УДК 519.6

ИСПРАВЛЕНИЕ ПЕРЕСЕЧЕНИЙ В ПОВЕРХНОСТНЫХ ТРЕУГОЛЬНЫХ СЕТКАХ В ПРЕПРОЦЕССОРЕ ПАКЕТА ПРОГРАММ "ЛОГОС" ПРИ ПОДГОТОВКЕ ЗАДАЧ АЭРО- И ГИДРОДИНАМИКИ

Е. О. Евстифеева
(ФГУП "РФЯЦ-ВНИИЭФ", г. Саров Нижегородской области)

Приводится описание алгоритма исправления пересечений в поверхностных треугольных сетках различного типа. Алгоритм применяется в цепочке подготовки расчетной модели в препроцессоре пакета программ "Логос" при решении задач аэро- и гидродинамики. В отличие от большинства существующих подходов, которые ориентированы на определенный тип исходной сетки и применяют глобальное перестроение сетки для исправления, предложенный алгоритм не имеет ограничений на входные данные и изменяет сетку только в локальных областях, где были диагностированы пересечения.

Алгоритм состоит из двух последовательных частей: сначала производится исправление пересечений с помощью локальных операций над элементами сетки (треугольными ячейками и их вершинами), затем в случае наличия неисправленных пересечений применяются удаление области треугольных ячеек, заполнение образованного отверстия новыми треугольниками и оптимизация новой области для повышения качества сетки.

Ключевые слова: пакет программ "Логос", препроцессор, генератор поверхностной треугольной сетки (поверхностный генератор), пересечения ячеек, качество сетки.

Введение

В настоящее время во ФГУП "РФЯЦ-ВНИИЭФ" ведется разработка отечественного пакета программ (ПП) "Логос", предназначенного для компьютерного моделирования, а также анализа результатов при решении разных типов задач, в частности аэро- и гидродинамики [1, 2]. Для проведения моделирования используется расчетная дискретная модель, которая предварительно подготавливается в препроцессоре ПП "Логос".

Исходная модель может быть загружена в препроцессор в фасеточном [3] или параметрическом представлении [4]. По ней с помощью генератора поверхностных сеток [5] выполняется построение треугольной сетки (состоящей из треугольных ячеек), которая, в свою очередь, является исходной для генераторов объемных сеток [6—8] и должна быть замкнутой, иначе процесс подготовки модели будет остановлен. Для выполнения этого требования необходимо, в частности, чтобы в поверхностной сетке отсутствовали пересечения треугольных ячеек (далее для простоты *треугольников*) (рис. 1). Треугольники образуют пересечение, если они имеют общие точки, не принадлежащие в точности их ребрам и вершинам.

Если в загруженной в препроцессор модели в фасеточном представлении имеется большое количество ошибок — вырожденных треугольников (имеющих площадь, близкую к нулю), пересечений, наложений (пересечений треугольников, лежащих в одной плоскости), свободных и многосвязных ребер и др. [9], — лучшим решением при подготовке стартовой для поверхностного генератора модели является использование *генератора замкнутой оболочки* (*враппера*) [10, 11], который перестраивает сетку глобально.

Данная статья посвящена разработанному алгоритму автоматического устранения пересечений в поверхностной треугольной сетке, моди-

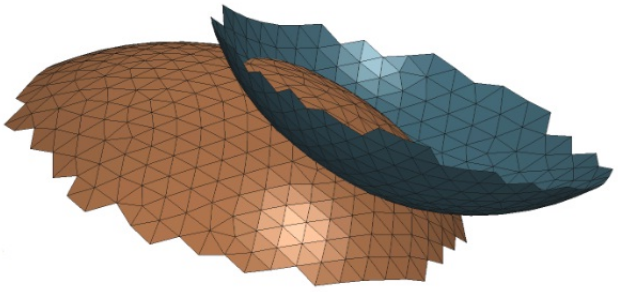


Рис. 1. Фрагмент поверхностной сетки с пересечением

фицирующему сетку в локальных областях, что позволяет использовать его как перед построением поверхностной сетки, так и после.

Пересечения могут появиться в сетке после перестроения вследствие задания крупного целевого размера, не соответствующего размерам определенных областей модели, что влечет за собой большое отклонение от исходной поверхности. Особенно часто такие коллизии встречаются при построении сетки для моделей, содержащих мелкие детали, выступы и скругленные поверхности (рис. 2).

Если построение сетки выполняется по модели в фасеточном представлении, в которой имеются пересечения, то эффективность построения существенно снижается: в большинстве случаев пересечения остаются в результирующей сетке либо сетку совсем не удается перестроить ввиду особенностей внутренних алгоритмов генерации. Таким образом, исправление пересечений на начальном этапе генерации поверхностной сетки также является актуальной задачей.

Существующие по данной проблематике работы [12—15] описывают подходы к исправлению пересечений, ориентированные на определенный тип исходной сетки, а также могут накладывать ограничения на исходную сетку, например, в некоторых случаях предполагается, что сетка не содержит свободных и многосвязных ребер или вырожденных треугольников.

В большинстве рассматриваемых статей при определении типа сетки делается акцент на распределении треугольников, т. е. соотношении площадей смежных треугольников. С точки зрения области рассматриваемых задач более всего интересуют изотропные и анизотропные поверхностные треугольные сетки [16] как два противоположных случая типов сеток по распределению и качеству [17] треугольников. Изотропная сетка характеризуется равномерным распределением треугольников и высоким качеством (рис. 3). Анизотропная сетка имеет неравномерное распределение треугольников и зачастую представляет собой сетку невысокого качества за счет вытянутых ячеек, а также может содержать разные виды ошибок (рис. 4). Процесс получения анизотропной сетки по модели в параметрическом представлении называется тесселяцией.

Сетки, полученные с помощью генератора замкнутой оболочки, не имеют четкого вида распределения треугольников и могут состоять из треугольников разного качества (рис. 5).

