Improved Mesh Generation: Not Simple but Good [*]

Friedhelm Neugebauer and Ralf Diekmann

Department of Computer Science, University of Paderborn,
Fürstenallee 11, D-33102 Paderborn, Germany
{freddy, diek}@uni-paderborn.de

Abstract.

An improved algorithm for two-dimensional triangular mesh generation in arbitrary polygonally bounded domains with holes and isolated interior points is presented. The algorithm is quad-tree based and follows the ideas of Bern, Eppstein and Gilbert [1]. Using a rhomboid structure of the quad-tree and a number of new ideas for warping and centering points and for shifting grids, we are able to generate meshes with provably good angle bounds between 30° and 90° (except probably smaller angles of the polygonal boundary given as input).

Keywords. Unstructured Mesh Generation, Rhomboidal Quad-Tree, Provable Angle Bounds

1 Introduction

High quality mesh generation is a difficult but never the less very important task in nearly all areas of numerical simulation and image generation. It is estimated, that the USA industry would be able save up to 75% of their budget spent for numerical simulation, if automatic mesh generators were available [4]. As the structural complexity of simulation domains increases, there is a growing need for generating unstructured meshes. Triangulations or even mixed element discretizations are much more flexible than the often used square grids and can approximate complex boundaries with larger accuracy [3, 7, 9].

Since the work of Bern, Eppstein and Gilbert [1], quad-tree based unstructured meshing techniques for the 2D-case exist which generate "optimal" triangular meshes in arbitrary polygonally bounded domains with holes. Mitchell and Vavasis generalized the technique to three dimensions [10]. The 2D method guarantees a minimal angle of 15.25° and an *aspect ratio* of at most five [1]. It generates triangulations of minimal size in the sense that no other mesh meeting the same bounds on smallest angle and aspect ratio can have (asymptotically) less elements. It has been implemented showing its practical relevance (cf. e.g. [6]).

The quad-tree based meshing divides a *root box* covering the whole domain of interest until no box of the tree contains more than one element of the polygonal boundary. While dividing boxes into four smaller ones, a balancing condition ensures that neighboring boxes differ in their size by a factor of at most two. After the tree is constructed, nodes lying near the boundary are moved and shifted in a certain way to guarantee that the domain boundary is part of the final triangulation. Afterwards, the quad-tree boxes are triangulated especially taking care of those whose nodes were shifted [1].

Using square boxes as the basis for the tree construction causes a large number of right angled triangles in the final mesh. If then corner points of boxes have to be moved, small angles are unavoidable. This is the main reason why Bern, Eppstein and Gilbert's method (and also of Mitchell and Vavasis') produce small angles.

In our algorithm we use rhombs instead of squares as the basis for the quad-tree construction. Like squares, rhombs can be divided into four smaller ones. But additionally, they can be split into two perfect triangles, if they are equilateral and if the interior angles are chosen to 60° and 120°. This gives more flexibility for grid shifting and thus allows to meet angle bounds of 30° to 90° in the final mesh. We modify the way how quad-tree nodes near the boundary are shifted. Despite of corners of the boundary and single interior nodes, a point of the tree is never moved to a boundary segment, but always away from it to the interior of the domain (see Sec. 5.2). The connection between tree and boundary is made by inserting so called *border-boxes*.

[*]This work is partly supported by the DFG Sonderforschungsbereich 376 "Massive Parallelität", by the Heinz Nixdorf Institut Paderborn (Project "Netze"), and the EC ESPRIT Long Term Research Project 20244 (ALCOM-IT).

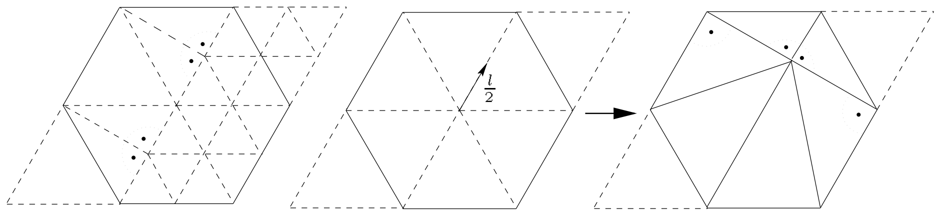

Figure 1: Advantages of the rhomboid mesh.

Of course, the improved angle bounds are not for free. As the basic tree construction differs not very much from Bern, Eppstein and Gilbert's, we conjecture that our method, too, generates meshes of minimal size. But the use of rhombs instead of squares removes one axis of symmetry which increases the number of cases to be considered.

The angle bounds of 30° to 90° which are met by our mesh generator improve the flexibility when using adaptive methods. The most common refinement strategies divide the smallest angle of the mesh into halves [8, 11]. It is known that in a number of finite element formulations, small angles cause stability problems [2]. With our algorithm, the smallest angle is larger than 30° which allows adaptive methods to be used without problems.

Additionally to polygons with holes, we allow isolated points lying in the interior of the polygonal region. This increases the complexity of the algorithm only slightly, but allows the user to control the mesh density to a certain extend.

The next section briefly summarizes the main ideas of the method. It is intended to give an intuition on how and why the algorithm works. A more formally description starts with Section 3 giving some basic definitions. It is followed by a section describing the algorithm (Sec. 4) and a part giving some ideas how the angle bounds are met (Sec. 5). In general, proofs are omitted as they mainly result from the constructions. The full paper will give missing details and proofs.

2 The Basic Idea

The algorithm presented here is able to triangulate polygonally bounded two-dimensional regions with polygonally bounded holes and single interior points. All angles of the generated mesh are (provably) larger than 30° and smaller than 90°. Like known methods from e.g. [1], it generates a quad-tree to separate the points and segments of the boundaries from each other. Differing to known methods, it uses rhomboid boxes instead of square ones. An equilateral rhomb with interior angles of 60° and 120° can be divided into two equilateral triangles (cf. Fig. 1). We will see other advantages of the rhomboid structure later on.

The generation of the quad-tree follows the standard technique as already described in [1]. A box is called *crowded* if it contains two or more segments or points of the domain which are *foreign*[1] to each other but reachable from each other through the interior of the domain without leaving the currently considered box. Boxes of the quad-tree are split into four equal sized smaller ones as long as they are crowded. A balancing condition ensures that neighboring boxes (those lying next to each other) differ in their size by a factor of at most two. The balancing condition used in our construction is slightly sharper than the one of [1]: A rhomb is split into four smaller ones if more than two of its neighbors are divided. The quad-tree construction ends up with a rhomboid mesh of the following properties:

1. A rhomb shares edges with neighbors of half, equal or double size.
2. At most two of the neighboring rhombs are of half size.
3. A box containing an isolated interior point is surrounded by rhombs of the same size.
4. A box containing a corner of the polygonal boundary is surrounded by rhombs of the same size up to a distance of three.
5. A rhomb containing a part of a segment (or a corner) has a distance of at least two (three) to rhombs containing parts of the boundary which are not directly connected to this segment (or corner).

[1]Segments or points are *foreign* if they do not have a common point.

