УДК 681.3

# ВЫБОР УРОВНЯ ДЕТАЛЬНОСТИ ПРИ НЕПРЕРЫВНОМ УПРОЩЕНИИ ПОВЕРХНОСТЕЙ ПОЛИГОНАЛЬНЫХ ОБЪЕКТОВ

**С. С. Садыков**[1], **А. А. Захаров**[1]

В статье рассматривается способ выбора уровня детальности, связывающий геометрическую сложность и расстояние от объекта до наблюдателя. Приводится краткий обзор методов уменьшения геометрической сложности. Предложен способ упрощения полигональной модели, использующий последовательное удаление вершин объекта. Выведено соотношение между количеством удаляемых вершин и расстоянием между объектом и наблюдателем, позволяющее однозначно определить уровень детальности.

**Ключевые слова**: уровень детальности, полигональные объекты, компьютерная графика, триангуляция.

**Введение**. В современных системах компьютерной графики, особенно работающих в режиме реального времени, используется множество различных алгоритмов, позволяющих снизить вычислительные затраты. Одной из таких областей алгоритмической разработки является отображение одного и того же объекта на различных уровнях детальности, представляющих собой некоторую степень визуального разрешения. Каждый уровень детальности есть некоторая аппроксимация начальной модели объекта с использованием меньшего количества геометрических примитивов [1]. В этом случае по исходной модели объекта необходимо получить его упрощенное представление. При создании последующего уровня мелкие вырождающиеся детали модели, фактически не влияющие на изображение, заменяются на более крупные.

Трехмерные объекты в системах компьютерной графики обычно представляются сетью смежных треугольников. Каждый треугольник является элементарным примитивом, влияющим на характер изображения. При упрощении сети происходит сокращение числа треугольников и за счет этого увеличиваются размеры оставшихся полигонов. В графической системе должны быть представлены только те грани, которые проецируются на область, сравнимую с размером пикселя. В идеале одному треугольнику объекта должны соответствовать один-два пикселя экрана. Примитивы, меньшие пикселя экрана, не имеет смысла обрабатывать. Поэтому сокращение избыточной детальности позволяет разгрузить аппаратуру от генерации ненужных примитивов. Кроме уменьшения геометрической сложности полигональных объектов существуют также задачи перехода от одного уровня детальности к другому и выбор конкретного геометрического представления в текущий момент времени. Например, при удалении объекта от наблюдателя часто бывает целесообразно заменить детальное описание на более грубое, а при приближении — наоборот. При этом необходимо стремиться к тому, чтобы переход между уровнями был незаметен, а качество визуализируемых объектов не ухудшалось.

Подобные задачи возникают в системах архитектурного и ландшафтного проектирования, различного рода симуляторах и тренажерах, системах визуализации научных (в том числе медицинских) данных, объемных презентациях, компьютерных играх.

В статье дан краткий обзор известных алгоритмов упрощения, переключения и выбора уровня детальности. Также рассматривается способ упрощения полигональной сети при последовательном удалении вершин. Анализ таких алгоритмов выбора уровня детальности, как использование диапазонов, вычисление площадей проекций ограничивающих объемов, применение приоритетов, показал, что они не устанавливают количественной зависимости между числом примитивов объекта и текущими условиями визуализации. Поэтому в работе выводится численное соотношение между количеством удаляемых вершин объекта и расстоянием до наблюдателя, позволяющее однозначно выбирать уровень детальности.

**1. Упрощение полигональных моделей**. Построение 3D-объектов в компьютерной графике осуществляется в большинстве систем на основе полигонов [1, 7, 9, 11]. Многоугольники, образующие поверхность объекта являются гранями, стороны граней — ребрами, концы ребер — вершинами. Ребра, грани и вершины полигонального объекта связаны классической формулой Эйлера [8]

$$V - E + F = 2,$$

[1] Муромский институт Владимирского государственного университета, радиотехнический факультет, ул. Орловская, 23, 602245, г. Муром; e-mail: is@mivlgu.murom.ru

где $V$ — число вершин, $E$ — число ребер и $F$ — число граней. Соотношения между $V$, $E$, и $F$ можно также представить следующим образом:

$$E \leqslant 3V - 6, \quad F \leqslant \frac{2}{3}E,$$

$$F \leqslant 2V - 4. \tag{1}$$

Для эффективной работы с полигональными объектами необходима такая структура данных, чтобы она отражала связи между вершинами и многоугольниками в сети. Для каждой вершины в этом случае имеется указатель на смежные с ней грани. Структура трехмерного объекта, образованного полигонами, представлена на рис. 1.

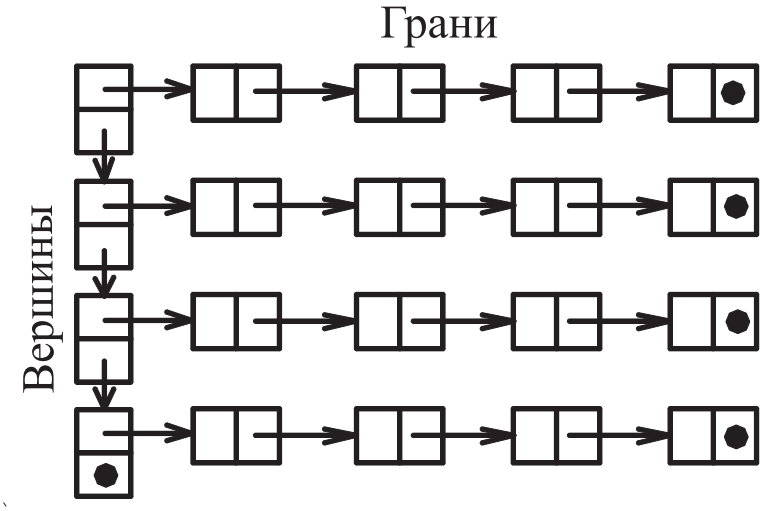


Рис. 1. Структура полигонального поля “вершина–грани”: получение для данной вершины списка соседних с ней граней

Чем большее количество полигонов используется для построения объекта, тем детальнее он будет представлен. Тем самым будет обеспечена плавность геометрии, а следовательно, и реалистичность синтезируемого изображения. Для снижения вычислительных затрат можно уменьшить количество полигонов, образующих полигональную поверхность. Этот процесс принято называть упрощением.

