

УДК 004.94

ПАРАЛЛЕЛЬНАЯ CUDA-РЕАЛИЗАЦИЯ АЛГОРИТМА СОПОСТАВЛЕНИЯ СТЕРЕОИЗОБРАЖЕНИЙ

В. А. Фурсов¹, Е. В. Гошин², А. П. Котов³

Поиск соответствующих фрагментов и точек на нескольких изображениях одной и той же сцены является одной из центральных проблем во многих прикладных задачах: автономная навигация по протяженным ориентирам, распознавание объектов на изображениях, реконструкция 3D-сцен и др. Для ее решения применяют различные корреляционные методы анализа сходства фрагментов. Алгоритмы на основе этих методов имеют высокую вычислительную сложность. Вместе с тем, как в указанных, так и многих других прикладных задачах одним из основных является требование оперативности принятия решений. Типичным примером является задача восстановления трехмерной сцены, в рамках которой реализуется алгоритм сопоставления стереоизображений. В настоящей статье для повышения быстродействия этого алгоритма предлагается вычислительная схема, допускающая его эффективную параллельную CUDA-реализацию. Высокая степень параллелизма достигается вследствие наличия большого числа однотипных операций при сопоставлении точек на эпиполярных линиях. Проведены эксперименты по реконструкции 3D-сцены по стереоизображениям с использованием предложенного параллельного алгоритма, получена оценка ускорения.

Ключевые слова: стереопара изображений, реконструкция 3D-сцены, проективная геометрия, эпиполярная геометрия, параллельные вычисления, графические процессоры, технология CUDA.

1. Введение. Алгоритмы поиска соответствующих фрагментов и точек на нескольких изображениях одной и той же сцены имеют широкий спектр приложений. Приведем несколько наиболее широко известных применений. В монографии [1] рассматривается задача поиска фрагментов изображений подстилающей поверхности Земли, наиболее подходящих на роль протяженных ориентиров. В этой, как и во многих других подобных задачах, ищется соответствие между текущим изображением и заданным эталоном. В системах компьютерного зрения разработано множество алгоритмов построения дескрипторов изображений, включая SIFT [2], LBP [3], HOG [4], PCA-SIFT [5], функционирование которых так же связано с нахождением и сопоставлением ключевых и/или соответствующих точек на изображениях. В задачах структурного редактирования изображений так же часто возникает необходимость нахождения соответствий при изменении местоположения объектов [6]. В задачах сегментации изображений, в частности при тематической обработке данных дистанционного зондирования Земли, алгоритмы сопоставления используются для нахождения границ сегментов изображения [7–9].

Важное место задача поиска соответствующих точек на разных изображениях одной и той же сцены занимает также в технологиях реконструкции 3D-сцен и построения цифровых моделей рельефа [10]. Этап нахождения соответствий в вычислительном отношении в этих технологиях является наиболее трудоемким. С другой стороны, задача реконструкции 3D-сцен чрезвычайно востребована в современных системах технического зрения. При этом часто ставится задача оперативного анализа обстановки в реальном времени [11]. В этой связи крайне актуальна задача построения быстродействующих алгоритмов и программного обеспечения нахождения соответствующих точек на стереоизображениях. В настоящей статье предлагается и исследуется параллельный алгоритм нахождения соответствий, реализованный в CUDA-среде.

¹ Самарский государственный аэрокосмический университет им. С. П. Королева, факультет базовой подготовки и фундаментальных наук, Московское шоссе, 34, 443086, г. Самара; зав. кафедрой, e-mail: fursov@ssau.ru

² Институт систем обработки изображений РАН, Молодогвардейская ул., 151, 443001, г. Самара; аспирант, e-mail: goshine@yandex.ru

³ Самарский государственный аэрокосмический университет им. С. П. Королева, факультет базовой подготовки и фундаментальных наук, Московское шоссе, 34, 443086, г. Самара; инженер, e-mail: antonykotov@gmail.com

2. Описание алгоритма, формулировка задачи. Технология реконструкции 3D-сцены основана на том, что для каждой точки на одном изображении стереопары осуществляется поиск соответствующей ей точки на другом изображении, затем по полученной паре точек определяются координаты их прообраза в трехмерном пространстве. Наиболее ресурсоемким в этой технологии является этап сопоставления фрагментов изображений с целью определения соответствующих точек. Поскольку обычно к системам видеонаблюдения предъявляются также требования низкой стоимости и компактности исполнения, актуальной является задача построения быстродействующих параллельных алгоритмов нахождения соответствующих точек и их реализация на сравнительно недорогих гибридных вычислительных устройствах, включающих в себя графические процессоры.