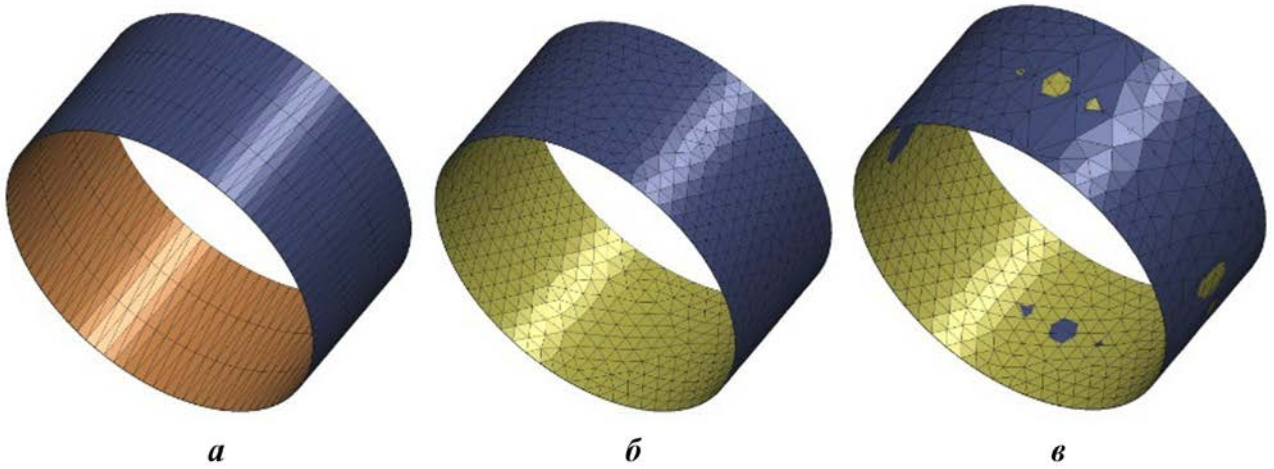


Рис. 2. Модель тонкостенного цилиндра: *а* — исходная; *б* — после перестроения без образования пересечений; *в* — после перестроения с образованием пересечений

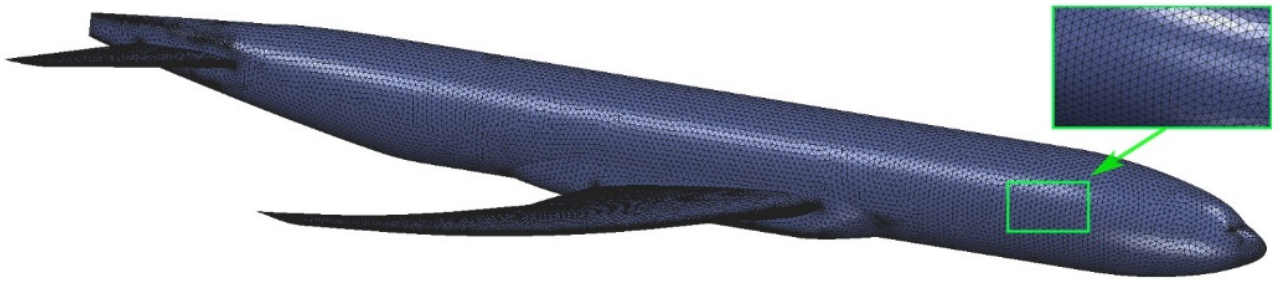


Рис. 3. Пример неструктурированной изотропной сетки, построенной для модели [14]

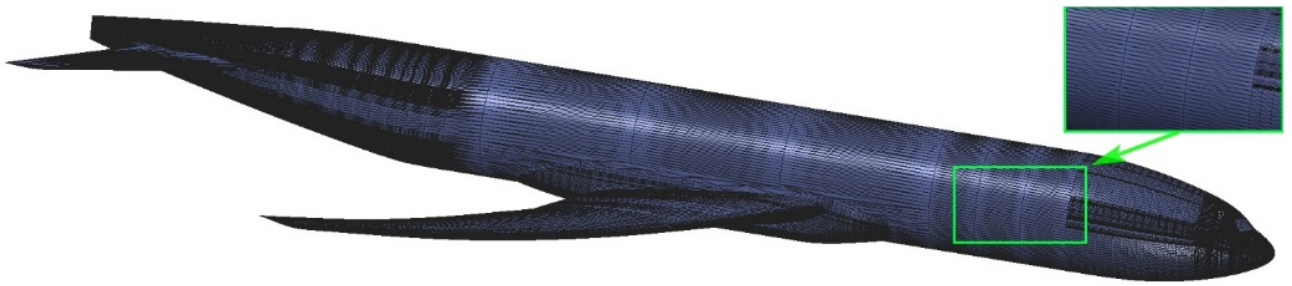


Рис. 4. Пример анизотропной сетки, полученной с помощью тесселяции для модели [18]

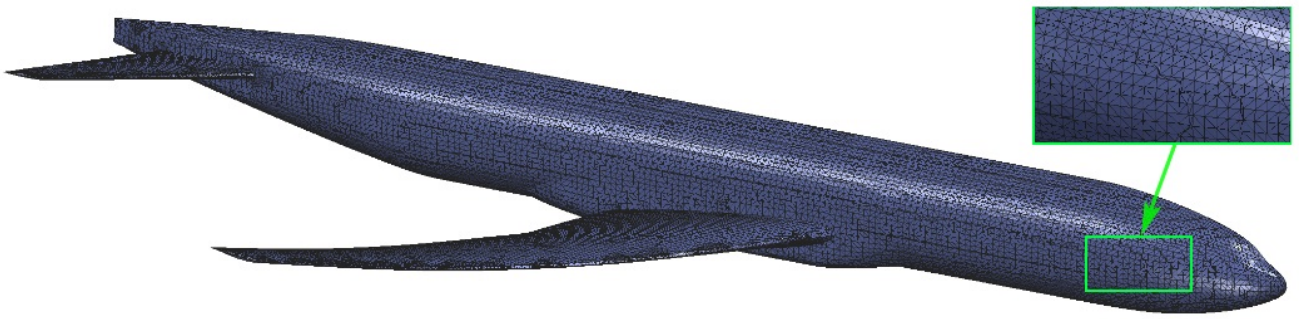


Рис. 5. Пример сетки, построенной с помощью генератора замкнутой оболочки по модели [14]

В работе [12] приведен алгоритм исправления пересечений для изотропных сеток, построенных по модели в параметрическом представлении, с помощью нескольких локальных операций. Данный алгоритм не накладывает дополнительных ограничений на исходную сетку.

