European Conference on Computational Fluid Dynamics
ECCOMAS CFD 2006
P. Wesseling, E. Oñate and J. Périaux (Eds)

ADAPTIVE CONTROL SPACE STRUCTURE FOR ANISOTROPIC MESH GENERATION

Tomasz Jurczyk and Barbara Głut

AGH University of Science and Technology
Institute of Computer Science, 30-059 Kraków, Polska
e-mail: {jurczyk,glut}@agh.edu.pl

Key words: Mesh Generation, Mesh Adaptation, Metric, Anisotropy, Control Space

Abstract. *The article concerns problem of automated generation of anisotropic finite element meshes on 3D surfaces. The meshes are created using a modification of Delaunay incremental insertion algorithm working in Riemannian space (using* metric transformation *approach). During the meshing process the generator relies on a special structure (called* control space*) to provide the required size and shape of elements at any point within the domain being discretized.*

Organization of this control space *structure directly influences both the quality of the obtained mesh and the efficiency of the meshing process. It should allow to combine sizing information gathered automatically from different sources (in both discrete and continuous form). The* quadtree grid *and* background mesh *structure are typically used for control space representation.*

This article describes an implementation of a adaptive control space structure with emphasis on efficiency and versatility necessary in automated adaptation process.

1 INTRODUCTION

Unstructured anisotropic mesh adaptation is now widely used in numerical simulations to improve the accuracy of the solution, reduce the computational time and properly capture the behavior of the simulated phenomena. With respect to mesh generation the adaptation task can be divided into two phases: preparing the element sizing map (storing information like required size, stretching, directionality, etc.) and discretizing the domain following the prescribed sizing map as closely as possible. This element sizing map (called usually as control space) can have various structure, typically a uniform grid, quadtree (octree) or background mesh [1, 2, 3].

In this article we present an automated procedure of adaptive creation of the control space which takes advantage of a number of geometrical sources of sizing information as well as potential input from the user or from the numerical solver. The sizing information for creation of surface meshes is stored in the unified form of metric within discrete nodes of the adaptive control space structure (either quadtree or background mesh) and is interpolated between the nodes. The method of interpolation has to be carefully selected since it influences both quality of the mesh and the efficiency of the meshing process.

In cases where sizing data is gathered from several different sources (e.g. model geometry, adaptation process, user input) we start with creation of separate control space structures for each type of source. Then the resultant control space is calculated using the intersection procedure, which takes into account both structure and metric data from the nodes of all control spaces.

In order to assure good quality of the created meshes the metric field within the control space can be subjected to an additional smoothing procedure which gives control over the maximum gradation ratio over neighboring elements within the mesh. For domains composed of several surface patches an iterative procedure of adjusting control space for adjacent patches was also implemented.

1.1 Surface mesh generation and adaptation

Starting from the BREP description of the domain, the developed generator is able to produce unstructured (anisotropic if necessary) triangular or quadrilateral meshes in 2D space or on three-dimensional surfaces, which can be used directly or as an intermediate step for 3D tetrahedral meshing. The process of mesh generation is hierarchical and consists of several subsequent phases. First, information about the required element size throughout the domain is automatically gathered and all contours are accordingly discretized. Each surface patch is then triangulated using a modification of the Delaunay incremental insertion algorithm[4]. If the quadrilateral mesh is requested, the conversion procedure is used[5]. Finally, several methods of mesh quality improvement are applied.

All 2D-meshing operations (triangulation, conversion to quadrilaterals and smoothing) are working in parametric space of the given surface patch with Riemannian metric introduced using the *metric transformation* approach [6]. During the meshing process the generator uses a special *control space* structure to provide the requested metric (governing the size, shape and directionality of elements) at any point within the patch being discretized. Each surface patch has a separate control space structure storing sizing information in a metric *transformation matrix* form in parametrical space of the given patch.

The sizing metric can be obtained directly from the user (in a number of convenient methods), retrieved from the solver as a part of an adaptation process (as a set of discrete points with sizing information) and/or gathered automatically from the geometrical and topological description of the discretized domain. The metric definition source can have different dimensions: 0D (e.g. for data from adaptation given in the nodes of a computational mesh), 1D (e.g. for curvature of boundary contours), 2D (e.g. for curvature of surface patch), and 3D (e.g. from user description). All available sources are processed and stored in a single adapted control space structure.

