A MESHFREE ISOSURFACE COMPUTATION METHOD FOR BOUNDARY ELEMENT METHODS

ANDRÉ BUCHAU & WOLFGANG M. RUCKER
Universität Stuttgart, Institut für Theorie der Elektrotechnik, Germany.

ABSTRACT

Isosurfaces are an appropriate approach to visualize scalar fields or the absolute value of vector fields in three dimensions. The nodes of the corresponding isosurface mesh are determined using an efficient and accurate isovalue search method. Then, these nodes are typically connected by triangular elements, which are obtained with the help of an adapted advancing front algorithm. An important prerequisite of an isovalue search method is that volume data of the examined field is available in total space. That means, the field values are precomputed in the nodes of an auxiliary post-processing volume mesh or a novel meshfree method is developed that enables both efficient computations of field values in arbitrary points and fast determination of domains with a defined range of field values. If the first approach is applied, a classical isovalue search method is to use an octree scheme to find relevant volume elements, which are intersected by the isosurface. Finally, the surface elements of the isosurface are constructed based on the intersection points of the isosurface with the volume elements. In that case, the accuracy and the computational costs are mainly influenced by the density of the post-processing volume mesh. In contrast, an innovative coupling of established isovalue search methods, fast boundary element method (BEM) techniques, and advancing front meshing algorithms is here presented to compute isosurfaces with high accuracy only using the original BEM model. This novel meshfree method enables very accurate isovalue search methods along with nearly arbitrarily adjustable resolution of the computed isosurface. Furthermore, refinements of the isosurface are also possible, for instance in dependency of the current viewing position. The main idea to realize this meshfree method is to directly combine an octree-based isovalue search method with the octree-based fast multipole method (FMM).

Keywords: applied boundary element methods, fast multipole methods, isosurface computations, meshfree post-processing, visualization.

1 INTRODUCTION

The computation of isosurfaces is a very attractive method to detect and visualize domains with a given range of scalar field values or the absolute value of vector field values in three dimensions. Furthermore, the characteristic properties of an examined field are sufficiently described by only a few isosurfaces in many practical applications. However, an efficient isovalue search, which is the fundamental step of isosurface computations, is still a challenging task, especially in the context of boundary element methods (BEMs). On the one hand, field values must be available in total space. A classical approach to provide volume data in the case of BEM is to compute the examined field values in the nodes of an additional auxiliary post-processing mesh. On the other hand, fast search methods are necessary to find isovalues efficiently. An established isovalue search method is an octree-based scheme Wilhelms and Gelder [1]. A pointer-less organization of the octree enables very efficient storage of the tree along with fast navigation in the tree in the case of dense octrees Wilhelms and Gelder [1]. Further enhancements improve the accuracy in the case of complex-shaped isosurfaces in combination with unstructured grids for field values Anderson *et al.* [2], Schreiner *et al.* [3], and Kazhdan *et al.* [4]. It is also possible to not only compute correct isosurfaces but to obtain surface elements, which can be used for further field computations Dey and Levine [5]. Modern implementations of isovalue search methods run massively

ISSN: 2046-0546 (paper format), ISSN: 2046-0554 (online), http://www.witpress.com/journals
DOI: 10.2495/CMEM-V5-N5-647-658

parallel on graphics processing units (GPU) Martin *et al.* [6]. However, the total accuracy and efficiency of all these approaches depend significantly on the density of the post-processing volume mesh for the field values. There, a compromise between accuracy and computational costs must be found to obtain correct isosurfaces.

The computation of field values in a large number of evaluation points is an expensive task, if classical BEM is used. Fortunately, compression techniques like the fast multipole method (FMM) Greengard and Rokhlin [7] reduce the computational costs significantly both for the solution of the linear system of equations and for the post-processing Buchau *et al.* [8]. Furthermore, it is possible to compute very efficiently field values for visualization objects, for instance streamlines, without an auxiliary mesh by a bidirectional coupling of BEM with visualization approaches Buchau and Rucker [9] and Buchau and Rucker [10].

