секунд даже на современных компьютерах.

Поэтому предпринимаются попытки для ускорения работы метода. Зачастую это – распараллеливание метода. Например, в [2] описывалась реализация метода на FPGA (Field Programmable Gate Arrays). С ростом вычислительной способности современных видеокарт все более популярным становится их использование.

При этом для их программирования использовались как неудобные методы языков низкого уровня [3], так и, с появлением удобных средств взаимодействия с видеокартой, эти средства. Наиболее популярным на данный момент является использование технологии CUDA [4]. Она применялась для распараллеливания метода PIV, например, в [2, 5]

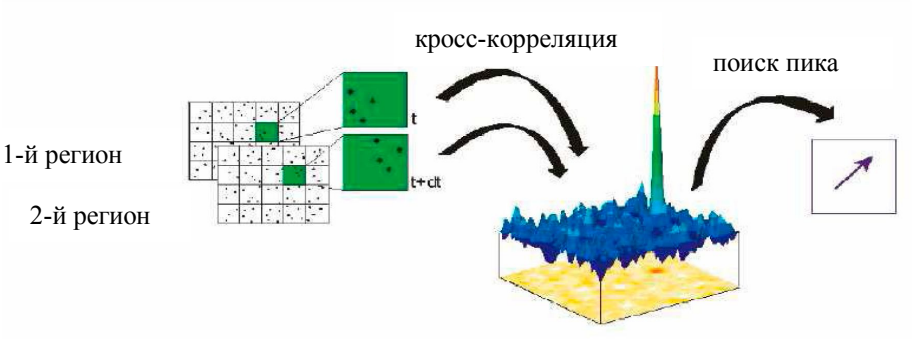


Рис. 1. Схема классического алгоритма PIV

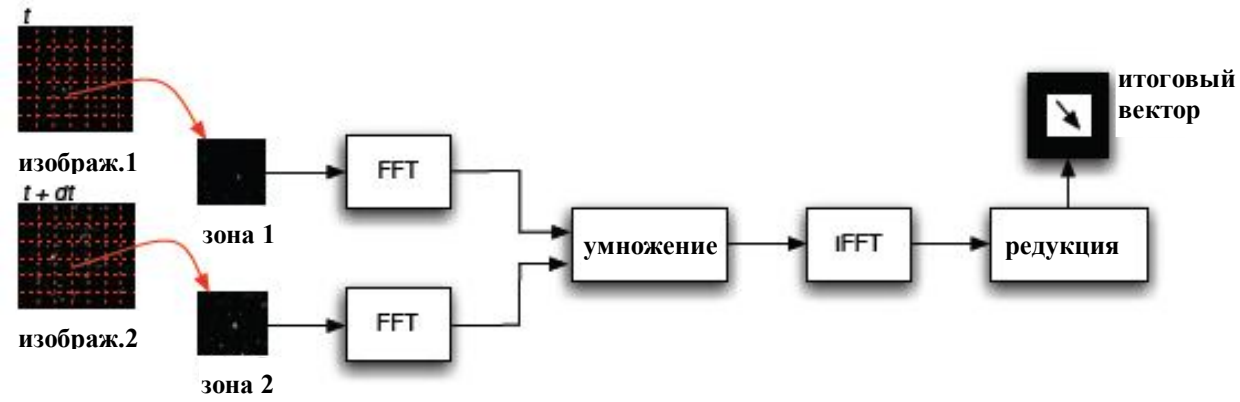


Рис. 2. Схема алгоритма PIV, используемая в современных системах

Однако большинство данных систем являются платными и не поддерживают некоторых необходимых функций, таких как непосредственное взаимодействие с видеокамерой.

А бесплатно распространяемое программное обеспечение [6] настолько медленно, что его использование без модификации не представляется возможным. Поэтому возникла необходимость в модификации существующих инструментальных средств.