2 CONTROL SPACE FOR SINGLE PATCH

In order to efficiently process metric sources of different form, a hierarchy of control space structures has been created. Most basic, abstract *ControlSpace* class defines the essential functionality of a control space, which is to return the required metric at any given point within the domain being discretized.

The form of metric which is stored in the control space depends on the selected meshing method. In our case, the *transformation matrix* $\mathbf{M_t}$ was selected, which is used during discretization to transform coordinates of a given point P to the Riemannian space:

$$P' = \mathbf{M_t}(P)P \; . \tag{1}$$

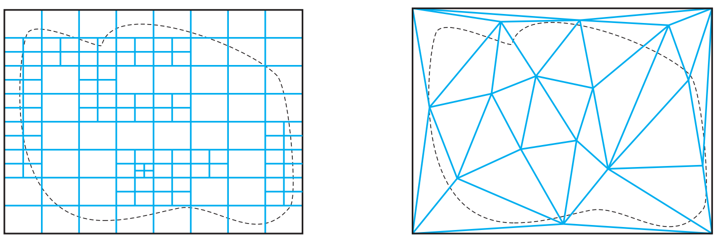
$\mathbf{M}_t$ is a two-dimensional symmetrical matrix related to *metric tensor* $\mathcal{M}$ as follows:

$$\mathcal{M}(P) = \mathbf{M_t}(P)\mathbf{M_t}^T(P) \tag{2}$$

and for which operations like interpolation, minimization or comparison can be efficiently calculated[6]. For patches on non-planar surfaces the parameterization matrix $\mathbf{M_p}$ is additionally required.

2.1 Adaptive control space

Two adaptive structures of control space were created, implementing balanced quadtree structure and background mesh (Fig. 1). Both structures implement adaptivity and are able to initialize the control space from either continuous (e.g. curvature of surface or analytical equations) or discrete sources (e.g. discrete nodes or segments with metric). In case of insufficient data the metric in some control nodes has to be interpolated or extrapolated. The structure of the control space can be further adapted to variation of metric or parameterization.

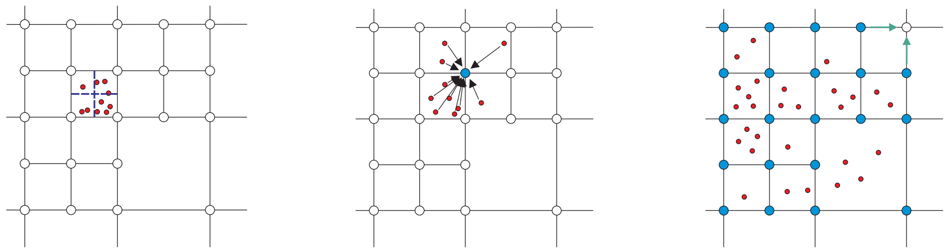


(a) quadtree (b) background mesh

Figure 1: Structure of adaptive control space

2.2 Introducing discrete data

For metric source in a form of a set of discrete data, the control space structure is accordingly adapted in order to avoid situation where any control space element (quadtree leaf or mesh triangle) contains data of too different metric description (Fig. 2(a)). After adaptation of the structure, the nodes of control space are initialized with the weighted average of metric from the adjacent nodes (Fig. 2(b)). Finally an iterative procedure of extrapolation of metric for still uninitialized control nodes (having no source points with metric description in the incident elements) is performed (Fig. 2(c)).

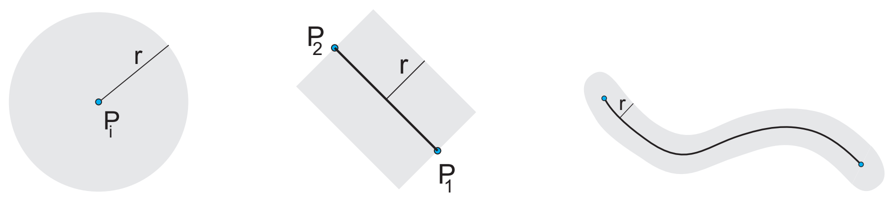


(a) structure adaptation (b) control nodes initialization (c) metric extrapolation

Figure 2: Initializing control space structure using discrete data

In order to facilitate precise and convenient definition of size and stretching of elements throughout the domain, the metric can be defined in each discrete node in a few ways (Fig. 3)[7]:

- (P, M, r) – metric M is treated as constant in the neighborhood of the point P with the radius $r \geq 0$. Outside this circle the metric is interpolated using control nodes lying nearby.
- (P_1, P_2, M, r) – along the segment P_1P_2 and its neighborhood with length r the constant metric M is assumed.
- $(c(t), t_1, t_2, M, r)$ – constant metric M is assumed along the given parametrical curve $c(t)$ and its neighborhood with length r.