Here, an innovative coupling of a meshfree BEM post-processing with an efficient isosurface computation is presented. The basis is an adaptive octree scheme, which is used both for the isovalue search and for a FMM accelerated BEM computation of field values. A direct evaluation of the series expansions of the FMM enables very accurate field value computations in relatively large domains using only a small number of coefficients. Hence, intersection points of isosurfaces with octree cubes can be determined more precisely and expensive computations of field values in the nodes of an auxiliary dense volume mesh are avoided. Furthermore, the presented approach enables an accurate refinement of the isosurface mesh with relatively low computational costs by computing additional intersection points inside the cubes on a set of lines parallel to the cubes edges. Finally, an adapted advancing front meshing algorithm is applied to construct triangular elements of the isosurface for an impressive visualization in a virtual reality environment.

2 NUMERICAL FORMULATION

2.1 Fast boundary element method

Here, three-dimensional problems, which are based on a solution of Laplace equations, are considered. An indirect BEM formulation is applied to obtain surface source densities on domain boundaries of piecewise homogeneous, linear, isotropic materials. In the case of electrostatic field problems, the scalar electric potential $u(\boldsymbol{r})$ at an arbitrary point $\boldsymbol{r}$ is directly obtained from the surface charge density σ on the surface A of conductors or dielectrics by an evaluation of

$$u(\boldsymbol{r}) = \frac{1}{\epsilon_0} \int_A \sigma(\boldsymbol{r}') G(\boldsymbol{r}, \boldsymbol{r}') \mathrm{d}A'. \tag{1}$$

G is Green's function of Laplace equation

$$G(\boldsymbol{r}, \boldsymbol{r}') = \frac{1}{4\pi} \frac{1}{|\boldsymbol{r} - \boldsymbol{r}'|} \tag{2}$$

and ϵ_0 is free space permittivity. A system of linear equations is obtained by an application of isoparametric second order quadrilateral boundary elements and the Galerkin method Buchau *et al.* [11]. The dense system matrix is compressed efficiently using the FMM Greengard and Rokhlin [7].

Here, isovalues of $u(\boldsymbol{r})$ are searched for a visualization of $u(\boldsymbol{r})$ with isosurfaces. The values of $u(\boldsymbol{r})$ can be directly obtained from (1). The evaluation of (1) is significantly accelerated, if the FMM is applied for the post-processing, too Buchau *et al.* [8]. Then, an octree scheme is used to group all boundary elements and evaluation points. The integral in eqn (1) is only evaluated for a small number of boundary elements in the near-field of an evaluation point $\boldsymbol{r}_{ep}$. However, singular and nearly singular integrals must be computed efficiently with satisfactorily high accuracy. The far-field of $\boldsymbol{r}_{ep}$ is taken into account by an evaluation of the local expansion in the octree cube of the evaluation point C_{ep}

$$u_f\left(\boldsymbol{r}_{ep}\right) = \frac{1}{4\pi}\sum_{n=0}^{L}\sum_{m=-n}^{n} L_n^m r^n Y_n^m\left(\theta,\varphi\right). \tag{3}$$

L_n^m are the so-called local coefficients in C_{ep}, L is the order of series expansions, and r, θ, φ are spherical coordinates of $\boldsymbol{r}_{ep}$ with origin in the center of C_{ep}. Y_n^m are spherical harmonics. Equation (3) can be regarded as a Taylor series expansion of $u_f\left(\boldsymbol{r}\right)$ in spherical coordinates, too. In practice, a good accuracy is obtained for $L = 9$ to $L = 12$ along with acceptable computational costs. That means, $u_f\left(\boldsymbol{r}\right)$ is approximated in C_{ep} with higher order polynomials and only a few coefficients. An approximation using linear volume elements would require a relatively large number of volume elements to achieve a comparable accuracy of $u_f\left(\boldsymbol{r}\right)$.

An octree scheme is the basis for a correct application of an FMM-accelerated BEM in practical applications. An octree is relatively easy to create and checks whether a point or an element is inside a specified cube are simple to perform by comparisons of Cartesian coordinates. FMM series expansions like eqn (3) are only applicable, if sources and evaluation points are definitely separated based on spherical coordinates. Hence, real dimensions of elements are taken into account by circumscribing spheres that result in an extra cube layer between corresponding cubes in practical implementations.

2.2 Meshfree isovalue search method

An isovalue search method is necessary to compute the nodes of the isosurface mesh. It requires to provide values of the examined field; here, the potential $u(\boldsymbol{r})$ in total space or in practice in a large but limited spatial range. An important property of BEM is that due to eqn (2) infinite space is taken exactly into account. The following method considers both aspects.