Соответствующие точки на стереоизображениях находятся на так называемых эпилюлярных линиях, которые могут быть определены с использованием заданной или вычисленной по тестовым точкам фундаментальной матрицы. Обычно этапу определения соответствующих точек предшествует этап ректификации изображений, в результате которого эпилюлярные линии преобразуются в параллельные прямые. В настоящей работе используется рассматривавшийся в [12] метод, в котором в явном виде ректификация не проводится. Этот метод обладает высокой степенью параллелизма, что делает его привлекательным с точки зрения CUDA-реализации.

Мы используем модель камеры-обскуры [13]. Предполагается, что реконструкция 3D-сцены осуществляется по стереоизображениям, полученным с двух камер. Пусть M — координаты некоторой точки в глобальной системе координат. Координаты этой точки в системах координат первой и второй камер определяются в следующем виде:

$$m_1 = K_1 \begin{bmatrix} R_1 \\ t_1 \end{bmatrix} M, \quad m_2 = K_2 \begin{bmatrix} R_2 \\ t_2 \end{bmatrix} M.$$

Здесь R_1 и R_2 — матрицы третьего порядка, описывающие поворот систем координат первой и второй камер относительно глобальной, t_1 и t_2 — координаты начала глобальной системы координат в системах координат первой и второй камер, а K_1 и K_2 — матрицы внутренних параметров камер:

$$K_1 = \begin{pmatrix} f_1 & 0 & u_{i0} \\ 0 & f_1 & v_{i0} \\ 0 & 0 & 1 \end{pmatrix}, \quad K_2 = \begin{pmatrix} f_2 & 0 & u_{i0} \\ 0 & f_2 & v_{i0} \\ 0 & 0 & 1 \end{pmatrix},$$

где f_i , $i = 1, 2$, — фокусные расстояния камер, (u_{i0}, v_{i0}) , $i = 1, 2$, — координаты главных точек камер в системах координат, связанных с камерами [14]. Зададим некоторую точку m_1 на первом изображении, тогда становится известной эпилюлярная плоскость Π и соответствующие этой точке эпилюлярные прямые l_1 и l_2 . Ясно, что при точном задании матриц проекций соответствующая точка m_2 на изображении второй камеры обязана лежать на прямой l_2 . Для отыскания всех соответствующих точек на эпилюлярных линиях l_1 и l_2 на одной из них, например l_1 , с некоторым шагом задается множество точек и формируются окружающие их фрагменты в виде единичных квадратов [14] со стороной, равной расстоянию между соседними пикселями. Из нескольких соседних единичных квадратов формируется так называемый составной фрагмент. На другой эпилюлярной линии l_2 ищутся заданные с тем же шагом соответствующие составные фрагменты (рис. 1).

В качестве признаков соответствия используются дескрипторы, составленные из значений яркости и норм величин градиентов:

$$f^{(k)}(x_1, y_1), \quad f^{(k)}(x_2, y_2), \quad k = \overline{1, N}; \quad \left| \text{grad } f^{(k)}(x_1, y_1) \right|, \quad \left| \text{grad } f^{(k)}(x_2, y_2) \right|, \quad k = \overline{1, N}.$$

Указанные дескрипторы вычисляются в N заданных вдоль эпилюлярных линий точках путем билинейной интерполяции значений этих характеристик в угловых точках соответствующих единичных квадратов. Достоинство этого дескриптора — простота реализации, поскольку указанные характеристики не зависят от направления эпилюлярной линии. Использование в составных фрагментах единичных квадратов позволяет существенно снизить ошибки, связанные с проективными искажениями.

В качестве соответствующих точек m_1 , m_2 принимаются точки на эпилюлярных линиях, соответствующие центральным единичным фрагментам на составных фрагментах. Оценка \hat{M} пространственных координат

