ПЛЕНАРНЫЕ ДОКЛАДЫ

БАЗОВЫЕ АЛГОРИТМЫ КОМПЬЮТЕРНОЙ ГРАФИКИ

А.А. Головнин

Тверской государственный технический университет, Тверь

Приведен обзор базовых алгоритмов компьютерной графики, изучаемых будущими специалистами в области программирования графических систем.

Ключевые слова: компьютерная графика, базовые алгоритмы, обзор.

BASIC ALGORITHMS OF COMPUTER GRAPHICS

A.A. Golovnin

Tver State Technical University, Tver

An overview of the basic algorithms of computer graphics, studying of future professionals in the field of graphical programming systems, from the standpoint of reading them, future users of computer graphics programs.

Keywords: computer graphics, basic algorithms, review.

Введение

Компьютерные технологии так стремительно видоизменили рабочее место и инструмент проектировщика и конструктора, что остается лишь констатировать и повсеместно учитывать этот факт. Они не просто обеспечили более высокую эффективность проектно-конструкторской работы, но и утверждают принципиально новые возможности. В первую очередь это работа с естественным для человека трехмерным пространством (непосредственно с геометрической 3D-моделью, а не с ее проекциями) и обеспечение в процессе конструирования информационной поддержки всего жизненного цикла изделия. Кроме того, достигнут качественно новый уровень визуализации конструкторской документации: технология WYSIWYG, презентационная графика, аудиовизуальные документы и многое др. Мало того, возможности нового конструкторского инструмента стандартами ЕСКД возведены в ранг требований.

В то же время объяснима и обеспокоенность пользователя относительно подконтрольности этого инструмента, учитывая, что работа про-

грамм не видна так, как мы привыкли при ручных геометрических построениях карандашом с линейкой и циркулем. Естественно, желание найти компактные обобщения и представления теоретической основы программных технологий и теории компьютерной лексики для описания традиционного языка техники – чертежа. В этой же плоскости находится проблема оптимального сочетания методов и положений традиционных технологий инженерной графики и современных информационных технологий, ставя задачу оптимизации образовательных программ и межпредметных связей.

Представляется, что именно с этих позиций будет интересен обзор образовательного пространства будущих специалистов в области программирования графических систем как занимающихся тем, «что за кнопками» в программах подготовки наших студентов. Для этого обратимся к базовым алгоритмам компьютерной графики, которые рассматриваются в дисциплине «Компьютерная геометрия и геометрическое моделирование».

В частности, согласно ФГОС-3 дисциплина «Компьютерная геометрия и геометрическое моделирование» относится к базовой части профессионального цикла по направлению подготовки 010200.62 «Математика и компьютерные науки», вместе с дисциплинами «Аналитическая геометрия» и «Дифференциальная геометрия и топология» участвует в формировании одних и тех же компетенций будущих бакалавров. С учетом совсем молодого по историческим меркам возраста компьютерной геометрии и графики ограничим информационную базу широко известными учебными изданиями, прошедшими апробацию в учебном процессе высших технических учебных заведений на протяжении одного – полутора десятка лет [1–3]. Обзор будет, конечно, неполным в силу динамического развития и постоянного совершенствования, обновления и пополнения содержания предмета обзора, к тому же опирающегося на учебную литературу, прошедшую этап включения в учебные программы и апробацию, что требует дополнительного времени.

1. Виды компьютерной графики и способы визуализации

Выделяют три вида компьютерной графики: растровую, векторную и фрактальную. Основным элементом изображения растровой графики является пи́ксель (англ. *pixel* – наименьший логический элемент двухмерного цифрового изображения, элемент матрицы дисплеев, формирующих изображение), в векторной графике основной элемент – это линия (прямая или

кривая). Во фрактальной геометрии (от лат. *fractus* – сломанный, разбитый) основным элементом изображения является бесконечно самоподобная геометрическая фигура, каждый фрагмент которой повторяется при уменьшении масштаба. Также в отдельный, четвертый, вид графики часто выделяют и 3D-, или трехмерную, графику. «От двухмерной она отличается тем, что подразумевает построение проекции трехмерной модели сцены (виртуального пространства) на плоскость» [4].

Наряду с видом графики различают способ визуализации. Наиболее известными способами визуализации являются растровый и векторный. Первый используется в таких графических устройствах, как телевизор, дисплей, принтер; второй – в векторных дисплеях, плоттерах, каттерах. Следует различать вид геометрического моделирования и процесс вывода изображения на экран, т.е. вид компьютерной графики и способ визуализации. Чаще всего вид компьютерной графики не совпадает со способом визуализации, для их совмещения требуется конвертация. Смешение этих понятий может привести к ложным выводам о том, что геометрическая модель тождественна ее аксонометрической проекции, или о том, что метод проецирования лежит в основе трехмерной графики.