(a) radial (b) segment (c) along curve

Figure 3: Extended metric definition

2.3 Introducing continuous data

For continuous data (e.g. metric in analytic form or gathered from the curvature of the surface) all nodes of the current control space structure are initialized with the available data (for already initialized nodes the minimum metric is calculated). Then each control space element (quadtree leaf or background mesh triangle) is checked for metric difference between its vertices and the element is recursively split if necessary.

2.4 Calculating minimum of two control spaces

Computing the minimum of two adaptive control space structures CS_1 and CS_2 consists of following phases:

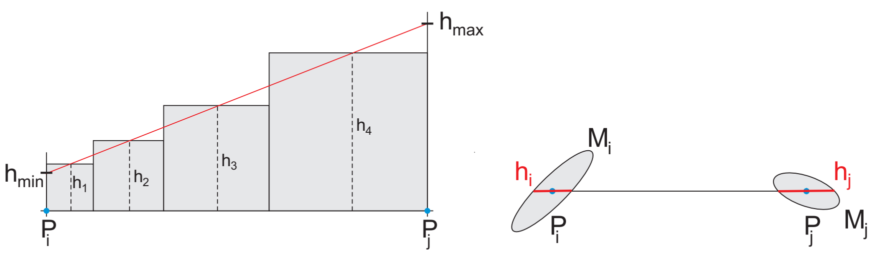
1. For each control node P_j of CS_1 the minimum metric $\min(M_j^{CS_1}, M^{CS_2}(P_j))$ is calculated.

2. All control nodes from the CS_2 are introduced into CS_1 as discrete metric sources which may modify the structure of the CS_1 structure. The refinement takes place if the difference between $M^{CS_1}(P_i)$ and $\min(M^{CS_1}(P_i), M_i^{CS_2}))$ is larger than some threshold, where $M_i^{CS_2}$ is the metric in the i^{th} control node of CS_2 and $M^{CS_1}(P_i)$ is the metric calculated from the CS_1 at the coordinates of the given control node P_i.

3. Operation described in the first step is performed again.

2.5 Smoothing of the metric field

Metric data come from different sources and the resulting metric map can have unacceptably large variation. Since the required smoothness of the mesh elements may depend on the application, the gradation parameter d_r was introduced. This gradation parameter simply limits the maximum increase of the required edge length (in any direction) between any two points of unitary distance (in metric space). Figure 4(a) presents an example case of maximum gradation of 1D elements length, where $d_r = h_{i+1}/h_i$.



(a) gradation ratio

(b) requested length along selected direction

Figure 4: Acceptable metric transition

2.5.1 Acceptable metric transition for a pair of nodes

For each pair of control nodes P_i and P_j we start with comparing the metrics M_i and M_j. If diff(M_i, M_j) is smaller than some threshold the rest of this procedure may be skipped. Otherwise, the requested length of elements along the selected direction P_iP_j is calculated as h_i and h_j (Fig. 4(b)). Maximum and minimum values are set as $h_{\max}$ and $h_{\min}$.

The coefficient a is calculated as

$$a = \frac{h_{\max} - h_{\min}}{d(P_i, P_j)} \tag{3}$$

and is compared with $a_{\max}$ which is defined by the gradation ratio:

$$a_{\max} = 2\frac{d_r - 1}{d_r + 1} \ . \tag{4}$$

If $a > a_{\max}$ the values are accordingly adjusted:

$$a \leftarrow a_{\max} \tag{5}$$

$$h_{\max} \leftarrow ad(P_i, P_j) + h_{\min} \ . \tag{6}$$

The maximum gradation factor s is established as:

$$s = 1 - \frac{(1 - d_r)(1 - a/2)d(P_i, P_j)}{h_{\min}} \ . \tag{7}$$

Using this factor metrics in both nodes can be adjusted (Fig. 5):

$$M_i \leftarrow \min(M_i, sM_j) \tag{8}$$

$$M_j \leftarrow \min(M_j, sM_i) \ . \tag{9}$$

In order to increase the efficiency of the whole control space smoothing procedure, the actual computation of the minimum metrics described above takes place only if the difference between metrics diff(M_i, M_j) is larger than some threshold proportional to the factor s.

