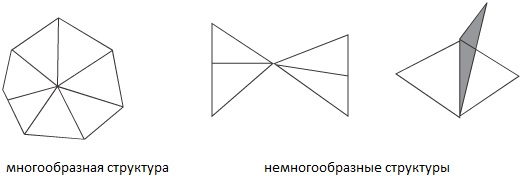
# ОБЗОР МАТЕМАТИЧЕСКИХ МЕТОДОВ

Существуют два основных подхода к решению задачи упрощения триангуляционных моделей: итеративный и глобальный. Итеративный подход позволяет последовательно упростить модель путем применения некоторого локального оператора. Глобальный же, в свою очередь, применяется к исходной модели в целом. Итеративный подход, как правило, используется более широко, так как с его помощью удобнее контролировать результат, степень упрощения сетки и т.п. Например, используя такой алгоритм, пользователь может устанавливать точное число граней результирующей модели или же выбирать в качестве критерия останова некое пороговое значение ошибки. Исходя из этого, мы будем рассматривать далее итеративные алгоритмы. Между собой их можно разделить на несколько категорий, так как каждый из алгоритмов может быть использован для своей специфической задачи. Одной из таких категорий является поддерживание упрощения немногообразных поверхностей.

## Немногообразные структуры

Тела, представляющие собой многообразия, можно иначе описать как тела с замкнутым объемом. Нарушениями условия многообразности являются, например, такие явления как: касание двух поверхностей в одной точке, касание двух поверхностей вдоль открытой или замкнутой кривой, два замкнутых объема с общей гранью, ребром или вершиной, ребро, выступающее из точки на поверхности. Примеры многообразных и немногообразных моделей представлены на рис. 1.1.

Рисунок 1.1 – Сравнение многообразных и немногообразных структур

Рассмотрим подробнее различие между моделями, являющимися многообразиями, и моделями, не являющимися таковыми. В многообразии каждая точка на поверхности является двумерной. Другими словами, хотя поверхность существует в трехмерном пространстве, с топологической точки зрения она является плоской, если рассматривать достаточно малый ее участок в окрестности любой заданной точки. В модели, не являющейся многообразием, окрестность некоторой точки на поверхности не обязана быть плоской. Точка может быть пересечением двух и более топологически плоских поверхностей или плоской поверхности и одномерной кривой.

## Алгоритмы упрощения многообразных триангуляционных поверхностей

Хоппе и др. [1] предложили алгоритм, цель которого заключается в том, чтобы получить такую сетку, которая бы наилучшим образом соответствовала множеству точек, расположенному по поверхности, и при этом содержала как можно меньшее количество вершин. Такое множество точек состоит из первоначальных вершин, составляющих поверхность, а так же дополнительных, которые размещаются на границах и равномерно по всей поверхности. Для того, чтобы достигнуть цели, необходимо найти такой симплициальный комплекс K и такое множество вершин V, определяющих сетку M = (K, V), которые минимизируют следующую энергетическую функцию:

E(K, V) = Edist(K, V) + Erep(K) + Espring(K, V).

Рассмотрим составляющие этой функции подробнее.

Edist(K, V) представляет собой сумму квадратов расстояний до сетки от точек X:

Стоит отметить, что изначально точки X лежат на поверхности сетки, однако в ходе итераций алгоритма и изменения топологии, сетка может отклоняться от этих точек на некоторое расстояние.

Erep(K), в свою очередь, это штрафная составляющая, которая увеличивается, если вершин становится больше, и описывается следующей формулой:

Erep(K) = crepm,

где m – количество вершин;

crep – параметр, определяемый пользователем и контролирующий компромисс между степенью упрощения сетки и точностью геометрического соответствия.

Espring(K, V) отвечает за растяжение ребер относительно их первоначального положения и определяется по формуле:

где k – коэффициент упругости, установленный заранее.

