Вычисление функций критериев и ограничений в задаче оптимизации пространственного рычажного механизма с распараллеливанием на графическом процессоре*

Е.С. Городецкий, В.Е. Турлапов

Рассматривается задача оптимизации пространственного рычажного механизма. Функции критериев и ограничений описывают характеристики изменения расстояний и углов между звеньями, осями кинематических пар, а так же препятствиями в процессе движения механизма с заданными параметрами. Вычисление значений целевых функций требует решения трудоёмкой задачи о положениях механизма, синтезированного по параметрам. В статье предлагается алгоритм и структуры данных для эффективного распараллеливания вычисления положений механизма и целевых функций на графических картах. Предлагаемый подход позволяет значительно ускорить решение задачи оптимизации механизма за счёт и параллельного расчёта положений с переносом трудоёмких векторных вычислений на графическую карту.

1. Введение

Пространственные рычажные механизмы являются важной составляющей современной техники и производственных технологий [5]. Механические системы разрабатываются для выполнения достаточно серьёзных задач, требующих высокой надёжности и точности. При конструировании механизма обычно требуется многократно выполнять полный расчет его движения для постепенной оптимизации модели.

Наибольший интерес для конструктора представляют технологии и инструменты для оптимального моделирования механических систем. Тем не менее, такие возможности практически отсутствуют в современных общедоступных системах автоматизированного проектирования. Задача параметрической оптимизации модели рычажного механизма была формализована и описана автором [3]. Дальнейшая работа ведётся в направлении автоматизации решения этой задачи с распараллеливанием векторных вычислений на графических картах.

2. Задача вычисления функций оптимизации рычажных механизмов

2.1. Введение в задачу о положениях пространственного рычажного механизма

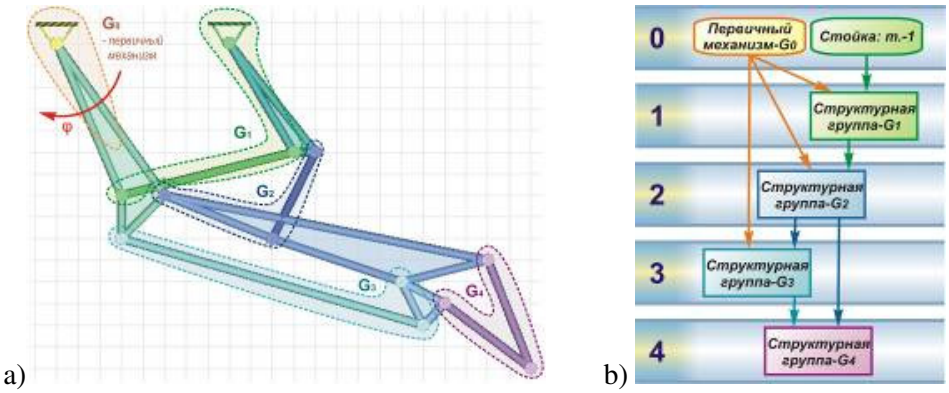
Под *механизмом* понимается совокупность взаимосвязанных твёрдых тел, предназначенная для преобразования движения входов на одном или нескольких твёрдых телах в движение на других твёрдых телах (выходах). Механизм может быть представлен своей структурной схемой, на которой выделяют *звенья* (твёрдые тела), соединяющиеся в подвижные *кинематические пары* с помощью различных типов *геометрических элементов* (вращательных, сферических, поступательных и др.). В основе методов конструирования механизмов и расчета кинематики лежит *принцип образования механизмов по Ассуру*, утверждающий что “Всякий механизм представляет собою совокупность одного или нескольких, *двухзвенных* (*первичных*) *механизмов* и одной или нескольких *групп нулевой подвижности* (*структурных групп*)” [5]. В работе рассматривается класс пространственных рычажных механизмов, которые могут быть образованы согласно принципу Ассура наслоением одноконтурных структурных групп.

Прямая задача о положении для механизмов ставится следующим образом. Задано движение всех входных звеньев, удовлетворяющее связям, которые накладывают кинематические пары, соответствующих первичных механизмов. Необходимо найти движение всех остальных

* Работа выполнена при финансовой поддержке Федерального агентства по науке и инновациям (ФЦП «Научные и научно-педагогические кадры инновационной России», госконтракт № 02.740.11.5018).