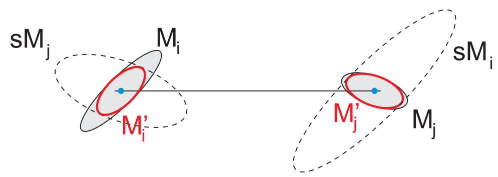
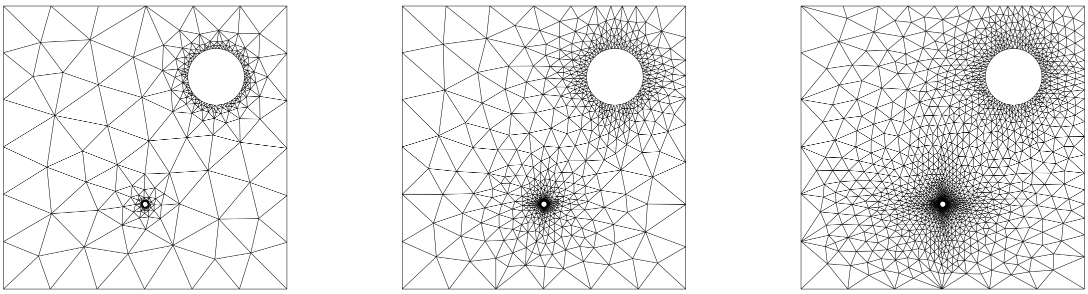


Figure 5: Constraining metrics

2.5.2 Global smoothing

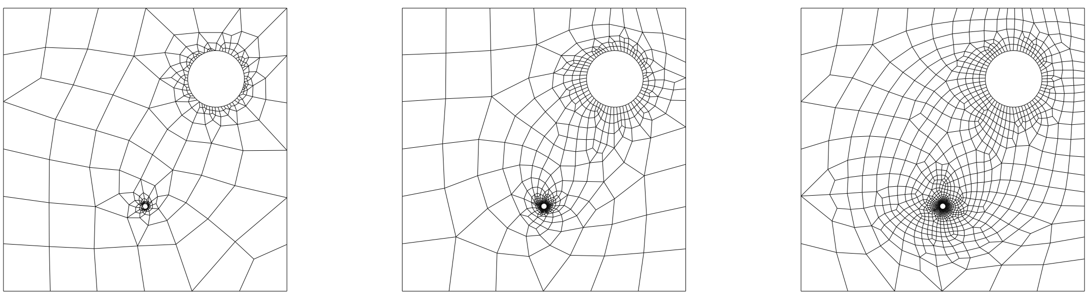
In order to ensure proper variation of metric throughout the whole mesh, the two-point gradation adjusting procedure has to be systematically extended to all nodes of the given control space. For quadtree structure a recursive method of tree traversing is used consisting of two steps. In each step the nodes of all tree elements are adjusted with the nodes of sub-elements. In the first step the tree is updated "upwards", propagating the smoothing from the leaves of the quadtree up to the main level. In the second step the reverse "downwards" direction is used. For background mesh structure of the control space an iterative procedure is implemented, where each node is adjusted with all incident nodes. The set of checked nodes contains initially all nodes of the background mesh. During the procedure each checked node is removed from the set, while nodes which had theirs metric changed are again included into this set.

Figures 6 and 7 present meshes of a simple rectangular domain with circular holes created for different values of the gradation ratio parameter and without smoothing.