Программы САПР оперируют линиями, т.е. относятся к программам векторной графики. В качестве теоретической основы геометрического моделирования выступают вариационное исчисление, дифференциальная и аналитическая геометрия, численные методы, теория B-сплайнов, топология и разделы вычислительной математики, методы моделирования различных кривых, поверхностей и тел, а также вычисление их геометрических характеристик и алгоритмы выполнения операций над ними, установление вариационных зависимостей параметров геометрических объектов. Геометрическое моделирование изучает методы построения кривых линий, поверхностей и твердых тел, методы выполнения над ними различных операций и методы управления численными моделями [5].

2. Координатный метод в компьютерной графике

Координатный метод без преувеличения можно назвать основным методом компьютерной графики. В координатах задают положение и форму всех геометрических объектов, выполняют их преобразования. В компьютерной графике предусмотрены самые широкие возможности по использованию координат. Применяются всевозможные прямолинейные и криволинейные координаты, все известные в аналитической геометрии способы задания.

Однако следует отметить, что привычный аппарат декартовых координат нельзя применять для решения некоторых важных задач в силу следующих соображений:

а) в декартовых координатах невозможно описать бесконечно удаленную точку, однако, если ввести понятие бесконечности, то многие математические и геометрические концепции значительно упрощаются;

б) в декартовых координатах нельзя провести различия между точками и векторами в пространстве с точки зрения алгебраических операций;

в) для выражения преобразований точек невозможно использовать унифицированный механизм работы с матрицами;

г) в декартовых координатах невозможно использовать матричную запись для задания перспективного преобразования (проекции) точек.

Решение этих проблем возможно при использовании однородных координат, которые были введены Плюккером в качестве аналитического подхода к принципу двойственности Жергонна–Понселе. Однородные координаты являются мощным математическим инструментом, который связан с определением положения точек в пространстве и находит свое применение в различных разделах компьютерной графики: геометрическом моделировании, визуализации, машинном зрении и т.д. Однородные координаты явно или неявно используются в любом графическом пакете [6].

Однородные координаты на плоскости имеют простую геометрическую интерпретацию. Преобразование из однородных координат (x, y, z) в евклидовы $(x/z, y/z, 1)$ эквивалентно проекции точки на плоскость $z = 1$ вдоль линии, соединяющей точку с началом координат.

К координатам можно применять двухмерные и трехмерные аффинные преобразования: повороты вокруг координатных осей, отражения относительно координатных плоскостей, перенос и композицию преобразований.

Общий вид аффинного преобразования на плоскости имеет вид

$$\begin{cases} X = Ax + By + C \\ Y = Dx + Ey + F \end{cases} \qquad \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} A & B & C \\ D & E & F \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix},$$

где (x, y) – двухмерная система координат; (X, Y) – координаты старой СК в новой системе координат; A, B, C, D, E, F – константы.

Важнейшей операцией при визуализации трехмерной графики является *проектирование*. Под проектированием понимают преобразование, которое ставит в соответствие точкам трехмерного пространства точки на некоторой плоскости, называемой *картинной.*

Два основных вида проектирования – параллельное и перспективное – используются в компьютерной графике. Как произвольное аффинное преобразование, так и параллельное и перспективное проектирования могут быть записаны при помощи матриц однородных преобразований. Для сравнения приводим *матрицы канонического уравнения параллельного проектирования*, осуществляемого на плоскость *Oxy* вдоль оси *Oz*, и *канонического уравнения перспективного проектирования*:

$$P_i = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad P_c = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}.$$

Обращает на себя внимание тот факт, что в программах компьютерной графики вычисления для получения параллельной и перспективной проекций похожи как по сути, так и по трудоемкости. Однако при ручных построениях алгоритмы значительно отличаются, особенно по трудоемкости.

3. Геометрические объекты

При описании геометрических объектов используются такие математические объекты и их свойства, как прямые и плоскости, кривые линии, двухмерные кривые, поверхности, кривизна линий на поверхности, криволинейные координаты. Все эти вопросы всесторонне разработаны и подробно освещены в курсе аналитической геометрии. Можно говорить о том, что компьютерная графика в части описания геометрических объектов имеет в лице аналитической геометрии готовый к использованию математический аппарат. В компьютерной графике могут решаться системы линейных и нелинейных уравнений. Это позволяет определять точки пересечения линий, линии и поверхности, построение линий пересечения поверхностей и другие вычисления.

Одним из важнейших инструментов систем автоматизированного проектирования и программ компьютерной графики стали сплайны.

