

УДК 004.274

## Исследование графической системы с вертикальной архитектурой для поиска пересечения луча с группой полигональных объектов

Р.В. Мальчева, Т.В. Завадская, В.И. Кудояр  
Донецкий национальный технический университет  
raisa\_malcheva@donntu.org

*Мальчева Р.В., Завадская Т.В., Кудояр В.И. Исследование графической системы с вертикальной архитектурой для поиска пересечения луча с группой полигональных объектов. Выполнен анализ архитектурной организации графических систем. Предложена модификация алгоритма поиска пересечения луча с графической сценой. Выполнено проектирование системы с вертикальной архитектурой с использованием UML. Проведены исследования параметров системы поиска пересечения луча с полигональными объектами.*

**Ключевые слова:** трассировка лучей, архитектура, параллельный поиск, проектирование, UML, исследование.

### Введение

Основным недостатком метода трассировки лучей является его трудоемкость. До недавнего времени существовали только системы с программной реализацией данного метода. Однако в последнее 10-летие усилилось внимание и к аппаратной реализации трассировки, в том числе и на FPGA [1, 2]. В полной мере особенностям трассировки лучей соответствуют массивно параллельные вычислительные способности графических процессоров NVIDIA, позволяя существенно увеличить скорость визуализации в многочисленных применениях. NVIDIA поддерживает компании в расширении применения GPU-вычислений на производстве и принимает во внимание их потребности при разработке своих будущих архитектур GPU, языков и инструментов [3, 4].

### Постановка задачи

Метод обратной трассировки лучей предполагает выполнение поиска пересечения луча, проведенного из глаза наблюдателя через каждый элемент окна, соответствующий пикселю экрана, с объектами сцены. Поэтому традиционно архитектура системы, основанной на этом методе, реализует принцип распараллеливания «от выходных данных» и по классификации [5] относится, в идеале, к типу «процессор на пиксель», а фактически - к «процессор на фрагмент изображения» [6, 7], хорошо реализуемый на кластерах [8].

Цель выполненных исследований – разработка системы с реализацией параллельного поиска пересечения трассируемого луча с

несколькими полигональными объектами, а также установление зависимости времени выполнения данного этапа алгоритма от числа параллельных процессов и сложности объектов. При разработке применено многоуровневое проектирование с использованием UML [9].

### Реализация блока поиска пересечения луча с полигональным объектом

На вход блока CR подаются параметры луча, заданные в системе координат объекта:  $X_v_o, Y_v_o, Z_v_o$  - координата начала луча;  $V_x_o, V_y_o, V_z_o$  – вектор направления луча. Объект аппроксимирован плоскими выпуклыми гранями, каждая из которых описывается массивом координат вершин  $\{x[n], y[n], z[n]\}$  и нормальным вектором  $N=\{n_x, n_y, n_z\}$ . Для анализа используется только 1-я по списку вершина каждой грани. В результате блок формирует признак наличия пересечения (cross), и, если пересечение есть, то номер грани, в которую входит луч (gin) и значение параметра  $t_0$  (рис.1).

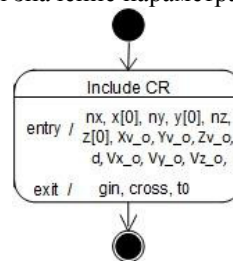


Рисунок 1 – Представление блока CR

Параметр  $t_0$  определяет пересечение луча с многогранником и равен отношению числителя ( $dev$ ) к знаменателю ( $div$ ), взятому с обратным

знаком. Числитель есть подстановка точки начала луча в уравнение плоскости грани (1).

$$\text{dev} = n_x \cdot x_v^o + n_y \cdot y_v^o + n_z \cdot z_v^o + d. \quad (1)$$

При этом свободный член уравнения плоскости,  $d$ , определяется из выражения (2) подстановкой координат любой вершины грани в уравнение.

$$d = -(n_x \cdot x[0] + n_y \cdot y[0] + n_z \cdot z[0]). \quad (2)$$

Знаменатель представляет собой скалярное произведение вектора луча на нормальный вектор грани (выражение 3).

$$\text{div} = S = v_x^o \cdot n_x + v_y^o \cdot n_y + v_z^o \cdot n_z. \quad (3)$$

Все три выражения можно рассчитывать параллельно, если выражение (1) заменить на выражения (4, 5).

$$\text{dev}_0 = n_x \cdot x_v^o + n_y \cdot y_v^o + n_z \cdot z_v^o; \quad (4)$$

$$\text{dev} = \text{dev}_0 + d. \quad (5)$$

UML-диаграмма блока CR поиска параметра пересечения луча с полигональным объектом приведена на рис.2.