звеньев механизма, либо сообщить о невозможности реализации положения, в тот или иной момент времени.

Согласно закону Ассура прямая задача о положениях механизма сводится к последовательному решению ряда более простых подзадач расчета положений структурных групп механизма. На рис. 1.a приведён пример механизма выдвижения закрылков самолёта. Мханизм состоит из одного первичного механизма и 4 структурных групп BBB (из 3 Вращательных пар), связанных друг с другом. Каждая группа имеет два входа, необходимых для расчета её положения. Исходя из структурной схемы механизма, строится последовательность расчета структурных групп, показанная на рис. 1.b - вычисления производятся последовательно по уровням 0, 1, 2, и т.д., сверху вниз (стрелками показаны соединения групп друг с другом). Положения структурных групп BBB вычисляются аналитически. Произведя вычисления положений механизма для разных углов φ поворота входного звена, мы получим движение всех звеньев механизма.



a) b)

Рис. 1: Последовательность расчета структурных групп механизма

Важно отметить, что аналитический или численный расчет положения структурной группы механизма требует применения базовых векторных и матричных операций, а также широкого набора геометрических функций на их основе.

Моделирование структурной схемы механизма, разбиение механизма на группы и определение последовательности вычисления структурных групп реализовано в программной системе “Mechanics Studio .NET” [4]. Параллельный расчёт серии положений механизма реализованный на графическом процессоре [1], выполняется по данным от системы моделирования.

2.2. Постановка задачи оптимизации пространственных рычажных механизмов

Процесс проектирования машины начинается с формулировки задачи и определения условий её функционирования. *Выходными параметрами* механизма являются законы движения его звеньев, определяемые их размерами (длины L_1, L_2,.. L_N) и направлениями осей кинематических пар $\psi_i=\angle(u_i, ej)$ – *внутренними параметрами* и *внешними параметрами* – законами движения входных звеньев $\varphi_j(t)$ (функции изменения углов или перемещений входных звеньев). Каждому из перечисленных выходных параметров соответствует свой критерий оптимальности: точность воспроизведения заданной траектории, наилучшее прохождение препятствий, КПД, и т.п. [5]. Задача оптимизации механизма по внешним параметрам имеет смысл лишь в том случае, если механизм имеет более одной степени подвижности. *Синтез механизма* заключается в поиске оптимальной совокупности значений его внутренних параметров, при заданных внешних параметрах.

Чтобы обеспечить корректный, с точки зрения конструктора, синтез механизма, при постановке задач оптимизации вводят различные ограничения, определяющие область варьируемых параметров. *Параметрические ограничения* включают ограничения на длины звеньев ($L_i^{min} \le L_i \le L_i^{max}$) и на углы поворота осей кинематических пар u_i ($\psi_l^{min} \le \psi_l \le \psi_l^{max}$). Среди *функциональных ограничений* рассматривают:

- Ограничения на расстояния между точками и прямыми звеньев:
  $\forall t \in [t_0,t_n]\ \rho_{i,j}^{min} \le \rho(P_i(t), P_j(t)) \le \rho_{i,j}^{max}$ или $\rho_{i,j}^{min} \le \rho(P_i(t), L_j(t)) \le \rho_{i,j}^{max}$,
  где P_i – точка звена механизма, а L_j – прямая, соединяющая две точки звена.

- Ограничения на углы между звеньями:
  $\forall t \in [t_0,t_n]\ \gamma_{i,j}^{min} \le \gamma(L_i(t), L_j(t)) \le \gamma_{i,j}^{max}$
- Ограничения на не пересечение звеньев с пространственными препятствиями R_i:
  $\forall t \in [t_0,t_n],\ \rho(R_i, L_j(t)) \ge \delta_b,\ i=1..M,\ j=1..N$

Функциональные ограничения также могут становиться критериальными, в случае выбора в качестве критерия соответствующей функции.

Проблема оптимального синтеза механизма, решаемая конструктором, выражается в выборе целевого критерия оптимизации и варьируемых параметров. Рассмотрим некоторые типы целевых критериев.