The total space is subdivided into two regions. One region, the inner domain Ω_i (red and green region in Fig. 1), contains the boundary elements in the red region in Fig. 1 and requires careful evaluations of compressed BEM integrals. The other region, the outer domain Ω_o (blue region in Fig. 1), is the surrounding free space, where the distance of all evaluation points to the boundary elements is large enough to only apply series expansions for the compressed evaluation of eqn (1). The cubes in the green and blue regions without boundary elements are necessary to compute and visualize isosurfaces not only between the boundary elements but also in the surrounding air domain. The number of cube layers in Ω_o is arbitrarily adjustable according to user's demands.

The first step of the adapted isovalue search method is to create two dense octrees, one for Ω_i and one for Ω_o. The root cube C_{ri} of the inner octree T_i is constructed from the bounding box B_e of all boundary elements. The centre of C_{ri} corresponds with the center of B_e and its edge length is the largest edge length of B_e. The inner octree is then extended by further cubes

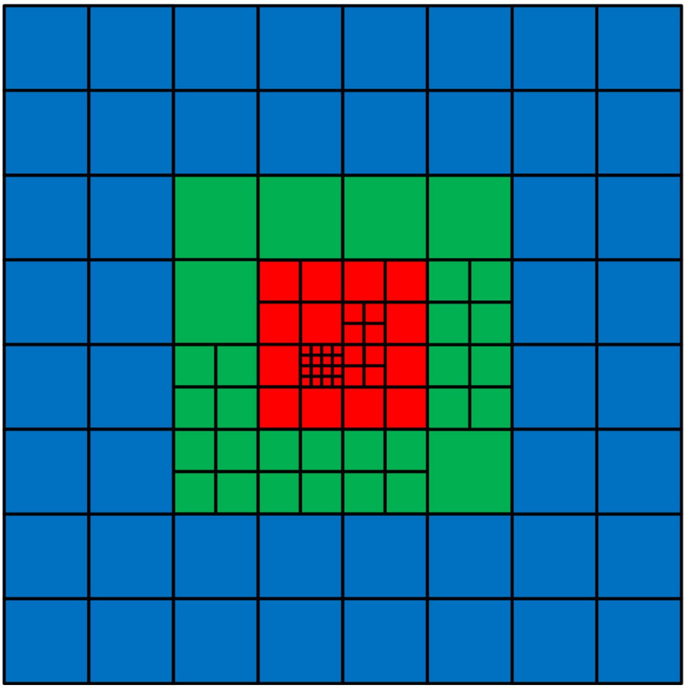

Figure 1: Octree scheme for geometrical subdivision of total space.

(green cubes in Fig. 1). Finally, outside of C_{ri} all prerequisites for an application of the so-called multipole expansion

$$u_f\left(\boldsymbol{r}_{ep}\right)=\frac{1}{4\pi}\sum_{n=0}^{L}\sum_{m=-n}^{n}M_n^m\frac{Y_n^m\left(\theta,\varphi\right)}{r^{n+1}} \tag{4}$$

are fulfiled. The origin of the spherical coordinates is in the centre of C_{ri}. The multipole coefficients M_n^m contain information of sources on all boundary elements, for instance the surface charge densities in electrostatics.

The cube C_{ri} is subdivided into eight child cubes according to the octree creation rules of the FMM. The significant difference to the classical FMM is that here empty cubes are created and stored, too. Empty cubes are only subdivided if boundary elements are assigned to their neighbour cubes and if these cubes are at a finer octree level than the empty cube.

The outer octree T_o (blue cubes in Fig. 1) is constructed in dependency of T_i. The centre of the root cube C_{ro} of T_o coincides with the center of C_{ri}. The edge length of C_{ro} is

$$l_{ro}=nl_{ri}, \tag{5}$$

where l_{ri} is the edge length of C_{ri} and n is a positive integer. The outer octree T_o is subdivided into empty child cubes until the level with edge length of the child cubes of C_{ri} is reached. The cubes, which would coincide with cubes of T_i are omitted, that means each $\boldsymbol{r}_{ep}$ is clearly assigned either to T_i or to T_o. Then, all cubes of T_o surround T_i and the field in all cubes of T_o can be computed using only one multipole expansion eqn (4). Of course, if very large surrounding air domains are considered, the order L in eqn (4) can be reduced for far distant cubes to reduce computational costs without significant loss in accuracy.