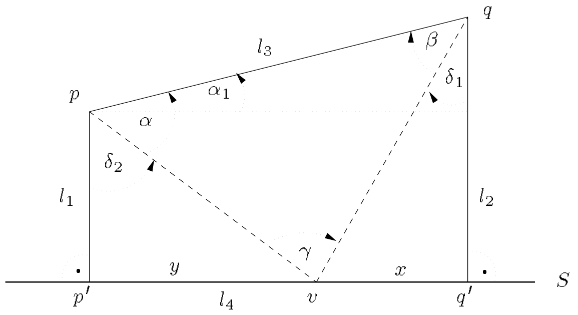

Figure 2: A border-box.

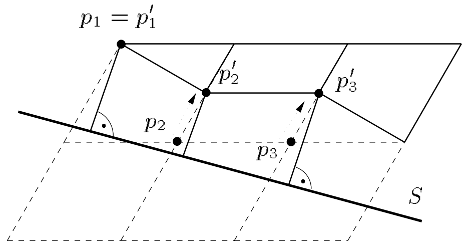

Figure 3: Creating border-boxes.

The mesh generation follows the basic structure from [1]. The quad-tree (of rhombs) is calculated, points of the tree lying near the domain boundary are shifted in a certain way (see section 5.2) and afterwards the individual boxes can be split into triangles independent of each other. Using a rhomboid mesh as the basis for the triangulation has a number of nice properties (cf. Fig 1):

1. A rhomb can be divided into two equilateral triangles.
2. A rhomb with at most two smaller neighbors can be triangulated into equilateral and right-angled triangles.
3. A corner of a triangulated rhomb with equal-sized neighbors and side length l can be moved by a distance of up to $l/2$ along any of its incident edges without violating the angle bounds of $30°$ to $90°$ (Fig 1).
4. Each corner of a rhomb can be considered as center of a hexagon (if it does not lie at the boundary of the quad-tree) and corners of neighboring rhombs are corners of the hexagon.

After the quad-tree is completely constructed, it is disconnected from the domain boundary by deleting all tree-edges cutting segments of the boundary. Corners of the tree lying near the boundary are shifted to the interior of the domain by using one of three possible directions (two sides and a diagonal of the rhomb pointing away from the boundary segment). The exact way how this shifting is performed, splits into a number of cases (depending on how the segment cuts the rhomb) and is described in Section 5.7.

Afterwards, the gap between the tree and the boundary is filled with so called *border-boxes*. They are constructed by inserting edges connecting outer corners of the tree to the boundary. These edges have to be perpendicular to the individual boundary segment. Figure 2 shows a border-box and Figure 3 shows how it is created. The corners p_2 and p_3 of the quad-tree lie to close to the segment S of the polygonal boundary. They are moved to the positions p_2' and p_3' and the edges (p_1', p_2') and (p_2', p_3') are inserted. Connecting p_1', p_2' and p_3' with S by edges which are perpendicular to S creates the border-boxes. Afterwards, all dashed lines can be deleted. The insertion of the edge (p_1', p_2') results in a rhomb which is degenerated to a triangle.

Border-boxes can be divided into three triangles by inserting the steiner point v on the boundary S (Fig. 2). Looking more closely at the way how corners of the quad-tree are shifted, it can be observed that l_1, l_2, l_3 and α_1 have a certain relation to each other (cf. Sec. 5.3). Thus, it can be shown that δ_1 and δ_2 lie between $15°$ and $60°$. Triangles of border-boxes with a δ-angle of less than $30°$ are melt with their neighboring triangle of the next border-box (by removing p' or q'). This results in the desired angle bounds of $30°$ to $90°$.

Corners of the domain boundary and isolated interior points have to be handled by their own. The quad-tree construction guarantees that a rhomb including an input point is surrounded by rhombs of the same size. Concerning only single points, it is not difficult to show that if the point is *centered* within a hexagon, the mid-corner of the hexagon can be moved to its position without running out of the angle bounds. Non-centered points can be centered by refining or derefining the quad-tree. Section 5.4 gives exact definitions and algorithms.

The triangulation of corner points of the boundary splits into a number of cases depending on the interior angle α between the two segments meeting at the corner. Three main cases can be identified:

1. $\alpha \leq 60°$:
   In this case the corner is cut off the rest of the domain by inserting two edges - one for each boundary segment - which are perpendicular to "their" segment and either meet in a common point or are connected by a third edge (cf. Fig. 12). The resulting polygon with four or five corners is triangulated by its own (without generating steiner points on the "cut-off-edges"). The edges to cut off the corner are chosen is a way to ensure that

the insertion of normal border-boxes can start right behind them. The details of the construction split into a number of cases depending on the relative position of the corner to the generated rhomboid mesh (see Sec. 5.6).

2. $60° < \alpha \leq 150°$:
   This case can be handled by shifting the rhomboid mesh in a certain way. The intersection of the bisector of α with the mesh gives a point which has to occur in the triangulation. Not only a single node of a neighboring rhomb is shifted to this point but the whole side of the rhomb. The amount of side shifting should not be to large. Therefore, the sides of neighboring boxes up to a distance of three are shifted, too. Afterwards, the insertion (and triangulation) of border-boxes in the normal way gives a valid triangular mesh meeting the angle bounds (cf. Sec. 5.7).
3. $150° < \alpha$:
   This case, again, splits into a number of parts. The triangulation is based on the rule to handle isolated interior points. The corner of the boundary is centered within a hexagon and the hexagon's mid-point is moved to the corner. The segments meeting at the corner then intersect triangles of the hexagon. The final triangulation depends on the angle between the segment and the side of the intersected triangle lying in the interior of the domain. Section 5.8 gives more details.

This was only a brief description of the technique. The next sections will give more details. It should be pointed out that the method as described here is only able to handle the interior *or* exterior of polygonally bounded area. Segments lying completely within the area of triangulation are not allowed because they would result in non-conforming meshes (because of the insertion of border-boxes at segments).

Additionally, is should be mentioned that the standard technique of *doubling* [1, 10] can be used to handle non-crowded boxes containing more than one segment (i.e. they contain two or more pieces of the boundary, but the pieces are not neighbored). In this case the triangulations of the different pieces do not influence each other and can be performed on different local copies of the box (one for each segment).

3 Definitions

Let P be a *polygonally bounded region*. The *boundary* δP of P is described by a set of *corners* $G_P := \{a_0, .., a_{n-1}\}$, $n \in I\!N$, and a set of *lines* $S_P := \{S[a_i, a_{(i+1) \bmod n}]\}$ where $S[a,b]$ defines a line between points a and b. Input to the mesh generator is a polygonally bounded region P, a set $H := \{H_1, .., H_h\}$, $h \in I\!N$, of holes, where each H_i is itself a polygonally bounded region, and a set $G := \{g_1, ..., g_j\}$, $j \in I\!N$, of single interior points. $\{P, H, G\}$ is a permitted input if the following conditions hold:

1. Each polygon has more than three corners and its border does not intersect itself.[2]
2. Each hole lies completely inside the region P and does not intersect other holes.
3. Each single point lies inside the polygonal region P and outside of the holes H.