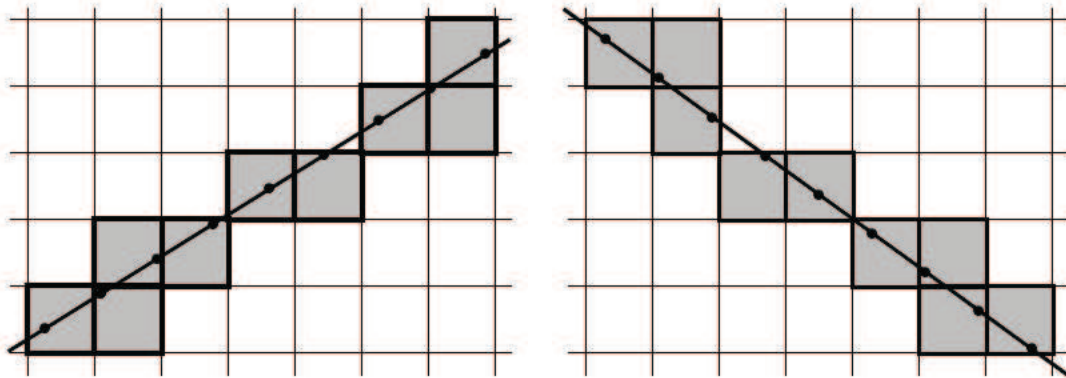


Рис. 1. Пример формирования составных фрагментов

точки M определяется как точка пересечения лучей (c_1, \hat{m}_1) и (c_2, \hat{m}_2) . Множество таких точек, дающих полное описание трехмерной модели сцены, получают путем “сканирования” трехмерного пространства пучками эпиполярных плоскостей с шагом, обеспечивающим требуемую точность.

В настоящей работе ставится задача эффективной реализации описанной выше технологии реконструкции 3D-сцен в CUDA-среде с использованием вычислительных систем, включающих в себя графические процессоры.

3. Оценка вычислительных затрат, параллельные схемы алгоритма. Проведем предварительную оценку объема вычислений, которые потребуются выполнить в рамках описанной технологии в предположении, что каждое из изображений стереопары имеет разрешение 800×600 . Если эпиполярные линии не горизонтальны, тогда на каждой линии будет около 10^3 точек. Для достижения достаточно точного восстановления 3D-сцены должно формироваться не менее 600 эпиполярных линий для каждого изображения стереопары. При выборе на соответствующих эпиполярных линиях пары точек, для каждой точки должно быть также задано восемь соседних точек (четыре справа и четыре слева). В результате будет сформирована пара фрагментов из девяти точек, которые затем сравниваются. Таким образом, для реализации алгоритма потребуется не менее $600 \times 10^3 \times 10^3 \times 9 = 5,4 \times 10^9$ элементарных вычислительных операций над данными. Ясно, что для оперативной (в том числе в реальном времени) реализации этой технологии в системах видеоконтроля и видеонаблюдения необходимо строить параллельный алгоритм. Далее рассматриваются две возможные параллельные схемы алгоритма.

На рис. 2 приведена укрупненная блок-схема алгоритма с декомпозицией исходных изображений по эпиполярным линиям. В этой схеме реализовано распараллеливание по данным, т.е. на каждом процессоре выполняются вычисления для одной пары эпиполярных линий. Достоинством такого способа является то, что вычисления для каждой пары эпиполярных линий выполняются независимо. Алгоритм с такой параллельной схемой реализован в работе [15]. В рамках этой схемы блок вычисления соответствующих точек для каждой пары эпиполярных линий был реализован на отдельном процессоре.

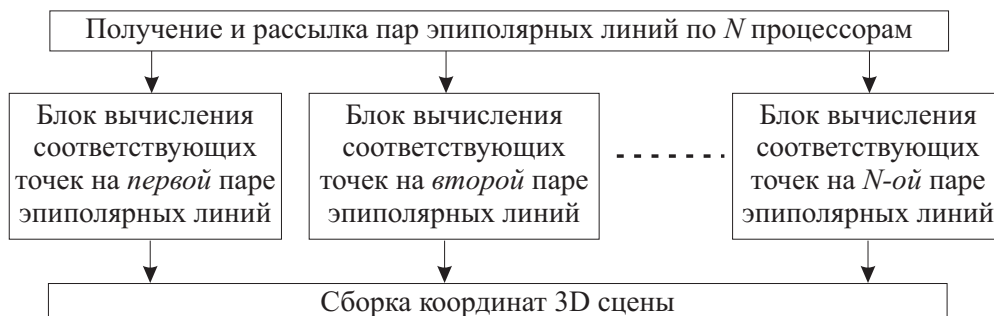


Рис. 2. Схема алгоритма с декомпозицией по эпиполярным линиям

На рис. 3 показана детальная схема алгоритма блока вычисления соответствующих точек на одной паре

эпиполярных линий. В данном случае вычисления, связанные с определением на них соответствующих точек, осуществляются последовательно. Если число соответствующих точек на одной паре эпиполярных линий, как указано выше, составляет 10^3 , то в соответствии с описанной технологией должны последовательно выполняться 9×10^6 однотипных операций сравнения составных фрагментов, что делает обоснованным применение CUDA-технологии.