Сокращение сетки происходит итеративно, путем выполнения допустимых операций на её ребрах: удаление ребра, поворот ребра (смена на ребро, соединяющего две противоположные вершины), добавление нового ребра путем расщепления текущего (добавление вершины в средину ребра и соединение её с оставшимися вершинами внутри текущего треугольника). Выбор допустимых действий производится посредством процесса оптимизации энергетической функции. На каждом шаге удаляется элемент, удаление которого дает наименьшее увеличение энергетической функции. Стоит также отметить, что при удалении ребра, новая вершина помещается либо в одну из координат вершин, расположенных на концах этого ребра, либо посередине.

Позже Хоппе [2] установил, что на самом деле для эффективного упрощения сеток достаточно всего одной операции: удаления ребра. Также предлагается модифицировать энергетическую функцию следующим образом:

E(M) = Edist(M) + Espring(M) + Escalar(M) + Edisc(M).

Edist(M) определяется аналогичным предыдущему алгоритму образом.

Для определения Espring(M) внесли некоторые изменения. Предлагается не задавать коэффициент упругости как константу, а менять его в ходе решения задачи. Таким образом, Espring(M) постоянно уменьшается и адаптируется при упрощении сетки.

Escalar(M) измеряет точность скалярных атрибутов сетки (например, таким атрибутом может быть цвет), а Edisc(M) – геометрическую точность кривых разрыва сетки. Введение в функцию таких составляющих вместо Erep(K), позволяет избежать неоднозначного параметра crep, а также позволяет указывать явное количество граней, которые необходимо получить в оптимизированной сетке. Также они дают возможность сохранить не только геометрию сетки, но и её внешний вид.

Из недостатков данного подхода можно выделить следующие. Рассмотрим ситуацию удаления ребра. Как было сказано ранее, новую вершину предлагалось поместить либо в одну из координат вершин, принадлежащих ребру, либо посередине. Можно предположить, что это не самое эффективное решение, и можно было бы вместо этого рассмотреть функцию ошибки, связанную с операцией сжатия, и попытаться минимизировать ее значение по сравнению с пространством возможных мест размещения вершин. Также, не смотря на очень хорошие результаты, которые дает алгоритм, вычисление оптимальной целевой позиции для данного сжатия является нелинейной проблемой и на практике довольно неэффективно, а также сложно в реализации. Кроме того, формулировка такой функции ошибки существенно ограничивает ее применение относительно многообразия сетки.

Другим подходом к упрощению многообразия сеток является реконструирование поверхности, которое включает в себя определение нового набора вершин, проецируемых на поверхность исходной сетки, а затем использование этих вершин для формирования новых граней поверхности. Алгоритм реконструирования поверхности, представленный Турком [3], проектирует новый набор вершин на поверхность сетки, как правило, по определенному шаблону, хотя кривизна поверхности может быть учтена. Точность этого подхода сильно зависит от метода, выбранного для проектирования нового набора вершин на поверхность.

Алгоритм упрощения, основанный на вокселях, был представлен Хи и др. [4]. Он сегментирует модель в сетку на основе вокселей, а затем применяет фильтр нижних частот для устранения вокселей с наименьшими перекрывающимися объемами с исходной сеткой. Затем алгоритм использует алгоритм маршевых кубов [5] для генерации новой сетки на основе оставшихся вокселей. Из-за подхода фильтрации, основанного на объеме, воксельное упрощение плохо работает на моделях с острыми краями и квадратными углами.

Коэн и др. [6] представили другой подход к упрощению многообразной сетки с использованием двух смещенных копий каждой поверхности, называемых упрощающими огибающими. Эти смещения использовались для обеспечения того, чтобы упрощенная поверхность оставалась в объеме, образованном огибающими. В то время как упрощающие огибающие могут гарантировать высокую степень точности, учитывая расстояние смещения, используемое для огибающих, расстояние также служит нижней границей для алгоритма и предотвращает радикальное упрощение сетки.

Таким образом, было предложено много подходов по упрощению многообразных моделей. Каждый подход имеет свои сильные и слабые стороны с точки зрения точности упрощенных сеток, способности определять степень ошибки, времени выполнения и сложности реализации.