(a) without smoothing (b) gradation ratio $d_r = 8$ (c) gradation ratio $d_r = 2$

Figure 6: Gradation ratio for smoothing of metric field



(a) without smoothing (b) gradation ratio $d_r = 8$ (c) gradation ratio $d_r = 2$

Figure 7: Gradation ratio for smoothing of metric field (quadrilateral mesh)

3 AUTOMATED CONSTRUCTION OF CONTROL SPACE

Algorithm 1 shows subsequent steps of automated creation and actualization of the control space structure. The domain being discretized is assumed to consist of one or more surfaces patches (possibly connected), where each surface provides a parametric representation. Each patch is triangulated separately in its parametric space. However, special care must be taken for assuring smooth variation of elements over adjacent patches.

foreach *surface patch* **do**
    create initial control space
mark all 3D-contours (c^{3d}) as *invalid*
while $\exists$ invalid c^{3d} ***or*** $\exists$ invalid *surface patch* **do**
    foreach invalid c^{3d} **do**
        mark all surface patches incident to this contour as *invalid*
    foreach invalid *surface patch* **do**
        update control space with length of discretization segments of c^{2d}
        I step of patch discretization – triangulation of boundary nodes only
        mark this patch as *valid-I*
        update control space with proximity of boundary segments
        foreach *incident* c^{3d} **do**
            if c^{3d} *discretization is too coarse for* Γ^{2d} **then** mark c^{3d} as *invalid*
foreach ***not*** valid-II *surface patch* **do**
    II step of patch discretization – insertion of inner nodes, conversion, smoothing, etc.
    mark this patch as *valid-II*

Algorithm 1: Preparation of control space during process of mesh generation

3.1 Creating the initial control space structure

The procedure starts with determining the bounding rectangle of the given surface patch (in its parametrical space) and creation of the main level grid. At he first step the structure of the control space is initialized and adapted for the curvature of the surface. For each control node in the initial main level grid of the control space structure the curvature data is retrieved from the parametrical description of the surface patch. The required metric in form $M = (\alpha, h_1, h_2)$ is calculated (from principal curvatures and directions of the surface patch) and adjusted using the curvature ratio coefficient c, maximum stretching $d_{\max}$, minimum and maximum length. In the adaptation phase in all elements of the control space the metric within element is calculated directly from the surface curvature and compared with the value obtained via the interpolation from element vertices. If the difference of metrics is larger than a given threshold the quadtree leaf or background mesh triangle is split accordingly and recursively checked. In case where the curvature of surface is not available for all nodes of the control space (e.g. the surface patch is note regular) an additional extrapolation procedure has to be performed.

If there is any existing data given by user (in any form) or obtained from the adaptation procedure, a separate control space (or control spaces) is created and minimum control space is computed.

In the next step the already existing control space is further adjusted using the curvature of contours. Along each curvilinear contour within the discretized domain the metric $M = (\alpha, h_1, h_2)$ is calculated, where h_1 is computed from the principal curvature, $h_2 = d_{\max} h_1$ and α is tangent to the contour. This information is introduced into the control space using the extended metric representation described in Sec. 2.2.

Since the nodes of the control space are using coordinates from the parametric space of the surface patch, the structure of the control space is further adapted with respect to the parameterization irregularity. This step is necessary if the interpolation of metric (used during meshing process) is to be calculated using shape functions in parametric space. Finally the whole control space is smoothed according to the given gradation ratio.

3.2 Updating control space

After discretization of contours and initial triangulation of boundary nodes for the given surface patch, additional sizing information becomes available and can be used to check and adjust the control space:

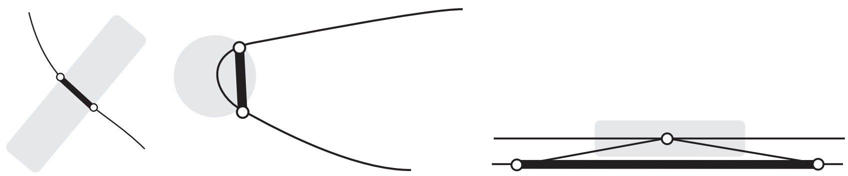
- *Length of boundary segments* (Fig. 8(a)). Length l_i^M of each boundary edge is calculated according to local metric space. If $l_i^M < \epsilon_1$ (in our work the parameter ϵ_1 is set to 0.8) the control space in the neighborhood is updated accordingly (using the length of this segment as a reference). Additionally if $l_i^M < \epsilon_1/2$ or $1/l_i^M < \epsilon_1/2$, the contour containing this edge is marked as *invalid* and should be discretized again.