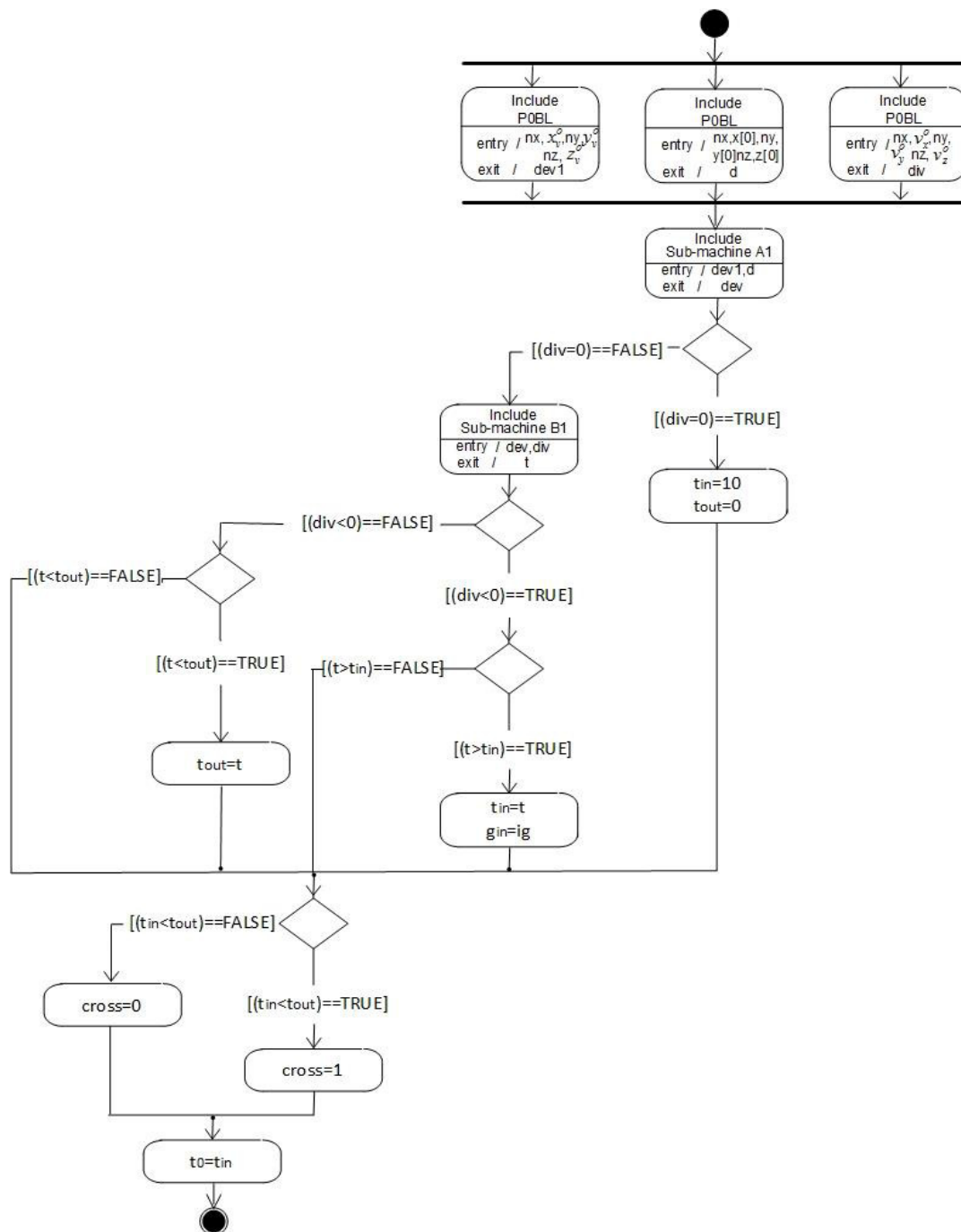


Рисунок 2 - UML-диаграмма блока CR поиска параметра пересечения луча с полигональным объектом

