Параллельная CUDA-реализация алгоритма сопоставления стереоизображений*

В.А. Фурсов[1,2], Е.В. Гошин[1,2], А.П. Котов[1]

Самарский государственный аэрокосмический университет им. академика С.П.Королёва[1], Институт систем обработки изображений РАН[2]

В работе рассматривается задача построения параллельного алгоритма сопоставления стереоизображений с целью последующего восстановления трехмерной сцены. Предлагается вычислительная схема, допускающая 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.

1. Введение

Задача реконструкции 3D-сцен чрезвычайно востребована в современных системах технического зрения. При этом часто ставится задача оперативного анализа обстановки в реальном времени. 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 графические процессоры.

Соответствующие точки на стереоизображениях находятся на так называемых эпиполярных линиях, которые могут быть определены с использованием заданной или вычисленной по тестовым соответствующим точкам фундаментальной матрицы. Обычно этапу определения соответствующих точек предшествует этап ректификации изображений, в результате которого эпиполярные линии преобразуются в параллельные прямые. В настоящей работе используется рассматривавшийся в работе [1] метод, в котором в явном виде ректификация не проводится. Этот метод обладает высокой степенью параллелизма, что делает его привлекательным с точки зрения CUDA-реализации.

2 Описание алгоритма, формулировка задачи

В работе используется модель камеры-обскуры [2]. Предполагается, что реконструкция 3D-сцены осуществляется по стереоизображениям, полученным с двух камер. Пусть $\mathbf{M}$ - координаты некоторой точки в глобальной системе координат. Координаты этой точки в системах координат первой и второй камер определяются как

$\mathbf{m}_1 = \mathbf{K}_1[\mathbf{R}_1 \vdots \mathbf{t}_1]\mathbf{M}$,

$\mathbf{m}_2 = \mathbf{K}_2[\mathbf{R}_2 \vdots \mathbf{t}_2]\mathbf{M}$.

Здесь $\mathbf{R}_1$, $\mathbf{R}_2$ - матрицы размерности 3×3, описывающие поворот систем координат первой и второй камер относительно глобальной, а $\mathbf{t}_1$, $\mathbf{t}_2$ – координаты начала глобальной систе-

*Работа выполнена при поддержке РФФИ (проекты № 12-07-00581, № 13-07-12030 офи_м, № 13-07-13166)

мы координат в системах координат первой и второй камер. $\mathbf{K}_1$, $\mathbf{K}_2$ – матрицы внутренних параметров камер:

$$\mathbf{K}_1 = \begin{bmatrix} f_1 & 0 & u_{i0} \\ 0 & f_1 & v_{i0} \\ 0 & 0 & 1 \end{bmatrix}, \mathbf{K}_2 = \begin{bmatrix} f_2 & 0 & u_{i0} \\ 0 & f_2 & v_{i0} \\ 0 & 0 & 1 \end{bmatrix},$$

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 [3].

Зададим некоторую точку $\mathbf{m}_1$ на первом изображении, тогда становится известной эпиполярная плоскость $\mathbf{\Pi}$ и соответствующие этой точке эпиполярные прямые $\mathbf{l}_1$, $\mathbf{l}_2$. Ясно, что при точном задании матриц проекций соответствующая точка $\mathbf{m}_2$ на изображении второй камеры обязана лежать на прямой $\mathbf{l}_2$. Для отыскания всех соответствующих точек на эпиполярных линиях $\mathbf{l}_1$ и $\mathbf{l}_2$ на одной из них, например $\mathbf{l}_1$, с некоторым шагом задается множество точек и формируются окружающие их фрагменты в виде единичных квадратов [3] со стороной, равной расстоянию между соседними пикселями. Из нескольких соседних единичных квадратов формируется так называемый составной фрагмент. На другой эпиполярной линии - $\mathbf{l}_2$ ищутся заданные с тем же шагом соответствующие составные фрагменты (рис. 1).