Алгоритм исправления пересечений, предложенный в [13], предполагает, что исходная сетка представляет собой оцифрованную сетку низкого качества. Для исправления пересечений, однако, добавляется условие, что треугольники сетки в основном не должны иметь большого различия в размерах, т. е. сетка должна иметь равномерное распределение треугольников. В отличие от многих других подходов здесь используются локальные модификации выделенных областей, которые необходимо перестроить для получения сетки без пересечений.

В статьях [14, 15] рассматривается исправление пересечений на анизотропных сетках, полученных с помощью тесселяции. В статье [14] предполагается, что в загруженной сетке есть информация о ее разбиении на границы, соответствующие граням исходной CAD[1]-модели, и пересечения образованы на стыках этих границ, причем по отдельности сетки на границах пересечений не имеют. Поскольку информация об исходных границах параметрической модели на момент загрузки сетки в фасеточном представ-

[1]CAD (Computer Aided Design) — система автоматизированного проектирования.

лении отсутствует, алгоритм из статьи неприменим.

В статье [15] описывается эффективный алгоритм исправления пересечений, который конвертирует исходную сетку в BSP-дерево[2]. Для корректного вычисления BSP-дерева необходимо, чтобы сетка не имела свободных ребер и вырожденных треугольников, что также не может быть гарантировано для произвольной загруженной сетки.

В данной статье предложен алгоритм исправления пересечений, состоящий из двух последовательных этапов. Предложенный алгоритм не накладывает на исходную сетку дополнительных ограничений, таких как отсутствие ошибок сетки, а также не предъявляет требований к ее типу (изотропная, анизотропная и др.). В заключение представлены примеры применения реализованного алгоритма и полученные результаты.

Основы алгоритма исправления пересечений

Поскольку препроцессор принимает на вход сетку произвольного типа, алгоритм исправления пересечений, соответственно, должен выполняться корректно для любых исходных данных.

Для того чтобы гарантировать, что в результате исправления модель не будет искажена с точки зрения отклонения от исходной поверхности и появления других ошибок сетки, необходимо ввести ряд дополнительных ограничений и рассмотреть допустимость применения операций, используемых для исправления пересечений.

Поиск пересечений осуществляется с помощью построения kd-дерева[3] и проверки пересекающихся треугольников в каждом из его блоков. Исходными данными для алгоритма исправления пересечений является список пар пересекающихся треугольников.

На первом этапе алгоритма к элементам сетки (треугольникам и их вершинам) применяются локальные операции. Второй этап алгоритма выполняется только в том случае, если в сетке остались пересечения после первого этапа. Он заключается в удалении области треугольников с пересечениями и последующем заполнении образованного отверстия новыми треугольниками. На обоих этапах сетка модифицируется локально, без глобального перестроения.

[2]BSP (Binary Space Partitioning)-дерево — представление объекта с помощью двоичного разбиения пространства.

[3]kd-дерево — особый вид BSP-дерева.

Исправление пересечений с помощью локальных операций

На первом этапе исправления пересечений в сетке используются две локальные операции над элементами сетки, описанные в статье [12], а именно смещение вершины и треугольника. Для возможности применения данных операций для всех типов сеток были добавлены дополнительные ограничения, модифицированы формулы поиска новой позиции при смещении вершины и треугольника. Для ускорения исправления операции применяются не последовательно к каждой паре пересекающихся треугольников, а к сформированным несвязным областям из треугольников, т. е. к областям, которые не имеют общих вершин.

Обязательным критерием успешности операции является сокращение количества пересечений и отсутствие новых пересекающихся треугольников.

Алгоритм является итерационным, максимальное количество итераций равно 15. Данное значение выбрано исходя из баланса между временными затратами и эффективностью алгоритма при исправлении пересечений.

Каждая итерация алгоритма состоит из следующих шагов:

1. Формирование из всех треугольников с пересечениями несвязных областей $region_i$, $i = \overline{1, m}$.
2. Цикл по областям $region_i$, $i = \overline{1, m}$:
   - оценка изменения объема исходной модели в результате применения каждой из локальных операций;
   - оценка изменения площади модифицированных треугольников в результате применения каждой из операций;
   - формирование очереди из операций, удовлетворяющих критерию изменения площади, сортировка очереди по возрастанию изменения объема.
3. Цикл по количеству операций:
   - применение первой в очереди операции для всех неисправленных областей $region_i$, $i = \overline{1, m}$;
   - поиск пересечений на всей сетке;

– для каждой области проверка успешности:
  - если в области не осталось пересечений, то удаление области из списка;
  - если в области сократилось количество пересечений и не было образовано новых, то пометка области как исправленной на данном шаге;
  - если количество пересечений не сократилось или были образованы новые, то отмена операции и удаление ее из очереди.

Для оценки изменения площади для каждой выделенной области вычисляется суммарная площадь треугольников, участвующих в модификации, до и после применения локальной операции и вычисляется отношение r_S минимального из этих двух значений к максимальному. При $r_S \geq \varepsilon_S$, где $0 < \varepsilon_S \leq 1$, операция считается успешной. Чем больше задано значение параметра ε_S, тем меньшее изменение площади допускается. При реализации было установлено значение $\varepsilon_S = 0{,}95$, при котором получаемые результаты характеризуются небольшим изменением поверхности и одновременно с этим высокой эффективностью исправлений.

На рис. 6, *а* приведен фрагмент анизотропной сетки низкого качества. Красным цветом выделены близкие к вырожденным треугольники, образующие область с пересечением. В данном случае применение локальных операций в области с вырожденными треугольниками сопровождается небольшим изменением объема, но при этом недопустимым изменением площади поверхности модели, как показано на рис. 6, *б* (значение r_S в данном случае равно 0,4). Предложенное ограничение на изменение площади поверхности позволяет сохранить более точное описание исходной модели.

Операция смещения вершины. Одной из модифицированных относительно [12] операций является смещение вершины. Пример фрагмента сетки с пересечением представлен на рис. 7, *а*, где выделена вершина, выбранная для смещения, а также стрелкой указано направление, по которому смещение производится. На рис. 7, *б* показан фрагмент сетки после выполнения локальной операции, где пересечение устранено.