Очевидно, что реализация каждого выражения требует параллельного выполнения трех умножений (устройство M1) и затем двух последовательных сложений (устройство A1) [10]. Соответствующий блок назван P0BL, его диаграмма на примере вычисления скалярного произведения приведена на рис. 3.

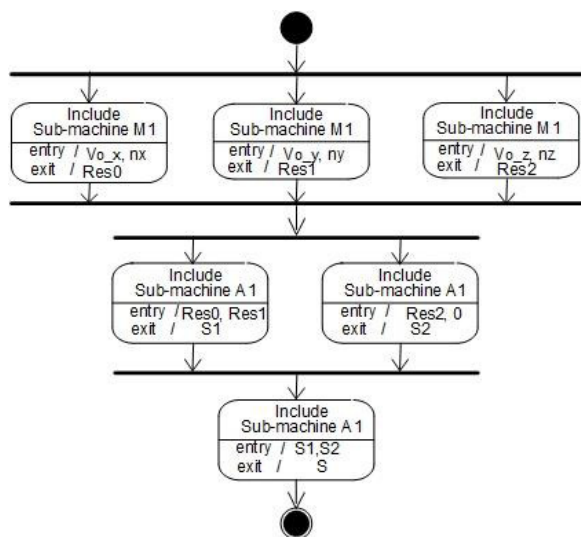


Рисунок 3 - UML-диаграмма блока вычисления скалярного произведения (P0BL)

Анализ наличия пересечения луча с объектом сводится к анализу  $t_{in} < t_{out}$ , т.е. расстояние до точки входа меньше, чем до точки выхода. При этом  $t_{in}$  выбирается между параметрами  $t$  видимых плоскостей ( $div < 0$ ), а  $t_{out}$  - между параметрами невидимых.

### Реализация параллельного поиска пересечения луча с группой объектов

Т.к. расчет выражений 1-5 выполняется в системе координат каждого объекта, то кроме блока CR предварительно необходимо выполнить расчет матрицы преобразования из глобальной системы и умножить на эту матрицу параметры луча. Расчет коэффициентов матрицы выполняется блоком MTR2 (рис. 4) по угловым координатам  $\{\psi, \Theta, \gamma\}$  вектора  $P_{ok}$  положения каждого объекта относительно глобальной системы координат. Входными параметрами параллельных вычислительных устройств являются синусы и косинусы углов, получаемые из расположенных в постоянной памяти таблиц:

$$\begin{aligned}
 sp &= \sin \psi; & cp &= \cos \psi; \\
 st &= \sin \Theta; & ct &= \cos \Theta; \\
 sg &= \sin \gamma; & cg &= \cos \gamma.
 \end{aligned}
 \tag{6}$$