## Алгоритмы упрощения триангуляционных поверхностей, поддерживающих немногообразие

Несмотря на хороший результат, который показывают алгоритмы, работающие только с многообразными моделями, существует довольно большое количество прикладных задач, в которых используются модели, содержащие немногообразные элементы. Рассмотрим наиболее известные алгоритмы, которые были разработаны в этой сфере.

### Vertex Decimation

Одним из первых алгоритмов упрощения сетки, способных обрабатывать немногообразные поверхности, был представлен Шроудером и др. [7]. Алгоритм прореживания работает, делая несколько проходов по всем вершинам модели. Во время каждого прохода для каждой вершины алгоритм определяет, можно ли удалить вершину без нарушения топологии локальной окрестности граней. Если результирующая поверхность будет находиться на заданном пользователем расстоянии от первоначальной сетки, алгоритм удаляет вершину и все связанные с ней грани. Полученное отверстие в сетке заполняется путем локальной триангуляции. Процесс удаления вершин повторяется с возможной корректировкой критерия удаления, пока не будет достигнуто условие выхода. Таким условием, например, может быть процентное уменьшение исходной сетки.

Представленный алгоритм децимации способен обрабатывать немногообразные поверхности в случае, если он не будет удалять немногообразные вершины во время итераций прореживания. Однако это правило накладывает ограничение снизу для алгоритма прореживания, так как он не может создать низкополигональную модель с меньшим количеством вершин, чем число немногообразных вершин. Кроме того, определяемое пользователем расстояние оказывает существенное влияние как на нижнюю границу, так и на точность алгоритма. Также минус такого подхода в том, что упрощение применяется равномерно ко всему набору данных, что приводит к размытию данных при неравномерной структуре сетки.

### Vertex Clustering

Другим подходом к упрощению немногообразных сеток является кластеризация вершин. Алгоритм кластеризации вершин, представленный Россиньяком и Боррелом [8], присваивает значение важности каждой вершине в зависимости от размера ее смежных граней и их кривизны. Затем алгоритм применяет сегментацию поверхности: модель помещается в некий параллелепипед, состоящий из ячеек, представляющий собой 3D сетку. В зависимости от размера ячейки, в каждую ячейку может попасть как одна, так и несколько вершин. В случае попадания нескольких, рассчитывается, в каком месте этой ячейки будет расположена новая точка, которая заменит собой старые. После того, как в каждой ячейке осталось максимум по одной точке, триангуляционная модель пересчитывается.

Минус такого подхода в том, что, не смотря на быстроту работы и свою обобщенность, сложно контролировать качество получаемой модели, и оно часто оказывается довольно низким. Также, довольно проблематично задать количество желаемых граней, которые должны быть получены на выходе. Кроме того, результат может зависеть от положения модели относительно окружающего её параллелепипеда.

### Hierarchical dynamic simplification (HDS)

Другим подходом к упрощению, основанным на кластеризации вершин, является алгоритм иерархического динамического упрощения, авторами которого являются Люэбке и Эриксон [9]. Алгоритм HDS представляет всю сетку как дерево вершин, которая представляет собой иерархию, состоящую из кластеров вершин. Узлы дерева могут быть свернуты в их родительские узлы, чтобы уменьшить общее число граней или развернуты в дочерние узлы для повторного получения исходной сетки. Поскольку дерево вершин не требует вершинной связности, алгоритм HDS поддерживает упрощение немногообразных сеток. Однако точность результата, получаемая с помощью данного алгоритма, как правило оказывается более низкой по сравнению с результатами других подходов.