**Определение 1**. Под *упрощением* полигональной сети понимается преобразование ее в сеть той же структуры, но имеющую меньшее количеством вершин.

**Определение 2**. *Уровень детальности* (Level of Detail — LOD) полигонального объекта представляет собой один из вариантов модели объекта, полученный в результате упрощения.

Следует отметить, что сложность представления объекта прямо пропорциональна объему вычислений, производимому в графической системе [6]. Поэтому для сокращения времени обработки необходимо переключать уровни детальности объекта.

**Определение 3**. *Переключение уровня детальности* подразумевает замену полигональной модели на модель с большей или меньшей геометрической сложностью.

**Определение 4**. *Выбор уровня детальности* предполагает определение конкретного геометрического представления объекта в соответствии с условиями визуализации.

Такими условиями могут быть расстояние от точки наблюдения до объекта [2 – 4, 9, 20, 23], приоритет объекта [20], площадь проекции ограничивающего объема на картинную плоскость [26], угол обзора камеры наблюдения [5].

**2. Обзор алгоритмов изменения уровня детальности**. Из-за того, что в настоящее время существует множество алгоритмов, обзор не претендует на абсолютную полноту, но в некоторой степени отражает основные направления работ, проводимых в рассматриваемой области. Все алгоритмы формирования объектов, представленных с различной степенью детализации, принято делить на три основные группы [12]: упрощение, переключение уровней детальности, выбор уровня детальности.

**2.1. Алгоритмы упрощения**. Алгоритмы этой группы формируют по исходному представлению объекта его упрощенное представление. Классификация таких алгоритмов производится на основе общих

принципов упрощения. Существуют следующие алгоритмы этой группы: кластеризация вершин, удаление вершин, стягивание ребра, использование ограничивающих поверхностей.

**2.1.1. Кластеризация вершин**. На вход алгоритма подается трехмерная решетка точек и каждая из вершин объекта переносится в ближайшую точку решетки. Те полигоны, у которых после выполнения операции стягивания оказывается менее трех различных вершин, отбрасываются [25]. Этот метод быстр в силу своей линейности и легко реализуем. Как правило, точность, с которой он воспроизводит модель, неотличима от оригинала, если расстояния между узлами решетки будут сравнимы с размерами граней объектов.

**2.1.2. Удаление вершин**. Алгоритмы, основанные на этом принципе, описываются в работах [13, 27, 28]. В этом случае удаляется одна из вершин сети. Вместе с удалением вершины удаляются грани, в которые входит эта вершина. Таким образом, остается отверстие, которое необходимо заполнить. Заполнение происходит в несколько этапов. Вначале определяется треугольник, вписанный в отверстие. Добавление треугольника создает три (или меньше) отверстия, которые также необходимо заполнить (рис. 2). Процесс продолжается до тех пор, пока все отверстия не будут заполнены.

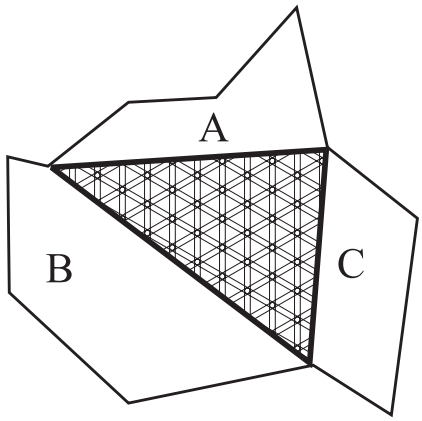


Рис. 2. Заполнение: добавление треугольника в отверстие создает три меньших отверстия

**2.1.3. Стягивание ребра**. Стягивание ребра представляет собой слияние двух вершин, образующих ребро, в некоторую точку, которая может быть найдена, например, как среднее арифметическое координат этих вершин (рис. 3). Алгоритмы, использующие этот принцип, описаны в работах [15, 17 – 19, 24].

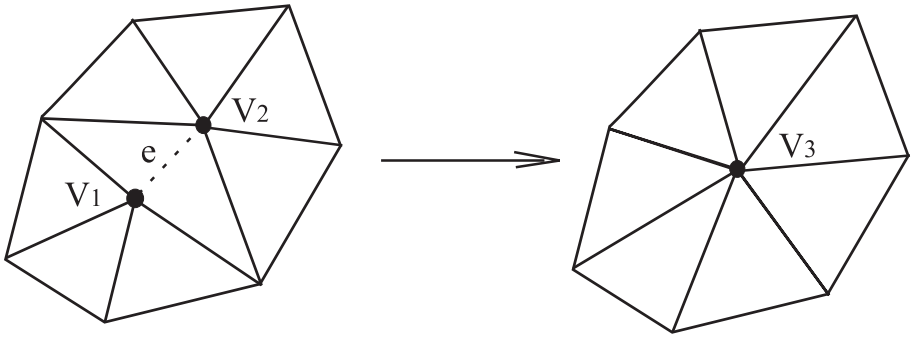


Рис. 3. Стягивание ребра

**2.1.4. Использование ограничивающих поверхностей**. Данный алгоритм позволяет максимально приблизить аппроксимированную поверхность к исходной путем введения двух ограничивающих поверхностей. На начальном этапе алгоритма формируются две поверхности, расположенные по обе стороны от первоначальной (рис. 4). Упрощение производится путем удаления вершин. Сохранение формы объекта основано на условии, что аппроксимированная поверхность не может пересекать ограничивающие [13].