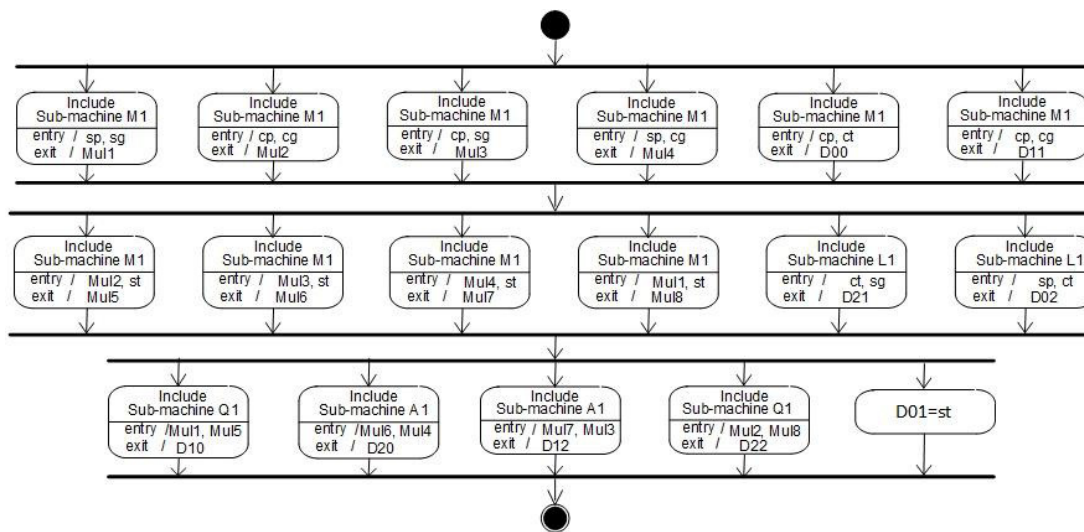


Рисунок 4 – UML-диаграмма блока MTR2 расчета матриц преобразований из глобальной системы координат в систему координат объекта

Далее блоком CC выполняется умножение координат начала луча, а блоком CCV1 – умножение параметров вектора направления луча на матрицу  $D[3][3]$ . Блоки CC и CCV1 включают в себя по 3 параллельных блока, аналогичных P0BL [10]. Обозначим такой объединенный блок TO.

UML-диаграмма блока CROSS приведена на рис. 5. В ней используются следующие обозначения:

- hit – число пересечений луча с объектами,
- gin – номер грани в точке входа луча,

$i_{ob}$  – индекс объекта  
 $K_{gr}$  – количество групп объектов, вычисляется автоматом деления  $D1$  по выражению (7):

$$K_{gr} = K_{ob} / kpr, \tag{7}$$

- igr – номер группы объектов, обрабатываемых параллельно,
- kpr – количество параллельно анализируемых объектов,
- ipr – номер объекта в группе.