## 2. Подключение высокоскоростной видеокамеры

Высокоскоростная видеокамера фирмы IDT [7] обладает способностью снимать со скоростью до 9000 кадров/сек. Для стандартного объектива размер пикселя готового изображения – 13,68x13,68 мкм, существует возможность как сохранения результатов в формате фильмов (AVI), так и в формате последовательности изображений (TIFF, JPEG, BMP). Фирмой-разработчиком разработан интерфейс для подключения камеры через прикладное программное обеспечение, в частности, интерфейс для подключения камеры через Microsoft Visual C++.

Видеокамера сохраняет данные на встроенном жёстком диске объемом до 16Гб, откуда они могут быть потом считаны клиентским программным обеспечением через интерфейс USB или 1000 Гб Ethernet. Существует возможность дистанционного подключения к камере через ЛВС Gigabit Ethernet.

Для камеры MotionPRO Y4-S2 разработано прикладное программное обеспечение, способное управлять камерой и считывать данные с камеры как в режиме одиночного кадра, так и в режиме последовательности изображений. Считанные данные непосредственно подаются на вход функции, реализующей метод PIV, что значительно убыстряет обработку данных.

## 3. Разработка программного обеспечения для отслеживания движения одиночного объекта

В случае, если на изображении наблюдается только один движущийся объект, расчет кросс-корреляции для всех регионов производить не имеет смысла: она будет изменяться только в районах, близких к движущемуся объекту. Поэтому имеет смысл заранее определить районы, в которых имеет смысл производить кросс-корреляцию. Для одного региона двух изображений можно посчитать расстояние между аналогичными пикселями изображений по формуле:

$$P_a = \frac{1}{N_x N_y} \sum_{i=0}^{N_x} \sum_{j=0}^{N_y} \sqrt{I'(i,j)^2 - I''(i,j)^2} \quad , (2)$$

где $I'(i,j)$ , $I''(i,j)$ - интенсивность пикселя (i,j) изображения.

Как показано на рис. 3, эта величина очень сильно взаимосвязана с величиной кросс-корреляции. Однако, в отличие от последней, для её вычисления не требуется три вичислительно затратных преобразования Фурье. Поэтому для отбора регионов, на которых следует осуществлять кросс-корреляцию, предлагается использовать эту величину.

Для определения физического размера порога расстояния в пикселях было получено её распределение в зависимости от величины, см. рис. 4. Распределение имеет обратный экспоненциальный вид, что логично. Число регионов, для которых расстояние больше, чем 0.7 максимально наблюдаемого для данной пары изображений, не более 4% от общего числа регионов изображения. Поэтому предлагается в качестве порога использовать величину 0.7 от максимально наблюдаемого значения расстояния между регионами.

Как показали эксперименты (см. табл. 2), введение порога расстояния убыстряет расчет изображения на 84%.

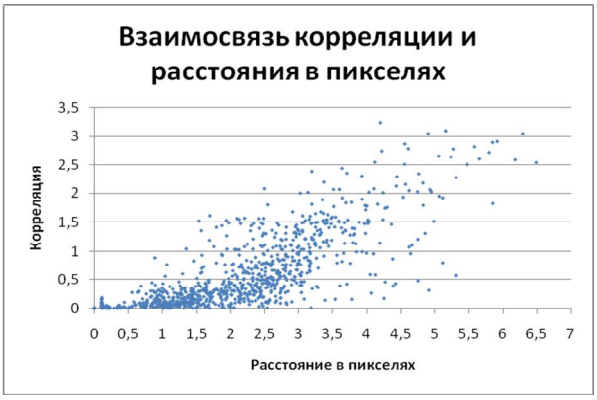


Рис. 3. Взаимосвязь кросс-корреляции и расстояния в пикселях

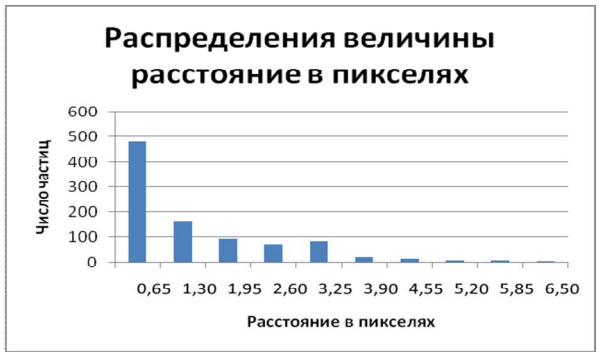


Рис. 4. Исследование распределения величины «Расстояние в пикселях»

Также актуальной прикладной задачей является определение скорости и направления вращения объекта вокруг своей оси. Эти величины могут быть посчитаны, если из всех векторов движения объекта (рис. 5) брать только проекцию на перпендикуляр к центру (рис. 6).