- *Proper representation of curvilinear boundaries* (Fig. 8(b)). Distance between the mesh segment and the underlying curvilinear contour is measured, and in case of too rough discretization the control space in this area is accordingly updated (using curvature of the contour directly) and the whole contour is marked as *invalid*.

- *Vicinity of boundaries* (Fig. 8(c)). Triangles of the initial Delaunay triangulation with metric area lower than some threshold ϵ_2 (equal to 0.2 in our work) are inspected and the control space is appropriately updated.

4 USING CONTROL SPACE DURING MESHING

During the mesh generation, each time the local metric has to be established at the given coordinates, the proper value is retrieved from the control space.



(a) boundary segment (b) boundary curve (c) boundary vicinity

Figure 8: Updating control space

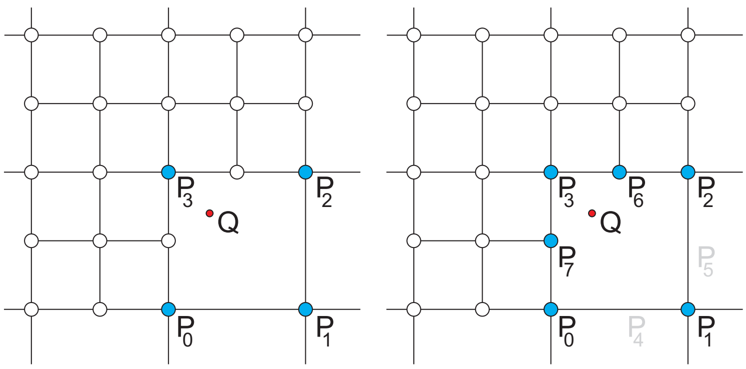
If the coordinates of the inspected point are close enough to the point, where the metric was recently calculated, the current metric may be considered valid, and no further operation would be needed. The maximum distance between points, which can be called "close enough" is calculated in the metric space (i.e. it depends on the required length of edges according to the current metric).

4.1 Quadtree grid

The procedure starts with localizing the leaf containing the point for which the metric value is required. Then the metric is calculated from the vertices of the rectangular leaf using linear or quadratic interpolation (Fig. 9):

$$\mathbf{M_t}(Q) = \sum_{i=0}^{n} N_i(\eta, \xi)\mathbf{M_t}(P_i) \tag{10}$$

where n is the number of nodes (4 for linear or 8 for quadratic "serendipity" interpolation) and $N_i(\eta, \xi)$ are the shape functions given in Table 1.



(a) linear (b) quadratic

Figure 9: Interpolation for regular grid

Table 1: Shape functions for quadtree nodes ($\eta, \xi \in [0, 1]$)

	linear	quadratic
N_0	$(1-\eta)(1-\xi)$	$(1-\eta)(1-\xi)(-2\eta-2\xi+1)$
N_1	$\eta(1-\xi)$	$\eta(1-\xi)(2\eta-2\xi-1)$
N_2	$\eta\xi$	$\eta\xi(2\eta+2\xi-3)$
N_3	$(1-\eta)\xi$	$(1-\eta)\xi(-2\eta+2\xi-1)$
N_4		$4(1-\eta)(1-\xi)\eta$
N_5		$4(1-\xi)\eta\xi$
N_6		$4(1-\eta)\eta\xi$
N_7		$4(1-\xi)(1-\eta)\xi$

4.2 Background mesh

The procedure starts with searching for the containing triangle, which is done by traversing the triangles in the mesh in the direction of the given point (an additional quadtree structure is used for selection of good starting triangle for this traversing algorithm). After the containing triangle is found the metric is interpolated from a local set V_* of vertices, depending on the selected method (detailed description can be found in [8]). From this set of nodes the resulting metric is calculated using the weighted average formula:

$$\mathbf{M_t}(Q) = \frac{1}{\sum_{P_i \in V_*} \omega_i} \sum_{P_i \in V_*} \mathbf{M_t}(P_i)\omega_i \tag{11}$$

where $\omega_i = d(Q, P_i)^{-2}$ is the inverse squared-distance weight, calculated from points coordinates in parametric space. If $d(Q, P_i) < \epsilon$, the metric from the point P_i is taken directly, skipping the interpolation procedure.