The mesh generator creates additional lines and points (those of the quad-tree and those needed for the final triangulation). A line or point is called *element*. Two elements E_1, E_2 are called *foreign* if they do not have a common point. The *distance* $dist(E_1, E2)$ is defined as the shortest Euclidean distance between the two elements. The *circle* $C(c, r)$ is defined as the circle of radius $r \in I\!R$ centered at $c \in I\!R^2$.

During our algorithm, we often need to insert edges between two elements E_1 and E_2. If at least one of them is a line, the edge has to be perpendicular to it. If E_1 (E_2) is a line, then let F_1 (F_2) be a point on this line. We define a *segment* (E_1, E_2) as:[3]

1.	$(E_1, E_2) := S[E_1, E_2]$	if E_1 and E_2 are points.
2.	$(E_1, E_2) := S[F_1, E_2]$	if E_1 is a line and E_2 is a point. In this case, F_1 is chosen such that $S[F_1, E_2] \perp E_1$ (if possible).
3.	$(E_1, E_2) := S[E_1, F_2]$	if E_1 is a point and E_2 a line. Here, the choice of F_2 ensures that $S[E_1, F_2] \perp E_2$ (if possible).
4.	$(E_1, E_2) := S[F_1, F_2]$	if E_1 and E_2 are lines. F_1 and F_2 are chosen arbitrarily such that $S[F_1, F_2] \perp E_1$ (if possible).

[2]The definition of *crowded* requires a polygon to have more than three corners (otherwise, a rhomb would probably not split sufficiently). Inserting an extra node on one of the edges of triangular input polygons can avoid any problems.

[3]Note that the defined perpendicular sides exist in all cases considered here.

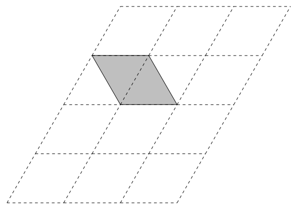
Figure 4: A virtual box.

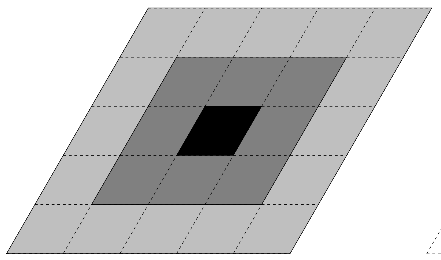
Figure 5: First neighbors.

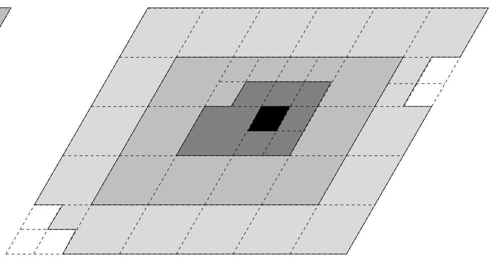
Figure 6: Third neighbors.

A *border-box* (cf. Fig. 2) is given by two points p, q and one line S of the boundary. It is defined by the segments $(p, q), (p, S), (q, S)$ and (q', p'), where p', q' are the endpoints of the segments (p, S) and (q, S).

The algorithm uses a *quad-tree*, a geometrical division of the plane into a tree of rhombs. Each rhomb is either a leaf of the tree or is split into four equal-sized children. During the construction, we sometimes work with *virtual* boxes which are possible neighbors of a rhomb but do not directly exist in the tree (because the rhomb lies at the boundary of the tree or the actual existing tree does not contain this box, see Fig. 4). A *neighbor* of a box might be a virtual box if not explicitly mentioned that is has to exist. A rhomb has four possible direct neighbors and we distinguish between five possible types of neighbors:

1. A neighbor of a rhomb is a rhomb of the same size sharing a side.
2. An extended neighbor is a rhomb of the same size sharing at least one corner.
3. A first neighbor of a rhomb B is an existing rhomb of the quad-tree which is not B, but shares at least one corner with B (Fig. 5).
4. A second neighbor of a rhomb B is an existing rhomb of the quad-tree which shares at least one corner with one of B's first neighbors but is not equal to B or one of B's first neighbors (Fig. 5 and 6).
5. A third neighbor of a rhomb B is an existing rhomb of the quad-tree which shares at least one corner with one of B's second neighbors but is not equal to B and not included in the set of B's first and second neighbors (Fig. 6).

The first, second and third neighbor definitions describe the rhombs around B with a distance of $0, 1$ and 2. A *corner* of a rhomb is one of its four vertices. The corners of the quad-tree are the points which are corners of its rhombs. The side of a rhomb is split if the extended neighbor sharing this side is split. All used quad-trees are *balanced*, i.e. each rhomb has at most two split sides and a first neighbor is of same or double size. This includes that, except the end points, at most one corner of the quad-tree is lying on each side of a rhomb. Such a corner is called *splitting point*. A rhomb B is *crowded* if the rhomb B' of size $x \cdot B$ centered at B contains at least two foreign elements and there exists a path between these elements lying completely in the interior of B' and P. The exact size of B' depends on the kind of elements included in B'. If it contains a polygonal corner, x has to be 7, otherwise, x is chosen to 5.

Boxes including points or polygonal corners must be surrounded by rhombs of the same size up to a distance which depends on the type of the corner (especially on the enclosing angle α between the two lines meeting at the corner). The *corner condition* for a rhomb B including the point p is given by:

1. p is a single point or $\alpha \leq 60°$: B and its first neighbors must have the same size.
2. $60° < \alpha \leq 150°$: B and its first, second and third neighbors must have the same size.
3. $\alpha > 150°$: B and its first, second, and third neighbors must have the same size.

Corners of the quad-tree lying near the polygon boundary are shifted into the interior of the domain. To be able to move nodes, certain *moving conditions* have to be satisfied which are rules on the refinement of the tree. Note that we can not avoid that neighboring boxes of different size are cut by boundary segments (otherwise, the mesh would become to fine). So we have to give rules on how to refine parts of the mesh to make nodes lying to close to the boundary "movable". Section 5.1 gives more details.

The *root box* is a rhomb containing the whole polygonally bounded region given as input.

During the quad-tree shifting, certain boxes, mainly around corners or single interior points, become *protected*. Protected boxes are split no further during the tree construction and the creation of border-boxes at boundary lines stops if a protected box is reached. This mechanism is used to prevent the creation of border-boxes from running into areas which are needed to handle the correct triangulation of corner points.