After the creation of the two octrees, the multipole expansions eqn (4) and local expansions eqn (3) of the FMM are computed in all cubes of T_i analogous to the classical FMM algorithm.

For an efficient octree based isovalue search, the value range of all cubes is determined first. Then, an efficient filtering of relevant cubes using the hierarchical tree structure is possible. Since extrema of $u(\boldsymbol{r})$ are lying on boundary elements, the values of $u(\boldsymbol{r})$ in all nodes of the boundary elements are determined first. Then the total range of $u(\boldsymbol{r})$ is known. The range of an octree cube is obtained from the values of $u(\boldsymbol{r})$ on boundary elements inside the cube and from the values of $u(\boldsymbol{r})$ in the corners of the considered cube. Note that it is necessary to take boundary elements into account that are assigned to the cube and boundary elements of neighbour cubes that extend into the cube.

If no boundary elements are lying in a cube or in its neighbour cubes, the local expansion in eqn (3) suffices to compute $u(\boldsymbol{r})$. A significant difference to well-known isovalue search methods is that $u(\boldsymbol{r})$ is not linearly interpolated between the corners of the cube but a higher order Taylor series expansion in eqn (3) is used. Hence, an optimization method, for instance a simple bisectional line search or a gradient-based search method, is applied to determine the intersection points of the isosurface with the octree cube edges. An advantage of the presented approach is that the intersection points are determined with high accuracy. Furthermore, additional intersection points inside the cube can be computed with a refinement of the cube and an isosurface with nearly arbitrarily fine adjustable resolution is obtained using further search lines on cube faces and inside the cube.

If boundary elements are lying in the considered cube or in its neighbour cube, the computation of intersection points is more expensive, because additional to the local expansion in eqn (3) nearly singular integrals in eqn (1) must be evaluated. In that case, a refinement of the isosurface is omitted. In any case, the resulting octree cubes are relatively small near boundary elements and a fine resolution of the isosurface is obtained automatically.

Two typical situations for refinement of the isosurface are depicted in Fig. 2 and in Fig. 3. First, the intersection points of the isosurface with the cube edges are determined. Then, the cube faces are refined using search lines at the faces. These points are important to obtain closed isosurface meshes. Further search lines inside the cube are used to obtain points of the isosurface between the cube edges and faces. These points are relevant for high-quality visualization of curved isosurfaces.

In Fig. 2, the red points are the intersection points at the cube edges. The green lines are additional search lines at cube faces for the blue intersection points. The intersection points of additional search lines inside the cube are depicted in Fig. 3.

Several strategies for positioning the additional search lines exist. One possibility is to place equidistant lines parallel to the cube edges. Radial search lines are an option for curved isosurfaces. In practice, an efficient and especially numerical robust method is to place all search lines parallel to cube edges but to use a mixture of directions. Then, almost regular distances of intersection points on curved isosurfaces are obtained along with low computational costs.

2.3 Isosurface meshing algorithm

The last step of isosurface computation is to create a meshed isosurface from the computed points obtained by the isovalue search method of the previous section. Linear triangular

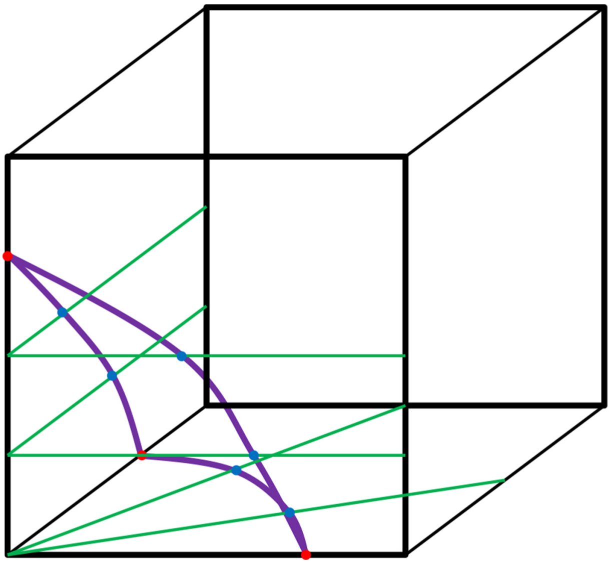

Figure 2: Computation of intersection points at cube faces and cube edges.