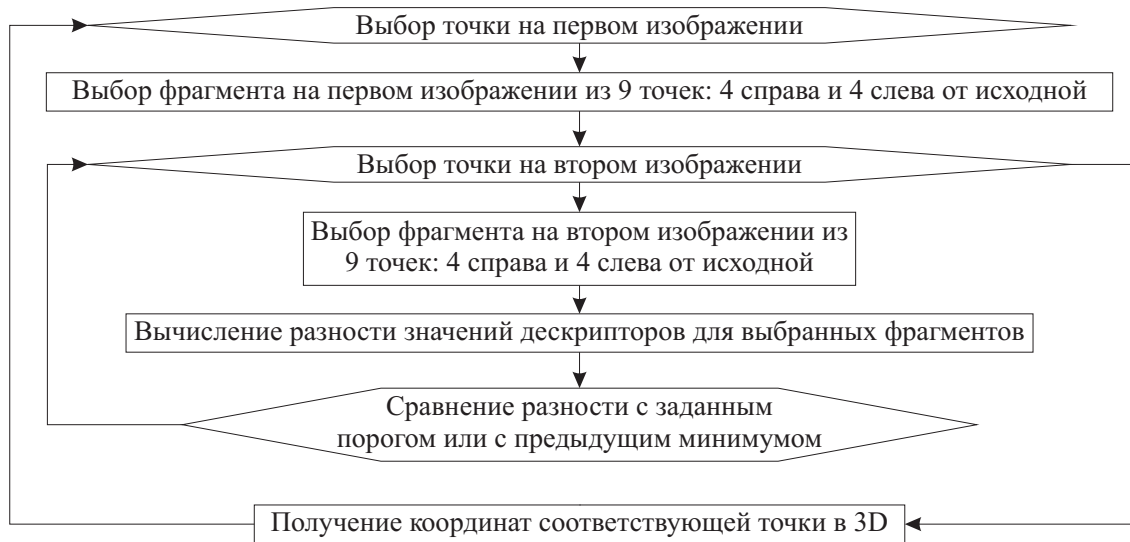


Рис. 3. Блок вычисления соответствующих точек на паре эпиполярных линий

Использование описанного подхода к распараллеливанию, основанного на декомпозиции по данным, при программировании на CUDA [16] нецелесообразно. При этом каждая из 600 нитей, соответствующих парам эпиполярных линий, будет оперировать с большим числом данных, что неэффективно. Nvidia CUDA представляет собой систему с общей памятью, ресурсы которой ограничены.

Наиболее предпочтительным подходом для реализации в CUDA-среде в данном случае является распараллеливание на более низком уровне, а именно распараллеливание блока вычислений соответствующих точек. Для этого модифицируем алгоритм, в частности разобьем общую схему алгоритма на два этапа. На рис. 4 показана двухэтапная параллельная схема реализации блока алгоритма на CUDA. В соответствии с приведенной схемой запуск CUDA-ядра осуществляется дважды.

На первом этапе формируется матрица сходства единичных фрагментов. В качестве элемента i -й строки j -го столбца формируется значение меры сходства дескрипторов i -й точки на первом изображении и j -й точки на втором изображении. Если дескрипторами являются значения яркости в данных точках, то в качестве меры сходства принимается разность между этими яркостями. В случае использования градиентов как дополнительных дескрипторов, в качестве меры сходства принимается сумма разностей значений дескрипторов.

На каждой нити вычисляется разность значений дескрипторов для выбранной точки на одном изображении и точки на втором изображении. Таким образом, одновременно будет задействовано 10^6 нитей. Результатом работы второго этапа является матрица сходства составных фрагментов, на основе которой осуществляется поиск всех соответствующих точек для выбранной пары эпиполярных линий.

4. Экспериментальные исследования. В качестве объекта эксперимента взята трехмерная модель плоскости с расположенными на ней предметами. Полученные координаты соответствующих точек моделируются в программе трассировки лучей POV-Ray (Persistence Of Vision Raytracer). Камеры имеют следующие параметры:

$$K_1 = K_2 = K = \begin{bmatrix} 400 & 0 & 400 \\ 0 & 400 & 300 \\ 0 & 0 & 1 \end{bmatrix}, \quad R_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad R_2 = \begin{bmatrix} 0.9848080 & -0.173648 \\ 0 & 1 & 0 \\ 0.1736480 & 0.984808 \end{bmatrix},$$