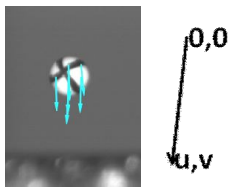


Рис. 5. Общий вид поля скоростей объекта

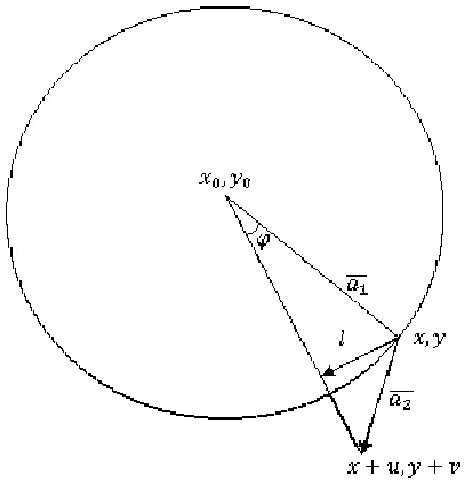


Рис. 6. Нахождение радиальной составляющей поля скоростей объекта

Для определения скорости вращения используются формулы.

$$\cos\varphi = \frac{\overline{a_1} * \overline{a_2}}{\left|\overline{a_1}\right| * \left|\overline{a_2}\right|} \quad (3)$$

$$\cos\varphi = \left|\overline{a_1}\right| * \sin\varphi \quad (4)$$

$$V_{rot} = \frac{1}{N} \sum_{i=1}^{N} l_i * Freq * Pix_{size} \quad (5)$$

Суммирование всех радиальных проекций скоростей даст суммарную скорость объекта вокруг своей оси.

## 4. Параллельная реализация алгоритма PIV

Алгоритм PIV обладает способностью к распараллеливанию, поскольку здесь требуется произведение одинаковых действий для большого числа кадров.

Автор уже имел опыт разработки параллельного алгоритма, в частности, параллельного генетического алгоритма [8]. В данном случае для разработки параллельного алгоритма выбрана среда программирования Microsoft Visual C++ и технология CUDA, очень популярная и перспективная на данный момент, а также принцип создания многопоточных приложений.

Было модифицировано программное обеспечение OpenPIV [6], для многопоточного выполнения. Один поток в качестве входных данных берет два изображения, которые затем считываются в массивы, и преобразуются для выполнения FFT. Средством обмена данными между главной программой и потоком во время выполнения потоков является механизм «сигнал-слот» объектной модели QT. Средством передачи результирующих данных от потока к главному приложению после окончания расчетов являются события Windows (event-ы).

Первоначально было создано многопоточное приложение, выполняющее FFT, необходимые для метода PIV, с помощью библиотеки FFTW [9]. Результаты исследований параллельного алгоритма показаны в табл. 1. Эксперименты проводились для двухъядерного процессора Intel Core I3-330M 2.17 ГГц в режиме «Debug» Visual C++: данные цифры не являются оптимальными и предназначены только для сравнения.

Таблица 1

Исследование введения многопоточности для PIV

| Время расчетов 50 изображений, 608x432 | мс. | % к пред. | % к перв. |
|---|---|---|---|
| Без многопоточности | 282206 | | |
| Многопоточность, 1 поток. | 242247 | 0,14 | 14,16 |
| Многопоточность, 2 потока | 178782 | 0,26 | 36,65 |
| Многопоточность, 3 потока | 183574 | -0,03 | 34,95 |
| Многопоточность, 4 потока | 203846 | -0,11 | 27,77 |

Как видно из табл. 1, многопоточность оправдывает себя уже с 1 потока. Поскольку приложение выполнялось на двухъядерном процессоре, как только кроме главного приложения появляется еще один поток, он выполняется уже на втором ядре, что дает прирост производительности. Максимальный прирост производительности наблюдается для 2 потоков, при дальнейшем увеличении числа потоков производительность падает. Это опять же связано с числом ядер микропроцессора. Однако, несомненно, в современных приложениях введение многопоточности даст значительное увеличение производительности работы.