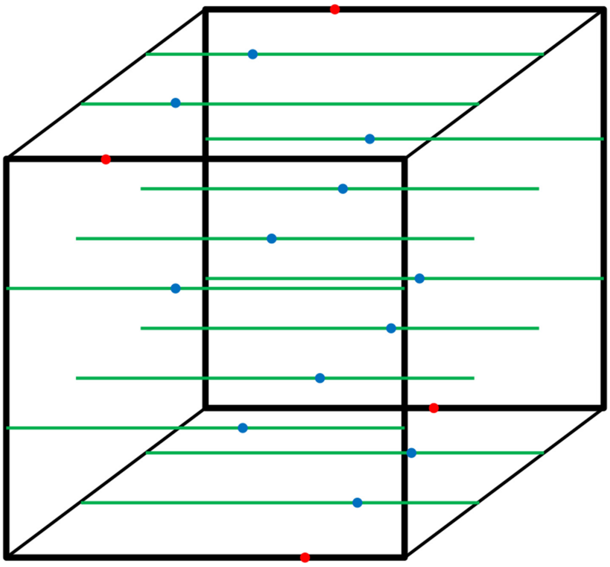

Figure 3: Computation of intersection points in a cube.

elements are typically used in visualization for performance and rendering reasons. Here, the task is to create a triangular mesh from already computed nodes. In contrast to classical meshing algorithms, where the position of the nodes is determined during the meshing procedure, the position of the nodes of the isosurface mesh is fixed and they have to be connected in a suitable way.

In principle, the approaches for watertight surface reconstructions from arbitrary point clouds could also be used here, Kobbelt and Botsch [12] and Hornung and Botsch [13]. However, they should be adapted to avoid deletion of some already computed nodes. Hence, an established surface mesh generation method, the advancing front technique Löhner [14], is more suited. It is applied with some adaptions to each cube C_{is}, which is intersected by the isosurface, and the meshes of the isosurface segments A_{is} are constructed separately for each cube C_{is}.

First, the edges E_{is} of A_{is} are defined using the intersection points at the edges of C_{is} and the additional intersection points at the faces of C_{is}. For the following algorithm, it is important that the intersection points along each E_{is} are sorted.

The first triangle is constructed from the first point of the first edge P_1, the second point on that edge P_2, and the next point P_3 on the second edge, which ends or starts at P_1 (Fig. 4). The order of these three nodes of the triangle defines the direction of the normal vector of the triangle. The direction of the normal vector is important during the construction of adjacent triangles. Hence, care has to be taken to ensure that the normal vectors of all elements of A_{is} point in the same direction.

Starting from the first triangle, element edges and triangles along the edges E_{is} are constructed (Fig. 4). Therefore, two sequent points of E_{is} are connected by an element edge $\boldsymbol{e}_e$. Then the third node of the triangle is searched according to the following triangle construction algorithm.

The next isosurface segment edge is the edge adjacent to the previous edge. The direction of the edge is adjusted in such a way that the positive direction of the right-hand rule is maintained to ensure correct orientation of normal vectors.

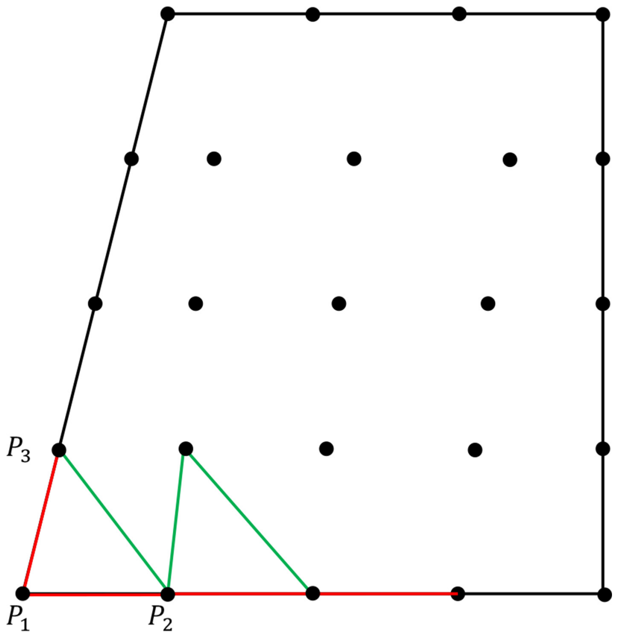

Figure 4: Construction of triangles starting at edges.