4 The Algorithm

Let $\{P, H, G\}$ be a permitted input. Throughout the algorithm, whenever a rhomb is split into four smaller ones, the balancing condition is propagated as far as necessary. The algorithm consists of the following steps:

1. Calculate a root rhomb.
2. As long as a crowded rhomb B exists split B into four equal sized smaller ones.
3. For each leaf rhomb including a corner point p ensure that the corner condition is valid. Split larger rhombs if necessary.
4. For each leaf rhomb B including a corner point p which belongs to a polygonal corner with an angle $\alpha > 150°$ do: If p is not *centered* in B then split B and all first, second and third neighbors of B. Let B' be the new rhomb including p. If p is not centered in B', three of the first neighbors of B' and B' are melt (cf. Sec. 5.4 for an exact description). Afterwards, p is definitely centered in the resulting larger rhomb $\tilde{B}$. Ensure that all first and second neighbors of $\tilde{B}$ are of the same size, i.e., melt rhombs is necessary, and protect $\tilde{B}$ and all its first and second neighbors.
5. For each leaf rhomb B including a single interior point p do the same centering as in 4). Protect $\tilde{B}$ and its first neighbors.
6. For each leaf box B including a corner point p which belongs to a polygonal corner with $\alpha \leq 60°$ do: Find the first rhomb B' lying completely inside of the polygonal corner (cf. Fig. 12) and protect all rhombs from B to B' and the first neighbors of B'. All protected rhombs must have the same size, otherwise they are split to satisfy this condition.
7. For each leaf rhomb B including a part of a segment do: Ensure that the moving condition is valid (cf. Sec. 5.1 for more details).
8. For each unprotected leaf rhomb including a part of a segment S do: Move the corner p which is nearest to S to the interior of the domain (if necessary, cf. Sec. 5.2 for details). Insert the edge (p, S). This process creates border-boxes. Rhomb edges intersecting the boundary can be deleted afterwards (cf. Fig. 3).
9. Delete all rhombs outside of the polygon.
10. Triangulate the remaining rhombs B:
    (a) B has degenerated to a triangle: Nothing has to be done.[4]
    (b) B has no split side: Inserting the shortest diagonal divides B into two triangles with angles bounded between $30°$ and $90°$.
    (c) B has one split side: Inserting the shortest diagonal divides B into two equilateral triangles. Connecting the splitting point with the opposite corner of B divides one of the triangles into two right-angled ones. The case of a rhomb with shifted corners having only one split side does not occur. Thus all angles are bounded between $30°$ and $90°$ (cf. Sec. 5.2).
    (d) B has two split sides: Inserting the shortest diagonal divides B into two equilateral triangles. If the splitting points are lying in different triangles both can be divided into right-angled ones. Otherwise, a new splitting point is inserted at the center of the diagonal. The two splitting points and the center point divide the triangle into four equilateral triangles and the other one can be divided into two right-angled triangles. Again, corners of such rhombs are not shifted (Sec. 5.2).
11. Triangulate border-boxes (cf. Fig. 2): Inserting a steiner point v on the segment S and connecting v to the corners p and q divides the box into three triangles. Section 5.3 shows that the angle bounds can be satisfied.
12. Triangulate protected rhombs B belonging to a single point p: Move the nearest corner of B to p. Inserting the shortest diagonal triangulates all protected rhombs (cf. Sec. 5.5).
13. Triangulate protected rhombs which belong to an angle $\alpha \leq 60°$ (cf. Sec. 5.6).
14. Triangulate protected rhombs which belong to an angle $\alpha \leq 60° \leq 150°$ (cf. Sec. 5.7).
15. Triangulate protected rhombs which belong to an angle $\alpha \geq 150°$ (see Sec. 5.8).
16. Merge neighboring triangles with an angle $\delta_1 < 30°$: If such a case occurs the corresponding triangle is right-angled and can be melt with its neighbor (cf. Sec. 5.3).

[4] A rhomb can degenerate to a triangle by either moving two of its corners to the same point or by deleting one of its sides and inserting a new edge. Both cases occur when nodes near the boundary are shifted to the interior (cf. Fig. 3, Sec. 2).

5 Angle Bounds

The following descriptions often need a quad-tree corner p and a corresponding hexagon. Let p be a corner of a rhomb which intersects the boundary. p is the center of a hexagon with the corners $a_i, \quad i \in \{0, ..., 5\}$ and the sides $(a_i, a_{(i+1) \bmod 6})$ of length l. Let v_i be the intersection point of the polygonal boundary and the sides (a_i, p) and let $dist(v_i, p) := \infty$ if v_i does not exist. Let $dist(v_j, p) \leq dist(v_i, p)\ \forall i \neq j$. Let $(\tilde{a}_i, p)$ be the side of double length which includes (a_i, p) and the part $(\tilde{a}_i, a_i)$ lies in the interior of the polygon (see Fig. 8).

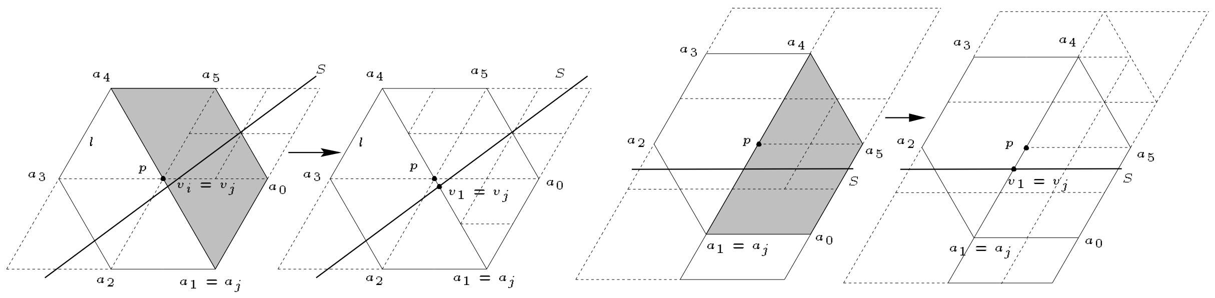

Figure 7: Moving conditions.

5.1 Moving Conditions

Step 7 of the algorithm requires to ensure that corners of the quad-tree lying near the boundary are movable. If p is a corner of four equal sized rhombs, it is movable and nothing has to be done (cf. Fig. 3).

If p is a splitting point or a corner of rhombs of different size, it is not necessarily movable and so the larger rhomb has to be split. Let l be the length of the largest rhomb belonging to p. The side (a_j, a_{j+3}) splits the hexagon around p into two equal parts. Figure 7 shows the situation with $j = 1$. If it is not possible to triangulate each part into equal sized triangles, the larger rhomb or only a part of it is split. This means that the larger rhomb is divided into two equal sized triangles and each triangle is split into four smaller ones if it intersects the considered part of the hexagon.