4.3 Storing the parameterization matrix

The second matrix required during the meshing process, the *parameterization matrix* can be calculated directly from the parametric representation of the surface patch or it could be stored within the control space. Storing this matrix within the control space structure allows to increase significantly the efficiency of the meshing process without visible degradation of the mesh quality.

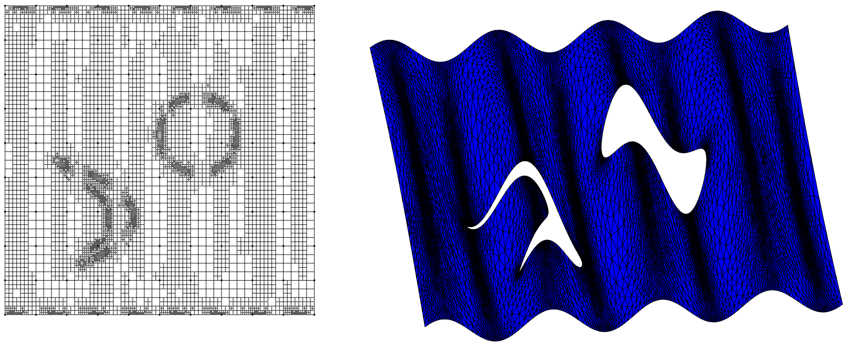
The parameterization matrix is stored in the nodes of the control space structure and is interpolated in precisely the same way as the metric transformation matrix.

5 EXAMPLES

Figure 10 presents the triangular mesh created for an analytical surface patch with inner holes. The parametric surface description is

$$\mathbf{p}_2 : (u, v) \to (u, v, 1.5e^{-0.001(u^2+v^2)} \sin(2u) \cos(0.2v)) \tag{12}$$

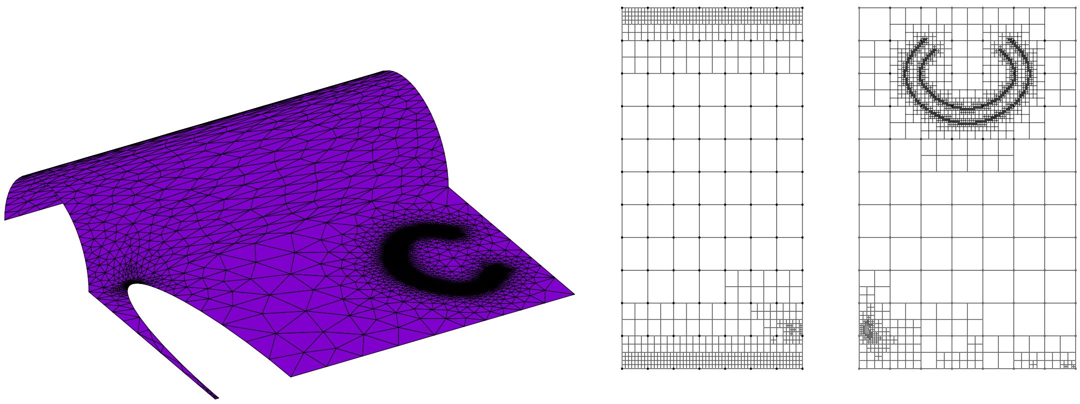
where $u, v \in [-6, 6]$.



(a) adapted control space (b) generated mesh

Figure 10: Analytical surface

Figures 11 and 12 present triangular meshes for domain consisting of two adjacent surface patches and control space structures for each of the patch with additional user-defined metric map along the inner contour in shape of letter C. Mesh in Fig. 11 was created using the procedure of control space adaptation, the other one without it which results in areas of mesh with unsatisfactory transitions of elements (shown in Fig.13 in more detail).