Сплайн – это гладкая кривая, которая проходит через две или более опорные точки, а также имеет расположенные вне ее управляющие точки, влияющие на форму сплайна. К наиболее общим типам сплайнов относятся кривые Безье и B-сплайны (B-spline curves). Неоднородные рациональные B-сплайны (non-uniform rational B-spline – NURBS) также являются типичным примером сплайнов. Сплайны состоят из вершин (vertices) и сегментов (segments). У каждой вершины сплайна имеются касательные векторы (tangents), на концах которых находятся управляющие точки, или маркеры (handels). С помощью маркеров касательных векторов можно управлять кривизной сегментов сплайна при входе в вершину, которой принадлежат касательные векторы, и при выходе из нее.

Кривые Безье были созданы для проектирования кузовов автомобилей в 60-х гг. XX в. Независимо друг от друга их разработали Пьер Безье из автомобилестроительной компании «Рено» и Поль де Кастельжо из компании «Ситроен».

Кривые Безье описываются в параметрической форме:

$$x = P_x(t), y = P_y(t),$$

где значение t выступает как параметр, которому отвечают координаты отдельной точки линии.

Многочлены Безье для P_x и P_y имеют следующий вид:

$$P_x(t) = \sum_{i=0}^{m} C_m^i t^i (1-t)^{m-i} x_i, \quad P_y(t) = \sum_{i=0}^{m} C_m^i t^i (1-t)^{m-i} y_i, \quad C_m^i = \frac{m!}{t!(m-t)!},$$

где C_m^i – сочетание m по i, а x_i, y_i – координаты точек ориентиров P_i. Значение m (1, 2, 3) можно рассматривать и как степень полинома, и как значение, которое на единицу меньше количества точек-ориентиров.

4. Методы построения поверхностей

В компьютерной графике самое широкое применение имеют плоскостные (полигональные) модели или полигональные сетки. Поверхность геометрического объекта выглядит в них как набор состыкованных друг с другом плоских полигонов. Традиционным считается иерархическое описание полигональной модели объекта, которое включает в себя списки ребер, вершин и полигонов объекта. Контур

полигона определяется вершинами, которые соединены отрезками прямых линий. В векторной форме полигон задается перечислением его вершин: $P = \{p_1, p_2, \ldots, p_n, p_1\}$. Очевидно, что главный недостаток полигональных моделей заключается в необходимости задания большого количества полигонов для представления сложных, особенно криволинейных, поверхностей. Это приводит к тому, что в режиме реального времени при синтезе динамических изображений необходимо пересчитывать геометрические параметры у большого числа примитивов, поэтому наряду с плоскими поверхностями в графических системах применяются криволинейные примитивы, в частности поверхности второго порядка.

Достаточно часто отображаемые объекты, прежде всего природные, имеют довольно сложную форму, не позволяющую применить универсального аналитического описания в целом. Форму таких объектов можно задать в виде набора характерных (опорных) точек, которые принадлежат поверхности объекта. Опорные точки можно получить разными способами: путем сканирования с помощью 3D-сканеров, с помощью замеров на реальных объектах или назначения характерных точек самими разработчиками.

Наиболее известным примером такого описания поверхности является составленная геодезистами карта высот участка земной поверхности. Если соединить опорные точки, то получится совокупность плоских элементов, т.е. полигональная модель. Следует отметить, что для реалистичного отображения объекта его полигональная модель должна состоять из тысяч и десятков тысяч полигонов, что требует как значительного объема памяти, так и повышенных требований к производительности графической системы. Применение квадрик также не приносит успеха, так как в этом случае возникает проблема их гладкой стыковки в единую поверхность. В настоящее время применение кусочно-полиномиальных функций – сплайнов позволяет наиболее полно представить поверхности неаналитических форм.

Достаточно часто в геометрическом моделировании используется бикубическая поверхность Безье. Прохождение поверхности через угловые точки характеристического многогранника и необходимость соблюдать заданные на его границах наклоны касательных являются ограничениями при моделировании с помощью поверхностей Безье. Кроме сплайнов Безье, широкое применение в компьютерной графике получили базовые сплайны, или B-сплайны (би-сплайны). Значи-

тельные изобразительные возможности характерны для рациональных бикубических сплайнов. В компьютерной графике широко применяются рациональные B-сплайны на неравномерной сетке (non-uniform rational B-splines – NURBS), в описание которых входят числовые параметры формы или весовые коэффициенты, позволяющие управлять формой поверхности [3].

Использование аналитических поверхностей позволяет существенно повысить наглядность и эффективность изучения свойств этих поверхностей как в исследовательских работах, так и в учебном процессе [7, 8].

5. Многоугольники (полигоны)

С полигонами связано большое количество вычислений, в том числе тестов. Тесты могут не давать исчерпывающий ответ на поставленный вопрос, но могут ограничить область поиска и сократить время и машинные ресурсы. Например, человек, не задумываясь, ответит на вопросы на основании того, что он видит. В компьютерной графике поиск ответов на вопросы формализован.