## 5. Использование CUDA для параллельной реализации алгоритма

Технология CUDA в настоящее время весьма популярна в связи с предоставляемым для разработчиков вполне удобным интерфейсом программирования на языке, основанном на C.

Идеология программирования на CUDA учитывает, что даже атомарные операции выполняются асинхронно, и, если следующая операция зависит от предыдущей и требует её завершения, все нити, выполнявшие предыдущую операцию, необходимо синхронизировать.

Технологию применять для расчетов PIV можно, поскольку в библиотеках CUDA существуют стандартные функции для расчета FFT. Результаты применения технологии CUDA для PIV представлены в табл. 2.

Как видно из таблицы, применение технологии CUDA для данной задачи вполне уместно и может дать до 40% экономии времени. Кроме этого, показано, что значительное время (также порядка 40%) занимает прорисовка результатов, без которой также можно обойтись.

Таблица 2

Результаты опытов для 1 изображения

| Время расчетов для пар изображений 608x432 | ms | % к пред. | % к перв. |
|---|---|---|---|
| Openpiv | 4539,9 | | |
| С порогом расстояние в пикселях (ПРП) | **696,1** | 85,67 | 84,67 |
| CUDA, первая реализация, ПРП | 1280,2 | -84,9 | 71,80 |
| CUDA модиф., ПРП | **595,5** | 14,45 | 86,88 |
| CUDA модиф., все регионы | **3327** | | **26,72** |
| Расчет ПРП, без прорисовки | 369,4 | 89,90 | 91,86 |
| CUDA, ПРП, без прорисовки | 264,1 | 29,51 | 94,18 |

В табл. 3 показаны результаты многопоточного запуска приложения, где для расчета PIV использовались средства CUDA.

Таблица 3

Многопоточный запуск с использованием CUDA

| Время расчета 50 изображений, 608x432 пикс. | мс. | % к пред. | % к перв. |
|---|---|---|---|
| Без многопоточности | 282206 | | |
| CUDA, 1 поток | 260011 | | 7,86 |
| CUDA, 2 потока | 242729 | 0,07 | 13,99 |
| CUDA, 1 поток, без прорисовки (БП) | 101722 | 0,58 | 63,95 |
| CUDA, 2 потока, БП | 47735 | 0,53 | 83,09 |
| CUDA, 3 потока, БП | 36287 | 0,24 | 87,14 |

Как показали эксперименты, решения, использующие CUDA, хорошо масштабируются (прирост быстродействия в 2 и 3 раза, соответственно). Однако дальнейший рост быстродействия связан с аппаратными ограничениями видеокарты. Обойти эти ограничения сложно, поскольку 94% времени расчетов занимает хорошо оптимизированное преобразование cuFFT. Копирование данных в- и из- видеокарты занимает только 6%, а любая оптимизация может касаться только этого процесса. Но перспективность использования видеокарт для расчетов данного метода неоспорима.

Данные исследования выполнены совместно с техническим университетом г. Гамбург, Германия (Technical University of Hamburg-Harburg, TUHH)

## Выводы

Доказана взаимосвязь величины кросс-корреляции и расстояния между пикселями региона изображений. Для случая одиночного движущегося объекта предложено использование порога расстояния между пикселями для уменьшения числа регионов, для которых необходимо искать кросс-корреляцию;

Доказано, что введение многопоточности при расчетах PIV дает убыстрение скорости расчетов до 37% (на двухъядерном процессоре);

Доказана эффективность переноса расчетов на видеокарту. Применение технологии CUDA дает линейное убыстрение скорости расчетов (пропорционально к числу потоков CUDA), которое ограничено только аппаратными ресурсами видеокарты.