Формула смещения вершины была модифицирована таким образом, чтобы, во-первых, минимизировать изменение сетки и, во-вторых, сделать операцию применимой на сетках, полученных с помощью тесселяции.

На каждой итерации для смещения выбирается та вершина области, которая принадлежит наибольшему количеству ребер, пересекающих

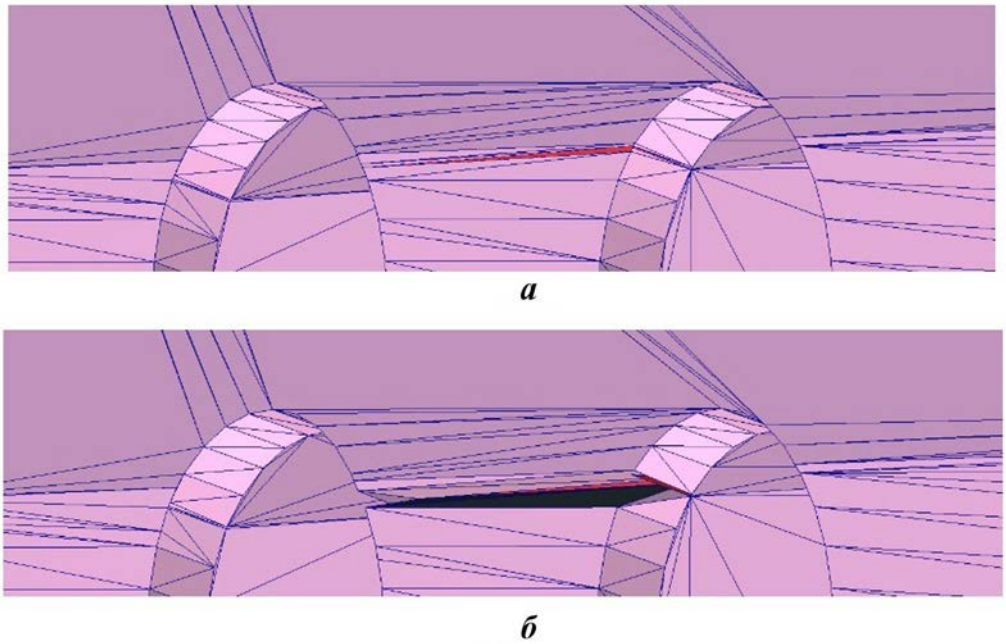


Рис. 6. Фрагмент анизотропной сетки: *а* — с выделенной областью пересекающихся треугольников; *б* — после применения операции с недопустимым изменением площади модифицированной области

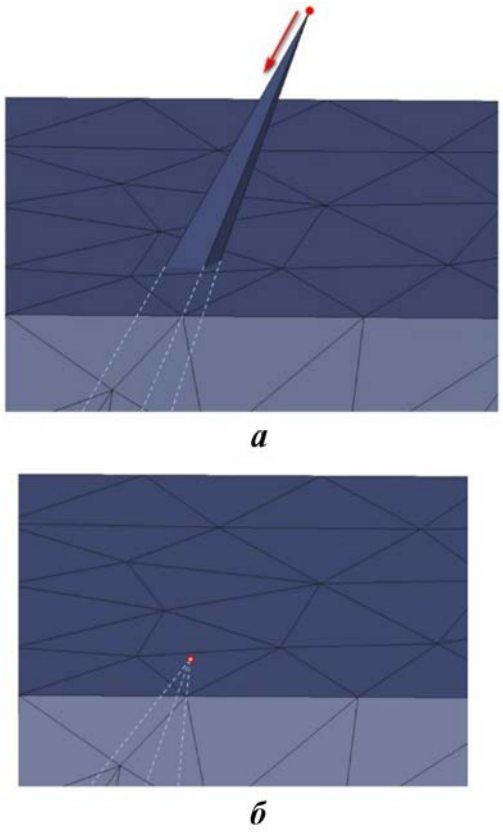


Рис. 7. Фрагмент сетки: *а* — до операции смещения вершины; *б* — после операции смещения вершины

треугольники. Новое положение смещенной вершины вычисляется как суперпозиция нескольких новых точек.

Для выделенной области $region_i$, $i = \overline{1,m}$, треугольников с пересечениями рассчитаем длину минимального и максимального ребер в этой области: l_i и L_i соответственно.

Пусть a — вершина, выбранная для смещения, а ребро ab_n ($n = \overline{1,N}$) является инцидентным вершине a ребром, пересекающим треугольники.

Введем параметр $0 \le t \le 1$, определяющий положение точки на отрезке (значениями $t = 0$ и 1 определяются концы отрезка), α и β — коэффициенты смещения. Положение точки смещения вычисляется следующим образом:

а) если ребро ab_n пересекается с одним треугольником в точке x, то точка для сдвига a'_n вычисляется по формуле[4]

$$\vec{a}\,'_n = t_n\left(\vec{b}_n - \vec{x}\right) + \vec{b}_n,$$

где t_n вычисляется в зависимости от выполнения условий:

– если $0{,}5\left|\overrightarrow{xb_n}\right| > l_i$, то

$$t_n = \begin{cases} \alpha, & l_i < \alpha\left|\overrightarrow{xb_n}\right| < \beta L_i; \\ \dfrac{l_i}{\left|\overrightarrow{xb_n}\right|} & \text{в других случаях;} \end{cases}$$

– если $0{,}5\left|\overrightarrow{xb_n}\right| \le l_i$, то $t_n = 0{,}5$.

б) если ребро ab_n пересекается с несколькими треугольниками в точках $x_1, x_2, \ldots, x_k$ и x_1, x_2 являются двумя точками пересечения, ближайшими к вершине a, то точка для сдвига a'_n вычисляется по формуле

$$\vec{a}\,'_n = t_n\left(\vec{x}_1 - \vec{x}_2\right) + \vec{x}_1,$$

где t_n вычисляется в зависимости от выполнения условий:

– если $0{,}5\left|\overrightarrow{x_1x_2}\right| > l_i$, то

$$t_n = \begin{cases} \alpha, & l_i < \alpha\left|\overrightarrow{x_1x_2}\right| < \beta L_i; \\ \dfrac{l_i}{\left|\overrightarrow{x_1x_2}\right|} & \text{в других случаях;} \end{cases}$$

– если $0{,}5\left|\overrightarrow{x_1x_2}\right| \le l_i$, то $t_n = 0{,}5$.