5.2 Moving Corners

The way how corners of the quad-tree near the polygonal boundary are moved to the interior depends on their distance from the boundary and on whether of not the rhombs meeting in that point are of the same size. Two different cases are possible:

1. p is a corner of four equal sized rhombs. The moving operation of p to p' is defined as:

   $p' := p$ if $dist(p, v_j) \geq 0.5l$
   $p' := (p, a_{(j+3)})^{‡}/2$ otherwise (see Figure 3)

2. p is a splitting point or a corner of rhombs of different size. Let l be the length of the largest rhomb side. The moving operation of p to p' is defined as:

   $p' := a_{j+3} := (v_j, \tilde{a}_{j+3})/2$ if $dist(p, v_j) < 0.25l$ (cf. Fig. 8)
   $p' := a_{j+3} := (p, a_{j+3})/2$ if $dist(p, v_j) \geq 0.25l$ (cf. Fig. 9)

   In this case moving p indicates a slightly different moving operation for its neighbors a_{j-1} or a_{j-2} where both have to be points of the smaller rhombs.[5]

   delete a_{j-1} and set
   $a_{j-2} := a_{j-2} - dist(v_{j-1}, a_{j-1})^{\dagger}$ if $dist(v_{j-1}, a_{j-1}) < l/8$ (Figure 8)
   $a_{j-1} := (a_{j-2}, a_{j-1})/2$ if $dist(v_{j-1}, a_{j-1}) < l/4$
   $a_{j-1} := (a_{j-2}, v_{j-1})/2$ if $dist(v_{j-1}, a_{j-1}) \geq l/4$ (Figure 9)

‡ $(p, q)/2 :=$ center of the segment (p, q).

[5] Depending on the position of large to smaller rhombs, the numbering is clock- or counter clockwise.

† Move a_{j-2} by distance $dist(v_{j-1}, a_{j-1})$ into the direction of the boundary (see Figure 8).

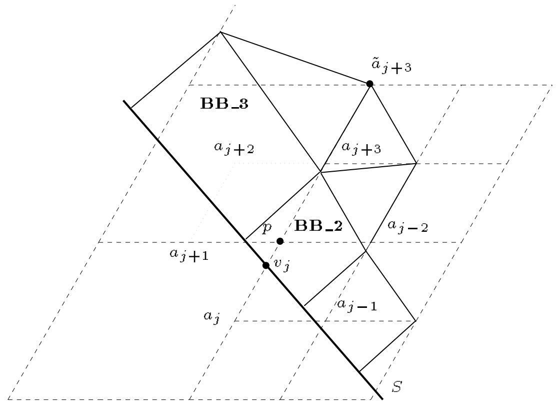

Figure 8: Moving Corners.

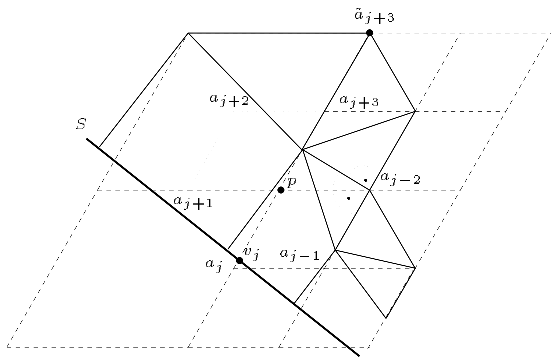

Figure 9: Moving Corners.

5.3 Triangulating Border-Boxes

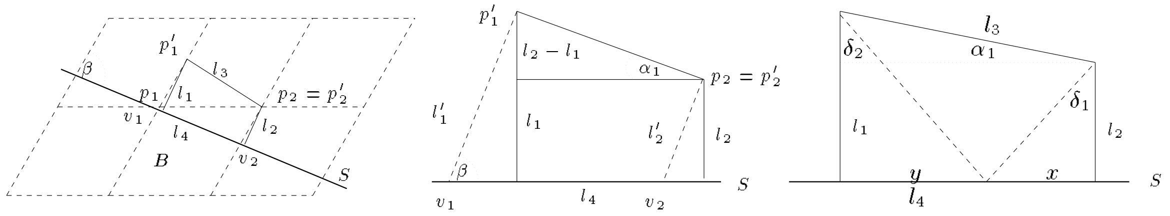

Figure 10: Bounds of a Border-Box.

To define a triangulation of a border-box we need some more information about the length of the sides l_1, l_2, l_3, l_4 and the angle α_1. All identifiers are shown in Figure 10 and w.l.o.g., we assume $l_2 \leq l_1$. The algorithm creates 7 different types of border-boxes:

BB_1 is created by a rhomb which is neighbored by rhombs of equal size (cf. Fig. 10).

BB_2 is created by a rhomb which is neighbored by a larger rhomb (cf. Fig. 8).

BB_3 is created by a rhomb which is neighbored by smaller rhombs (cf. Fig. 8).

BB_4 is created by a rhomb with two sides of length $l - x$, caused by grid shifting (cf. Fig. 13).

BB_5 is created by a rhomb with two sides of length $l + x$, caused by grid shifting (cf. Fig. 13).

BB_6 and **BB_7** are created during the triangulation of a corner with $\alpha \leq 60°$ (cf. Fig. 12).

Depending on the corner moving operation the length of the sides l'_1, l'_2 can be bounded. Depending on this bounds and the angle β between the side l'_1 and the segment S, the bounds of the sides l_1, l_2 are calculated. The results are shown in Table 1.

	$min(l_1)$	$max(l_1)$	$min(l_2)$	$max(l_2)$	$min(l_3)$	$max(l_3)$	$max(\alpha_1)$
BB_1	$0.433l$	$1.00l$	$0.433l$	$1.00l$	$0.866l$	$1.000l$	$30.00°$
BB_2	$0.433l$	$0.75l$	$0.433l$	$1.00l$	$0.866l$	$1.000l$	$30.00°$
BB_3	$0.650l$	$1.00l$	$0.866l$	$1.50l$	$0.866l$	$1.146l$	$40.90°$
BB_4	$0.433l$	$1.00l$	$0.433l$	$1.00l$	$0.741l$	$0.875l$	$37.43°$
BB_5	$0.433l$	$1.00l$	$0.433l$	$1.00l$	$0.991l$	$1.125l$	$29.78°$
BB_6	$0.433l$	$1.00l$	$0.433l$	$0.72l$	$0.567l$	$0.750l$	$45.00°$
BB_7	$0.433l$	$0.72l$	$0.433l$	$1.00l$	$0.750l$	$0.756l$	$37.09°$

Table 1: Bounds of Border-Boxes.

Now we have to calculate the bounds on the angle δ_1 and δ_2. We do not distinguish between the name of a side its length. The following general properties are available (cf. Fig. 10):

$$1)\ l_1 = \sin(\alpha_1)l_3 + l_2 \qquad 2)\ l_4 = \cos(\alpha_1)l_3 \qquad 3)\ x + y = l_4 \quad 4)\ \tan(\delta_1) = x/l_2 \qquad 5)\ \tan(\delta_2) = y/l_1$$

This leads to:

$$\frac{\cos(\alpha_1)l_3-(\tan(\delta_2)(\sin(\alpha_1)l_3+l_2))}{l_2} = \tan(\delta_1) \quad \text{and} \quad \frac{\cos(\alpha_1)l_3-\tan(\delta_1)l_2}{\sin(\alpha_1)l_3+l_2} = \tan(\delta_2)$$

Defining the angle[6]

$$\delta_1 := \arctan((\cos(\alpha_1)\cdot\frac{l_3}{l_2} - \tan(30°) + \sin(\alpha_1)) + 30° - \frac{\arctan(l_3)}{l_2} - \tan(30°) + \frac{l_2}{1.7}\cdot 30°$$

results to the bounds. The border-boxes **BB_3, BB_4, BB_5, BB_6, BB_7** are of a special type and sometimes are triangulated by inserting the shortest diagonal.