Свойства плоских многоугольников:

1. *Пересечение прямой линии с полигоном*. Прямая пересекает полигон, если существует хотя бы одна пара вершин, лежащих от нее по разные стороны (это свойство предполагает сравнение для всех имеющихся пар вершин, а не только смежных).

2. *Выпуклость полигона*. У выпуклого полигона все углы при вершинах $pi - 1$, pi, $pi + 1$ имеют одинаковый знак. Другими словами, при обходе выпуклого полигона по замкнутому контуру в произвольном направлении каждая вершина $pi + 1$ расположена относительно ребра $pi - 1$ pi одинаково для всех значений i: слева при положительном направлении обхода и справа при отрицательном.

3. *Самопересечение полигона*. Полигон является самопересекающейся замкнутой ломаной линией, если у него существует хотя бы одна пара пересекающихся отрезков. Два отрезка пересекаются друг с другом, если концы одного находятся по разные стороны от прямой другого, и наоборот (тестироваться должны все пары несмежных ребер полигона).

Тесты ориентации точки относительно полигона следующие: выпуклый тест, габаритный тест, угловой тест. Перечисленные тесты, будучи проведенными до начала вычислений, существенно сокращают время вычислений и вообще оптимизируют работу компьютера.

6. Базовые растровые алгоритмы

Алгоритмы вывода прямой. Растровые алгоритмы используются для рисования линий на экране компьютера. Изображение отрезка на экране должно отвечать некоторым минимальным требованиям:

– отрезки должны выглядеть прямыми;

– концы отрезка должны находиться в заданных точках;

– яркость вдоль отрезка должна быть постоянной и не зависеть от длины и наклона.

В силу дискретности ни одно из этих условий не может быть точно выполнено на растровом дисплее [9].

Растровые алгоритмы построения прямой призваны получить максимально возможное приближение к этим требованиям при минимальных ресурсах.

Прямое вычисление координат. Пусть заданы координаты конечных точек отрезка (x_1, y_1) и (x_2, y_2). Координаты внутренней точки отрезка вычисляются следующим образом: $y = F(x)$: $y = y_1 + (x - x_1) \times \times (y_2 - y_1)/(x_2 - x_1)$.

Для того чтобы свести к минимуму вычисления в цикле, все операции над константами выносятся из тела цикла: $k = (y_2 - y_1)/(x_2 - x_1)$, $yy = y_1 - x_1 \cdot k$.

В цикле вычисляется $y = yy + x \cdot k$.

С учетом того что вычисление дробей происходит с определенной погрешностью, возможна ситуация, когда на последнем шаге цикла x окажется неравным x_2.

Подход, позволяющий разрабатывать так называемые *инкрементные алгоритмы растеризации*, впервые был предложен в 1962 г. сотрудником компании IBM Дж.Е. Брезенхемом (Jack E. Bresenham). Алгоритм Брезенхе́ма – это один из старейших алгоритмов в компьютерной графике. Он позволяет определить, какие точки двухмерного растра нужно закрасить, чтобы получить близкое приближение прямой линии между двумя заданными точками.

Инкрементные алгоритмы выполняются как последовательное вычисление координат соседних пикселей путем добавления приращений координат. Приращения рассчитываются на основе анализа функции погрешности. В цикле выполняются только целочисленные операции сравнения и сложения/вычитания без использования умножения и деления.

Это позволяет повысить эффективность использования цифровых ЭВМ, которые заточены под целочисленные вычисления. Достаточно заметить, что y уменьшается от y_0, и за каждый шаг мы добавляем к x единицу и к y значение наклона $s = \frac{y_1 - y_0}{x_1 - x_0}$, которое можно вычислить заранее. Более того, на каждом шаге мы делаем одно из двух: либо сохраняем тот же y, либо уменьшаем его на 1 [10].

Следует отметить, что для построения кривых 2-го порядка существует обобщение алгоритма Брезенхема.

Алгоритм Брезенхема вывода окружности. Для вывода контура круга можно построить алгоритм прямого вычисления координат, используя соотношение между координатами X и Y для точек окружности $X^2 + Y^2 = R^2$. Однако в этом случае необходимо вычислять квадратный корень (как элемент бесконечной последовательности приближений).

Алгоритм вывода окружности пошагово генерирует очередные точки окружности, выбирая на каждом шаге для занесения пикселя точку растра Pi (Xi, Yi), ближайшую к истинной окружности, так, чтобы ошибка Ei (Pi) = ($Xi^2 + Yi^2$) – R^2 была минимальной. Причем, как и в алгоритме Брезенхема для генерации отрезков, выбор ближайшей точки производится с помощью анализа значений управляющих переменных, для вычисления которых не требуется вещественной арифметики. При выборе очередной точки достаточно проанализировать знаки.