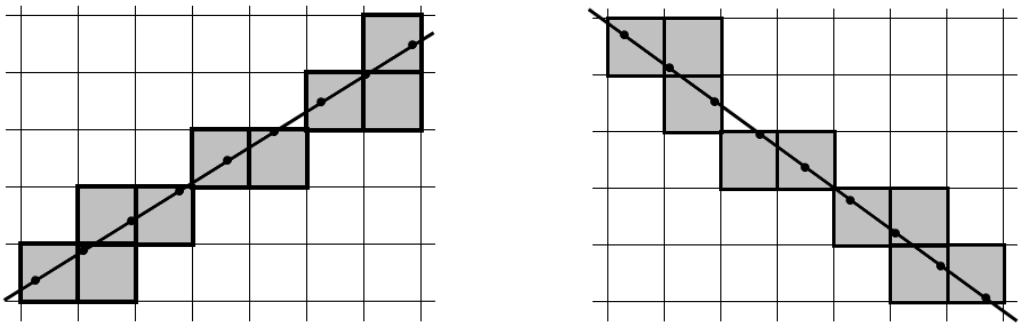


Рис. 1. Пример формирования составных фрагментов

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

$$f^{(k)}(x_1, y_1),\ f^{(k)}(x_2, y_2),\ k = \overline{1, N},$$

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

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

В качестве соответствующих точек $\mathbf{m}_1$, $\mathbf{m}_2$ принимаются точки на эпиполярных линиях, соответствующие центральным единичным фрагментам на составных фрагментах. Оценка $\hat{\mathbf{M}}$ пространственных координат точки $\mathbf{M}$ определяется как точка пересечения лучей ($\mathbf{c}_1$, $\hat{\mathbf{m}}_1$) и ($\mathbf{c}_2$, $\hat{\mathbf{m}}_2$). 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 графические процессоры.

3. Оценка вычислительных затрат, параллельные схемы алгоритма

Проведем предварительную оценку объема вычислений, которые потребуется выполнить в рамках описанной технологии в предположении, что каждое из изображений стереопары имеет разрешение 800x600. Если эпиполярные линии не горизонтальны, тогда на каждой линии будет около10^3 точек. Для достижения достаточно точного восстановления 3D-сцены должно формироваться не менее 600 эпиполярных линий для каждого изображения стереопары. При выборе на соответствующих эпополярных 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 $600\times10^3\times10^3\times9 = 5{,}4\times10^9$ элементарных вычислительных операций над данными. Ясно, что для оперативной (в т.ч. в реальном времени) реализации этой технологии в системах видеоконтроля и видеонаблюдения необходимо строить параллельный алгоритм. Далее рассматриваются две возможные параллельные схемы алгоритма.

На рисунке 2 приведена укрупненная блок-схема алгоритма с декомпозицией исходных изображений по эпиполярным линиям. В данной 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 [5]. В рамках этой схемы, блок вычисления соответствующих точек для каждой пары эпиполярных линий был реализован на отдельном процессоре.

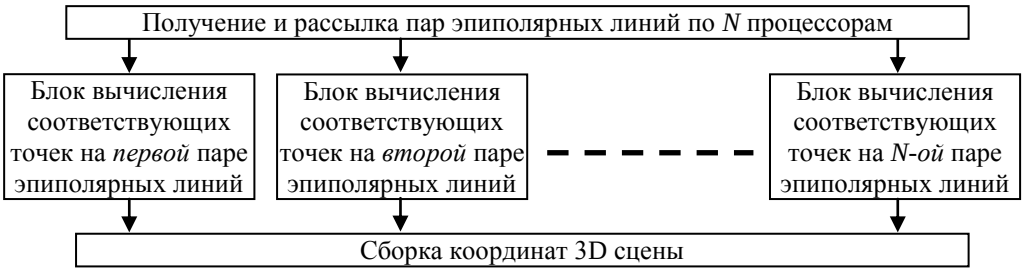


Рис. 2. Схема алгоритма с декомпозицией по эпиполярным линиям

На рисунке 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. При этом, 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.

Использование описанного подхода к распараллеливанию, основанного на декомпозиции по данным, при программировании на CUDA [4] 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.