### QEM

Возможно, наиболее распространенным подходом к упрощению немногообразных сеток является использование квадратичной метрики ошибки (QEM), впервые представленной Гарландом и Хекбертом [10]. QEM вершины представляет собой матрицу размером 4 × 4, которая определяется как сумма квадратов расстояний от вершины до плоскостей смежных граней. Когда вершина сливается с другой вершиной в пределах заданного пользователем порога расстояния, значение ошибки может быть вычислено как сумма матриц QEM объединенных вершин, которая становится QEM новой вершины. При объединении вершин алгоритм сохраняет отсортированную очередь приоритетов всех пар вершин-кандидатов на основе их объединенного QEM. Алгоритм удаляет пару вершин с наименьшей ошибкой из ​​верхней части очереди, объединяет вершины и затем обновляет значения ошибок тех пар вершин, которые были соединены с данной. Позиция новой вершины вычисляется на основе решения задачи минимизации функции ошибки.

Данный подход позволяет получить сетки с сохранением довольно высокой точности, даже при радикальной степени упрощения. Его отличительной особенностью является то, что он позволяет упрощать несвязные модели, а также позволяет работать с немногообразными поверхностями. Кроме того, как уже было сказано, в данном алгоритме вычисляется наиболее оптимальное положение новых вершин, что не было учтено большинством других авторов.

Одним из ограничений первоначального алгоритма QEM является то, что он имеет тенденцию к большим отклонениям на граничных ребрах из-за уменьшения числа смежных граней, что приводит к возникновению большого количества пропусков на границах (рис. 1).

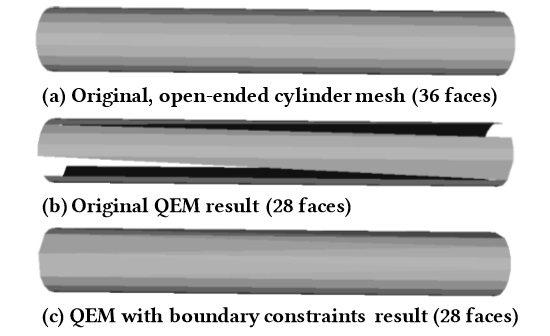


Рисунок 1 – Пример неспособности алгоритма QEM сохранить важные граничные ребра

Чтобы решить эту проблему и обеспечить сохранение границ, Гарланд и Хекберт [10] определили плоскость граничного ограничения, проходящую через каждое граничное ребро. Для каждой такой плоскости предлагается вычислить матрицу QEM, которая умножается на довольно большой постоянный весовой коэффициент, и добавить её к матрицам QEM реберных вершин. Различные исследователи также предложили взвешивать плоскость граничного ограничения как квадрат длины кромки. Тем не менее, все эти подходы приводят к тому, что граничные ребра становятся более приоритетными, чем все остальные, что приводит в результате к упрощенным сеткам с отверстиями на их поверхностях (рис. 2).

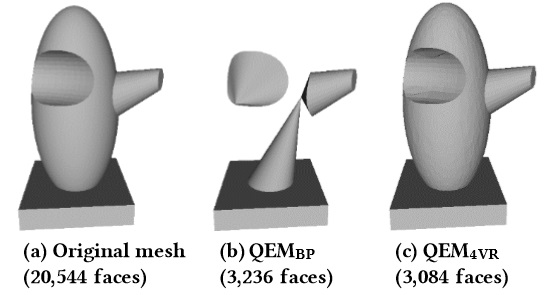


Рисунок 2 – Пример неспособности алгоритма QEMBP сохранять необходимые для поверхности ребра

Другим ограничением исходного алгоритма QEM является то, что он не учитывает свойства поверхности, такие как нормали, цвета и текстурные координаты. Гарланд и Хекберт [11] обобщили оригинальный алгоритм QEM, чтобы также обрабатывать свойства поверхности. Для обработки нормалей или цветов исходная матрица ошибок 4 x 4 может быть расширена до матрицы 6 x 6, которая содержит ошибку нормали вершины (abc) или ее цвета (rgb). Аналогично, матрица ошибок 5 × 5 может использоваться для фиксации ошибки текстурных координат вершины (st). Хоппе [12] также представил обобщенную QEM, которая занимает меньше места для хранения, чем у Гарланда и Хекберта [11]. С момента внедрения QEM многие исследователи изучили, как использовать его для различных приложений. Однако большинство вариаций QEM подходят только для конкретных типов сеток (например, для многообразных сеток с границами, многообразных сеток без границ, немногообразных сеток и т. д.), а при использовании для других типов сеток дают менее желательные результаты.