**2.2. Переключение уровней детальности**. По рассмотренным выше алгоритмам можно сфор-

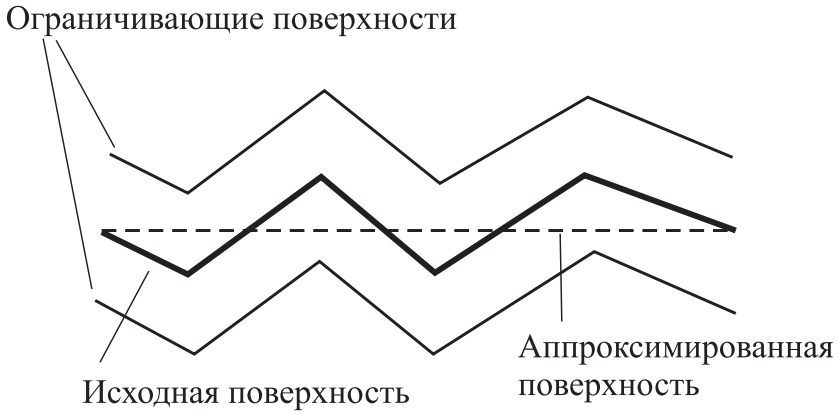


Рис. 4. Введение ограничивающих поверхностей

мировать объекты с различной степенью детальности. Однако для использования разных представлений надо заменять одно представление модели на другое. При этом необходимо стремиться к тому, чтобы процесс перехода с одного уровня на другой был бы как можно более незаметным для пользователя графической системы. Выделяют следующие алгоритмы переключения уровней детальности: дискретный LOD, LOD-смешение, альфа-LOD, непрерывный LOD и геоморфный LOD.

**2.2.1 Дискретный LOD.** При использования этого метода предварительно формируются несколько уровней детализации объекта [21]. В зависимости от расстояния до наблюдателя выбирается то или иное представление объекта в текущий момент времени. При использовании дискретных моделей происходит резкое переключение, что негативно сказывается при визуализации сцены.

**2.2.2. LOD-смешение**. Более приемлемый способ переключения уровней предполагает смешение (blend) двух соседних уровней детальности в течение короткого промежутка времени. Это выполняется для сглаживания контура объекта. Следует отметить, что с точки зрения вычислительных затрат такой процесс достаточно трудоемок, так как в один и тот же момент времени приходится отображать поверхности двух объектов. Это обстоятельство компенсируется тем фактом, что визуализация двух объектов происходит на незначительном временно́м отрезке. При этом переключение уровней детальности всех объектов сцены происходит в разное время. Алгоритмы, построенные на этом принципе, состоят в следующем. В системе RGBA, помимо трех основных цветовых компонент (R — красный, G — зеленый, B — синий), существует четвертый (альфа-компонент), который принято называть альфа-каналом. Он также сохраняется в буфере кадра, как и значения RGB-компонентов. Значения альфа-составляющей трактуется как значение параметров прозрачности (transparency) или непрозрачности (opacity). Прозрачность и непрозрачность являются двумя взаимно дополняющими параметрами, характеризующими оптические свойства среды [11]. Для системы заданное значение интенсивности компоненты A определяет коэффициент пересчета значений основных цветов при записи их в буфер кадра. В результате получается, что в код засветки пикселя вносят свой вклад несколько геометрических объектов и формируется смешанное (или комбинированное) изображение (рис. 5).

Рис. 5. Пример LOD-смешения

Обычно значение A меняется в диапазоне от 1 до 0, где 1 соответствует абсолютно непрозрачной поверхности и 0 — идеально прозрачной поверхности. Таким образом, LOD1 на первоначальной фазе переключения имеет коэффициент прозрачности 1, а LOD2 — 0. Постепенно эти коэффициенты изменяются, и таким образом происходит плавный переход с одного уровня представления на другой [16, 29].

**2.2.3. Альфа-LOD**. Этот метод считается наиболее простым. Для его реализации используется только один уровень подробности объекта [12]. При удалении от наблюдателя изменяется коэффициент A.

На определенном расстоянии коэффициент A становится равным 0. В этом случае объект оказывается невидимым и его обработка в графической системе перестает иметь смысл. Таким образом, удаление объекта из сцены происходит практически незаметно для пользователя.

**2.2.4. Непрерывный LOD (Continuous Level of Detail — CLOD)**. При непрерывном изменении детальности происходит последовательное удаление элементов полигонального объекта. В этом случае может происходить удаление ребра, вершины и т.д. Пусть визуализируемый объект находится на расстоянии 100 м, на этом расстоянии он имеет уровень подробности 1000 треугольников; тогда на расстоянии 101 м он будет состоять уже из 995 треугольных граней. Процесс упрощения при реализации таких алгоритмов характеризуется плавностью переходов между уровнями детальности [12].

**2.2.5. Геоморфный LOD**. В этом случае предварительно создаются несколько дискретных моделей объекта. При переключении между двумя предварительно сгенерированными уровнями вычисляются интерполированные значения координат вершин [19, 22], что также обеспечивает плавность переключения.

**2.3. Выбор уровня детальности**. Алгоритмы этой группы определяют, какой уровень детальности объекта используется в данный момент времени и когда необходимо осуществить переход на следующий уровень. Критерием использования того или иного уровня детализации является приемлемое качество изображения при наименьших вычислительных затратах. Известны несколько способов выбора уровня детализации, которые могут использовать диапазоны, перекрывающиеся диапазоны, площади проекций, приоритеты визуализируемых объектов.