Критерий воспроизведения заданных скалярных функций - задаёт целевую скалярную функцию $f^*(t)$, определяющую желаемое поведение некоторой *функции движения* механизма $f(t, \alpha)$ (например угла между звеньями или расстояния между точками звеньев). Необходимо за счёт выбора вектора варьируемых параметров $\alpha^* \in \Pi$, обеспечить максимальную близость характеристики f к целевой функции f^*, с учётом заданных ограничений, т.е.:

$$\Phi^I(\vec{\alpha}) = \sqrt{\sum_{i=0}^{n}\left(f(t_i,\vec{\alpha}) - f^*(t_i)\right)^2} \xrightarrow[\vec{\alpha}\in\Pi]{} \min,$$

$$\vec{\alpha} = (\alpha_1,..,\alpha_n) \in \Pi:\ \forall i \in \{0,..,n\}\ c_l \le g_l(t_i,\vec{\alpha}) \le d_l,\ l = 1,..,m$$

Критерий воспроизведения целевого движения, требует прохождения некоторой точки $P(t, \alpha)$ механизма через заданный набор положений $(P^*_1, P^*_2, \ldots, P^*_k)$ в определённые моменты времени $(t^*_1, t^*_2, \ldots, t^*_k)$. Т.е. за счёт выбора вектора варьируемых параметров $\alpha^* \in \Pi$, необходимо минимизировать наибольшую невязку между точками $P(t^*_i, \alpha^*)$ и P^*_i, при всех $i=1..k$, с учётом заданных ограничений:

$$\Phi^{II}(\vec{\alpha}) = \max_{i\in\{1..k\}}\left\|\vec{P}(t_i^*,\vec{\alpha}) - \vec{P}_i^*\right\| \xrightarrow[\vec{\alpha}\in\Pi]{} \min,$$

$$\vec{\alpha} = (\alpha_1,..,\alpha_n) \in \Pi:\ \forall i \in \{0,..,n\}\ c_l \le g_l(t_i,\vec{\alpha}) \le d_l,\ l = 1,..,m$$

2.3. Проблемы вычисления функций ограничений и критериев

Для эффективного решения описанных задач оптимизации механизмов важно выполнять быстрый расчет функций критериев и ограничений. Для вычисления каждого значения целевых функций требуется выполнить полный расчет N положений синтезированного рычажного механизма с заданными параметрами, что является довольно трудоёмкой задачей. Для ускорения вычисления целевых функций, можно выполнять расчет всех положений параллельно с последующим параллельным вычислением функций движения для каждого из положений. Чтобы реализовать такие параллельные расчеты эффективно, требуется использовать архитектуры с сотнями процессоров, по возможности поддерживающих векторные операции. Современные графические процессоры отвечают этим требованиям.

Задача данной работы заключается в разработке структур данных, алгоритмов и программы для графических процессоров, основанных на модели SIMD (Single Instruction Multiple Data), которая бы выполняла расчет положений механизма и вычисление необходимых функций движения для критериев и ограничений по заданным параметрам.

3. Вычисление функций движения и по параметрам механизма на графической карте

3.1. Технология программирования вычислений общего назначения

Для программирования вычислений общего назначения на графических процессорах на сегодняшний день существует несколько высокоуровневых языков и технологий управления памятью [1]: NVidia CUDA, ATI Stream OpenCL, Microsoft DirectX Compute Shader. Для вычисления положений и функций движения рычажных механизмов было решено использовать технологию DirectX 11 Compute Shader (CS) [7-8], т.к. она ориентирована на работу с графическими картами разных производителей (ATI и NVidia) и позволяет задействовать возможности векторных вычислений языка HLSL. На текущий момент все возможности технологии Compute

Shader реализованы только в графических картах серии ATI Radeon 5-го поколения и частично поддерживаются DirectX 10 совместимыми картами NVidia GeForce. В ближайшее время NVidia планирует выпустить графические карты Fermi с поддержкой технологии DirectCompute, а также отладчик Nexus для программ на CS и CUDA.