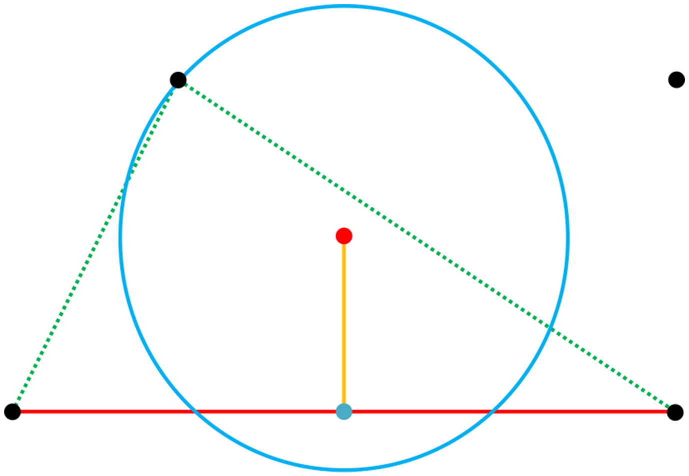

Figure 5: Construction of a new triangle from an edge.

The same triangle construction algorithm is applied to all free edges (green edges in Fig. 4) after the construction of triangles along the edges E_{is}. Finally, a completely meshed isosurface segment is obtained.

A new triangle is constructed starting from an element edge (red line in Fig. 5). Its normal vector is estimated using the normal vector of neighbour triangles. If the element edge is at the edge of the isosurface segment, the mean value of the normal vectors of all triangles, which have a common corner with the considered edge, is computed. The normal vector of the triangle adjacent to the element edge is used otherwise. The search point for the third node of the triangle (red point in Fig. 5) is constructed using the midpoint of the element edge (blue point in Fig. 5) and the weighted cross product of edge direction and normal vector. Then, the distances of all nodes to the red search point are computed and the nodes are sorted by this distance in ascending order. The closest node (node intersected by the blue circle in Fig. 5) is checked, whether it could be used as the third node of the triangle. The found node must be a free node or it must belong to an element with a free edge towards the red edge. The third node is finally connected using two edges (dotted green lines in Fig. 5) with the red edge. If these new edges are already part of existing triangles, these edges are assigned to the new triangle, too. Otherwise, new element edges are created.

During the construction phase, it is necessary to store not only the triangles including their nodes but also all element edges due to performance optimization. For each edge, a flag whether the edge is a free edge is recorded, too. To optimize the element edge search, a list of all edges, which end at a node, is maintained.

The triangles of all isosurface segments are merged in a common element container. Therefore, a renumbering of element nodes is required. Finally, one element set is transferred to the visualization software and rendered.

3 NUMERICAL EXAMPLES

The presented isosurface computation method was implemented in our in-house post-processing BEM software using the programming language C# including the .NET library. There, the focus was on testing the isovalue search method with respect to required computer resources and accuracy as well as the meshing algorithm to create high-quality visualizations of the computed isosurface. The aim was to evaluate the potential of meshfree isosurface computations. Performance optimization of the code have been omitted due to desired flexibility of the code.

The computation of isosurfaces of the electric potential of a plate capacitor was chosen as a numerical example. Although the geometrical configuration is relatively simple, it is very well suited for testing isovalue and isosurface computations. The isosurface in the symmetry plane is completely flat and isosurfaces near the plates of the capacitor are highly curved. Furthermore, the surrounding air domain is relevant for an accurate post-processing, too.

The two plates of the capacitor were discretized with in total 800 second order quadrilateral elements. The resulting linear system of equations with 2,562 degrees of freedom was solved using four parallel threads on an Intel Xeon CPU E3-1275v5 with 3.6 GHz and 64 GB main memory at 2,133 MHz in 9 s. GMRES with a Jacobi preconditioner was used as iterative solver and convergence with an error smaller than 10^{-6} was achieved after 56 iteration steps. The matrix of the linear system of equations was compressed using the FMM with $L = 12$.