Для простоты и без ограничения общности генерируют 1/8 окружности, центр которой лежит в начале координат. В вычислениях используется свойство окружности, что после закрашивания на экране пикселя самой верхней точки окружности на 1/8 ее части следующий пиксель, который необходимо закрасить, может быть только один из двух соседних – справа или справа снизу. Остальные части окружности могут быть получены последовательными отражениями.

Аналогично можно построить эллипс.

Алгоритм Ву – это алгоритм разложения отрезка в растр со сглаживанием. Был предложен в 1991 г. У. Сяолинем (*Xiaolin Wu*), отсюда устоявшееся в русском языке название алгоритма. Алгоритм сочетает высококачественное устранение ступенчатости и скорость, близкую к скорости алгоритма Брезенхема без сглаживания.

Отличие алгоритма Ву от алгоритма Брезенхема состоит в том, что в нем на каждом шаге устанавливается не одна, а две точки. В случае, если за основную ось принять ось X, то будут рассматриваться точки

с координатами (x, y) и (x, y + 1). Величина ошибки показывает, как далеко ушли пиксели от идеальной линии по неосновной оси. В зависимости от этого будет распределяться интенсивность между этими двумя точками. Чем больше удалена точка от идеальной линии, тем меньше ее интенсивность. Сумма интенсивности двух пикселей всегда равна единица, т.е. это интенсивность одного пикселя, в точности попавшего на идеальную линию. Такое распределение на всем протяжении линии придает ей одинаковую интенсивность, создавая при этом иллюзию, что точки расположены вдоль линии не по две, а по одной.

7. Цвет в компьютерной графике

Современные САПР работают в цветном представлении изображений. Для описания цветов, которые получаются с помощью устройств, основанных на принципе излучения, используется аддитивная цветная модель RGB. В этой модели работают мониторы и бытовые телевизоры.

Кроме модели RGB, применяется модель HSV, в которой цвет описывается следующими параметрами: цветовой тон H (hue), насыщенность S (saturation), яркость, светлота V (value).

Для описания цвета при получении изображений на устройствах, которые реализуют принцип поглощения (вычитания) цветов, используется субтрактивная цветовая модель CMYK. Цветовыми компонентами этой модели являются не основные цвета, а те, которые получаются в результате вычитания основных цветов из белого: голубой (cyan), пурпурный (magenta), желтый (yellow). Существуют также и другие цветовые модели.

8. Визуализация изображений. Основные понятия

Изображение на экране монитора персонального компьютера выполняется растровой графикой путем закрашивания всех пикселей в определенный цвет. На задание цвета для каждого пикселя экрана в памяти компьютера может быть отведено 1 или 2 бита. В соответствии с этим получено 2- или 4-цветное изображение (мониторы CGA, единственные, содержащие в своем названии слово «цветной», были именно 4-цветными). При дальнейшем увеличении отведенной памяти в 1, 2, 3 байта может быть передано 256, 65 536 или 16 777 216 цветами [11]. Интересно, что человеческий глаз различает примерно 16 млн оттенков цветов.

Возможное разрешение мониторов непрерывно растет и составляет 7680×4800 пикселей (при привычном 1280×1024, хотя лет 10 назад

было 800×600). При таком количестве точек, которые необходимо закрасить, без специальных средств компьютер можно было бы загрузить одной только этой задачей. Задача решается как аппаратно (совершенствованием видеокарт, процессоров и т.п.), так и программно путем разработки эффективных счетных алгоритмов.

9. Алгоритмы удаления скрытых линий и поверхностей

Наиболее сложной в компьютерной графике является задача удаления невидимых линий и поверхностей. Алгоритмы удаления невидимых линий (поверхностей) служат для определения линий ребер, поверхностей или объемов, которые являются видимыми или невидимыми для наблюдателя, находящегося в заданной точке пространства.

Именно сложностью решения задачи удаления невидимых объектов обусловлено появление большого числа различных способов ее решения, многие из которых ориентированы на специализированные приложения. Для общей задачи удаления невидимых линий и поверхностей наилучшего решения не существует. При моделировании процессов, протекающих в реальном времени, например для авиатренажеров, необходимо применение быстрых алгоритмов, порождающих результаты с частотой видеогенерации (30 кадр./с). Если же речь идет о компьютерной мультипликации, то требуются алгоритмы, позволяющие генерировать сложные реалистические изображения. В таких изображениях могут быть представлены тени, прозрачность и фактура, в которых эффекты отражения и преломления цвета можно реализовать в мельчайших оттенках. Подобные алгоритмы требуют значительных временных затрат – от нескольких минут до нескольких часов. Если подходить строго, то учет таких эффектов, как прозрачность, фактура, отражение и т.п., не входит в задачу удаления невидимых линий или поверхностей. Гораздо естественнее считать их частью процесса визуализации изображения, который является интерпретацией или представлением изображения или сцены в реалистической манере.