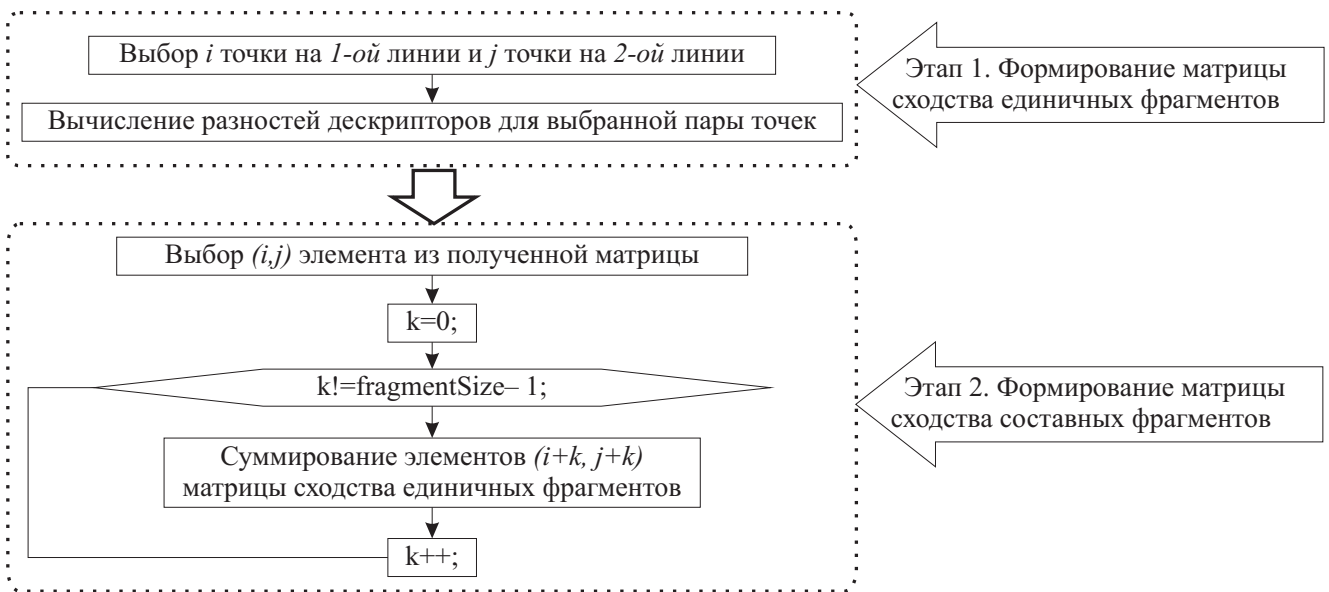


Рис. 4. Параллельная схема реализации блока алгоритма на CUDA

$$C_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad C_2 = \begin{bmatrix} -1.25 \\ 0 \\ -1.25 \end{bmatrix}.$$

На рис. 5 приведена пара полученных стереоизображений.

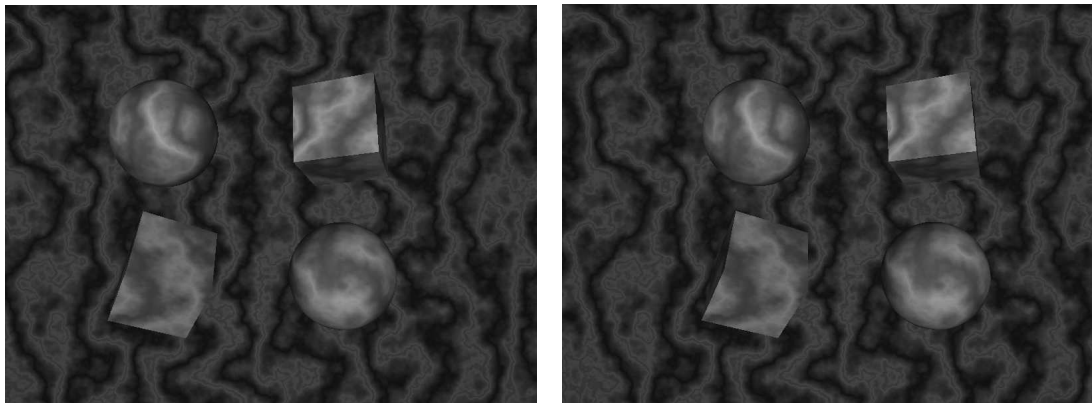


Рис. 5. Пара стереоизображений

По известным параметрам камер была вычислена фундаментальная матрица

$$F = \begin{bmatrix} 0 & -5.725 \times 10^{-5} & 0.017175 \\ 4.94192 \times 10^{-5} & 0 & -0.0540064 \\ -0.0148258 & 0.050673 & 1 \end{bmatrix}.$$

С помощью показанного на рис. 4 алгоритма было сформировано множество пар соответственных точек и вычислены соответствующие этим точкам координаты сцены в трехмерном пространстве. На рис. 6а и 6б для иллюстрации работы алгоритма приведены изображения исходной и восстановленной сцены.

