УДК 004.922

ИСПОЛЬЗОВАНИЕ 3D РЕПРОЕКЦИИ ДЛЯ ОПТИМИЗАЦИИ ГЕНЕРАЦИИ НОВЫХ КАДРОВ

Романовский М.Д. [1], студент гр.353501, Могилевец Д.Э. [2], студент гр.353505

Белорусский государственный университет информатики и радиоэлектроники[1]
г. Минск, Республика Беларусь

Примичева З.Н. – канд. физ.-мат. наук

Аннотация. В данной работе рассматривается алгоритм генерации промежуточных кадров в пространстве при изменении положения камеры, требующий меньшее количество вычислительных операций по сравнению с полным рендером сцены с нуля. Предложенный алгоритм основан на использовании информации, содержащейся в карте глубины сцены после рендера, а также на возможности восстановления с помощью этой информации перспективной координаты.

Ключевые слова. Рендер изображений, промежуточные кадры, матрицы преобразований

Введение

Рендеринг 3D изображений — процесс создания кадров (изображений) на основе трехмерных объектов. Данная технология применяется в различных приложениях компьютерной графики, виртуальной и дополненной реальности, а также в компьютерных играх. Все объекты при 3D-рендеринге разбиты на небольшие треугольники, вершины которых имеют три координаты. Эти треугольники сначала переносятся из пространства относительно объекта в пространство относительно камеры, затем проецируются на двухмерную поверхность, а потом закрашиваются в соответствии с различными параметрами такими, как текстура объекта или освещение.

В основном сцены содержат большое количество различных объектов, и, следовательно, треугольников, на которые они разбиты. 3D-рендеринг является крайне требовательной с вычислительной точки зрения операцией. Если пользователь меняет угол или положение камеры, то требуется время для того, чтобы создать новый кадр, в котором это изменение заметно. Наиболее ощутимой эта проблема становится в технологиях виртуальной и дополненной реальности, при использовании которых пользователь часто сдвигает голову, хотя такие сдвиги и не значительны, чтобы создавать новые кадры достаточно быстро. Исследования показывают, что в случае снижения частоты кадров ниже 60 в секунду, пользователь испытывают дезориентацию и головокачание [1].

Генерация новых кадров с использованием 3D репроекции

Вместо того, чтобы создавать новый кадр с нуля, можно использовать предыдущие кадры для создания промежуточных кадров с новой позицией камеры, повторив преобразования в обратном порядке. Такой кадр не будет полностью совпадать с настоящим (созданным с нуля) кадром, однако он будет предоставлять достаточно похожую иллюзию для того, чтобы пользователю казалось, что камера движется максимально плавно и без задержек. Так как отсутствует необходимость заново вычислять цвет точек и сами преобразования достаточно просты, создание такого кадра происходит значительно быстрее, чем рендеринг сцены с нуля. Предлагаемый алгоритм выглядит следующим образом:

1. Отрендерить кадр классическим способом, дополнительно получая при этом карту глубин.
2. Представить каждый пиксель как точку в пространстве относительно исходного положения камеры, используя для этого координаты пикселя на изображении и значение глубины.
3. Преобразовать точки из пространства изначальной камеры в пространство новой камеры.
4. Спроецировать эти точки на плоскость, создавая тем самым новый кадр.

Рассмотрим, как происходит преобразование координат вершин треугольника относительно объекта в координаты, которые будут использоваться при проецировании на плоскость:

1. К трем координатам вершин треугольника относительно объекта вводится четвертая, которая равна единице и будет использована для создания эффекта перспективы.
2. Отображение из пространства 3D объекта в пространство мира производится посредством умножения на матрицу преобразования 4x4 (при этом четвертая компонента вектора координат равна 1).
3. Поскольку простое проецирование треугольника на плоскость создает ортогональное изображение, в большинстве приложений требуется перспективное. Для этого производится проективное преобразование, изменяющее координаты объекта в зависимости от его удалённости от камеры путём умножения на матрицу преобразования перспективы (при этом четвёртая координата изменяется аналогично).

4. Совершается деление первых трёх координат всех векторов на четвёртую координату. Первые две полученные координаты используются как координаты пикселя на экране, а третья заносится в карту глубины для определения перекрытия рассматриваемых объектов другими объектами, находящимися близи от них.

После выполнения описанных выше операций создаётся двумерное изображение, а также двумерная карта глубины, хранящая информацию о расстоянии до каждой из точек, полученных в результате преобразований, которое можно считать длиной отрезка расстояния от камеры до точки в пространстве камеры и видеть на рисунке 1.

Рисунок 1 – Пример изображения (слева) и карты глубин (справа)

Поскольку такой алгоритм достаточно прост и будет выполняться ровно столько раз, сколько пикселей есть в кадре, он имеет более высокую производительность, чем традиционный подход, в котором каждый пиксель на финальном кадре зачастую обрабатывается несколько раз, и является значительно менее затратным с вычислительной точки зрения методом.