Однако все алгоритмы удаления невидимых линий (поверхностей) включают в себя сортировку. Порядок, в котором производится сортировка координат объектов, вообще говоря, не влияет на эффективность применения этих алгоритмов. Основная сортировка ведется по величине геометрического расстояния от точки, ребра, поверхности или тела до точки наблюдения. Главная идея сортировки по расстоянию состоит в том, что чем дальше от точки наблюдения располагается объект, тем

больше возможность полного или частично заслонения его объектами, которые расположены ближе к точке наблюдения. После определения расстояний или приоритетов по глубине пространства проводится сортировка по горизонтали и по вертикали. Это необходимо для определения расположения рассматриваемого объекта относительно других объектов и решения вопроса: действительно ли данный объект будет заслонен объектом, расположенным ближе к точке наблюдения?

Алгоритмы удаления невидимых частей сцены можно классифицировать по следующим признакам:

1) выбор удаляемых частей: удаление невидимых линий, ребер, поверхностей, объемов;

2) порядок обработки элементов сцены: удаление в порядке, определяемом процессом визуализации, или в произвольном порядке;

3) в зависимости от системы координат: алгоритмы, работающие в пространстве объектов, когда каждая из N граней объекта сравнивается с остальными $N - 1$ гранями (объем вычислений растет как N^2); алгоритмы, работающие в пространстве изображения, когда для каждого пикселя изображения определяется, какая из N граней объекта видна (при разрешении экрана $M \times M$ объем вычислений растет как $M^2 \times N$).

Алгоритмы удаления линий. Алгоритмы удаления линий применяются в векторных устройствах. Для ускорения процесса визуализации они могут применяться и в растровых устройствах. Однако при таком применении не используется основное ценное качество растрового дисплея – возможность закраски поверхностей.

Наиболее известным ранним алгоритмом является алгоритм Робертса (1963 г.), который может работать только с выпуклыми телами в пространстве объектов. При этом каждый объект сцены представляется в виде многогранного тела, которое получается в результате пересечения плоскостей. Таким образом, тело описывается как список граней, состоящих из ребер, которые, в свою очередь, образованы вершинами.

Сначала из описания каждого тела удаляют нелицевые плоскости, которые экранированы самим телом. Затем для определения видимости каждое из ребер сравнивается с каждым телом. Таким образом, объем вычислений растет как квадрат числа объектов в сцене. Последними вычисляют новые ребра, которые получаются при пересечении тел друг с другом.

Алгоритм плавающего горизонта чаще всего используется для удаления невидимых линий в случае трехмерного представления функций, описывающих поверхность в виде $F(x, y, z) = 0$.

Подобные функции возникают во многих приложениях в естественных науках, технике, математике и других дисциплинах.

Главная идея данного метода заключается в понижении размерности задачи – от трехмерной к двухмерной. Для этого исходную поверхность пересекают последовательностью параллельных секущих плоскостей, которые имеют постоянные значения координат x, y или z.

В дальнейшем алгоритм предусматривает упорядочивание плоскостей z = const по возрастанию расстояния до них от точки наблюдения. Затем для каждой плоскости, начиная с ближайшей к точке наблюдения, строится кривая, лежащая на ней. Таким образом, для каждого значения координаты x в пространстве изображения определяется соответствующее значение y. Для удаления невидимой линии применяется следующий алгоритм. Если при некотором заданном значении x на текущей плоскости соответствующее значение y на кривой больше значения y для всех предыдущих кривых при данном значении x, то текущая кривая будет видимой в этой точке.

Фактически такой алгоритм удаления невидимых линий работает каждый раз только с одной линией.

Алгоритм удаления поверхностей с Z-буфером. Алгоритм был предложен Эдом Кэтмулом и представляет собой обобщение буфера кадра. Обычный буфер кадра хранит в пространстве изображения коды цвета для каждого пикселя. Идея алгоритма удаления поверхностей с Z-буфером состоит в том, чтобы для каждого пикселя дополнительно хранить величину глубины или координату Z. Когда очередной пиксель заносится в буфер кадра, происходит сравнение значения его Z-координаты с координатой Z пикселя, который уже имеется в буфере. Атрибуты нового пикселя и его Z-координата заносятся в буфер, если он ближе к наблюдателю, т.е. если Z-координата нового пикселя больше, чем координата старого.