Бахират и др. [12] предложили алгоритм, специально разработанный для виртуальной реальности (VR). Данный алгоритм позволяет качественно обрабатывать сетки с большим количеством граничных областей, а также поддерживает возможность упрощения немногообразных поверхностей, что является ключевым требованием при работе с VR. Кроме того, при разработке алгоритма были устранены недостатки оценки квадратичной ошибки (QEM). Также, разработанный алгоритм QEM4VR позволяет сохранить ключевые свойства поверхности, такие как нормали, текстурные координаты, цвета и материалы.

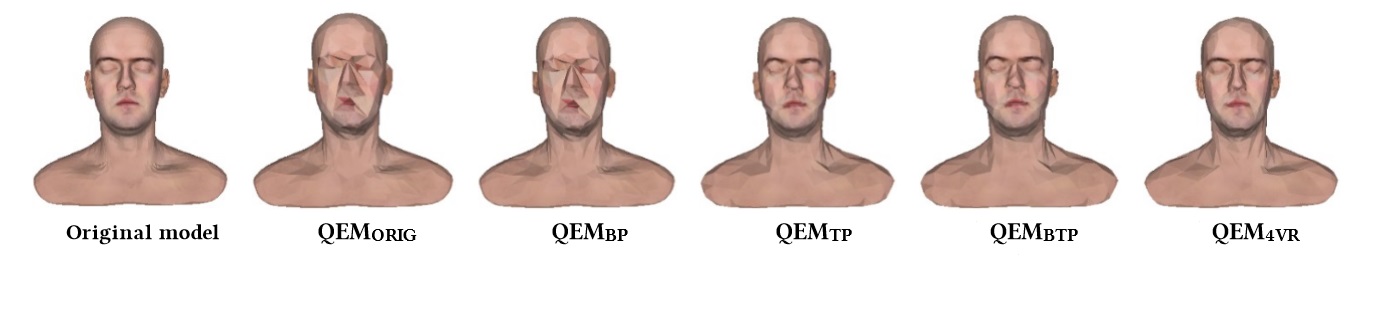
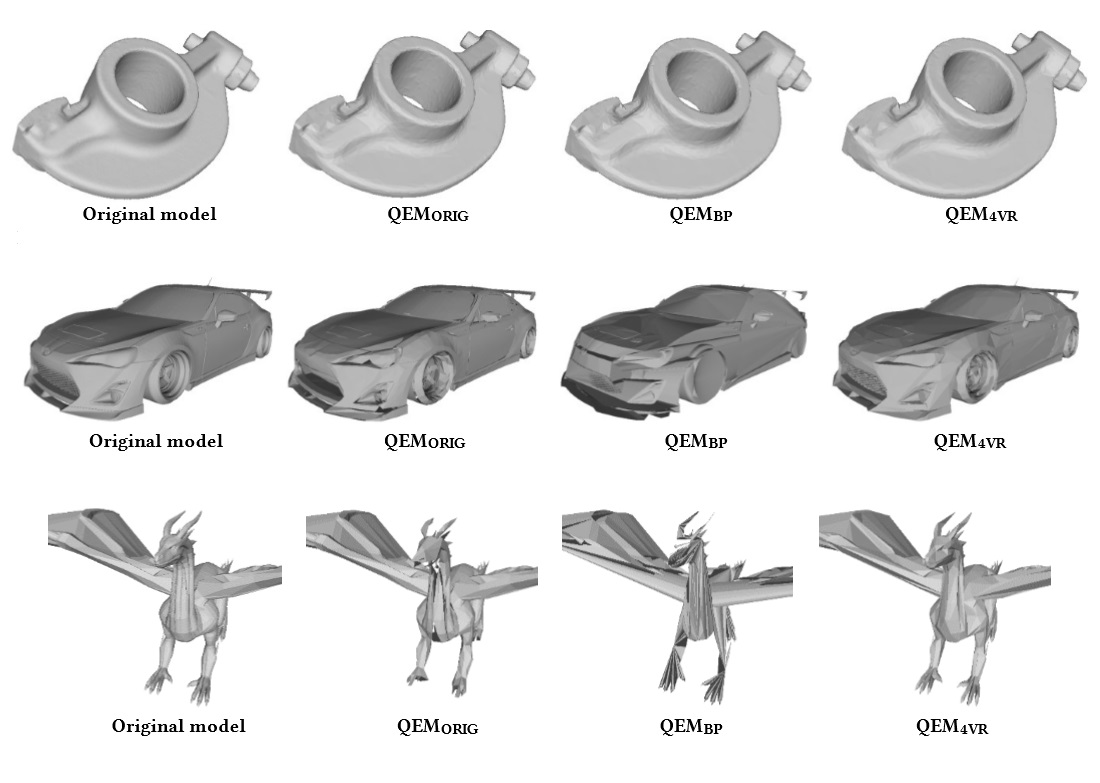
Рассмотрим подробнее подход сохранения границ на основе кривизны. Сначала определяют граничные ребра при инициализации. Для многообразных и немногообразных поверхностей, если ребро принадлежит двум или большему количеству треугольников, оно считается внутренним. Если ребро принадлежит только одному треугольнику, оно рассматривается как граничное, а соответствующие конечные точки рассматриваются как граничные вершины. Для начала рассмотрим случай многообразных поверхностей. Каждая граничная вершина будет иметь ровно две соседние граничные вершины. Рассмотрим граничную вершину 𝑣1 и ее соседние граничные вершины 𝑣2 и 𝑣3. Вычислим кривизну границы в вершине 𝑣1 как:

где 𝑘 - кривизна граничной кривой в .

Для немногообразных поверхностей предположение о точности наличия двух соседних граничных вершин может оказаться неверным. Следовательно, кривизна, рассчитанная с учетом любой случайной пары соседних граничных вершин, приведет к неправильному суждению о вершине. Чтобы обрабатывать такие случаи, предлагается отмечать немногообразные граничные вершины как сложные вершины и не упрощать их, присваивая им большие веса. Аналогично оригинальному алгоритму QEM, предлагается вычислить плоскость граничного ограничения и соответствующую квадрику аналогичным образом. Вычисление квадрики для плоскостей граничных ограничений позволяет сохранять границы без изменения функциональности исходного алгоритма. Затем следует добавить квадрику плоскости граничных ограничений к обоим концам граничного края. Вместо того, чтобы умножать её на постоянный весовой коэффициент, следует умножать квадрику на значение кривизны в каждой конечной точке независимо, а затем добавлять ее к начальным квадрикам соответствующих вершин. Эта схема взвешивания на основе кривизны для сохранения границ присваивает больший вес граничным вершинам с высокой кривизной. Граничным вершинам, имеющим меньшую кривизну (например, вершине на линейной границе), присваивается меньший вес. Это позволяет избежать смещения алгоритма в сторону граничных вершин.