## Литература

*1. Adrian, R.J. Twenty years of particle image velocimetry [Text] / R.J. Adrian. – Experiments in Fluids, 2005. – Vol. 39, Issue 2. – P. 159 – 169.*

*2. Venugopal, V. Accelerating Particle Image Velocimetry Using Hybrid Architectures [Text] / Vivek Venugopal, Cameron D. Patterson, Kevin Shinpaugh // Symposium on Application Accelerators in High Performance Computing. – 2009 NCSA-UIUC, Champaign, IL. – P. 155 – 157.*

*3. Schiwietz T. GPU-PIV [Text] / T. Schiwietz, R. Westermann // VMV04 (2004). – P. 151 – 158.*

*4. CUDA zone [Электронный ресурс] / Nvidia corporation. – США, 2012. – Режим доступа: http://www.nvidia.ru/object/cuda_home_new_ru.html. – 12.03.2012 г.*

*5. Accelerating Computer Vision algorithms with Graphics Processing Units [Text] / Tamas K. Lengyel, James Gedarovich, Antonio Cusano, Thomas J. Peters // Electrical and Electronic Engineering, Miscellaneous Papers. – Nov. 2010. – Vol. 2, Is. 1. – P. 233 – 239.*

*6. Open Source Software for Particle Image Velocimetry [Электронній ресурс]. – Режим доступа: http://www.openpiv.net. – 12.03.2012 г.*

*7. IDT: Slow Motion Cameras, High Speed Digital Video. Камеры серии Y4. [Электронный ресурс] / IDT Company. – Режим доступа: http://www.idtpiv.com/imaging/y4.php. – 12.03.2012 г.*

*8. Хмелевой, С.В. Параллельная реализация эволюционного алгоритма для создания базы знаний на основе нечетких логических контроллеров [Текст] / С.В. Хмелевой // Інформаційно-керуючі системи на залізничному транспорті. – 2007. – № 4 (66). – С. 120 – 123.*

*9. FFTW Home Page. A fast, free C FFT library; includes real-complex, multidimensional, and parallel transforms [Электронный ресурс]. – Режим доступа: http:// www.fftw.org. – 12.03.2012 г.*

*Поступила в редакцию 12.03.2012*

**Рецензент:** д-р техн. наук, проф. Г.Ф. Кривуля, Харьковский национальный университет радиоэлектроники, Харьков, Украина.

## ВИКОРИСТАННЯ GPU ДЛЯ РОЗРАХУНКУ ШВИДКОСТЕЙ ГАЗО-ЖИДКІСНИХ СЕРЕДОВИЩ ЗА ДОПОМОГОЮ МЕТОДА PIV

***С.В. Хмелевой***

Розглядається реалізація методу PIV (particle image velocimetry), який являється оптичним методом для отримання миттєвих макроскопічних параметрів потоків газо-рідинних систем. На даний час навіть сучасні CPU не здатні розраховувати цей метод з достатньою швидкістю. Тому для розрахунків методу підключаться GPU, зокрема, відеокарти NVIDIA, які підтримують технологію CUDA, пропонуються різноманітні засоби зменшення розрахункової складності алгоритму. В роботі наведені результати багатопоточного запуску додатку. Доведена ефективність переносу розрахунків на відеокарту.

**Ключові слова:** CUDA, PIV, потоки, крос-кореляція/

## USING GPU FOR SPEED-CALCULATIONS OF GAZO-LIQUID SYSTEMS BY MEANS OF PIV METHOD

***S.V. Khmilovyi***

The implementation of method PIV (particle image velocimetry) is considered. PIV is an optical method for reception of instant macroscopical parameters of streams in gazo-liquid systems. Modern CPU on the present time is not in a condition to calculate this method with sufficient speed. Therefore GPU is used for method calculations, in particular, video cards NVIDIA which supporting CUDA technology. Various approaches for reduction the computing complexity of algorithm are offered. The results of multiflow application running are presented. The effectiveness of porting the calculations on video card is proved.

**Keywords:** CUDA, PIV, threads, kross-correlation/

**Хмелевой Сергей Владимирович** – канд. техн. наук, доцент, доцент кафедры автоматизированных систем управления Донецкого национального технического университета, Донецк, Украина.