Главное преимущество алгоритма заключается в его простоте, однако для его реализации требуется большой объем памяти.

Алгоритм, использующий список приоритетов. Алгоритмы, использующие список приоритетов, пытаются получить преимущество посредством предварительной сортировки по глубине или приоритету. Тогда можно записать все элементы в буфер кадра поочередно, начиная с элемента, наиболее удаленного от точки наблюдения. Более близкие к наблюдателю элементы будут «затирать» информацию о более далеких элементах в буфере кадра. Эффекты прозрачно-

сти можно включить в состав алгоритма путем не полной, а частичной корректировки содержимого буфера кадра с учетом атрибутов прозрачных элементов.

Метод иногда называют алгоритмом художника, особенно для простых элементов сцены, таких как многоугольники. Алгоритм аналогичен способу создания картины художником. Сначала художник рисует фон, затем предметы, лежащие на среднем расстоянии, и, наконец, передний план картины [12].

Алгоритм разбиения области Варнока. Алгоритм работает в пространстве изображения и анализирует область на экране дисплея (окно) на наличие в нем видимых элементов. В том случае, если в окне нет изображения, оно просто закрашивается фоном. Если же в окне имеется элемент, то проверяется, достаточно ли он прост для визуализации. Если объект сложный, то окно разбивается на более мелкие окна, каждое из которых проверяется на отсутствие и/или простоту изображения. Рекурсивный процесс разбиения может продолжаться до тех пор, пока не будет достигнут предел разрешения экрана [1, 13, 14].

Алгоритм определения видимых поверхностей путем трассировки лучей. Оценка эффективности всех перечисленных алгоритмов удаления невидимых поверхностей зависит от определенных характеристик когерентности той сцены, для которой ведется поиск ее видимых участков.

В отличие от них, трассировка лучей является методом грубой силы (специфика обрабатываемого объекта не учитывается). В основе этого метода лежит идея о том, что наблюдатель видит любой объект посредством испускаемого неким источником света, который падает на этот объект и затем каким-то путем доходит до наблюдателя. Свет может достигать наблюдателя, отразившись от поверхности, преломившись или пройдя через нее. Проследив за лучами света, выпущенными из источника, можно убедиться в том, что не многие из них дойдут до наблюдателя. Следовательно, этот процесс был бы вычислительно неэффективен. Артур Аппель в 1968 г. первым предложил определение видимых или скрытых поверхностей отслеживать (трассировать) лучами обратного направления, т.е. от наблюдателя к объекту.

Впоследствии Кей и Уиттед реализовали алгоритмы трассировки лучей с использованием общих моделей освещения. Эти алгоритмы учитывают эффекты отражения одного объекта от поверхности другого, преломления, прозрачности и затемнения [15].

10. Кратко о презентационной графике

При формировании изображения сцен в презентационной графике используют законы геометрической оптики, преломляющие свойства материалов, эффекты смешения цветов и т.д. Производится также устранение ступенчатости.

Методы закрашивания объектов, моделируемых многогранниками и полигональными сетками, основаны на моделях отражения света. При изображении объектов обычно моделируют сочетание зеркального и диффузного рассеивания в пропорции, характерной для конкретного материала. Интенсивность отраженного света можно определить как сумму диффузного и зеркального компонентов (из физики) с учетом рассеяния света с расстоянием $I_{\text{отр}} = I_a K_a + I (K_d \cos q + K_s \cos pa) / (R + k)$, где константы K_d, K_s определяют отражательные свойства материала; R – расстояние от центра проекции до поверхности; k – константа, подбираемая эмпирически.

Этот метод предназначен для создания иллюзии гладкой криволинейной поверхности, описанной в виде многогранников или полигональной сетки с плоскими гранями. Простое увеличение числа граней приводит к существенному замедлению визуализации. Преодолеть это противоречие призваны методы Гуро и Фонга.

Метод Гуро основывается на идее закрашивания каждой плоской грани не одним цветом, а плавно изменяющимися оттенками, вычисляемыми путем билинейной интерполяции цветов примыкающих граней. Существенным недостатком метода Гуро является невозможность получения качественных бликов на блестящих поверхностях.

Закраска Фонга требует больших вычислительных затрат, однако она позволяет разрешить многие проблемы метода Гуро. При закраске Гуро вдоль сканирующей строки интерполируется значение интенсивности, а при закраске Фонга – вектор нормали. Затем он используется в модели освещения для вычисления интенсивности пикселя. При этом достигается лучшая локальная аппроксимация кривизны поверхности, следовательно, получается более реалистичное изображение. В частности, более правдоподобно выглядят зеркальные блики [16].

Выводы

Можно сказать, что обстоятельно исследованы сегменты геометрических преобразований и описаний кривых и поверхностей. Изучены, но все еще продолжают развиваться методы растрового сканирования,

отсечение, удаление линий и поверхностей, цвет, закраска, текстура и эффекты прозрачности.