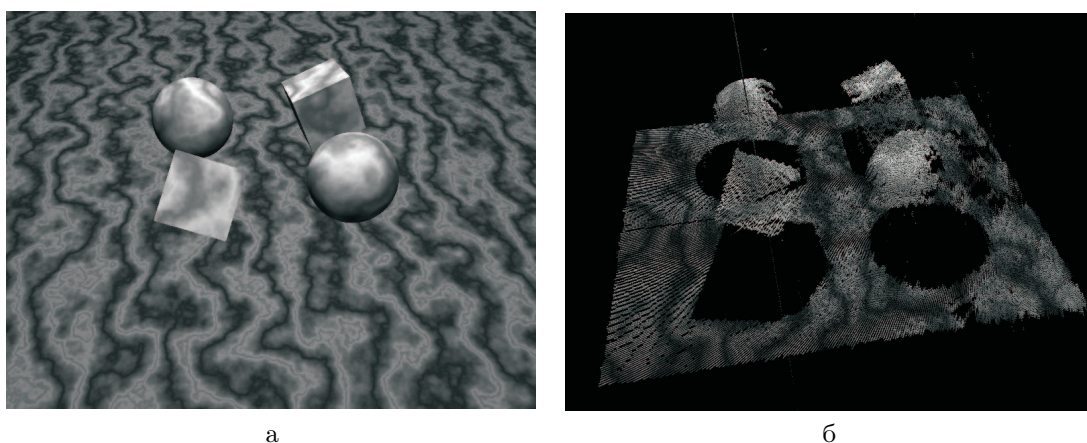


Рис. 6. Исходное (а) и восстановленное (б) изображения

Для проверки эффективности CUDA-реализации параллельного алгоритма были проведены следующие эксперименты. Алгоритм реализовывался на одном CPU (Intel (R) Core i5-3470). Для сравнения тот же алгоритм был реализован с использованием CPU+GPU на видеокартах GeForce GTX 550 Ti и Nvidia Tesla c2070. Время выполнения, полученное в результате экспериментов, приведено в таблице. Для строк 2 и 3 через дробь указаны видеокарты GeForce GTX 550 Ti и Nvidia Tesla c2070 соответственно.

Время работы последовательной и параллельной реализаций

№ п/п	Выполняемая процедура	Время работы (мс)	
		CPU	CPU+GPU
1	Предварительный этап формирования дескрипторов эпиполярных линий	635	
2	Формирование матрицы сходства единичных фрагментов	776	55 / 37
3	Формирование матрицы сходства составных фрагментов	817	50 / 36
4	Выбор соответственных точек	155	

Из таблицы видно, что при выполнении алгоритма только на CPU наиболее затратными по времени являются процедуры, указанные в строках 2 и 3 (этапы 1 и 2 на рис. 4), осуществляющие вычисление мер сходства составных фрагментов. Суммарное время работы этих этапов на CPU составляет 1.59 с. Применение для этих этапов параллельной схемы реализации на CUDA позволило значительно сократить время их выполнения. На GPU GeForce GTX 550 Ti результат составил 105 мс, а на GPU Nvidia Tesla c2070 73 мс. Таким образом, получено ускорение в 15 раз на GPU GeForce GTX 550 и в 21 раз на GPU Nvidia Tesla c2070.

5. Заключение. Результаты исследований показали, что время работы алгоритма при распараллеливании процесса нахождения соответствующих точек для каждой пары эпиполярных линий в CUDA-среде существенно меньше. Дальнейшее повышение быстродействия может быть достигнуто за счет включения блока, показанного на рис. 4, в параллельную схему алгоритма с декомпозицией по данным, показанную на рис. 2. Такая схема реализации возможна на гибридной вычислительной системе, имеющей большое число графических процессоров.

Работа выполнена при поддержке РФФИ (проекты 12-07-00581, 13-07-12030 офи_м и 13-07-13166).

Статья рекомендована к публикации Программным комитетом Международной научной конференции “Параллельные вычислительные технологии” (ПаВТ-2014; <http://agora.guru.ru/pavt2014>).