Предложенный метод смещает вершину a в вершину a', определяемую как суперпозиция найденных вершин:

$$\vec{a}\,' = \sum_n \frac{\vec{a}\,'_n}{n}, \quad n = \overline{1,N}.$$

Схематичный пример операции смещения вершины приведен на рис. 8.

Применение операции по приведенным формулам гарантирует, что расстояние между точкой смещения и точкой пересечения для каждого пересеченного ребра не слишком мало относительно длины минимального ребра и не слишком велико относительно длины максимального ребра области. При этом образованного просвета между новой точкой и точкой пересечения до исправления достаточно, чтобы обеспечить отсутствие пересечения после данной модификации. Коэффициенты α и β можно варьировать. При реализации применялись следующие значения: $\alpha = 0{,}25$; $\beta = 0{,}1$. Результаты исправления пересечений с данными коэффициентами показали свою эффективность, и при этом отклонение

[4]Здесь и далее в формулах запись $\vec{p}$ означает радиус-вектор точки p, а $|\overrightarrow{pq}|$ — расстояние между точками p и q.

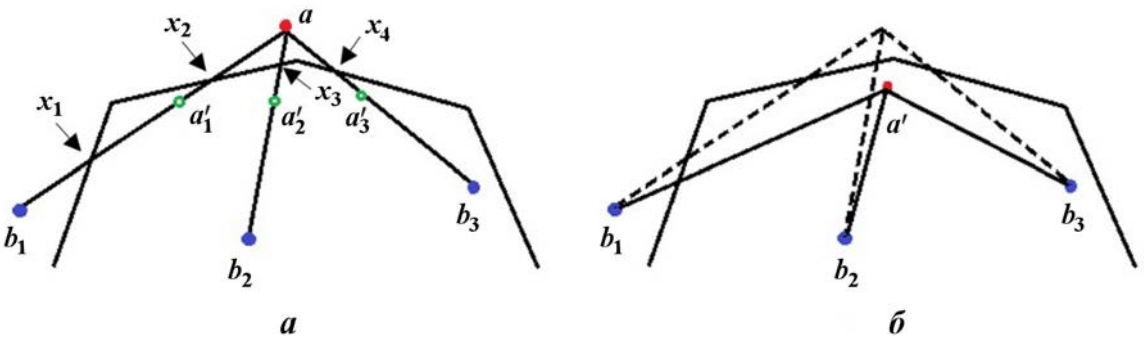


Рис. 8. Схематичное изображение операции смещения вершины: *а* — до смещения вершины a; *б* — после смещения вершины a в вершину a'

от начальной точки было сокращено, что существенно для сеток, полученных с помощью тесселяции.

Операция смещения треугольника. Данная операция смещает треугольник вдоль нормали к его плоскости в двух направлениях. На каждой итерации для каждой области $region_i$, $i = \overline{1,m}$, выбирается треугольник с наибольшим количеством пересечений.

Предположим, что треугольник T с вершинами v_1, v_2, v_3 пересекается с одним или более ребрами и вершины p_1, p_2, ..., p_k инцидентны этим ребрам. Вектор $\vec{n}$ — вектор единичной нормали для T.

Минимум и максимум расстояний (взятых со знаками) от вершин p_i, $i = \overline{1,k}$, до плоскости треугольника T обозначим как $d_{\min}$ и $d_{\max}$ соответственно и будем вычислять по формулам

$$d_{\min} = \min_i \left\{ (\vec{p}_i - \vec{v}_1) \cdot \vec{n} \right\};$$
$$d_{\max} = \max_i \left\{ (\vec{p}_i - \vec{v}_1) \cdot \vec{n} \right\}.$$

Предложенный метод перемещает треугольник T на две возможные позиции с помощью смещения каждой его вершины от начального положения по формулам

$$\begin{aligned} \vec{v}\,'_i &= \vec{v}_i + \vec{n}\,(d_{\max} + \varepsilon_{move}), \\ \vec{v}\,'_i &= \vec{v}_i + \vec{n}\,(d_{\min} - \varepsilon_{move}), \quad i = \overline{1,3}, \end{aligned} \tag{1}$$

где ε_{move} — выбранное смещение.

Выбор смещения зависит от точности исходной модели ε, если она задана. При отсутствии заданной точности модели предлагается ее вычисление по формуле

$$\varepsilon = \begin{cases} 0{,}05 l_{\min}, & 0{,}05 l_{\min} > 10^{-16}; \\ 10^{-16}, & 0{,}05 l_{\min} \le 10^{-16}, \end{cases}$$

где $l_{\min}$ — минимальная длина среди всех ребер исходной сетки. Тогда смещение ε_{move} вычисляется по формуле

$$\varepsilon_{move} = \begin{cases} \max\left\{10^{-16}, \delta\varepsilon\right\}, & \delta\varepsilon \le \gamma L_i; \\ \gamma L_i, & \delta\varepsilon > \gamma L_i, \end{cases}$$

где L_i — максимальная длина ребра среди треугольников области $region_i$, $i = \overline{1,m}$; δ, γ — коэффициенты смещения.

При реализации использовались следующие значения коэффициентов смещения: $\delta = 0{,}5$; $\gamma = 0{,}001$.