**2.3.1. Использование диапазонов**. Является самым простым способом выбора LOD. В этом случае все наблюдаемое пространство разбивается на диапазоны. Начало первого диапазона находится в точке наблюдения. Далее каждому диапазону присваивается свой уровень представления объекта (рис. 6).

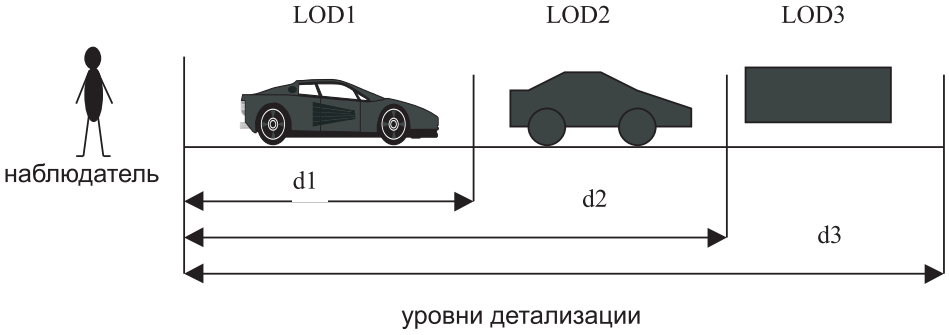


Рис. 6. Диапазоны LOD

**2.3.2. Перекрывающиеся диапазоны**. Для того чтобы избежать частого нежелательного переключения между уровнями детальности, используются перекрывающиеся диапазоны [20, 23]. На рис. 7 верхние значения диапазонов представлены для случая, когда расстояние между объектом и наблюдателем увеличивается, нижние значения — уменьшаются.

**2.3.3. Выбор, основанный на площади проекций**. В этом случае оценивается проектируемая область ограничивающего объема. Ограничивающий объем может быть представлен сферами, параллелепипедами, эллипсоидами и т.д. Принцип подобных алгоритмов основан на том факте, что при удалении от наблюдателя перспективная проекция ограничивающего объема на экранную плоскость также уменьшается (рис. 8).

Один из таких алгоритмов использует ориентированные прямоугольные оболочки (OBB — Oriented Bounding Box). Его идея состоит в том, что определяются те грани ориентированного объема, которые видны с точки зрения наблюдателя (рис. 9). Этот процесс выполняется с использованием специальных таблиц (Look-up Table). Зная вершины контура проекции, можно определить площадь проекции ограничивающего объема на картинную плоскость [26].

Другой алгоритм выбора уровня детальности использует угол наклона между вектором направления взгляда и плоскостью полигона, а также расстояние между объектом и наблюдателем (рис. 10).

Размер видимой части объекта определяется соотношением

$$r = \frac{a(n \cdot v)}{d \cdot d},$$

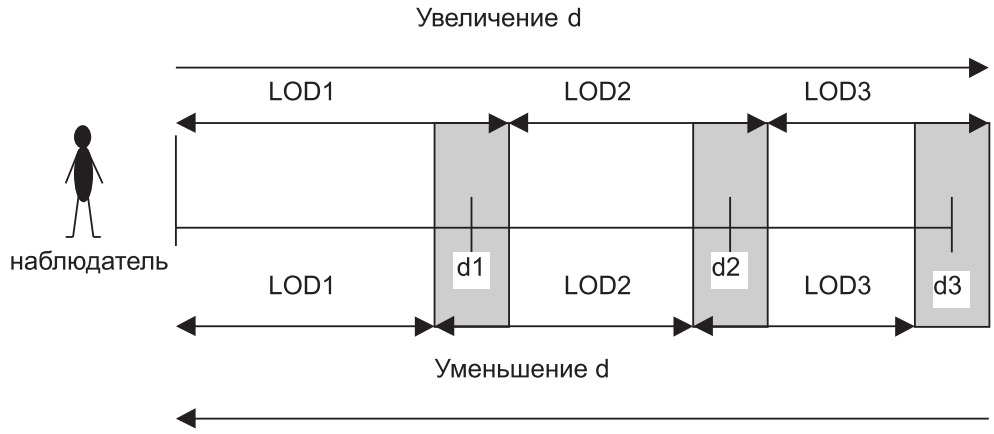


Рис. 7. Перекрытие диапазонов

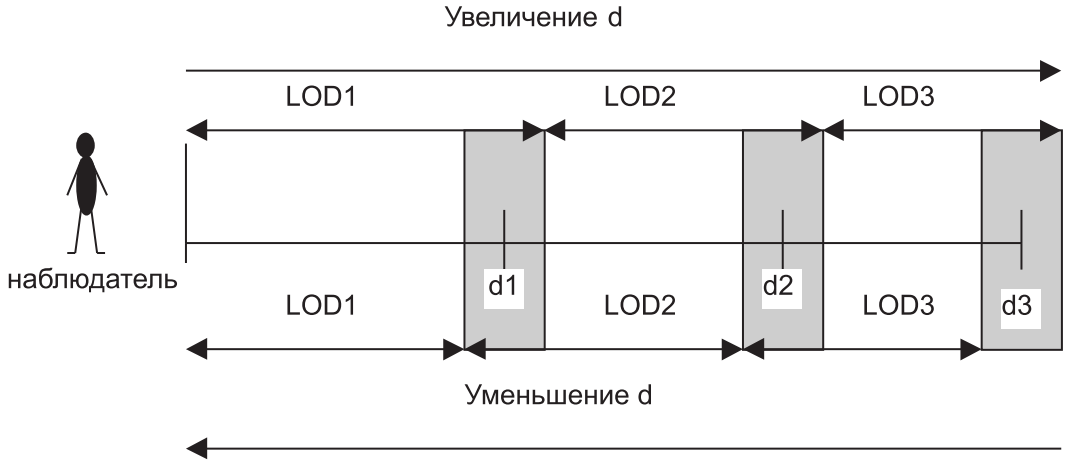


Рис. 8. Изменение площади проекции ограничивающего объема на картинную плоскость

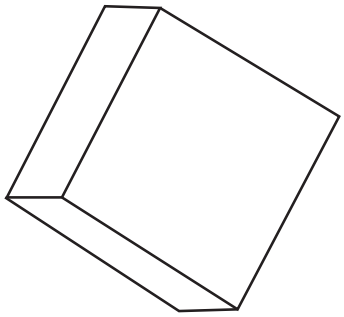

Рис. 9. Ориентированный прямоугольный объем (OBB)