СПИСОК ЛИТЕРАТУРЫ

1. Аншаков Г.П., Голяков А.Д., Петрищев В.Ф., Фурсов В.А. Автономная навигация космических аппаратов. Самара: ЦСКБ-Прогресс, 2011.
2. Lowe D.G. Object recognition from local scale-invariant features // Proc. Seventh IEEE Int. Conf. on Computer Vision. Vol. 2. New York: IEEE Press, 1999. 1150–1157.

3. *Ojala T., Pietikainen M., Harwood D.* Performance evaluation of texture measures with classification based on Kullback discrimination of distributions // Proc. 12th IAPR Int. Conf. on Pattern Recognition. Vol. 1. New York: IEEE Press, 1994. 582–585.
4. *Dalal N., Triggs B.* Histograms of oriented gradients for human detection // IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR'05). Vol. 1. New York: IEEE Press, 2005. 886–893.
5. *Ke Y., Sukthankar R.* PCA-SIFT: a more distinctive representation for local image descriptors // Proc. 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Vol. 2. New York: IEEE Press, 2004. 506–513.
6. *Barnes C., Shechtman E., Finkelstein A., Goldman D.* PatchMatch: a randomized correspondence algorithm for structural image editing // ACM Trans. Graph. 2009. **28**, N 3. Article No. 24.
7. *Klaus A., Sormann M., Karner K.* Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure // Proc. 18th Int. Conf. on Pattern Recognition. Vol. 3. New York: IEEE Press, 2006. 15–18.
8. *Bleyer M., Rother C., Kohli P., Scharsteinh D., Sinha S.* Object stereo–joint stereo matching and object segmentation // Proc. 2011 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). New York: IEEE Press, 2011. 3081–3088.
9. *Zitnick C. L., Kang S.B.* Stereo for image-based rendering using image over-segmentation // Int. J. of Computer Vision. **75**, N 1. 2007. 49–65.
10. *Yang Q., Ahuja N.* Stereo matching using epipolar distance transform // IEEE Trans. on Image Processing. 2012. **21**, N 10. 4410–4419.
11. *Kowalczyk J., Psota E.T., Perez L.C.* Real-time stereo matching on CUDA using an iterative refinement method for adaptive support-weight correspondences // IEEE Trans. on Circuits and Systems for Video Technology. 2013. **23**, N 1. 94–104.
12. *Фурсов В.А., Гошин Е.В., Бибииков С.А.* Реконструкция 3D-сцен на пучках эпиполярных плоскостей стереоизображений // Мехатроника, автоматизация, управление. 2013. № 9. 19–24.
13. *Форсайт Д., Понс Ж.* Компьютерное зрение. Современный подход. М.: Вильямс, 2004.
14. *Грузман И.С., Киричук В.С., Косых В.П., Перетягин Г.И., Спектор А.А.* Цифровая обработка изображений в информационных системах. Новосибирск: Изд-во НГТУ, 2002.
15. *Фурсов В.А., Бибииков С.А., Гошин Е.В., Жердев Д.А.* Параллельная реализация модифицированного алгоритма реконструкции трехмерной сцены по стереоизображениям // Тр. Междунар. научной конференции “Параллельные вычислительные технологии”, 1–5 апреля 2013 г., г. Челябинск. Челябинск: Изд-во ЮУрГУ, 2013. 624.
16. *Боресков А.В., Харламов А.А.* Основы работы с технологией CUDA. М.: ДМК Пресс, 2010.

Поступила в редакцию
28.01.2014

Parallel CUDA Implementation of a Stereo Matching Algorithm

V. A. Fursov¹, E. V. Goshin², and A. P. Kotov³

¹ *Korolev Samara State Aerospace University, Faculty of Basic Training and Fundamental Sciences; shosse Moskovskoe 34, Samara, 443086, Russia; Professor, Head of Department, e-mail: fursov@ssau.ru*

² *Image Processing Systems Institute, Russian Academy of Sciences; ulitsa Molodogvardeiskaya 151, Samara, 443001, Russia; Graduate Student, e-mail: goshine@yandex.ru*

³ *Korolev Samara State Aerospace University, Faculty of Basic Training and Fundamental Sciences; shosse Moskovskoe 34, Samara, 443086, Russia; Engineer, e-mail: antonykotov@gmail.com*

Received January 28, 2014

Abstract: Searching for the corresponding fragments and points on several images of the same scene is one of the central problems in many applications: autonomous navigation extended landmarks, pattern recognition, 3D-scene reconstruction, etc. To solve this problem, various correlation methods for the similarity analysis of fragments are used. Algorithms based on these methods have a high computational complexity. In this paper we consider a stereo matching algorithm for 3D-scene reconstruction. We propose a computational scheme that improves the performance of this algorithm. This computational scheme is implemented using CUDA technology. A high degree

of parallelism is achieved due to a large number of the same operations for corresponding points on epipolar lines. Numerical experiments were carried out using the proposed parallel algorithm. The resulting speed-up is estimated.