3.2. Структуры данных для расчёта серии положений механизма на GPU

Для расчёта положений механизма на графической карте, данные упаковываются в структуры, представленные на рис. 2, передаются на графическую карту и используются программой вычислительного шейдера:

1. CS_MECHANISM – константный буфер, хранящий размеры других буферов;
2. *LinksBuffer* – буфер звеньев с индексами их точек и осей;
3. *GroupsBuffer* – буфер групп с идентификатором типа, индексами точек и осей пар гуппы, знаками направления осей и индексами геометрических констант.
4. *GeomConstsBuffer* – буфер геометрических констант для вычисления положений.
5. *OriginalPositionBuffer* – буфер векторов точек и осей исходного положения мех-ма.
6. *IterationsBuffer* – буфер для записи векторов точек и осей итераций механизма (свёртка элементов типа CS_ITERATION в одномерный массив векторов);

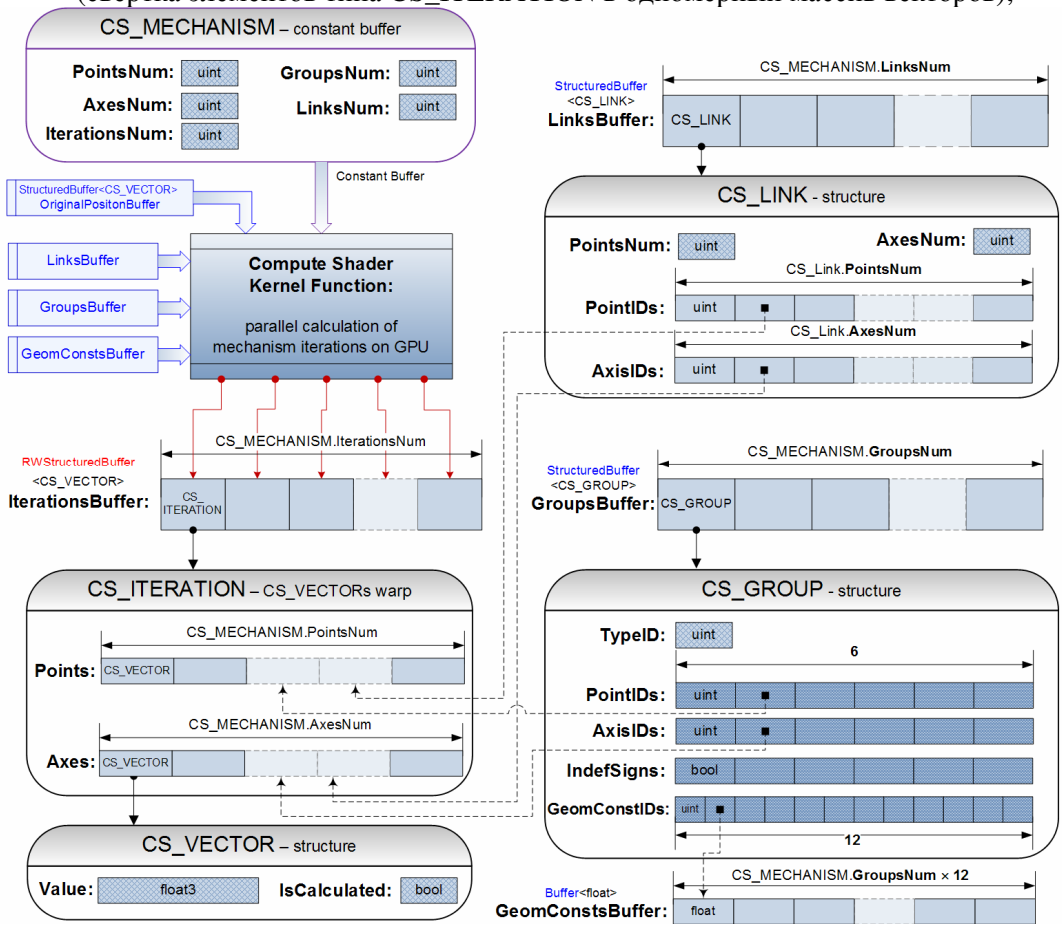


Рис. 2 Структуры данных для вычисления положений механизма на GPU

3.3. Параллельный расчёт серии положений механизма на GPU

Программа вычислительного шейдера расчета положений механизма исполняется на графической карте на потоках, количество которых равно *числу итераций* – *IterationsNum*, т.е. числу положений механизма. Каждый поток шейдера независимо вычисляет вектора точек и осей i-го положения механизма и записывает их в *IterationsBuffer*.