где $a$ — площадь полигона, $n$ — нормаль к полигону, $v$ — вектор направления взгляда, $d$ — расстояние между точкой наблюдения и полигоном. Из этого соотношения следует, что размер видимой части будет максимален, когда плоскость грани перпендикулярна вектору направления взгляда. Кроме того, величина $r$ обратно пропорциональна квадрату расстояния от объекта до наблюдателя [14].

**2.3.4. Приоритет объекта**. Этот способ выбора уровня детальности основан на предположении о том, что не обязательно все объекты сцены следует отображать с одинаковой детальностью. Наиболее детально должны быть представлены те объекты, на которых в текущий момент времени сконцен-

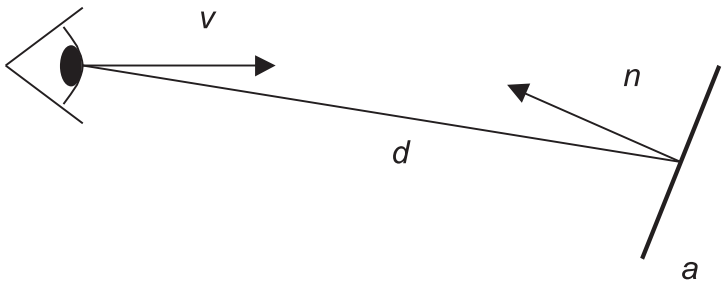


Рис. 10. Зависимость размера видимой части от угла наклона и расстояния до наблюдателя

трировано внимание пользователя. Остальные объекты могут быть представлены с меньшей степенью подробности [20].

**3. Алгоритм упрощения полигонального поля на одну вершину за шаг**. В применяемом непрерывном алгоритме упрощения происходит удаление одной вершины за шаг, чем обеспечивается последовательное изменение геометрической сложности. Грани, в которые входит вершина, также удаляются (рис. 11).

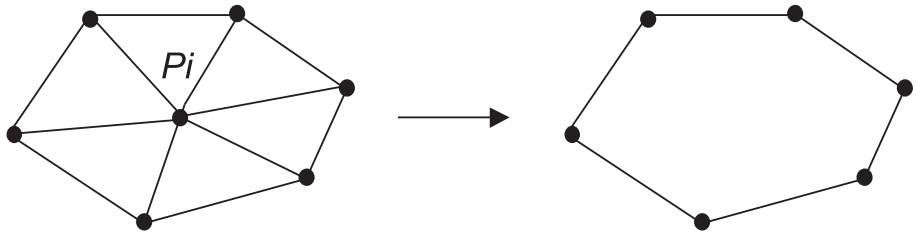


Рис. 11. Удаление вершины и граней, в которые она входит

Оставшийся после удаления вершины контур подвергается триангуляции. Следует стремиться к тому, чтобы полученные треугольники становились равноугольными. Наилучший результат по этому критерию дает триангуляция Делоне [8]. Для триангуляции используется пошаговый алгоритм, в котором выбирается некоторая базовая начальная линия [10]. В качестве базовой линии берется один из отрезков контура. После этого для базовой линии необходимо найти ближайший узел контура. В пошаговом алгоритме для поиска соседа нужно выбрать среди всех точек $P_i$ такую, чтобы угол $AP_iB$ был максимальным (рис. 12). Найденный сосед соединяется с концами базовой линии и образует $\Delta AP_iB$. Новые ребра $AP_i$ и $BP_i$ помечаются как новые базовые линии и процесс поиска продолжается.

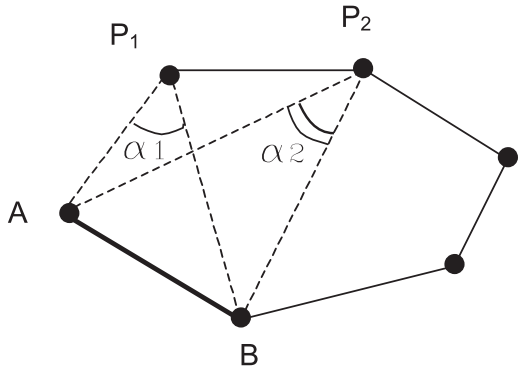


Рис. 12. Триангуляция контура с использованием пошагового алгоритма

**4. Выделение опорных вершин**. Следует отметить, что необходимо удалять только те вершины,

которые не влияют существенно на форму объекта. Вершины, в которых поверхность объекта значительно изменяется, будем называть опорными. Такие вершины выделяются на стадии предварительной обработки и постоянно присутствуют при синтезе объекта. Выделение экстремальных вершин происходит следующим образом.