Для реализации предлагаемого алгоритма необходимо более подробно рассмотреть используемые в нём преобразования. Во-первых, матрица перспективного преобразования может быть записана [2] в следующем виде:

$$P = \left(\frac{n}{r}\ 0\ 0\ 0\ 0\ \frac{n}{t}\ 0\ 0\ 0\ 0\ -\frac{f+n}{f-n}\ -\frac{2fn}{f-n}\ 0\ 0\ -1\ 0\right),$$

в котором n координата по оси z ближайшей плоскости, ограничивающей область видимости, f — дальней, t — ближней границы плоскости, ограничивающей область видимости сверху, r — справа. Будем полагать, что область видимости симметрична, то есть левая и правая, верхняя и нижняя границы равноудалены от центра, как показано на рисунке 2.

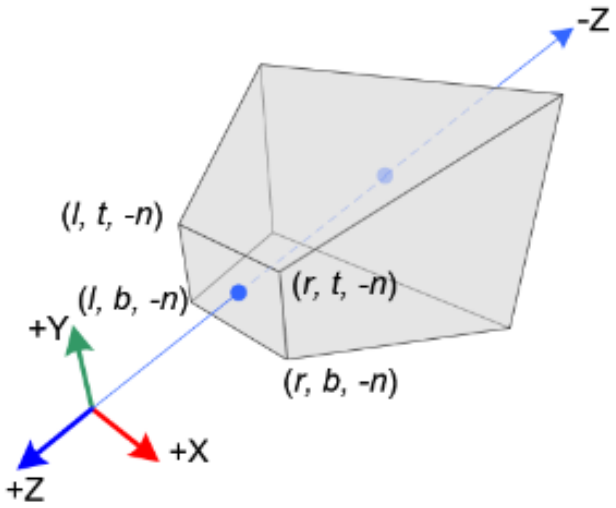

Рисунок 2 – Область видимости (стоит считать $l = r, t = b$)

До перспективного преобразования четвёртая координата каждой точки остаётся равной 1, то есть координаты произвольной точки имеют вид:

$$(x\ y\ z\ 1).$$

Тогда после перспективного преобразования вектор координат точки будет иметь вид

$$(P_{11}x\ P_{22}y\ P_{33}z + P_{34}\ P_{43}z\ ),$$

где P_{ij} — элемент матрицы P, стоящий в i -ой строке и в j-ом столбце.

Отсюда конечные координаты точки, как $x^{'}, y^{'}, z^{'}$, получаются из координат точки после перспективного преобразования путём деления первых трёх компонент на четвёртую, то есть:

$$x^{'} = \frac{P_{11}x}{P_{43}z},$$

$$y^{'} = \frac{P_{22}y}{P_{43}z},$$

$$z^{'} = \frac{P_{33}z+P_{34}}{P_{43}z}.$$

Заметим, что координаты $x^{'}, y^{'}$ используются как координаты пикселя на экране, а значение $z^{'}$ сохраняется в карту глубины.

Выразим значение z через значение $z^{'}$:

$$P_{43}zz^{'} = P_{33}z + P_{34},$$
$$z = \frac{P_{34}}{P_{43}z' - P_{33}}. \quad (1)$$

Значения x и y могут быть выражены через значения $x^{'}$ и $y^{'}$ как:

$$x = \frac{P_{43}z}{P_{11}}x',$$
$$y = \frac{P_{43}z}{P_{22}}y',$$

значение z совпадает с вычисленным по формуле (1).НЕ ВИДНО ФОРМУЛЫ

Поэтому прообраз точки

$$\left(x^{'}\ y^{'}\ z^{'}\right)$$

до применения проективного преобразования P может быть записан в виде:

$$\left(\frac{P_{43}z}{P_{11}}x^{'}\ \frac{P_{43}z}{P_{22}}y^{'}\ \frac{P_{34}}{P_{43}z^{'} - P_{33}}\ 1\right).$$

Таким образом, каждой точке на экране можно сопоставить её прообраз при помощи представленного преобразования:

$$\left(x^{'}\ y^{'}\ z^{'}\right) \rightarrow \left(\frac{P_{43}z}{P_{11}}x^{'}\ \frac{P_{43}z}{P_{22}}y^{'}\ \frac{P_{34}}{P_{43}z^{'}-P_{33}}\ 1\right).$$

Обозначим данное преобразование как $F(v)$, где v вектор координат исходной точки. Заметим, что v вектор из трёх компонент, а $F(v)$ вектор из четырёх компонент, при этом четвёртая компонента равна 1.