Keywords: stereo matching, 3D reconstruction, projective geometry, epipolar geometry, parallel computing, graphics processors, CUDA technology.

References

1. G. P. Anshakov, A. D. Golyakov, V. F. Petrishchev, and V. A. Fursov, *Autonomous Navigation of Spacecrafts* (Progress, Samara, 2011) [in Russian].
2. D. G. Lowe, "Object Recognition from Local Scale-Invariant Features," in *Proc. 7th IEEE Int. Conf. on Computer Vision, Kerkyra, Greece, September 20–27, 1999* (IEEE Press, New York, 1999), Vol. 2, pp. 1150–1157.
3. T. Ojala, M. Pietikainen, and D. Harwood, "Performance Evaluation of Texture Measures with Classification Based on Kullback Discrimination of Distributions," in *Proc. 12th IAPR Int. Conf. on Pattern Recognition, Jerusalem, October 9–13, 1994* (IEEE Press, New York, 1994), Vol. 1, pp. 582–585.
4. N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," in *Proc. 2005 IEEE Conf. on Computer Vision and Pattern Recognition, San Diego, June 20–26, 2005* (IEEE Press, New York, 2005), Vol. 1, pp. 886–893.
5. Y. Ke and R. Sukthankar, "PCA-SIFT: A More Distinctive Representation for Local Image Descriptors," in *Proc. 2004 IEEE Conf. on Computer Vision and Pattern Recognition, Washington, DC, June 27–July 2, 2004* (IEEE Press, New York, 2004), Vol. 2, pp. 506–513.
6. C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman, "PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing," *ACM Trans. Graph.* **28** (3) (2009), Article No. 24.
7. A. Klaus, M. Sormann, and K. Karner, "Segment-Based Stereo Matching Using Belief Propagation and a Self-Adapting Dissimilarity Measure," in *Proc. 18th Int. Conf. on Pattern Recognition, Hong Kong, August 20–24, 2006* (IEEE Press, New York, 2006), Vol. 3, pp. 15–18.
8. M. Bleyer, C. Rother, P. Kohli, et al., "Object Stereo-Joint Stereo Matching and Object Segmentation," in *Proc. 2011 IEEE Conf. on Computer Vision and Pattern Recognition, Colorado Springs, June 20–25, 2011* (IEEE Press, New York, 2011), pp. 3081–3088.
9. C. L. Zitnick and S. B. Kang, "Stereo for Image-Based Rendering Using Image Over-Segmentation," *Int. J. Comput. Vis.* **75** (1), 49–65 (2007).
10. Q. Yang and N. Ahuja, "Stereo Matching Using Epipolar Distance Transform," *IEEE Trans. Image Process.* **21** (10), 4410–4419 (2012).
11. J. Kowalczyk, E. T. Psota, and L. C. Perez, "Real-Time Stereo Matching on CUDA Using an Iterative Refinement Method for Adaptive Support-Weight Correspondences," *IEEE Trans. Circuits Syst. Video Techn.* **23** (1), 94–104 (2013).
12. V. A. Fursov, E. V. Goshin, and S. A. Bibikov, "3D-Scene Stereo Reconstruction on Sheaves of Epipolar Planes," *Mekhatron. Avtomat. Upravl.*, No. 9, 19–24 (2013).
13. D. A. Forsyth and J. Ponce, *Computer Vision: A Modern Approach* (Prentice Hall, Englewood Cliffs, 2003; Williams, Moscow, 2004).
14. I. S. Gruzman, V. S. Kirichuk, V. P. Kosykh, G. I. Peretyagin, and A. A. Spektor, *Digital Image Processing in Information Systems* (Novosibirsk Gos. Tekh. Univ., Novosibirsk, 2002) [in Russian].
15. V. A. Fursov, S. A. Bibikov, E. V. Goshin, and D. A. Zherdev, "Parallel Implementation of a Modified Algorithm for Reconstruction of Three-Dimensional Scenes Using Stereoimages," in *Proc. Int. Conf. on Parallel Computational Technologies, Chelyabinsk, April 1–5, 2013* (South Ural State Univ., Chelyabinsk, 2013), p. 624.
16. A. V. Boreskov and A. A. Kharlamov, *Fundamentals of CUDA Technology* (DMK Press, Moscow, 2010) [in Russian].