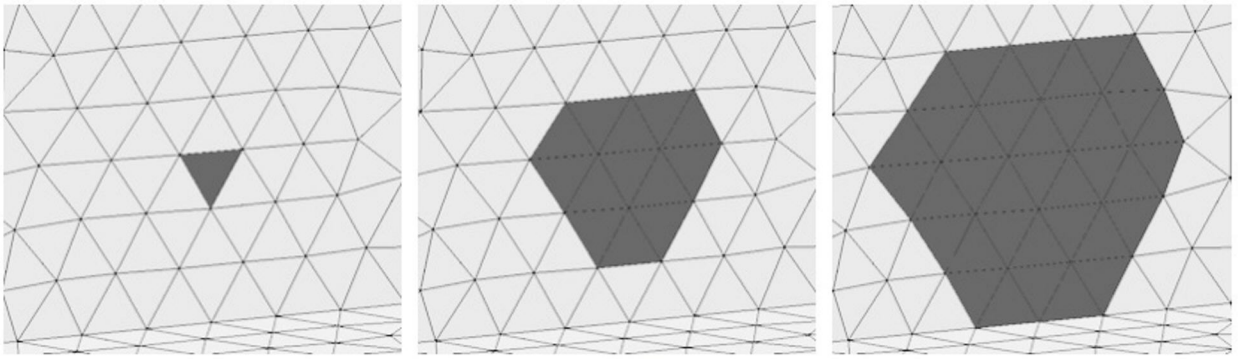
Каждое из смещений, заданных формулами (1), определяет отдельную операцию, которая добавляется в очередь локальных операций.

Второй этап исправления пересечений

После применения локальных операций в сетке все еще могут присутствовать пересечения. В качестве второго этапа исправления пересечений был разработан алгоритм, основанный на работе [13].

Алгоритм является итерационным, максимальное количество итераций равно трем. Как и на первом этапе, предварительно выделяются все пары пересекающихся треугольников t_i, $i = \overline{1,M}$ из них формируется начальное множество треугольников $T = \{t_1, \ldots, t_i\}$, $i = \overline{1,M}$ для обработки алгоритмом. В цикле по $j = \overline{0,2}$ выполняются следующие шаги алгоритма:

1. Наращивание для множества треугольников T слоя окружения (рис. 9) и добавление его треугольников к текущему множеству T;



а *б* *в*

Рис. 9. Наращивание слоя окружения: *а* — слой окружения нулевого порядка (выделенный треугольник); *б* — слой окружения первого порядка; *в* — слой окружения второго порядка

2. Формирование из T несвязных областей $region_i$, $i = \overline{1,m}$;
3. Для каждой области $region_i$, $i = \overline{1,m}$:
   1) удаление всех треугольников области, формирование контура из свободных ребер;
   2) заполнение образованного отверстия новыми треугольниками с использованием только вершин свободного контура [5];
   3) адаптивное измельчение треугольников [19], сформированных на предыдущем шаге;
   4) сглаживание сетки [20], построенной на предыдущем шаге;
   5) проверка допустимости изменения площади поверхности, отсутствия многосвязных и свободных ребер:
      – если хотя бы одна из проверок не успешна, то возвращение исходных треугольников области.
4. Поиск пересечений после модификации сетки.
5. Для каждой модифицированной области $region_i$, $i = \overline{1,m}$, проверка сокращения количества пересечений, а также отсутствия новых пересечений:
   – если хотя бы одна из проверок не успешна, то замена сетки области на исходную,
   – если все проверки успешны, то удаление из множества T всех его треугольников и удаление области из списка $region_i$, $i = \overline{1,m}$.
6. Проверка критерия останова алгоритма:
   – если в сетке не осталось пересечений или достигнуто максимальное количество итераций, то завершение алгоритма;
   – иначе переход на шаг 1.

После выполнения второго этапа исправления пересечений добавлена проверка на появление несвязных областей сетки. Такие ситуации возможны при сложной конфигурации сетки в области с пересечением, например, при наличии в области других ошибок. На данном этапе возможно удаление таких областей, если они малы по количеству треугольников и площади поверхности относительно всей модели.

Внедренные этапы адаптивного разбиения треугольников и сглаживания сетки выполняются после формирования новой области, чтобы она как можно точнее аппроксимировала исходную поверхность. Предложенные проверки допустимости изменения площади поверхности, а также отсутствия многосвязных и свободных ребер предотвращают искажение поверхности и появление новых ошибок сетки.

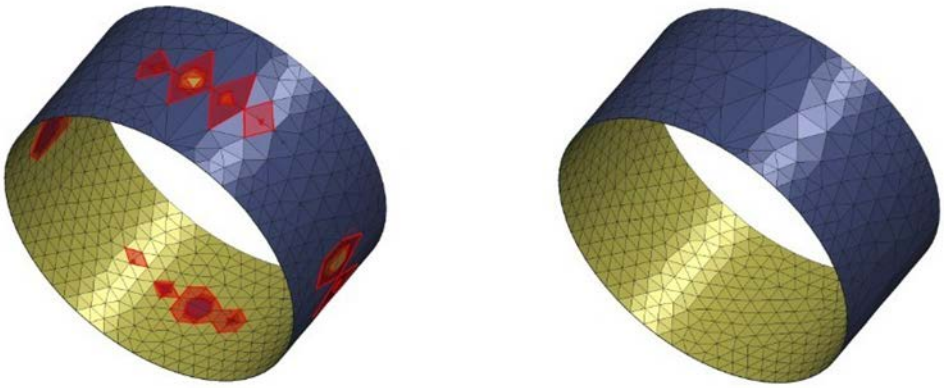
Результаты

Реализованный алгоритм был протестирован на большой базе моделей с пересечениями. База моделей состоит из сеток различного типа, используемых при подготовке моделирования задач аэро- и гидродинамики. Часть тестовых случаев представляет собой реальные модели задач аэро- и гидродинамики, на которых были обнаружены пересечения при подготовке к моделированию, другая часть — искусственно созданные сетки.

Алгоритм показал свою эффективность при исправлении сеток различного типа. В случае изотропных сеток хорошего качества исправление пересечений в большинстве случаев проходит успешно. На рис. 10 приведен фрагмент тестовой изотропной сетки до и после исправления пересечений. Не всегда полностью удается исправить пересечения в областях, содержащих другие сеточные ошибки, а также в сетках, исходная модель которых изначально неправильно спроектирована (например, состоит из несвязных деталей, которые пересекаются).