Алгоритм вычисления положений механизма описан блок-схемой на рис. 3. Алгоритм состоит в проходе по списку групп *GroupsBuffer*, последовательном решении задачи о положении для каждой структурной группы и расчёту положений тех звеньев из *LinksBuffer*, которые стало возможно посчитать по точкам и осям полученным из положений пар группы.

Расчёт положения структурной группы выполняется по определённым формулам для каждого типа групп [6], на основе данных из буфера *GroupsBuffer* и геометрических констант из *GeomConstsBuffer*. Расчёт положения звена выполняется путём преобразования его точек и осей из исходного положения *OriginalPositionBuffer*, путём умножения на матрицу поворота-переноса, полученную по изменению 3-х известных векторов звена в текущем положении: 3 точкам или 2 точкам и 1 оси или 1 точке и 2 неколлинеарным осям.

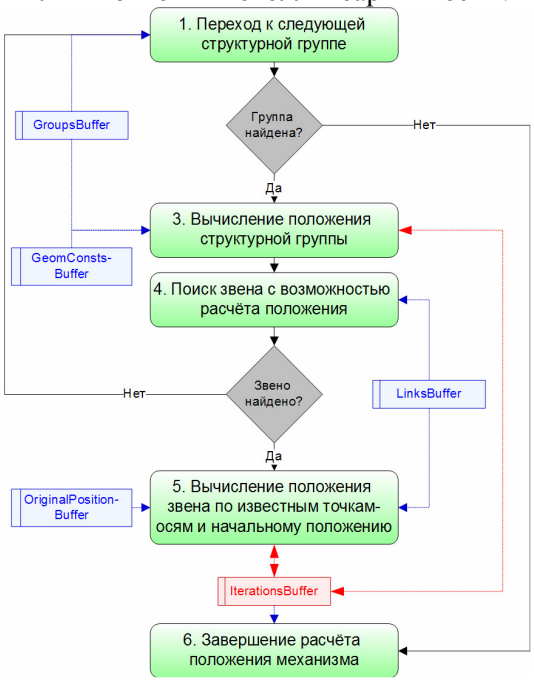


Рис. 3 Алгоритм вычисления положения рычажного механизма на GPU

3.4. Синтез механизма по параметрам и вычисление функций движения на GPU

Вернёмся к задаче вычисления функций критериев и ограничений для рычажного механизма с заданными параметрами. Общая схема вычислений с чтением и записью данных в буферы графической карты показана на рис. 4. При решении задачи оптимизации, структура механизма (буферы c1, b1, b2, b5, b7, b8), его исходное положение (b4), серии положений входных звеньев (b3), а также описания параметров (b6) и функций движения (b9, b10), являются неизменными данными и могут быть записаны в память графической карты один раз во время инициализации вычислений (шаг 0). Для вычисления критериев и ограничений на графическую карту нужно повторно передавать только новые значения параметров механизма (c2, шаг 1) и считывать полученные значения функций движения (a3, шаг 5). Пересчитываемые вектора положений механизма (a1) и геометрические константы (a2) записываются в буферы карты и используются как промежуточные данные для вычисления функций движения (шаг 5), но не загружаются в основную память. Такая организация программы позволяет значительно сократить дорогостоящие передачи данных между основной памятью и памятью графической карты.

Для синтеза механизма с заданными параметрами (длин звеньев и углов поворота осей) и расчета его положений, выполняются две подготовительные процедуры:

1. Синтез положений входных звеньев механизма и запись их в буфер итераций, путём масштабирования исходных точек входных звеньев и поворота их исходных осей по соответствующим параметрам (шаг 2);
2. Обновление геометрических констант структурных групп, соответствующих параметризованным звеньям и осям: изменение длин отрезков звена и пересчёт косинусов постоянных углов в группе (шаг 3).

Вычисление положений синтезируемого механизма с новыми параметрами выполняется в соответствии с описанной на рис. 3 схемой, но с одним отличием на шаге 5, при вычислении положения звена. В дополнение к повороту и переносу исходного звена необходимо выполнить масштабирование его точек в соответствие с соответствующим параметром длины и повернуть его оси в соответствии с параметрами углов изменения их направлений.