The potential in 1,681 points of a plane intersecting the capacitor was computed using a single thread in 5 s (Fig. 6). The result was visualized with the visualization tool COVISE Lang and Wössner [15], which was originally developed at the high performance computing centre (HLRS) of University of Stuttgart and which is now an open source software. The communication between COVISE and our in-house post-processing code is established based on OASIS Open Data Protocol (OData) [16] to enable a bidirectional coupling between the visualization tool and the code for numerical field computations.

An inner octree with 3,209 cubes and an outer octree with 65 cubes have been constructed to compute isovalues. The outer octree adds one additional layer of cubes to the inner octree to take field values outside the bounding box of the capacitor into account, too. The data

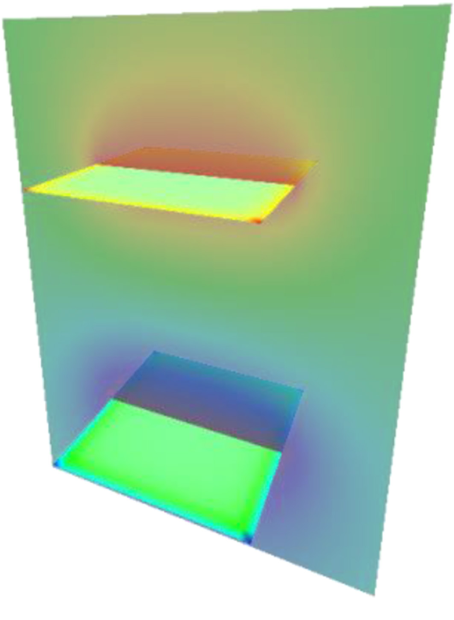

Figure 6: Surface charge density on the capacitor plates and potential in a plane.

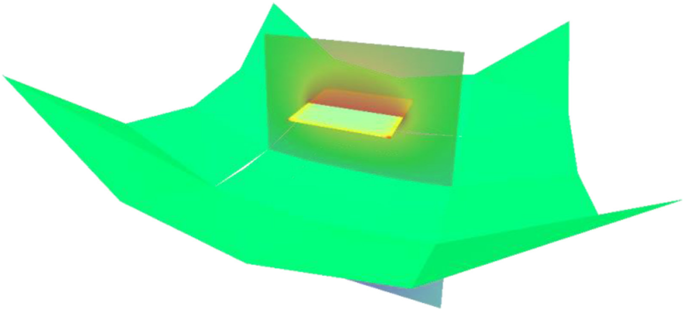

Figure 7: Isosurface without refinement.

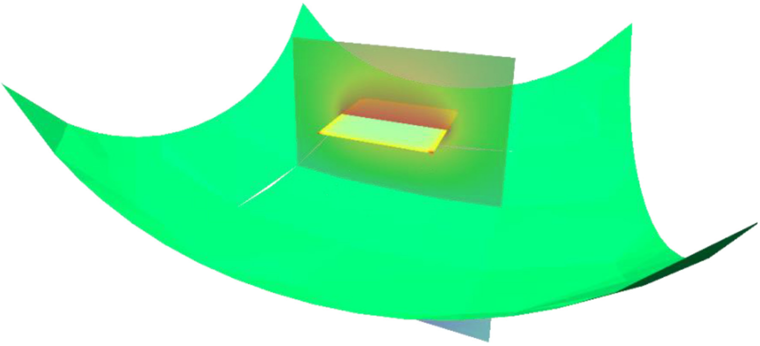

Figure 8: Refined isosurface.

range of each cube is determined first. Then, 28 relevant cubes of the inner octree and 16 relevant cubes of the outer octree have been selected for isosurface computation. That means, detailed computations of the potential have been performed in 1.3% of the cubes only. This corresponds to a significant reduction of computational costs in comparison to a volume meshed based post-processing.

A line search method using bisection is applied to find the isovalue along a search line. On average, 20 iterations are needed to find the point on the line with the defined isovalue with a relatively small error.

If no refinement is applied, the octree structure is visible in the computed isosurface (Fig. 7). The coarse isosurface consists of 168 nodes and 80 linear triangular elements. It is obvious that the visualized isosurface is not smooth enough. In total, 29,984 computations of the electric potential in the corners of the cubes, in the nodes of the boundary elements, and in points on the search lines were necessary. The CPU time on a single core was about 10 minutes. An example of a refined isosurface is depicted in Fig. 8.