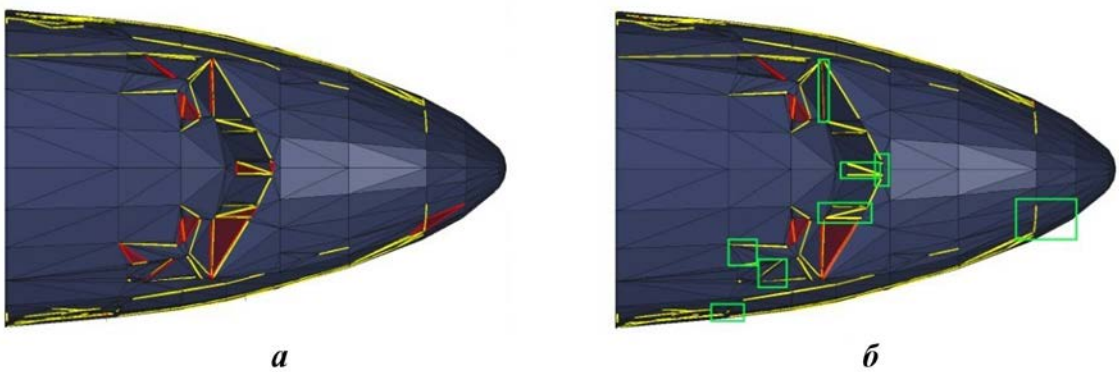
При тестировании алгоритма на анизотропных сетках процент успешных исправлений составил 79 % в случае отсутствия большого количества ошибок других типов.

В случае анизотропной сетки низкого качества с большим количеством ошибок предполагается предварительная обработка сетки с помощью генератора замкнутой оболочки: в данном случае требуется ее глобальное исправление. Однако, поскольку препроцессор работает с любым типом входных данных, тестирование алгоритма также было проведено на подобных сетках. Исправление пересечений выполнялось достаточно эффективно, даже несмотря на сложность конфигурации сетки из-за наличия других ошибок. На рис. 11, *а* приведен фрагмент анизотропной сетки [21] низкого качества с большим количеством ошибок: красным цветом выделены пересекающиеся треугольники, желтым цветом — вырожденные треугольники, оранжевым — наложения треугольников. На рис. 11, *б* — фрагмент той же сетки после исправления пересечений: исправлено 8 областей из 12, неисправ-



а *б*

Рис. 10. Фрагмент изотропной сетки: *а* — с пересечениями (выделены красным цветом); *б* — после исправления пересечений с помощью алгоритма



а *б*

Рис. 11. Фрагмент анизотропной сетки с пересечениями, наложениями и вырожденными треугольниками: *а* — до исправления пересечений; *б* — после исправления пересечений (области исправления выделены рамками зеленого цвета)

ленные области содержат наложения и вырожденные треугольники.

Скорость выполнения алгоритма зависит от количества треугольников в сетке, количества пересечений и их конфигурации, определяющей степень "запутывания" сетки, которая повышается из-за наличия других видов ошибок.

Важным вопросом при разработке алгоритма исправления пересечений является допустимость модификации сетки. С одной стороны, пересечение нарушает замкнутость сетки, но с другой, применение исправления означает отступление от исходного описания модели. В данной работе считается приоритетным исправление пересечений, поскольку при их наличии в поверхностной сетке блокируется построение объемной сетки. Более того, при выполнении каждой модификации сетки по описанному алгоритму вычисляется величина, контролирующая относительное отклонение измененной поверхности от исходной, а также проверяются введенные ограничения на изменение площади, при невыполнении которых изменение не производится. Сравнение сеток до и после исправления проводилось с помощью программы metro [22]. Максимальное отклонение не превышает $0{,}0086d$, где d — диагональ габаритной коробки модели.

Таким образом, применение разработанного алгоритма обеспечивает исправление большинства пересечений, где это допустимо в рамках введенных ограничений, а исправление оставшихся ошибок, требующее детальной проработки, возлагается на пользователя.

Заключение

В статье указаны основные источники пересечений треугольников в сетке, типы поверхностных сеток, для которых предполагается использование алгоритма исправления пересечений, а также существующие подходы к исправлению пересечений. Приведено описание алгоритма исправления пересечений, разработанного с целью подготовки поверхностной треугольной сетки, удовлетворяющей требованиям генератора объемных сеток. Данный алгоритм внедрен в препроцессор ПП "Логос" и применяется в цепочке подготовки расчетной сетки для моделирования задач аэро- и гидродинамики.

До реализации алгоритма проведение сквозного построения поверхностной и объемной сеток могло быть прервано из-за образования пересечений в поверхностной сетке. Сетку надо было исправлять с помощью интерактивных средств препроцессора вручную, что является трудоемким и затратным по времени процессом. Реализованный автоматический инструмент позволил проводить исправление пересечений в процессе построения сетки без участия пользователя, что существенно снизило влияние данных ошибок на результат построения и сократило временные затраты на подготовку расчетной сетки.

Предложенный алгоритм показал свою эффективность, при этом изменение сетки контролируется дополнительными ограничениями и введенной величиной, обеспечивающей минимальное относительное отклонение от исходной поверхности.

Список литературы

1. *Погосян М. А., Савельевских Е. П., Стрелец Д. Ю., Корнев А. В., Шагалиев Р. М., Козелков А. С.* Использование отечественных суперкомпьютерных технологий при проектировании новых образцов авиационной техники // Авиационная промышленность. 2013. № 3. С. 3—7.
Pogosyan M. A., Savelyevskikh E. P., Strelets D. Yu., Kornev A. V., Shagaliev R. M., Kozelkov A. S. Ispolzovanie otechestvennykh superkompyuternykh tekhnologiy pri proektirovanii novykh obraztsov aviatsionnoy tekhniki // Aviatsionnaya promyshlennost. 2013. № 3. S. 3—7.
2. *Козелков А. С. Дерюгин Ю. Н., Зеленский Д. К., Полищук С. Н., Лашкин С. В., Жучков Р. Н., Глазунов В. А., Яцевич С. В., Курулин В. В.* Многофункциональный пакет программ "Логос": физико-математические модели расчета задач аэро-, гидродинамики и теплообмена: Препринт № 111. Саров: РФЯЦ-ВНИИЭФ, 2013.
Kozelkov A. S., Deryugin Yu. N., Zelenskiy D. K., Poloshchuk S. N., Lashkin S. V., Zhuchkov R. N., Glazunov V. A., Yatsevich S. V., Kurulin V. V. Mnogofunktsionalnyy paket programm "Logos": fiziko-matematicheskie modeli raschyeta zadach aero-, gidrodinamiki I teploobmena: Preprint № 111. Sarov: RFYaTs-VNIIEF, 2013.
3. *Cohen-Steiner D., Morvan J.-M.* Restricted delaunay triangulations and normal cycle // Proc. Annual Symposium of Computational Geometry. California, 2003. P. 312—321.