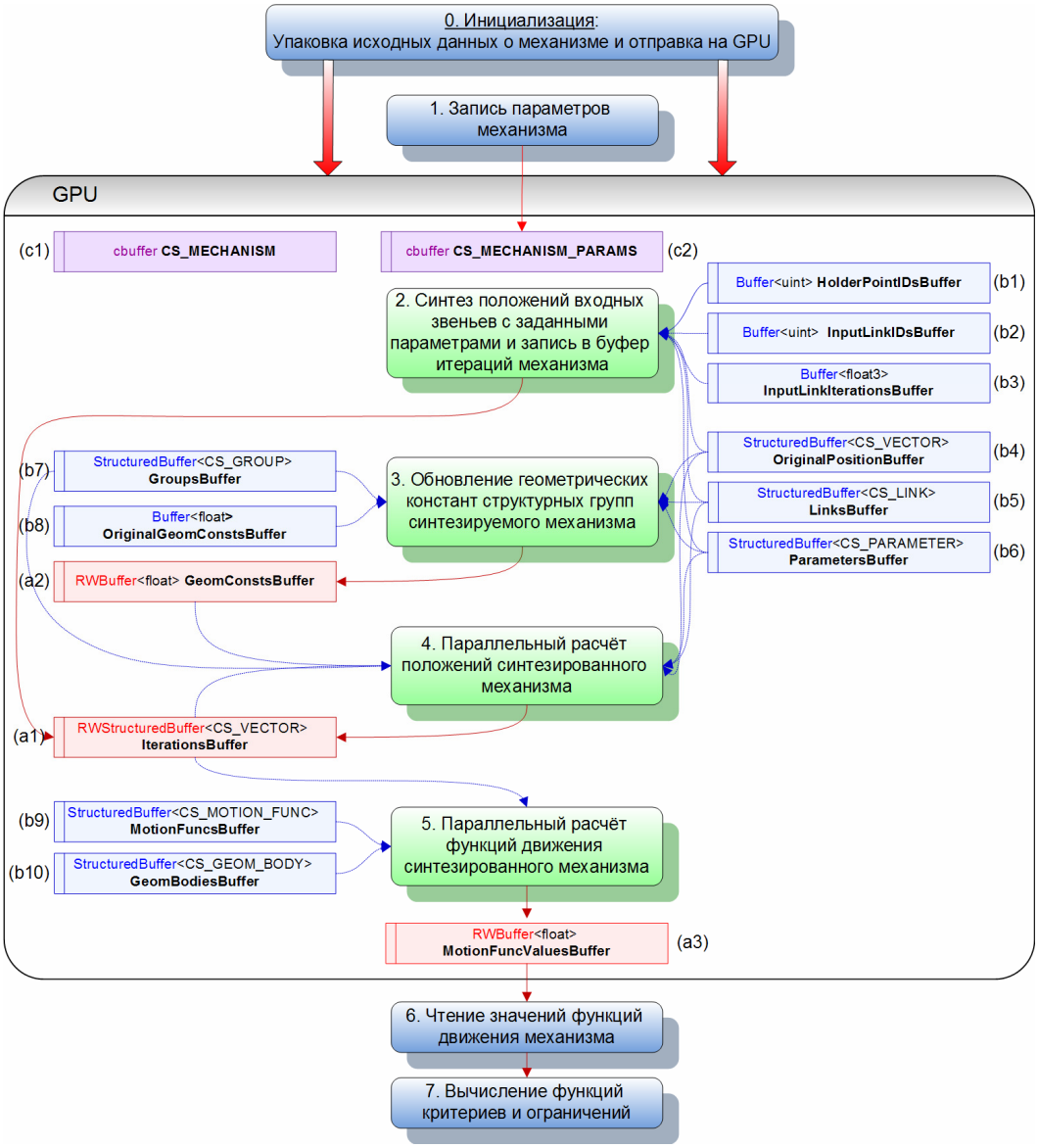


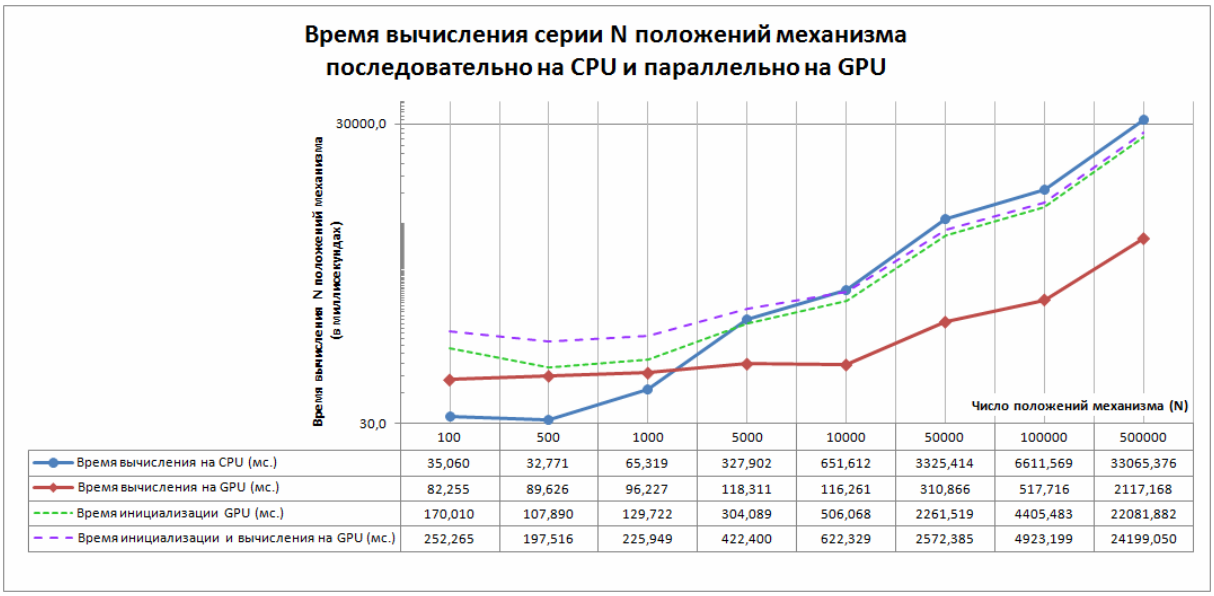
Рис. 4 Схема вычисления функций движения механизма на графической карте

Расчёт функций движения механизма выполняется независимо и параллельно для каждого положения механизма по буферу с описаниями функций расстояний и углов (b9) и буферу с описаниями ограничивающих геометрических тел: цилиндров и параллелепипедов (b10). Значения функций движения записываются в буфер (a3) и передаются в основную память CPU для подстановки в функции критериев и ограничений.

3.4. Результаты вычислительного эксперимента

Для подтверждения эффективности предложенного подхода был проведён эксперимент по сравнению времени вычисления серии из N положений пространственного механизма (на примере механизма закрылков, см. рис. 1.а) на однопоточной реализации алгоритма для центрального процессора и многопоточной SIMD-реализации для графического процессора. Далее приведена подробная информация об условиях проведения данного эксперимента и его результаты на графике (рис. 5):