Шаг 1. Строится ограничивающий объем, представляющий собой параллелепипед, стороны которого параллельны осям координат. Координаты вершин параллелепипеда находятся как значения минимальных и максимальных вершин объекта по трем осям:

$$x_{\min} = \min(x_1, x_2, \ldots, x_n), \quad x_{\max} = \max(x_1, x_2, \ldots, x_n),$$
$$y_{\min} = \min(y_1, y_2, \ldots, y_n), \quad y_{\max} = \max(y_1, y_2, \ldots, y_n),$$
$$z_{\min} = \min(z_1, z_2, \ldots, z_n), \quad z_{\max} = \max(z_1, z_2, \ldots, z_n).$$

Шаг 2. Строится октодерево путем последовательного разбиения каждой области на две части (рис. 13). Координаты ограничивающих объемов находятся из предыдущих значений следующим образом:

$$x_{c_k} = \frac{x_{\min} + x_{\max}}{2k_x}, \quad y_{c_k} = \frac{y_{\min} + y_{\max}}{2k_y}, \quad z_{c_k} = \frac{z_{\min} + z_{\max}}{2k_z},$$

где $k_x$, $k_y$, $k_z$ — количество разбиений по осям $X$, $Y$, $Z$.

Шаг 3. Определяется принадлежность вершин к каждой области разбиения путем проверки условий

$$x_{\min} \leqslant x_i \leqslant x_{\max}, \quad y_{\min} \leqslant y_i \leqslant y_{\max}, \quad z_{\min} \leqslant z_i \leqslant z_{\max}.$$

Шаг 4. Определяется квадрат расстояния от вершин области до центров областей разбиения:

$$d_i^2 = (x_{c_{k+1}} - x_i)^2 + (y_{c_{k+1}} - y_i)^2 + (z_{c_{k+1}} - z_i)^2.$$

Шаг 5. Находятся минимальные и максимальные квадраты расстояний. Вершины, расположенные на минимальном и максимальном расстояниях, удаляться не будут.

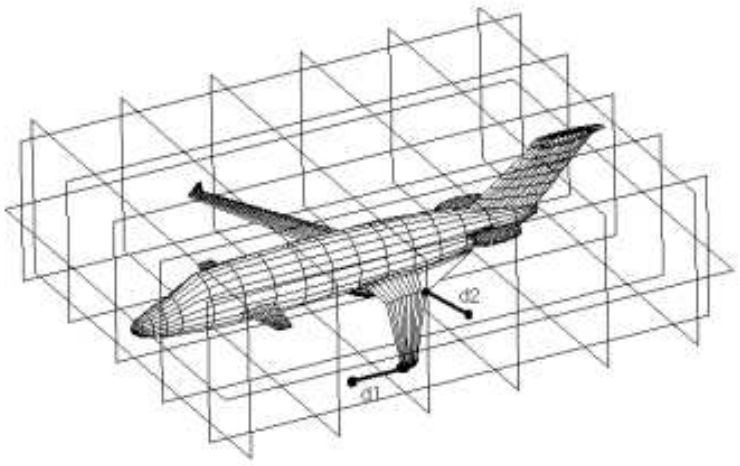


Рис. 13. Обнаружение опорных точек на основе октодерева

Рассмотрим графики, демонстрирующие зависимость площади поверхности объекта от количества присутствующих в нем граней (рис. 14). Кривая 1 показывает изменение площади поверхности объекта от количества граней при уменьшении детализации без определения опорных точек. Кривая 2 показывает ту же зависимость при использовании опорных точек, найденных по максимальному квадрату расстояния. На кривой 3 отображено изменение детальности, когда опорными являются точки, найденные как по максимальному, так и по минимальному критериям.

**5. Выбор уровня детальности**. Выше было представлено, как можно получить модель объекта с любым уровнем детализации. Однако, кроме этого, необходимо автоматически переключать уровни детализации в соответствии с расстоянием от точки наблюдения до объекта. Для нашего алгоритма необходимо

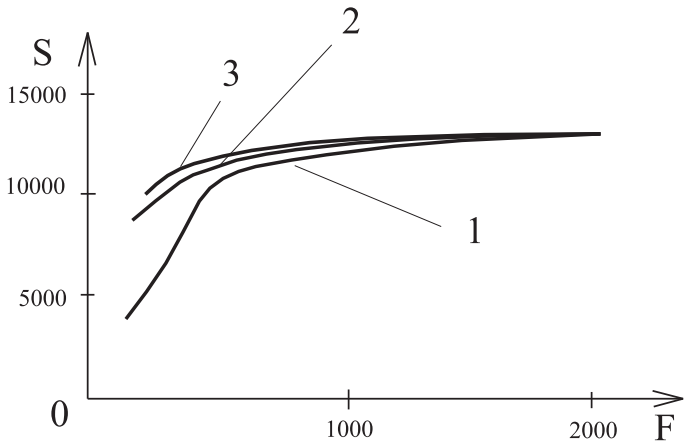


Рис. 14. Зависимость площади объекта от количества граней

определить соотношение между количеством удаляемых точек и расстоянием до объекта. Вычисления, на которых основан алгоритм, являются приближенными, так как грани, образующие поверхность, не могут иметь в точности одинаковые размеры, и площадь поверхности по мере упрощения незначительно уменьшается. Поэтому, например, погрешность в два-три десятка граней не скажется на качестве отображения объекта, представленного тысячей полигонов.

На рис. 15 изображено перспективное изменение размеров объекта в соответствии с расстоянием от него до точки наблюдения. Ось $Y$ характеризует высоту объекта, а ось $Z$ — расстояние от объекта до наблюдателя. Плоскость проекции помещена перед центром проецирования на расстоянии $d$ перпендикулярно к оси $Z$. Тогда точка с координатами $(h, z)$ переместится в точку $(h_p, d)$. Из подобия треугольников, показанных на этом рисунке, справедливо соотношение

$$h_p = \frac{h}{z/d}, \tag{2}$$

где $h$ — высота объекта, $h_p$ — размер объекта по оси $Y$ на плоскости проекции, $z$ — расстояние от точки наблюдения до объекта, $d$ — расстояние от точки наблюдения до картинной плоскости.