4. *Piegl L., Tiller W.* The NURBS Book. Second Edition. Springer, 1997.

5. *Борисенко О. Н., Лукичев А. Н., Евстифеева Е. О., Панкратов Д. М., Цалко Т. В., Гиниятуллина А. Г.* Алгоритмы обработки особенностей геометрических моделей при построении поверхностных треугольных сеток в препроцессоре пакета программ "Логос" // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 2020. Вып. 3. С. 40—52.
Borisenko O. N., Lukichyev A. N., Evstifeeva E. O., Pankratov D. M., Tsalko T. V., Giniyatullina A. G. Algoritmy obrabotki osobennostey geometricheskikh modeley pri postroenii poverkhnostnykh treugolnykh setok v preprotsessore paketa programm "Logos" // Voprosy atomnoy nauki i tekhniki. Ser. Matematicheskoe modelirovanie fizicheskikh protsessov. 2020. Vyp. 3. S. 40—52.

6. *Смолкина Д. Н., Борисенко О. Н., Черенкова М. В., Гиниятуллина А. Г., Кузьменко М. В., Чухманов Н. В, Потехина Е. В., Попова Н. В., Турусов М. Р.* Автоматический генератор неструктурированных многогранных сеток в препроцессоре пакета программ "Логос" // Там же. 2018. Вып. 2. С. 25—39.
Smolkina D. N., Borisenko O. N., Cherenkova M. V., Giniyatullina A. G., Kuzmenko M. V., Chukhmanov N. V., Potekhina E. V., Popova N. V., Turusov M. R. Avtomaticheskiy generator nestrukturirovannykh mnogogrannykh setok v preprotsessore paketa programm "Logos" // Tam zhe. 2018. Vyp. 2. S. 25—39.

7. *Попова Н. В, Борисенко О. Н., Корнеева И. И., Чухманов Н. В., Потехина Е. В., Лазарев В. В., Гиниятуллина А. Г.* Автоматический генератор неструктурированных тетраэдральных сеток с призматическими слоями в препроцессоре пакета программ "Логос" // Там же. 2020. Вып. 1. С. 43—57.
Popova N. V., Borisenko O. N., Korneeva I. I., Chukhmanov N. V., Potekhina E. V., Lazarev V. V., Giniyatullina A. G. Avtomaticheskiy generator nestrukturirovannykh tetraedralnykh setok s prizmaticheskimi sloyami v preprotsessore paketa programm "Logos" // Tam zhe. 2020. Vyp. 1. S. 43—57.

8. *Попова Н. В.* Автоматический генератор неструктурированных многогранных сеток на основе тетраэдральных сеток с призматическими слоями // Там же. 2021. Вып. 3. С. 70—83.
Popova N. V. Avtomaticheskiy generator nestrukturirovannykh mnogogrannykh setok na osnove tetraedralnykh setok s prizmaticheskimi sloyami // Tam zhe. 2021. Vyp. 3. S. 70—83.

9. *Attene M., Campen M., Kobbelt L.* Polygon mesh repairing: an application perspective // ACM Computing Surveys. 2013. Vol. 45, No 2. P. 1—33.

10. *Kobbelt L. P., Vorsatz J., Labsik U., Seidel H.-P.* A shrink wrapping approach to remeshing polygonal surfaces // Computer Graphics Forum, 2000. Vol. 18. P. 119—130.

11. *Никитин В. А., Шурыгин А. В., Новиков И. Г., Егоров А. В., Соколов С. С., Панов А. И.* Программный модуль генерации замкнутой поверхностной триангуляционной сетки в пакете программ "Логос" // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 2021. Вып. 2. С. 70—79.
Nikitin V. A., Shurygin A. V., Novikov I. G., Egorov A. V., Sokolov S. S., Panov A. I. Programmnyy modul generatsii zamknutoy poverkhnostnoy triangulyatsionnoy setki v pakete programm "Logos" // Voprosy atomnoy nauki i tekhniki. Ser. Matematicheskoe modelirovanie fizicheskikh protsessov. 2021. Vyp. 2. S. 70—79.

12. *Yamakawa S. and Shimada K.* Removing self intersections of a triangular mesh by edge swapping, edge hammering, and face lifting // Proc. 18th Int. Meshing Roundtable. Carnegie Mellon University, 2009. P. 13—29.

13. *Attene M.* A lightweight approach to repairing digitized polygon meshes // The Visual Computer. 2010. Vol. 26. P. 1393—1406.

14. *Bischoff S., Kobbelt L.* Structure Preserving CAD Model Repair // Eurographics. 2005. Vol. 24, No 3. P. 527—536.

15. *Campen M., Kobbelt L.* Exact and robust (self-)intersections for polygonal meshes // Ibid. 2010. Vol. 29, No 2. P. 397—406.

16. *Sadrehghighi I.* Mesh generation in CFD. A review — CFD Open Series, Patch 1.86.5. Annapolis, 2019.

17. *Knupp P. M.* Algebraic mesh quality metrics // Siam J. Sci. Comput. 2001. Vol. 23, No 1. P. 193—218.

18. 3rd AIAA CFD Drag Prediction Workshop. San Francisco, June 2006 // http://aiaa-dpw.larc.nasa.gov/Workshop3/workshop3.html.

19. *Liepa P.* Filling holes in meshes // Eurographics Symposium on Geometry Processing. The Eurographics Association, 2003. P. 200—206.

20. *Botsch M., Sorkine O.* On linear variational surface deformation methods // IEEE Transactions on Visualization and Computer Graphics. 2008. Vol. 14(1). P. 213—230.

21. Free3D: https://free3d.com/3d-models/stl-aircraft.

22. *Cignoni P., Rocchini C., Scopigno R.* Metro. Measuring Error on Simplified Surfaces. Technical Report B4–01–01–96. Istituto I. E. I.–C. N. R. Pisa, Italy, January 1996.

Статья поступила в редакцию 21.07.22.