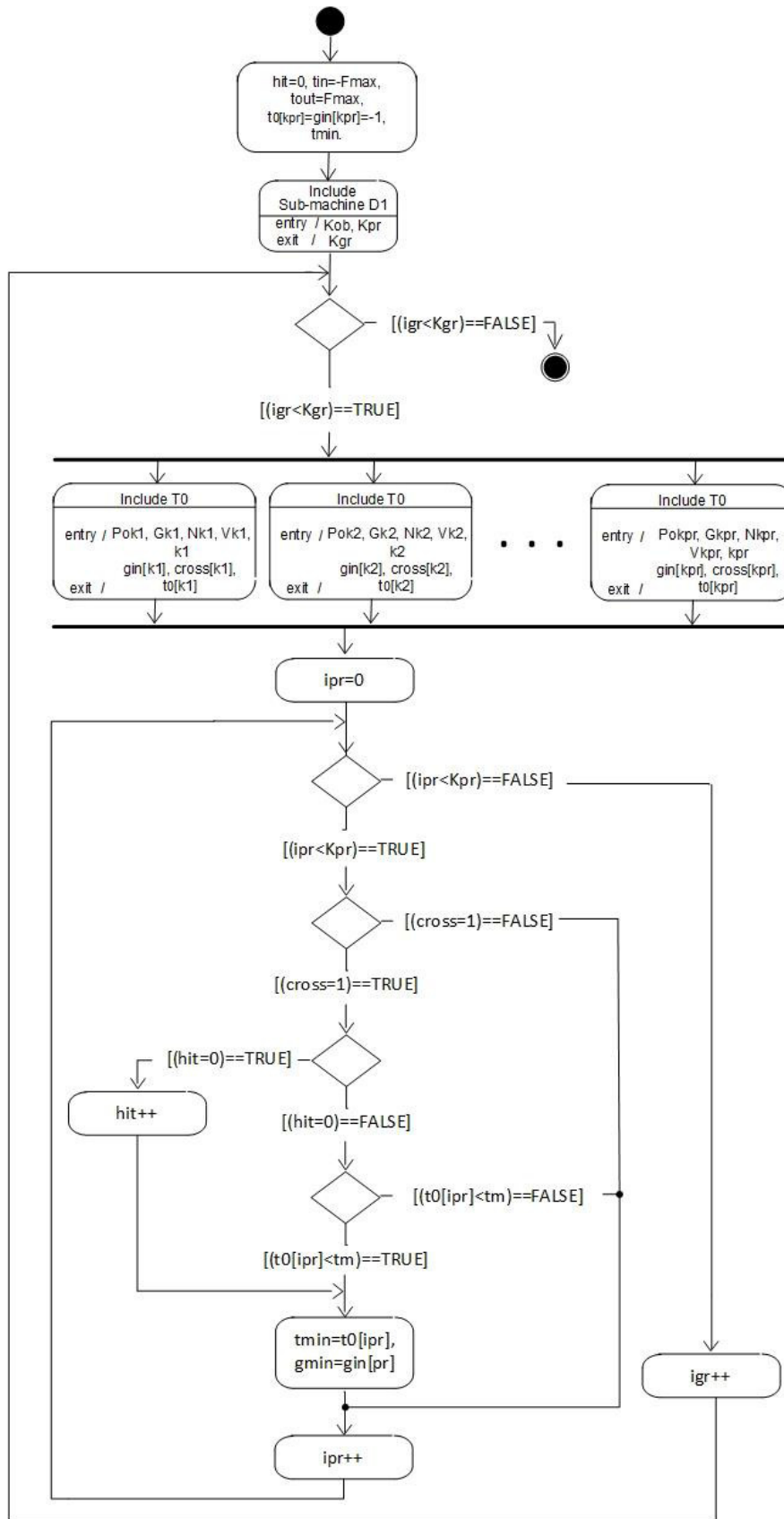


Рисунок 5 – UML-диаграмма блока CROSS поиска пересечения луча с объектами кадра

Если луч пересекает более одного объекта, то в качестве точки входа луча выбирается

минимальное значение из  $t_0$  объектов группы. В поиске минимума участвует значение, полученное для предыдущих групп объектов.

### Исследование системы

Анализ UML-диаграммы блока CROSS показывает, что система поиска пересечения луча с группой полигональных объектов имеет вертикальную архитектуру [5], в ней происходит распараллеливание «от входных данных».

Для поведения исследований на базе UML-диаграмм разработаны модели устройств с использованием языка Verilog [9, 11]. В качестве входных данных использованы базы данных по 5 миллионов полигональных объектов.

В исследованиях, результаты которых приведены на рис. 6, варьируется количество объектов в группе (соответственно, число параллельных блоков  $T_0$  kpr, заданное по оси абсцисс). По оси ординат приведено полученное время трассировки сцены,  $T$ , машинных циклов.

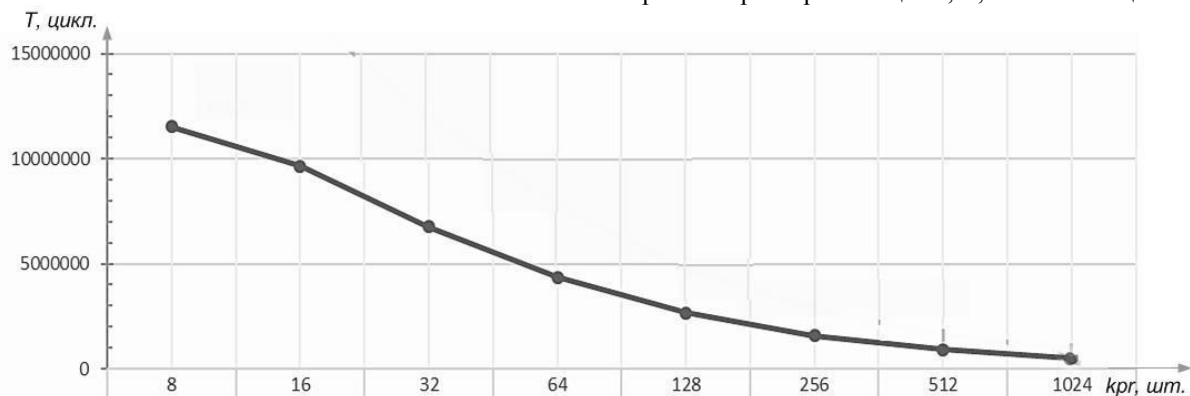


Рисунок 6 – Зависимость времени трассировки сцены от количества параллельных блоков  $T_0$

В исследованиях, результаты которых приведены на рис. 7, время трассировки,  $T$ , задано в миллионах машинных циклов. При этом кроме количества объектов в группе варьируется

сложность полигональных объектов. Число граней принято равным 4, 8, 12, 24 и 36 – им соответствующие кривые, обозначенные  $C_4 - C_{36}$ .