Нужно ли знание внутренних алгоритмов работы графических компьютерных программ САПР инженеру-конструктору? В профессиональной деятельности вряд ли. Но грамотному преподавателю для расширения кругозора и эрудиции на понятийном уровне ознакомление с ними необходимо. Кроме того, в программах компьютерной графики (возможно, не во всех) предусматривается возможность для программирования на уровне пользователя. Использование такой возможности в сочетании со знанием внутренних алгоритмов может повысить производительность и значительно расширить возможности использования программ. Есть примеры, когда преподаватели, владеющие графическими программами на уровне программирования в них, успешно передают эти знания своим студентам [7].

В то же время алгоритмы компьютерной геометрии – это один из разделов математики, которую, по меткому выражению М.В. Ломоносова, «уже затем учить надо, что она ум в порядок приводит».

Список литературы

1. Вельтмандер П.В. Машинная графика [Электронный ресурс]: учеб. пособие: в 3 кн. Кн. 2. Основные алгоритмы компьютерной графики / Новосиб. гос. ун-т. – Новосибирск, 1997. – URL: http://ermak.cs.nstu.ru/kg_rivs/kg02.htm#tth_sEc0.10.1 (дата обращения: 14.01.2016).

2. Снижко Е.А. Компьютерная геометрия и графика: конспект лекций / Балт. гос. техн. ун-т. – СПб., 2005. – 132 с.

3. Косников Ю.Н. Поверхностные модели в системах трехмерной компьютерной графики: учеб. пособие / Пенз. гос. ун-т. – Пенза, 2007. – 60 с.

4. URL: http://cpu3d.com (дата обращения: 14.01.2016).

5. Голованов Н.Н. Геометрическое моделирование. – М.: Изд-во физ.-мат. лит., 2002. – 472 с.

6. Игнатенко А. Однородные координаты. Компьютерная графика и мультимедиа // Сетевой журнал. – 2003. – № 1 (5). – URL: http://cgm.computergraphics.ru/content/view/51 (дата обращения: 10.02.2016).

7. Саморуков А.В., Хейфец А.Л., Самойлов С.П. Аналитические поверхности в курсе компьютерной графики для архитекторов [Электронный ресурс] // Проблемы качества графической подготовки студентов в техническом вузе в условиях ФГОС ВПО: материалы II Между-

нар. науч.-практ. интернет-конф., Пермь, февраль – март 2011 г. – Пермь: Изд-во Перм. гос. техн. ун-та, 2011. – С. 207–211. – URL: http://dgng.pstu.ru/media/files/%D0%A1%D0%B1%D0%BE%D1%80%D0%BD%D0%B8%D0%BA_%D0%9A%D0%93%D0%9F-2011.pdf (дата обращения: 10.02.2016).

8. Хейфец А.Л. Развитие курса инженерной 3d компьютерной графики в новом учебнике [Электронный ресурс] // Проблемы качества графической подготовки студентов в техническом вузе: традиции и инновации: материалы V Междунар. науч.-практ. интернет-конф., Пермь, февраль – март 2015 г. – Пермь: Изд-во Перм. нац. исслед. политехн. ун-та, 2015. – С. 476–490. – URL: http://dgng.pstu.ru/media/files/%D0%A1%D0%B1%D0%BE%D1%80%D0%BD%D0%B8%D0%BA_%D0%9A%D0%93%D0%9F-2015.pdf (дата обращения: 14.02.2016).

9. URL: http://algolist.manual.ru/graphics/painting/line.php (дата обращения: 21.02.2016).

10. URL: https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%91%D1%80%D0%B5%D0%B7%D0%B5%D0%BD%D1%85%D1%8D%D0%BC%D0%B0 (дата обращения: 10.02.2016).

11. URL: https://ru.wikipedia.org/wiki/%D0%93%D0%BB%D1%83%D0%B1%D0%B8%D0%BD%D0%B0_%D1%86%D0%B2%D0%B5%D1%82%D0%B0#HighColor (дата обращения: 11.02.2016).

12. URL: http://www.mari-el.ru/mmlab/home/kg/Lection10/1.html (дата обращения: 14.01.2016).

13. URL: http://sergeypacuk.narod.ru/glava2.html (дата обращения: 14.01.2016).

14. URL: http://compgraph.tpu.ru/warnock.htm (дата обращения: 14.01.2016).

15. URL: http://www.mari-el.ru/mmlab/home/kg/Lection10/5.html (дата обращения: 14.01.2016).

16. URL: http://www.mari-el.ru/mmlab/home/kg/Lection11/4.html (дата обращения: 14.01.2016).