- Реализации алгоритма вычисления положений:
  - **Однопоточная реализация для CPU**: язык C++/CLI, компилятор Microsoft VC 9.0, режим mixed-mode, release для платформ x86 и .NET 2.0;
  - **Многопоточная реализация для GPU**: язык HLSL 5, компилятор Microsoft HLSL Shader Compiler 9.27, режим компиляции CS 5.0, optimized.
- Аппаратное обеспечение:
  - **CPU**: Intel Core 2 Quad Q6600 (Kentsfield), 4 ядра @ 2400 MHz, L1-кэш 32 Kb x 4, L2-кэш 4Mb x 2, оперативная память DDR2 2 Gb @ 400 Mhz;
  - **GPU**: ATI Radeon HD 5850 (Cypress), 1440 потоковых ядра @ 725 Mhz, ширина шины 256 бит, 5 модель шейдеров, видео память GDDR5 @ 1000 Mhz;
- Программное обеспечение:
  - **ОС**: Microsoft Windows 7 Ultimate
  - **Драйвер GPU**: atiumdag 8.14.10.0716 (ATI Catalyst 9.12)
  - **Исполняющая среда**: .NET Framework 3.5 SP1
  - **Библиотеки**: DirectX February 2010
  - **Среда разработки**: Microsoft Visual Studio Team Suite 2008