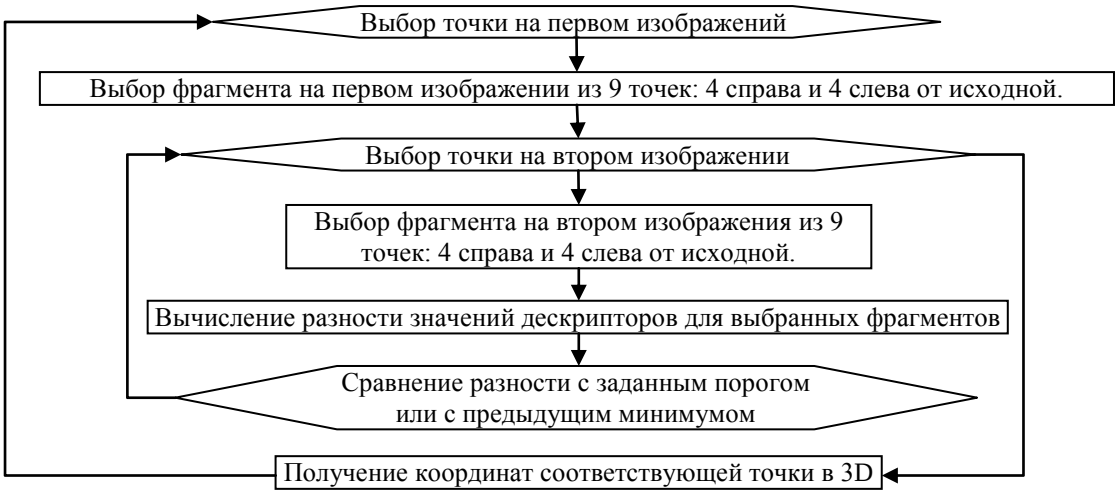


Рис. 3. Блок вычисления соответствующих точек на паре эпиполярных линий

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 рисунке 4 показана двухэтапная параллельная схема реализации блока алгоритма на CUDA. В соответствии с приведенной схемой запуск CUDA -ядра осуществляется дважды.

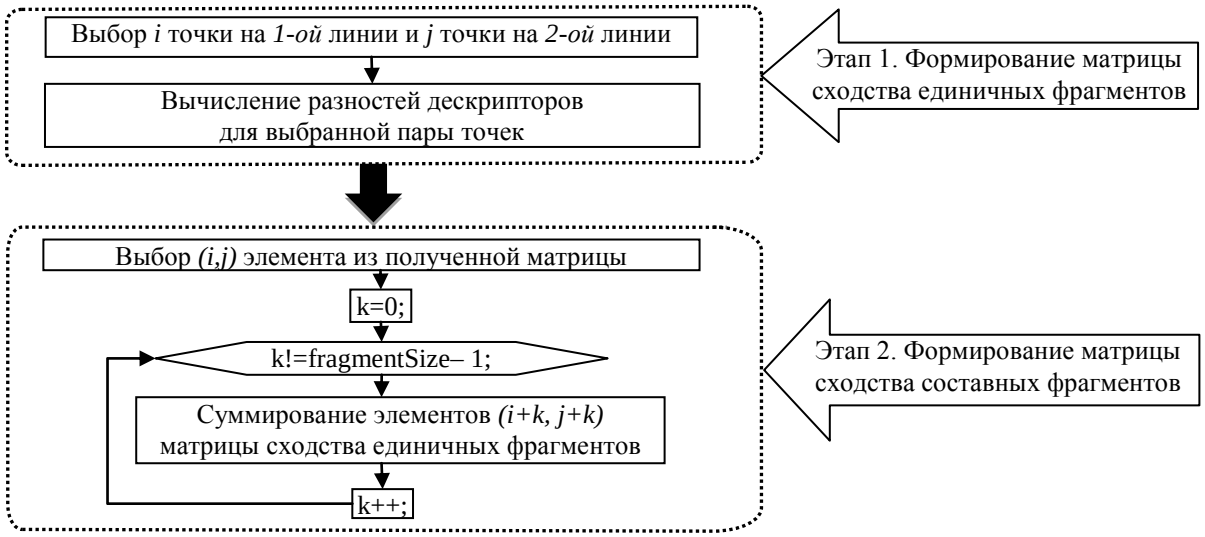


Рис. 4. Параллельная схема реализации блока алгоритма на CUDA

На первом этапе формируется матрица сходства единичных фрагментов. В качестве элемента *i*-ой строки *j*-ого столбца формируется значение меры сходства дескрипторов *i*-ой точки на первом изображении и *j*-ой точки на втором изображении. Если дескрипторами являются значение 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. Камеры имеют параметры $\mathbf{K}_1 = \mathbf{K}_2 = \mathbf{K} = \begin{bmatrix} 400 & 0 & 400 \\ 0 & 400 & 300 \\ 0 & 0 & 1 \end{bmatrix}$, $\mathbf{R}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $\mathbf{R}_2 = \begin{bmatrix} 0.984808 & 0 & -0{,}173648 \\ 0 & 1 & 0 \\ 0{,}173648 & 0 & 0.984808 \end{bmatrix}$, $\mathbf{C}_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$, $\mathbf{C}_2 = \begin{bmatrix} -1.25 \\ 0 \\ -1.25 \end{bmatrix}$.

На рисунке 5 приведена пара полученных стереоизображений.