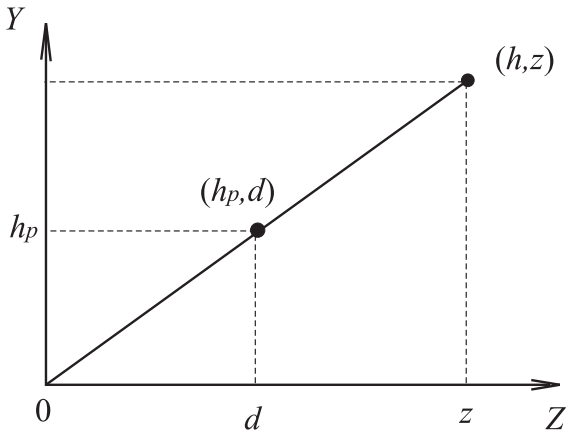


Рис. 15. Перспективная проекция

Таким образом, на определенном расстоянии размер проекции грани на картинную плоскость будет меньше размера пикселя экрана и обработка данной грани не будет иметь смысла. Предлагается за счет сокращения количества граней увеличивать их размеры, оставляя тем самым размер проекции грани на картинную плоскость приблизительно постоянным. Из графиков, показанных на рис. 14, видно, что площадь поверхности объекта является приблизительно постоянной величиной на различных уровнях

детализации. Это означает, что можно сократить геометрическую сложность объекта в четыре раза, а площадь поверхности практически не изменится. Предполагается, что все грани имеют приблизительно одинаковые размеры. Тогда длина ребра грани равна $h$ и из площади треугольника ее значение можно определить как

$$h = \sqrt{2S_F}, \tag{3}$$

где $S_F$ — средняя площадь одной грани, которая приближенно равна

$$S_F \approx \frac{S}{F}, \tag{4}$$

где $S$ — общая площадь поверхности объекта, $F$ — количество граней. Учитывая выражения (2), (3) и (4), средний размер проекции грани, расположенной перпендикулярно оси $Z$, имеет вид

$$h_p \approx \frac{\sqrt{2S/F}}{z/d}.$$

Рассмотрим два уровня детализации одного и того же объекта, расположенного на расстояниях $z_1$ и $z_2$ от наблюдателя, со следующими размерами проекций ребер:

$$h_{p_1} \approx \frac{\sqrt{2S/F_1}}{z_1/d}, \quad h_{p_2} \approx \frac{\sqrt{2S/F_2}}{z_2/d}.$$

На разных расстояниях размеры проекций ребер должны быть одинаковыми: $h_{p_1} \approx h_{p_2}$. Отсюда получаем соотношение между числом граней и расстоянием от объекта до наблюдателя на двух уровнях детализации:

$$F_2 \approx F_1 \left(\frac{z_1}{z_2}\right)^2. \tag{5}$$

Изменение количества граней в зависимости от расстояния демонстрирует график на рис. 16.

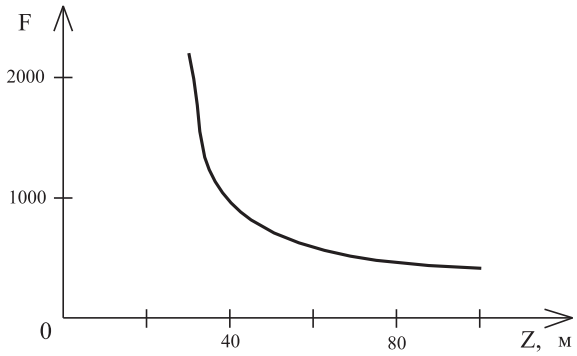


Рис. 16. Зависимость количества граней от расстояния между объектом и наблюдателем

Поскольку в алгоритме удаляется по одной вершине за шаг, то найдем количество вершин полиэдра, которые надо удалить при перемещении объекта по оси $Z$. Обозначим соотношение количества вершин на разных уровнях детальности следующим образом:

$$V_2 = V_1 - T, \tag{6}$$

где $T$ — количество удаляемых вершин. Подставляя выражение (6) в (1) получим

$$F_2 \leqslant 2(V_1 - T) - 4. \tag{7}$$

Из выражений (5) и (7) находим

$$T \approx (V_1 - 2)\left(1 - \left(\frac{z_1}{z_2}\right)^2\right).$$

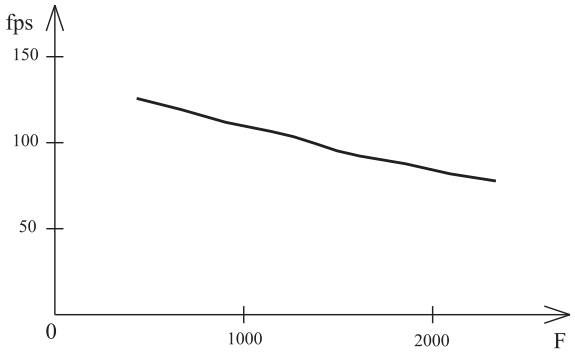


Рис. 17. Зависимость числа кадров в секунду от геометрической сложности объекта

**6. Результаты**. Эффект от применения уровней детализации заключается в следующем. При общем визуальном подобии визуализируемого объекта общее число граней заметно сокращается и таким образом уменьшается общее количество вычислений, выполняемых системой. Результаты проведенных экспериментов демонстрирует график на рис. 17. На нем отображена зависимость количества кадров в секунду (frames per second — fps), воспроизводимых графической системой, от числа граней, образующих объект.

На рис. 18 приведены поверхности одного и того же объекта при различных уровнях детализации, полученных по описанному выше алгоритму.

Рис. 18. Уровни детализации объекта

**Заключение**. Таким образом, можно сказать, что алгоритм выбора уровня детальности устанавливает зависимость между расстоянием от объекта до наблюдателя и количеством граней, образующих поверхность. Кроме того, использование уровней детализации с применением разработанного алгоритма значительно сокращает количество вычислений. Однако необходимо отметить, что описанный способ применим только для объектов, размеры которых значительно меньше размеров синтезируемых пространств (так, например, невозможно использовать приведенный алгоритм для протяженной поверхности ландшафта).