Другим ключевым вкладом алгоритма QEM4VR является его способность обрабатывать несколько свойств поверхности, такие как нормали, текстурные координаты, цвета и материалы. Некоторые предшествующие варианты QEM [11, 12] также позволяли сохранять свойства поверхности, однако эти подходы рассматривали только тот случай, когда вся модель сопоставляется с одним текстурным материалом, так что между вершинами и координатами текстуры существует взаимно однозначное соответствие. Но на практике несколько текстурных материалов могут быть связаны с одной 3D-моделью. Кроме того, нескольким значениям из одной текстурной карты также может быть присвоена заданная вершина. Следовательно, модель может состоять из вершин с несколькими наборами текстурных координат. Для обработки этих случаев было предложено создать новую структуру данных для каждой вершины, которая содержит в себе информацию о соседних гранях и соответствующих текстурных координатах, связанных с данной вершиной. Эти структуры обновляются во время удаления ребер. Также был расширен подход сохранения границ, чтобы сохранять ребра, которые образуют границы между несколькими текстурами. Ребра, связанные с несколькими текстурами, представляют собой материальные границы. Если модель имеет подмножество граней, которые не имеют информации о свойствах поверхности, их значения устанавливаются равными нулю. Это гарантирует, что порядок удаления соответствующих ребер основывается исключительно на геометрической ошибке, а не на свойствах поверхности, которых нет. Чтобы определить границы материала, используется вышеупомянутая структура, которая содержит список граней в окрестности. Если какие-либо две грани в этом списке имеют разный материальный индекс, можно считать, что данной вершине присвоены два разных материала. Следовательно, её можно идентифицировать как материальную граничную вершину. Кроме того, если любая вершина связана с несколькими текстурными значениями, то она считается критической вершиной. Затем применяется определяемый пользователем весовой коэффициент 𝑊𝑡 к квадрическим метрикам, связанным с вершинами на материальных границах. Таким образом, материальные границы остаются неизменными, а материальные показатели должным образом сохраняются.

Далее рассмотрим сравнение работы алгоритмов QEM на примерах.