5.4 Centering Points

Let B be a rhomb and p a single point in the interior of B. All other identifiers are shown in Figure 11. Let l be the side length of the rhomb, $l = dist(a, b)$. W.l.o.g. $dist(p, a) \geq dist(p, b) \leq dist(p, c)$.[7] p is centered in B if

$$1.\ C((a,c)/2, 0.5l) \cap p = \emptyset^{\ddagger} \qquad 2.\ C(a, 0.6l) \cap p = \emptyset \qquad 3.\ C(c, 0.6l) \cap p = \emptyset$$

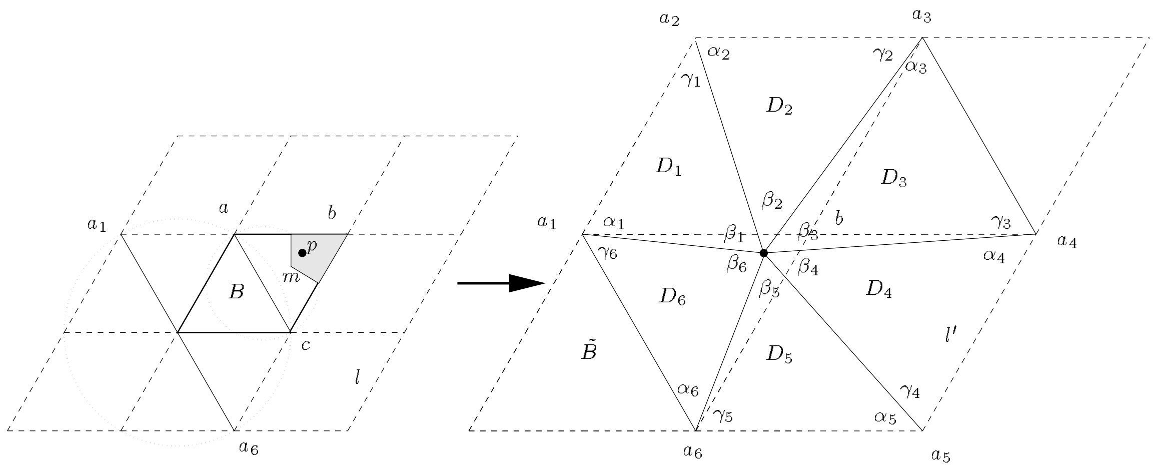

Figure 11: Center a point and the resulting triangulation.

Let $\tilde{B}$ be a rhomb of side length $2l = l'$ including B and b is a corner of $\tilde{B}$.

Claim: If p is not centered in B than p is centered in $\tilde{B}$.

Proof: $dist(p, (a_1, a_6)) \geq dist(m, (a_1, a_6)) = 1.299l = 0.6485l' > 0.6l'$ ∎

$\tilde{B}$ and the extended neighbors of $\tilde{B}$ include the rhombs to be melt in step 4 of the algorithm (Sec. 4).

[6] All values l_i are divided by l.
[7] p could lie anywhere inside of the shaded area in Figure 11.
[‡] p does not lie in the interior of a circle of radius $0.5l$ which is centered in the middle of edge (a, c).

5.5 Triangulating Rhombs with a Single Interior Point

Let B be a rhomb and p be a single point which is centered in B. All other identifiers are shown in Figure 11. Let b be the corner nearest to p. b is the center of a hexagon which could be triangulated with equilateral triangles. Moving point b to the single point p and using the condition that p is centered in B gives the following bounds to the length of the sides:

$$\begin{array}{ccccc|ccccc|ccccc}
0.5l & \leq & (a_1,p) & \leq & l & 0.866l & \leq & (a_2,p) & \leq & 1.33l & l & \leq & (a_3,p) & \leq & 1.5l \\
l & \leq & (a_4,p) & \leq & 1.5l & 0.866l & \leq & (a_5,p) & \leq & 1.33l & 0.5l & \leq & (a_6,p) & \leq & l
\end{array}$$

These length together with the fact that p is centered in B result in the desired angle bounds of 30° to 90°. The calculated bounds are displayed in Table 2.

i	$min(\alpha_i)$	$max(\alpha_i)$	$min(\beta_i)$	$max(\beta_i)$	$min(\gamma_i)$	$max(\gamma_i)$
1	60°	90°	49.11°	90°	30°	60°
2	60°	90°	40.89°	60°	40°	60°
3	60°	80°	35.26°	60°	60°	80°
4	40°	60°	40.89°	60°	60°	90°
5	30°	60°	49.11°	90°	60°	90°
6	30°	60°	60.00°	90°	30°	60°

Table 2: Angle bounds of the triangulation.

5.6 Triangulating Corners with Angle $\alpha \leq 60°$

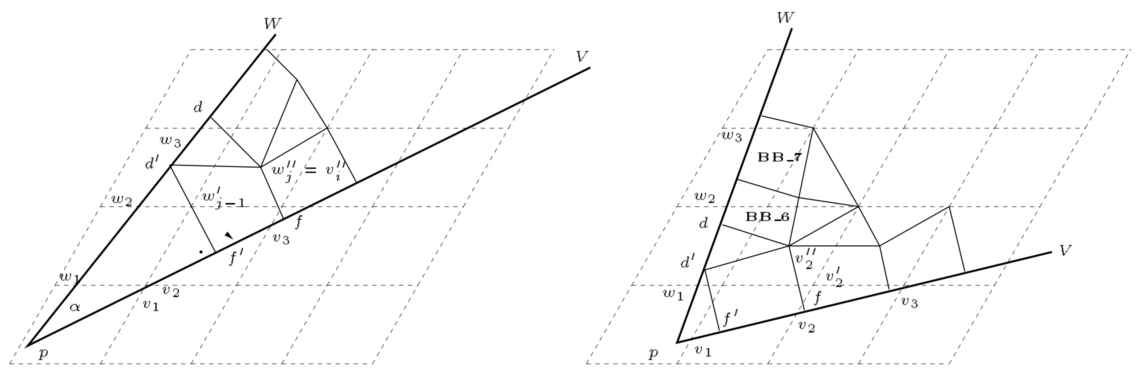

Figure 12: Triangulating a corner with an angle $\alpha \leq 60°$.

Let W, V be the segments with the common end-point p and the enclosing angle α. Let $v_1, ..., v_n$ be the intersecting points of V with the rhombs in sorted order. Let v'_i be the corner of a rhomb B which lies in the interior of P and (v_i, v'_i) is a part of a side of B. Let v''_i be the corner v'_i after corner moving. w_j, w'_j, w''_j are defined in the same way.