	100	500	1000	5000	10000	50000	100000	500000
Время вычисления на CPU (мс.)	35,060	32,771	65,319	327,902	651,612	3325,414	6611,569	33065,376
Время вычисления на GPU (мс.)	82,255	89,626	96,227	118,311	116,261	310,866	517,716	2117,168
Время инициализации GPU (мс.)	170,010	107,890	129,722	304,089	506,068	2261,519	4405,483	22081,882
Время инициализации и вычисления на GPU (мс.)	252,265	197,516	225,949	422,400	622,329	2572,385	4923,199	24199,050

Рис. 5: График сравнения времени вычисления серии N положений механизма для последовательной реализации на CPU и параллельной реализации на GPU при разных N

Результаты эксперимента демонстрируют 10-ти кратное ускорение вычислений на графической карте, по сравнению с однопоточными вычислениями на центральном процессоре, при больших количествах вычисляемых положений (более 10000). Необходимо отметить, что учитывая значительные временные затраты на инициализацию и передачу данных в память графической карты и чтение результатов из неё, преимущество использования графического процессора теряется. Это говорит о необходимости минимизировать объём передаваемых данных при реализации вычислений целевых функций в задаче оптимизации.

Заключение

Описанные структуры данных и схема вычисления функций критериев и ограничений позволяют ускорить решение задачи оптимизации кинематики пространственного рычажного механизма за счёт распараллеливания расчётов на графической карте. На данный момент полностью реализована и испытана программа параллельного вычисления положений [1], написанная с помощью DirectX 11 Compute Shader, подтверждающая эффективность предложенного подхода. В дальнейшем планируется продолжить её развитие с добавлением возможностей параметрического синтеза механизма и параллельного расчёта функций движения на основе описанных принципов.

Литература

1. Городецкий Е.С. Реализация параллельного вычисления серии положений пространственного рычажного механизма на графическом процессоре с помощью DirectX 11 Compute Shader // Высокопроизводительные параллельные вычисления на кластерных системах: Материалы девятой международной конференции-семинара. Владимир: Изд-во Владимирского государственного университета, 2009г. - стр. 131-135;
2. Городецкий Е.С. Алгоритм параллельного вычисления серии положений пространственного рычажного механизма // Технологии Microsoft в теории и практике программирования: Материалы конференции. Н. Новгород: Изд-во Нижегородского госуниверситета, 2009г. - стр. 102-108;
3. Городецкий Е.С. Задача параметрической оптимизации пространственных механизмов с параллельными вычислениями кинематики на графической карте // Высокопроизводительные параллельные вычисления на кластерных системах: Материалы седьмой международной конференции-семинара. Н.Новгород: Изд-во Нижегородского госуниверситета, 2007г. – стр. 117-124;
4. Городецкий Е.С. Система проектирования кинематики пространственных механизмов Mechanics Studio .NET на основе Managed DirectX // Технологии Microsoft в теории и практике программирования: Материалы конференции. Н. Новгород: Изд-во Нижегородского госуниверситета, 2007г. – стр. 48-52;
5. Заблонский К.И., Белоконев И.М., Щекин Б.М. “Теория механизмов и машин: учеб. для вузов” - К.: Выща шк. Головное изд-во, 1989г.- 376с.
6. Турлапов В.Е. “Задача о положениях и классификация одноконтурных структурных групп пространственных рычажных механизмов” // Электронный ж. "Прикладная геометрия". Вып.2. №2. МАИ. 2000г. С.1-22;
7. Nick Thibieroz (AMD) Shader Model 5.0 and Compute Shader: GDC Conference materials, 2009 [www.gdconf.com/conference/Tutorial Handouts/100_Advanced Visual Effects with Direct3D/100_Handout 6.pdf]
8. Microsoft DirectX Software Development Kit (February 2010) [msdn.microsoft.com/en-us/directx/default.aspx]