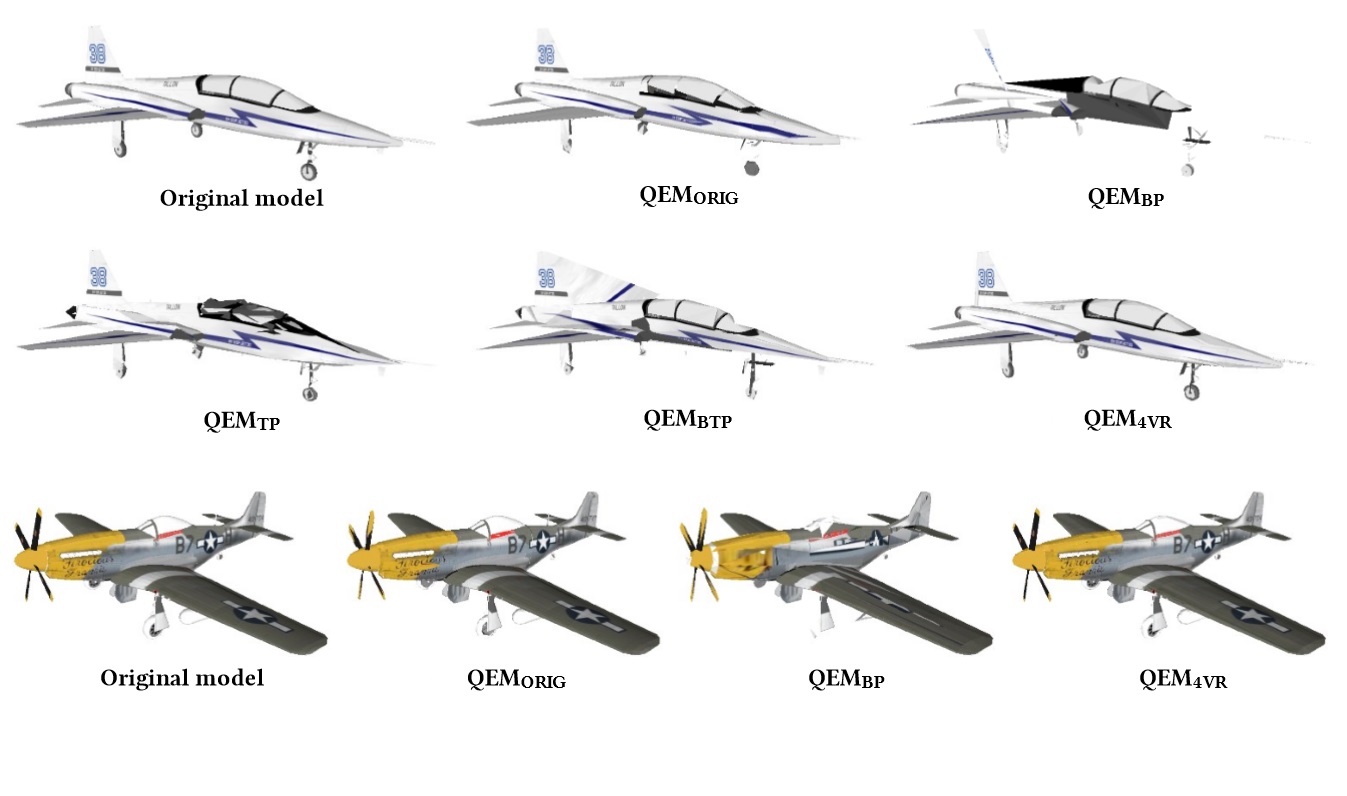


Рисунок 3 – Сравнение результатов упрощения 3D моделей различными QEM алгоритмами

На рис. 3 представлены 3D модели различных типов: деталь и машина – многообразные модели с границами, дракон – немногообразная модель, человек – многообразная модель с наложенной текстурой без использования границ, реактивный самолет – текстурная многообразная модель с границами, самолет – текстурная немногообразная модель с границами. Как мы можем видеть, некоторые предшествующие алгоритмы дают неплохие результаты для упрощения моделей определенных типов, однако в других ситуациях показывают себя хуже. В свою очередь, алгоритм QEM4VR показывает хорошие результаты для всех типов представленных моделей.

# СПИСОК ЛИТЕРАТУРЫ

1. H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, “Mesh optimization,” in ACM Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH), 1993, pp. 19-26.
2. H. Hoppe, “Progressive meshes,” in ACM Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH), pp. 99-108, 1996.
3. G. Turk, “Re-tiling polygonal surfaces,” ACM SIGGRAPH Computer Graphics, vol. 26, no. 2, pp. 55-64, 1992.
4. T. He, L. Hong, A. Kaufman, A. Varshney, and S. Wang, “Voxel based object simplification,” in IEEE Conference on Visualization, 1995, pp. 296-303.
5. W. E. Lorensen, and H. E. Cline, “Marching cubes: A high resolution 3D surface construction algorithm,” ACM SiGGRAPH Computer Graphics, vol. 21, no. 4, pp. 163-169, 1987.
6. J. Cohen, A. Varshney, D. Manocha, G. Turk, H. Weber, P. Agarwal, F. Brooks, and W. Wright, “Simplification envelopes,” in ACM Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH), 1996, pp. 119-128.
7. W. J. Schroeder, J. A. Zarge, and W. E. Lorensen, “Decimation of triangle meshes,” in ACM SIGGRAPH Computer Graphics, vol. 26, no. 2, pp. 65-70, 1992.
8. J. Rossignac, and P. Borrel, “Multi-resolution 3D approximations for rendering complex scenes,” in Modeling in Computer Graphics, 1993, pp. 455-465.
9. D. Luebke, and C. Erikson, “View-dependent simplification of arbitrary polygonal environments,” in ACM Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH), 1997, pp. 199-208.
10. M. Garland, and P. S. Heckbert, “Surface simplification using quadric error metrics,” in ACM Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH), 1997, pp. 209-216.
11. M. Garland, and P. S. Heckbert, “Simplifying surfaces with color and texture using quadric error metrics,” in IEEE Conference on Visualization, 1998, pp. 263-269.
12. H. Hoppe, “New quadric metric for simplifiying meshes with appearance attributes,” in IEEE Visualization, 1999, pp. 59-66.
13. K. Bahirat, C. Lai, R. P. McMahan, and B. Prabhakaran, “A Boundary and Texture Preserving Mesh Simplification Algorithm for Virtual Reality”, in ACM Multimedia Systems Conference on Multimedia Systems, 2017, pp. 50-61.