The first points v''_i, w''_j which satisfy the distance condition: $0.433l \leq dist(x, Y) \leq l$ (where $x \in \{v''_i, w''_j\}, Y \in \{V, W\}$) could be used for the triangulation. Two cases are possible:

1. $v''_i = w''_j$
   In this case the two points are common end-point of two border-boxes where (v''_i, V) and (w''_j, W) are perpendicular sides of the box. Also, v''_i must lie in the center of rhomb B and therefore, v'_{i-1} or w'_{j-1} must be a corner of B. Let w'_{j-1} be this corner. The segment (d', V) which includes (w'_{j-1}) splits the corner of the polygon into a triangle with angles $\alpha, 90° - \alpha$ and $90°$. The other part is split into a triangle and a border-box by the segment (d', v''_i). To maximize the angle between (d', V) and (d', v''_i), (d', V) was chosen only if

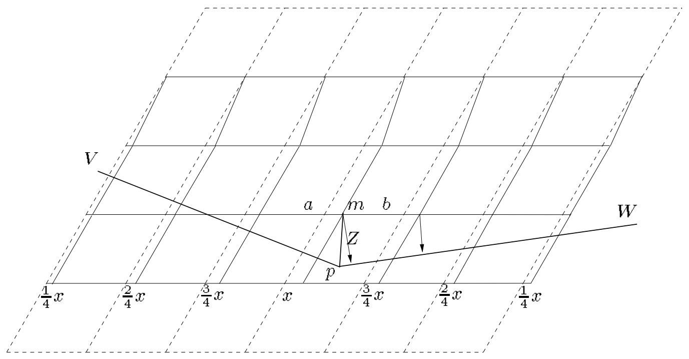

Figure 13: Shifting the rhomb mesh.

$dist(v_i'', V) \geq dist(w_j'', W)$. Otherwise the construction is done with the segment (f', W) (see Figure 12). A triangle (d', f', w_{j-1}) can be inserted if this construction fails.

2. $v_i'' \neq w_j''$
   In this case only one of the two points have to lie in the center of a rhomb B. Take this point to generate a triangulation in the same way as in case 1. A different situation occurs because there is only one regular border-box available for this point and so the second border-box has to be constructed. This gives two more cases which create small border-boxes. The situation is displayed in Figure 12(ri). If this construction fails, the corner can be split up with three segments $(W, w_j''), (w_j'', v_j''), (V, v_j'')$ triangulated by its own.

5.7 Triangulating Corners with Angle $60° \leq \alpha \leq 150°$

Let W, V be the segments with the common end-point p and the enclosing angle α. In this case the corner of the polygon must not be centered because the rhomb mesh will be adapted to the situation. This will generate rhombs with different sides. First calculate the bisector Z which divides the angle α into equal parts. Detect the first intersecting point m of Z and the rhomb mesh which satisfies the distance condition. m has the same distance to V and W. Let a and b be the corners of the intersected rhomb side with the point m (cf. Fig. 13). W.l.o.g., $x := dist(a, m) \leq dist(b, m)$. Rhomb sides with corner a and not parallel to the intersected side are moved by distance x. This includes that the point a is moved to m. The rhomb sides opposite of the moved sides are moved in the same direction but only by a distance of $\frac{3}{4}x$. This moving operation also occurs for the next two rhombs but the moving distance will be $\frac{2}{4}x$ and $\frac{1}{4}x$. The quad-tree generation ensures that all shifted sides have the same length. After shifting there are three rhombs with a side length l and $l - \frac{x}{4}$ and three rhombs with side length l and $l + \frac{x}{4}$ in each row. The triangulation of the shifted grid can be done in the normal way (creating border-boxes). This will create two equal triangles with angles $\alpha/2, 90°$ and $90° - \alpha/2$ at the polygonal corner. If an angle smaller than $30°$ occurs ($\alpha > 120°$) the triangle can be melt with its neighbor from the border-box (see Figure 13).

5.8 Triangulating Corners with Angle $\alpha \geq 150°$

Let W, V be the segments with the common end-point p and the enclosing angle α. In this case, p is centered within a rhomb B and the triangulation which would be done, if p was a single interior point is known. This triangulation is used to describe how to handle the obtuse corner (but does not appear in the final triangulation, cf. Fig. 14). Let $a_i, i = \{1..6\}$ be the corners of the (point) triangulation. Depending on the angle γ between the segment W and the side of the intersected triangle (the side must lie inside of the polygon and is given by (a_i, p)) a triangulation of the interior will be described. Three different cases occur:

1. $30° \leq \gamma \leq 60°$
   Inserting the edge (a_i, W) creates a triangle with angles $90°, \gamma$ and $90° - \gamma$.
2. $60° < \gamma \leq 90°$
   Inserting the edge (a_i, W) creates a triangle with angles $90°, \gamma$ and $90° - \gamma$. Because the angle $90° - \gamma$ is smaller than $30°$ the triangulation of the neighboring border-box has to create a triangle which can be melt with it.

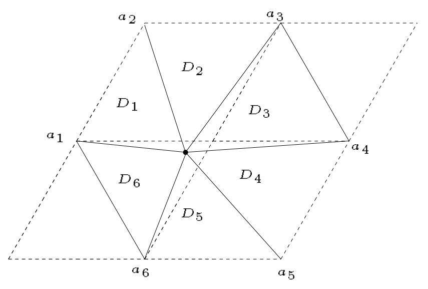

Figure 14:

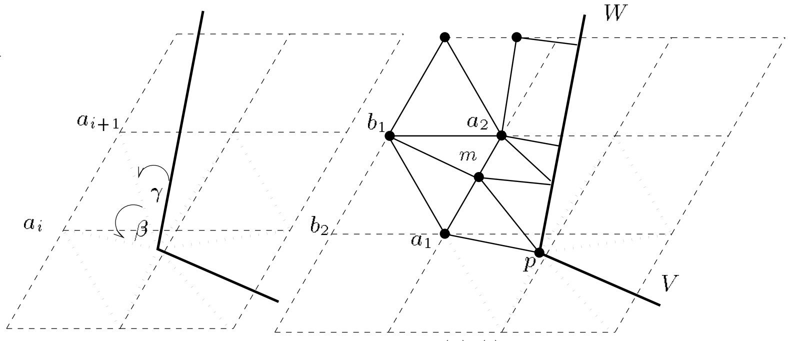

Figure 15: 3.(c).(i)

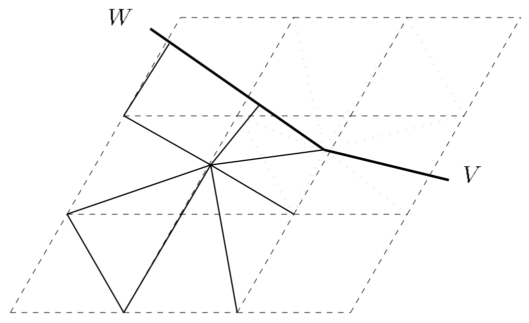

Figure 16: 3.(a)

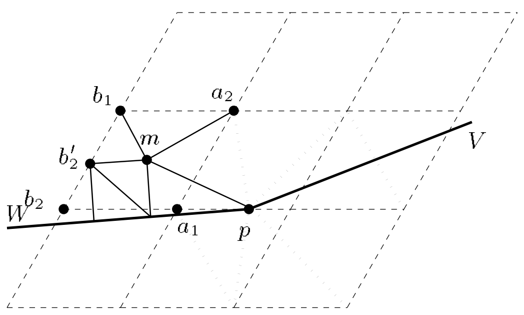

Figure 17: 3.(b)