Полагая матрицу преобразования камеры за C, а обратную к ней за C^{-1}, получим, что координаты прообраза v до преобразования матрицей камеры могут быть записаны в виде:

$$C^{-1} * F(v).$$

Введем новую матрицу камеры как $C^{'}$. Тогда окончательное преобразование старых координат v вершины на экране в новые $v^{'}$ примет вид:

$$v^{'} = P * C^{'} * C^{-1} * F(v).$$

Реализация описанного алгоритма

Для демонстрации предлагаемого подхода на практике было разработано приложение на языке C++, использующее OpenGL. Во-первых, сцена рендерится стандартными способами в две текстуры:

одна текстура для хранения данных цвета каждого пикселя, вторая — для хранения глубины. Во-вторых, для создания промежуточных кадров программа выполняет отрисовку стольких точек, сколько пикселей имеется на исходном изображение. Для каждой точки происходит определение её позиции, используя вершинный шейдер, выполняющий описанные выше вычисления. Вершинный шейдер принимает на вход текстуры цвета и глубины, матрицу перспектив, матрицу камеры в изначальном кадре и новую матрицу камеры. Такой подход позволяет максимально эффективно использовать GPU, минимизируя медленную коммуникацию с процессором [3]. Окно приложения представлено на рисунке 3:

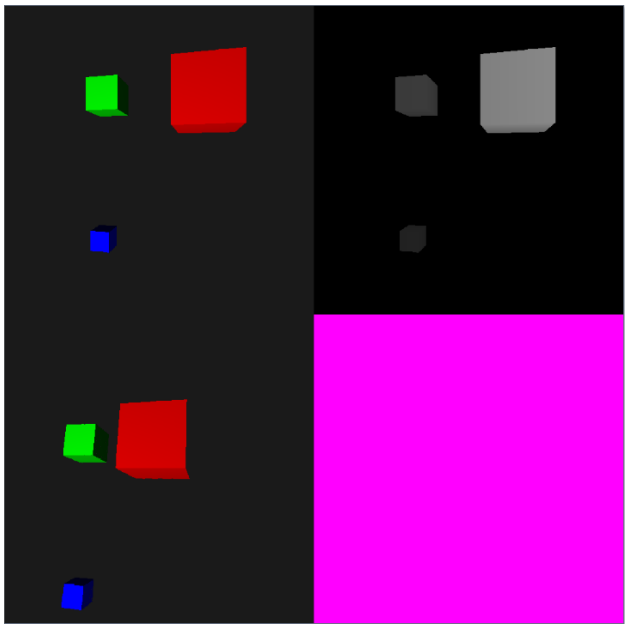

Рисунок 3 – Окно демонстрационного приложения

Демонстрационное приложение предоставляет возможность перемещения и поворота камеры с помощью клавиатуры. Окно приложения разделено на 3 части. Верхние две отображают текстуры, созданные стандартным способом, цвет слева и глубина справа. Снизу же отображён кадр, созданный репроекцией. Для лучшей демонстрации эффекта скорость генерации кадров обычным способом искусственно замедлена. На рисунке 2 в верхней половине показан кадр и соответствующая карта глубин, созданные классическим способом с нуля, а ниже слева представлен кадр, созданный описанным в работе алгоритмом, с камерой в новой позиции.

Заключение

Предложенный алгоритм описывает эффективный способ генерации дополнительных кадров для приложений, использующих *3D* графику. Кроме того, благодаря невысокой сложности операций, уменьшению задержек и повышению качества пользования данными технологиями, данный подход может быть реализован на конечных устройствах, например, на гарнитурах виртуальной реальности, и клиентами для облачного гейминга. Отметим, что существуют направления для улучшения данного метода, например, использование информаций о передвижениях объектов для улучшения плавности анимаций и рассмотрение реализации алгоритма на ПЛИС для достижения минимальных задержек.

Список использованных источников:

1. *The Importance of Frame Rates [Электронный ресурс]. – Режим доступа: https://help.irisvr.com/hc/en-us/articles/215884547-The-Importance-of-Frame-Rates/. – Дата доступа: 26.04.2024.*
2. *OpenGL Projection Matrix [Электронный ресурс]. – Режим доступа: https://www.songho.ca/opengl/gl_projectionmatrix.html. – Дата доступа: 26.04.2024.*
3. *Framebuffers – Advanced OpenGL [Электронный ресурс]. – Режим доступа: https://learnopengl.com/Advanced-OpenGL/Framebuffers. – Дата доступа: 26.04.2024.*

UDC 004.922

OPTIMIZING GENERATION OF NEW FRAMES BY USING 3D REPROJECTION

Ramanouski M.D. [1], *Mahiliavets D.E.* [2]

Belarusian State University of Informatics and Radioelectronics[1], *Minsk, Republic of Belarus*

Primicheva Z.N. – PhD in Physics and Mathematics

Annotation. This paper presents an algorithm for generating intermediate frames when camera changes position, while requiring less computational power compared to a full rendering of a scene from scratch. The proposed algorithm is based on the use of information contained in the depth map, as well as possibility to recover perspective coordinates based on available information.

Keywords. 3D Rendering, intermediate frames, transformation matrix.