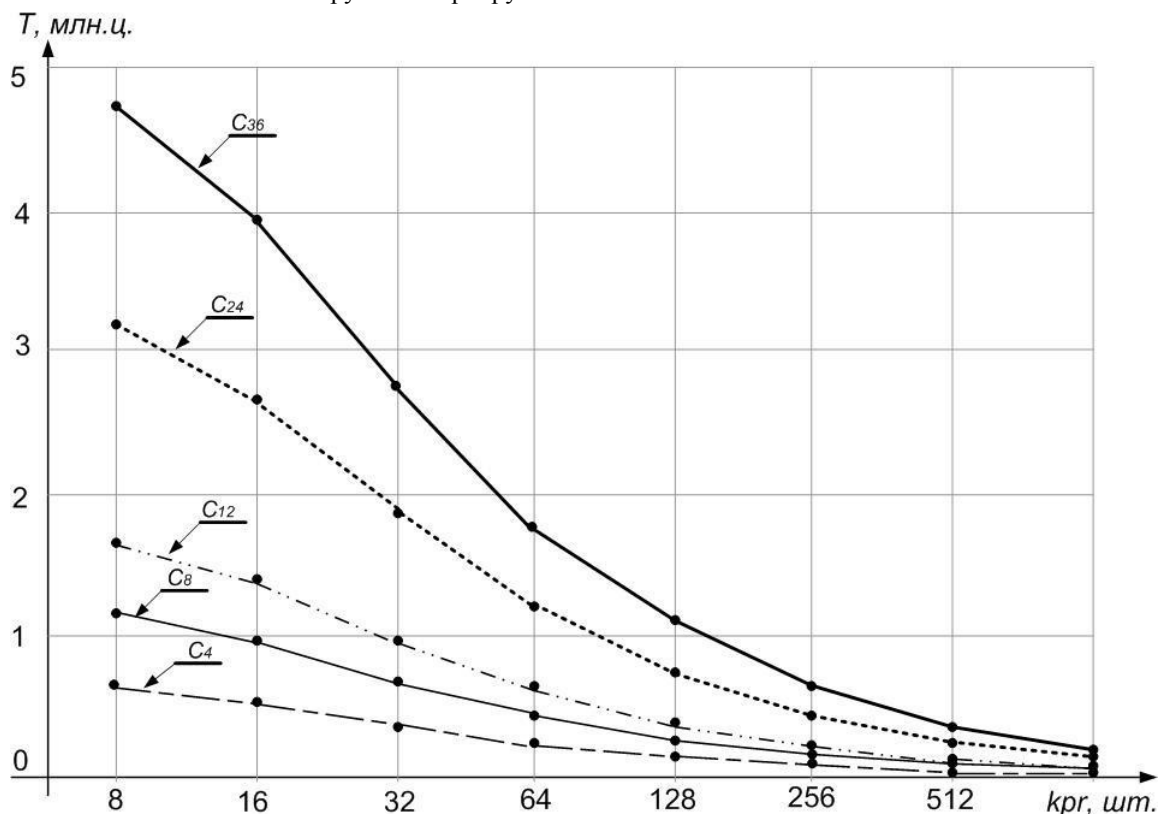


Рисунок 7 – Зависимость времени трассировки сцены от количества параллельных блоков  $T_0$  и сложности объектов

## Заклучение

Анализ зависимостей, приведенных на рис. 6 и рис. 7, показывает, что увеличение количества параллельных блоков T0 в 2 раза уменьшает время трассировки менее, чем в 2 раза. Причем, с увеличением количества объектов в группе выигрыш в производительности становится все менее заметен. Это связано с дополнительными затратами на распределение объектов по процессам вертикального конвейера. Более 512 вычислительных процессов применять не имеет смысла, что вполне соответствует размерам матриц технологии CUDA.