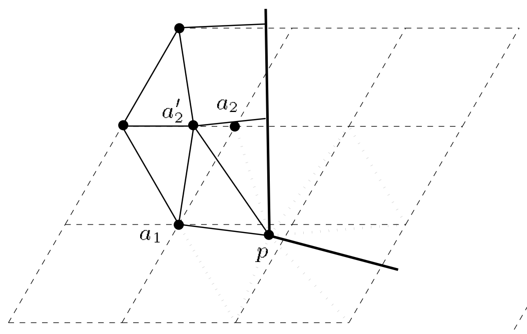

Figure 18: 3.(c).(ii)

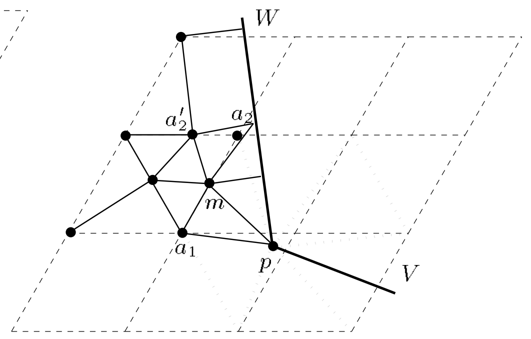

Figure 19: 3.(c).(iii)

3. $\gamma < 30^\circ$
In this case the edge (a_i, p) instead of the edge $(a_{(i-1) \bmod 6}, p)$ or $(a_{(i+1) \bmod 6}, p)$ has to be used and the angle $\gamma' = \gamma + \beta$ has to be considered (cf. Fig. 15). Since $30^\circ \leq \beta \leq 90^\circ$, the angle γ' has bounds between 60° and 120°. If $\gamma' \leq 90^\circ$, the triangulation from above can be used. $\gamma' > 90^\circ$ implies $\beta > 60^\circ$ and this only can happen using one of the three different triangles with an angle $\beta_i > 60^\circ$ which splits into five different cases: Let D_1, D_5, D_6 are be the possible triangles (using the same description as in Figure 11) (cf. Fig. 14).
   (a) The segment W intersects D_1 [or D_5] and D_6 lies inside the polygon (cf. Fig. 16).
   (b) The segment W intersects D_6 and D_1 [or D_5] lies inside the polygon. Let B be the rhomb sharing a side with triangle D_1 and b_1, a_2, a_1, b_2 are corners of B. Inserting the center m of B and the edges $(m, W), (m, p), (m, a_2)$ and (m, b_1) gives a possible triangulation (cf. Fig. 17).
   (c) The segment W intersects D_2 and D_1 lies inside the polygon [or the segment W intersects D_4 and D_5 lies inside the polygon].
      i. The corner $a_2[a_6]$ will not be moved (cf. Fig. 15).
      ii. The corner $a_2[a_6]$ will be moved to the point $a_2'[a_6']$ and the edge $(a_2', p)[(a_6' p)]$ divides the angle γ' into two angles greater or equal to 30° (cf. Fig. 18).
      iii. The corner $a_2[a_6]$ will be moved to the point $a_2'[a_6']$ and the edge $(a_2', p)[(a_6', p)]$ divides the angle γ' into one angle smaller than 30° (cf. Fig. 19).
   (d) In all other possible cases, γ' will be smaller than 90°.

The same triangulation can be done with segment V and each one only effects the intersected triangle and its neighbor. This gives the solution that the triangulation with W and V can be combined to an independent triangulation if $\alpha \geq 150^\circ$. All not triangulated rhombs can be triangulated in the normal way.

6 Conclusions

A new algorithm for two dimensional triangular mesh generation of a polygonal region with holes and single points was presented. All angles of the generated mesh are bounded between 30° and 90°, except possibly smaller input angles. The algorithm uses a quad-tree of rhombs to separate elements of the given boundary from each other. Shifting nodes of the tree lying near the boundary to the interior and inserting border-boxes at the boundary allows the triangulation to meet the angle bounds. Corners of different type are handled by their own resulting in a number of distinct cases.

Allowing isolated interior points in the input gives a possible user the chance to control the mesh density.

Missing up to now is an analysis of the size of the resulting mesh and the running time of the algorithm. We conjecture that the meshes will be of minimal size but a proof has to be given. Also missing is an analysis of where to place the steiner point while triangulating a border-box to obtain optimal angles. Details will appear in a full paper.

Up to now, only parts of the method are implemented. We are working on this, as well as on a parallel implementation on distributed memory machines.

Acknowledgments

Thanks are to our colleagues Petra Berenbrink, Stephan Blazy, Uwe Dralle, Klaus Meyer, Burkhard Monien, Olaf Schmidt, and especially to Ralf Weickenmeier for many helpful hints, comments and suggestions.

Additionally, we thank the anonymous referees for many useful comments.

References

[1] M. Bern, D. Eppstein, J. Gilbert: Provably Good Mesh Generation. *Proc. 31st Symp. on Foundations of Computer Science (FOCS '90), pp. 231-241, 1990.*

[2] S. Blazy: Personal communication.

[3] P.L. George: *Automatic Mesh Generation.* John Wiley & Sons, 1993.

[4] *The CUBIT Mesh Generation Research Project.* Sandia National Lab, 1995.
WWW: `http://www.cs.sandia.gov/HPCCIT/cubit.html`

[5] R. Diekmann, D. Meyer, B. Monien: Parallel Decomposition of Unstructured FEM-Meshes. *Proc. of IRREGULAR '95, Springer LNCS 980, pp. 199-215, 1995.*

[6] R. Diekmann, U. Dralle, F. Neugebauer, T. Römke: PadFEM: A Portable Parallel FEM-Tool. *Proc. Int. Conf. on High-Performance Computing and Networking (HPCN-Europe '96), LNCS 1067, Springer-Verlag, pp. 580-585, 1996.*

[7] K. Ho-Le: Finite Element Mesh Generation Methods: A Review and Classification. *Computer Aided Design, Vol. 20, No. 1, pp. 27-38, 1988.*

[8] M.T. Jones, P. Plassmann: Adaptive Refinement of Unstructured Finite-Element Meshes. MSC Preprint P562-0296, Argonne Nat'Lab., to appear in *J. of Finite Elements in Analysis and Design.*

[9] E. Melissaratos, D.L. Souvaine: Coping with Inconsistencies: A New Approach to Produce Quality Triangulations of Polygonal Domains with Holes. *Proc. 8th Annual ACM Symposium on Computational Geometry, 1992.*

[10] S.A. Mitchell, S.A. Vavasis: Quality Mesh Generation in Three Dimensions. *Proc. 8th ACM Conf. on Comp. Geometry, pp. 212-221, 1992.*

[11] M.-C. Rivara: Mesh refinement processes based on the generalized bisection of simplices. *SIAM Journal on Numerical Analysis, 21(3), pp. 604-613, 1984.*