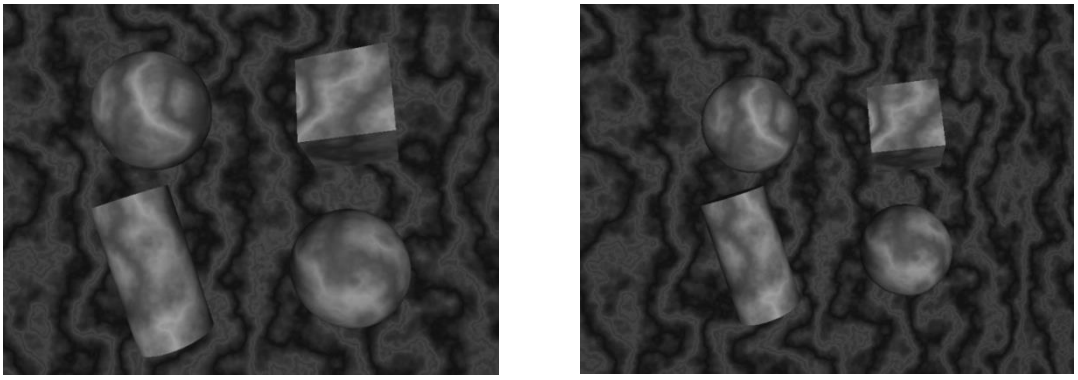
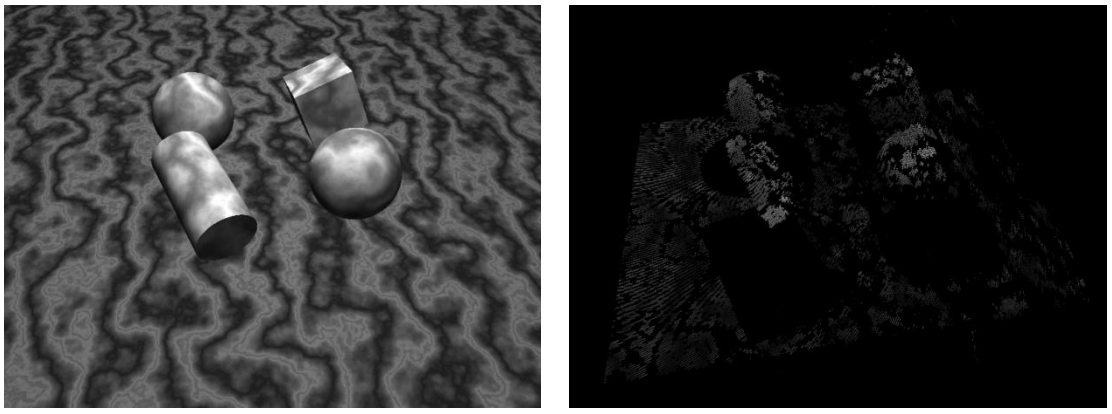


Рис. 5. Пара стереоизображений

По известным параметрам камер была вычислена фундаментальная матрица $\mathbf{F} = \begin{bmatrix} 0 & -5.725 \cdot 10^{-5} & 0.017175 \\ 4.94192 \cdot 10^{-5} & 0 & -0.0540064 \\ -0.0148258 & 0.050673 & 1 \end{bmatrix}$.

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



а б

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 видеокаортах GeForce GTX 550 Ti и Nvidia Tesla c2070. Время выполнения, полученное в результате экспериментов, приведено в таблице 1. Для строк 2 и 3, через дробь, указаны видеокарты GeForce GTX 550 Ti и Nvidia Tesla c2070 соответственно.

Таблица 1. 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 1 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 рисунке 4), осуществляющие вычисление мер сходства составных фрагментов. Суммарное время работы этих этапов на CPU – 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 рисунке 4, в параллельную схему алгоритма с декомпозицией по данным, показанную на рисунке 2. Такая схема реализации возможна на гибридной вычислительной системе, имеющей большое число графических процессоров.

Литература

1. Фурсов В.А., Гошин Е.В., Бибиков С.А.Реконструкция 3D-сцен на пучках эпиполярных плоскостей стереоизображений // Мехатроника, автоматизация, управление, 2013, №9 (150), С. 19-24
2. Форсайт Д., Понс Ж. Компьютерное зрение. Современный подход. М.: Издательский дом "Вильямс", 2004. - 928 с.
3. Грузман И.С., Киричук В.С., Косых В.П., Перетягин Г.И., Спектор А.А. Цифровая обработка изображений в информационных системах: Учебное пособие. Новосибирск: Изд-во НГТУ, 2002. - 352 с.
4. Боресков А.В., Харламов А.А. Основы работы с технологией CUDA. М.: Изд-во ДМК Пресс, 2010. – 234 с.
5. Бибиков С.А., Гошин Е.В., Жердев Д.А., Фурсов В.А. Параллельная реализация модифицированного алгоритма реконструкции трехмерной сцены по стереоизображениям //Параллельные вычислительные технологии (ПаВТ'2013): труды международной научной конференции (1–5 апреля 2013 г., г. Челябинск).Челябинск: Издательский центр ЮУрГУ, 2013. С. 624.