## Список использованной литературы

1. Schmittler J. Realtime Ray Tracing of Dynamic Scenes on an FPGA Chip // Computer Science, Saarland University, 2004. - PP. 8.
2. Malcheva R. An Acceleration of FPGA-based Ray Tracer / R. Malcheva, M. Yunis // European Scientific Journal, 2014. - Vol.10, N7. - PP. 186-190.
3. Компании, использующие GPU для трассировки лучей [Электронный ресурс] – Режим доступа: <http://www.nvidia.ru/object/gpu-ray-tracing-ru.html>
4. NVIDIA обеспечивает широкую доступность физически корректного рендеринга [Электронный ресурс] – Режим доступа: <http://www.nvidia.ru/object/blog-nvidia-pbr-gtc-ru.html>
5. Computer Graphics: Principles and Practice, Third Edition / John F. Hughes, Andries van Dam, Morgan McGuire, David Sklar, James D. Foley. - Addison-Wesley, 2014. – 1209 p.
6. Bashkov E.A. Synthesis of an image of enclosing circumstances in view of a physical condition of an atmosphere / E.A. Bashkov. – Electronic simulation, 1996. –Vol.18. - N3. - PP. 45-51.
7. Bashkov E.A. Accelerating Search for Ray-Object Intersection Point in the Ray Tracing Method. - Engineering Simulation, 1996. - Vol.13. - N3. - PP. 369 - 380.
8. Мальчева Р.В. Реализация модифицированного алгоритма трассировки лучей на кластере NeClus / Р.В. Мальчева, М. Юнис // Наукові праці Донецького національного технічного університету. Серія: «Обчислювальна техніка та автоматизація». – Донецьк: ДонНТУ, 2013. – №1(24) 2013. - С. 263-268.
9. Malcheva R. Application of multilevel design on the base of UML for digital system developing // in book “Design of Digital Systems and Devices. Series: Lecture Notes in Electrical Engineering. – Springer-Verlag Berlin Heidelberg, 2011. – Vol. 79. - PP. 93-117.
10. Попова А.Н. Разработка устройства поиска пересечения луча с полигональным объектом / А.Н. Попова, А.В. Селинова, Р.В. Мальчева // «Компьютерная и программная инженерия. Сборник материалов международной научно-практической конференции студентов, аспирантов и молодых ученых» (г. Донецк, ДонНТУ, 2015 г.) – С. 242-247.
11. Хаханов В.И. Проектирование и верификация цифровых систем на кристаллах. Verilog & System Verilog / В.И. Хаханов, И.В. Хаханова, Е.И. Литвинова, О.А. Гузь. – Харьков: ХНУРЭ, 2010. – 528 с.

*Malcheva R.V., Zavadskaya T.V., Kudoyar V.I. An investigation of a graphical system with a vertical architecture for searching for intersection of a ray with a group of polygonal objects. The analysis of the architectural organization of graphic systems is carried out. A modification of the search algorithm for the intersection of a ray with a graphic scene is proposed. A system with a vertical architecture is designed using UML. The parameters of a system for searching for intersection of a ray with a group of polygonal objects are investigated.*

**Keywords:** ray tracing, architecture, parallel search, design, UML, investigation.

*Мальчева Р.В., Завадська Т.В., Кудояр В.І. Дослідження графічної системи з вертикальною архітектурою для пошуку перетину промінню з групою полігональних об'єктів. Виконаний аналіз архітектурної організації графічних систем. Запропонована модифікація алгоритму пошуку перетину промінню з графічною сценою. Виконано проектування системи з вертикальною архітектурою з використанням UML. Проведені дослідження параметрів системи пошуку перетину промінню з полігональними об'єктами.*

**Ключові слова:** трасування промінів, архітектура, паралельний пошук, проектування, UML, дослідження.

Статья поступила в редакцию 20.09.2017

Рекомендована к публикации д-ром техн. наук В.Н. Павлышом