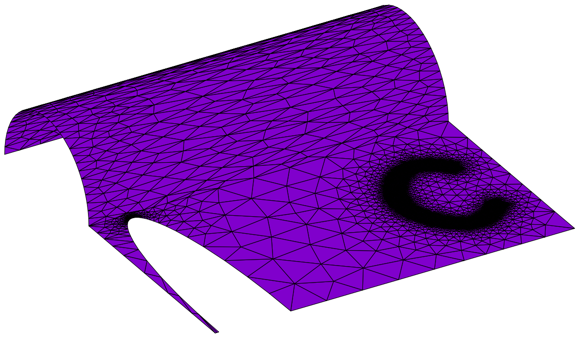


(a) triangular mesh (b) control space

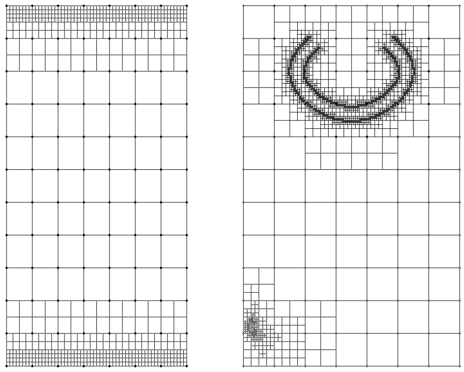
Figure 11: Mesh of two-patch surface with adaptation of control space

6 CONCLUSIONS

The procedure of automated creation of the control space for generation of surface meshes on multi-patch domains was presented. This procedure together with described implementation of adaptive control space structure allows to take advantage of metric description obtained from different sources which can be than efficiently combined and

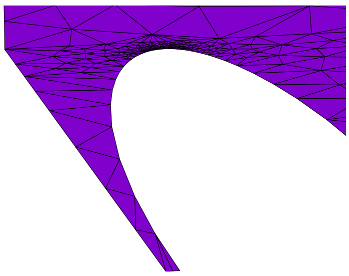


(a) triangular mesh

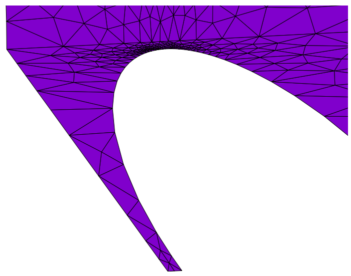


(b) control space

Figure 12: Mesh of two-patch surface without adaptation of control space



(a) without control space adaptation



(b) with control space adaptation

Figure 13: Closeup of the generated meshes

used during the meshing process.

In fully automated mesh generator all phases of the meshing process (including creation of sizing control space) shouldn't require any interaction from the user. At the same time the user should have the possibility of influencing the meshing process in a way he deems necessary depending on the purpose of the created meshes. The described method fulfills these requirements by combining automatically gathered geometrical sizing information with metric prescribed by user (in a discrete or analytical way) or gathered from the adaptation procedure.

The future work will concentrate on extending this iterative procedure for three-dimensional mesh generation.

Acknowledgments.

The partial support of the Polish Ministry of Scientific Research and Information Technology (MNII) Grant No. 3T11F00828 is gratefully acknowledged.

REFERENCES

[1] PJ. Frey, P-L. George, *Mesh generation. Application to finite elements. Hermès*, Paris, 2000.

[2] J. Zhu, T. Blacker, R. Smith, Background overlay grid size functions. *In Proc.* 11^{th} *International Meshing Roundtable*, Sandia National Laboratories, 65-74, 2002.

[3] K-F. Tchon, M. Khachan, F. Guibault, and R. Camarero, Three-dimensional anisotropic geometric metrics based on local domain curvature and thickness. *Computer-Aided Design*, **37**, 173–187, 2005.

[4] T. Jurczyk and B. Głut, Metric 3D Surface Mesh Generation Using Delaunay Criteria. *In Proc. 2006 Int. Conf. on Computational Science, Lecture Notes in Computer Science*, **3992**, 302–309, Reading, UK, 2006. Springer.

[5] T. Jurczyk and B. Głut, Generation of Good Quality Quadrilateral Elements for Anisotropic Surface Meshes. *In Proc. of the ADMOS2003 Conference on Adaptive Modeling and Simulation*, Göteborg, Sweden.

[6] T. Jurczyk and B. Głut, Metric 3D Surface Mesh Generation Using Coordinate Transformation Method. *In Proc. CMS'05 Conference on Computer Methods and Systems*, **1**, 395–405, 2005.

[7] J. Zhu, A new type of size function respecting premeshed entities. *In Proc.* 12^{th} *International Meshing Roundtable*, Sandia National Laboratories, 403–413, 2003.

[8] B. Głut and T. Jurczyk, Definition and interpolation of discrete metric for mesh generation on 3d surfaces. *Computer Science, Annual of University of Science and Technology*, in press, 2005.