СПИСОК ЛИТЕРАТУРЫ

1. *Гусев А.В., Ивашин С.Л., Талныкин Э.А.* Математические модели сцен в синтезирующих системах визуализации реального времени // Автометрия. 1985. № 4. 3–9.
2. *Захаров А.А., Масанов А.Н.* Некоторые задачи представления местности для тренажеров наземного транспорта. Деп. в ВИНИТИ РАН 28.03.2002. № 561-В2002.
3. *Захаров А.А. Брагин П.А.* Разработка методов изменения геометрической сложности графических объектов для систем генерации визуальной обстановки // Методы и устройства передачи и обработки информации. Вып. 2. СПб.: Гидрометеоиздат, 2002. 88–92.
4. *Захаров А.А., Масанов А.Н.* Синтез изображений протяженных участков местности // Методы и устройства передачи и обработки информации. Вып. 2. СПб.: Гидрометеоиздат, 2002. 98–102.
5. *Захаров А.А.* Имитация кратности приборов наблюдения для задач тренажеров транспортной техники // Системы управления и информационные технологии. Воронеж: Центрально-Черноземное книжное издательство, 2002. 145–149.
6. *Мартинес Ф.* Синтез изображений. Принципы, аппаратное и программное обеспечение. М.: Радио и связь, 1990.

7. *Палташев Т.Т., Климина С.И., Лях А.С.* Технология визуализации в компьютерном синтезе реалистичных изображений // Зарубежная радиоэлектроника. 1984. № 8. 64–79.
8. *Препарата Ф., Шеймос М.* Вычислительная геометрия. М.: Мир, 1989.
9. *Роджерс Д.Ф.* Алгоритмические основы машинной графики. М.: Мир, 1989.
10. *Скворцов А.В.* Алгоритмы построения триангуляции с ограничениями // Вычислительные методы и программирование. 2002. **3**, № 1. 86–96 (http://num-meth.srcc.msu.su).
11. *Эйнджел Э.* Интерактивная компьютерная графика. Вводный курс на базе OpenGl. М.: Издательский дом "Вильямс", 2001.
12. *Akenine-Miller T., Haines E.* Real-time rendering. Natick: A.K. Peters Ltd., 2002.
13. *Cohen J., Varshney A., Manocha D., Turk G., Weber H., Agarwal P., Brooks F., Wright W.* Simplification envelopes // Proc. of SIGGRAPH-96. New York: ACM SIGGRAPH, 1996. 119–128.
14. *Coorg S., Teller S.* Real-time occlusion culling for models with large occluders // Proc. 1997 Symposium on Interactive 3D-Graphics. New York: ACM SIGGRAPH, 1997. 83–90.
15. *Garland M., Heckbert P.S.* Surface simplification using quadric error metrics // Proc. of SIGGRAPH-97. New York: ACM SIGGRAPH, 1997. 209–216.
16. *Gigus Z., Canny J., Seidel R.* Efficiently computing and representing aspect graphs of polyedral objects // IEEE Transactions On Pattern Analysis and Machine Intelligence. 1991. **13**, N 6. 542–551.
17. *Gueziec A.* Surface simplification inside a tolerance volume // Second Annual International Symposium on Medical Robotics and Computer Aided Surgery. Yorktown: Yorktown Heights, 1995. 132–139.
18. *Hoppe H., Derose T., Duchamp T., Mcdonald J., Stuetzle W.* Mesh optimization // Proc. of SIGGRAPH-93. New York: ACM SIGGRAPH, 1993. 19–26.
19. *Hoppe H.* Progressive meshes // Proc. of SIGGRAPH-96. New York: ACM SIGGRAPH, 1996. 99–108.
20. *King Y.* Never let'em see your pop-issues in geometric level of detail selection // Game Programming Gems. New York: Academic Press, 2000. 432–438.
21. *Luebke D.P.* A developer's survey of polygonal simplification algorithms // IEEE Computer Graphics and Applications. 2001. **21**, N 3. 24–35.
22. *Melax S.* A simple, fast, and effective polygon reduction algorithm // Game Developer. 1998. **5**, N 11. 44–49.
23. *Rohlf J., Helman J.* IRIS performer: a high performance multiprocessing toolkit for real-time 3D Graphics // Proc. of SIGGRAPH-94. New York: ACM SIGGRAPH, 1994. 381–394.
24. *Ronfard R., Rossignac J.* Full-range approximation of triangulated polyhedra // Computer Graphics Forum (Proc. of Eurographics-96). 1996. **15**, N 3. 67–76.
25. *Rossignac J., Borrel P.* Multi-resolution 3D approximations for rendering complex scenes // Modeling in Computer Graphics. Berlin: Springer-Verlag, 1993. 455–465.
26. *Schmalstieg D., Tobler R.F.* Fast projected area: computation for three-dimensional bounding boxes // Journal of Graphics Tools. 1999. **4**, N 2. 37–43.
27. *Schroeder W.J., Zarge J.A., Lorensen W.E.* Decimation of triangle meshes // Computer Graphics Forum (Proc. of SIGGRAPH-92). 1992. **26**, N 2. 65–70.
28. *Turk G.* Re-tiling polygonal surfaces //Computer Graphics Forum (Proc. of SIGGRAPH-92). 1992. **26**, N 2. 55–64.
29. *Znou Kun, Pan Zhi-Geng, Shi Jiao-Ying.* Smooth transition between levels of detail of models // Aided Design and Computer Graphics. 2000. N 6. 463–467.

Поступила в редакцию
02.12.2002