4 CONCLUSION

An innovative approach for a meshfree computation of isosurfaces has been presented. A direct coupling of octree-based isovalue search methods, FMM accelerated BEMs, and advancing front meshing algorithms enable fast computations of isosurfaces with nearly arbitrarily adjustable resolution. The numerical results show impressively that the presented approach reduces both the number of expensive BEM computations and storage requirements significantly. In total, a flexible and powerful visualization of scalar fields and values in a virtual reality environment is possible along with relatively low computational costs, which is very important in practical applications of BEMs.

REFERENCES

[1] Wilhelms, J. & van Gelder, A., Octrees for faster isosurface generation. *ACM Transactions on Graphics*, **11**(3), pp. 201–227, 1992.
http://dx.doi.org/10.1145/130881.130882

[2] Anderson, J.C., Bennett, J.C. & Joy, K.I., Marching diamonds for unstructured meshes. *IEEE Visualization*, pp. 423–429, 2005.
http://dx.doi.org/10.1109/visual.2005.1532825

[3] Schreiner, J., Scheidegger, C.E. & Silva, C.T., High-quality extraction of isosurfaces from regular and irregular grids. *IEEE Transactions on Visualization and Computer Graphics*, **12**(5), pp. 1205–1212, 2006.
http://dx.doi.org/10.1109/TVCG.2006.149

[4] Kazhdan, M., Klein, A., Dalal, K. & Hoppe, H., Unconstrained isosurface extraction on arbitrary octrees. *Proceedings of Eurographics Symposium on Geometry Processing*, 2007.

[5] Dey, T.K. & Levine, J.A., Delaunay meshing of isosurfaces. *The Visual Computer*, **24**(6), pp. 411–422, 2008.
http://dx.doi.org/10.1007/s00371-008-0224-1

[6] Martin, S., Shen, H.W. & McCormick, P., Load-Balanced isosurfacing on multi-GPU clusters. *Proceedings of the 10th Eurographics Conference on Parallel Graphics and Visualization*, pp. 91–100, 2010.

[7] Greengard, L. & Rokhlin, V., A new version of the fast multipole method for the laplace equation in three dimensions. *Acta Numerica*, pp. 229–269, 1997.
http://dx.doi.org/10.1017/S0962492900002725

[8] Buchau, A., Rieger, W. & Rucker, W.M., Fast field computations with the fast multipole method. *COMPEL*, **20**(2), pp. 547–561, 2001.
http://dx.doi.org/10.1108/03321640110383861

[9] Buchau, A. & Rucker, M., Meshfree computation of field lines across multiple domains using fast boundary element methods. *IEEE Transactions on Magnetics*, **51**(3), 2015.
http://dx.doi.org/10.1109/TMAG.2014.2359520

[10] Buchau, A. & Rucker, W.M., Feasibility of a meshfree post-processing for boundary element methods. *Boundary Elements and Other Mesh Reduction Methods,* XXXVIII, pp. 327–338, 2015.
http://dx.doi.org/10.2495/BEM380261

[11] Buchau, A., Rieger, W. & Rucker, W.M., BEM computations using the fast multipole method in combination with higher order elements and the galerkin method. *IEEE Transactions on Magnetics*, **37**(5), pp. 3181–3185, 2001.
http://dx.doi.org/10.1109/20.952572

[12] Kobbelt, L.P. & Botsch, M., An interactive approach to point cloud triangulation. *EUROGRAPHICS*, **19**(3), 2000.

[13] Hornung, A. & Kobbelt, L., Robust reconstruction of watertight 3D models from non-uniformly sampled point clouds without normal information. *Eurographics Symposium on Geometry Processing*, 2006.

[14] Löhner, R., Progress in grid generation via the advancing front technique. *Engineering with Computers*, **12**, pp. 186–210, 1996.
http://dx.doi.org/10.1007/BF01198734

[15] Lang, U. & Wössner, U., Virtual and augmented reality developments for engineering applications. *Proceedings of ECCOMAS*, Jyväskylä, 2004.

[16] Open Data Protocol (OData), OASIS, Advancing open standards for the information society, available at: www.oasis-open.org