Efficient structural topology optimization system using the ground structure method

Vinícius Gama Tavares, Waldemar Celes
Instituto Tecgraf/PUC-Rio
Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio)
Rio de Janeiro, Brasil
{gama,celes}@tecgraf.puc-rio.br

***Abstract*—This work proposes a new method for the ground structure generation, which is necessary for the discrete structural topology optimization of bars based on the ground structure method (GSM). The main improvement of this method is to only require a domain contour description, in contrast to previous works that require a discretized model as input. Also, our method is valid for arbitrary two and three dimensional domains. In order to verify the method, we implement a solver for linear programming problems and compare the results with benchmark examples available in the literature.**

I. Introduction

Topology optimization aims to find the most efficient distribution of material in a specified domain without violating user-defined design constraints. The ground structure method (GSM) [1] is a discrete approach technique in the topology optimization field that can approximate an optimal Michell structure [2], [3]. The first step required by the method is the generation of a *ground structure* (set of nodes highly interconnected by bars); then, it can be applied a linear programming method to remove unnecessary bars (decreasing the final structure's volume), without changing node positions.

The initial *ground structure* (GS) is a critical part for the final topology optimization quality in the GSM. Ideally, the ground structure must be fully dense, that is, the domain should be completely covered by nodes and all the nodes should be connected with each other. But, in practice, this only adds computational cost without an effective gain in the final structure [4]. Because of this, GS are generated based on two principles: (i) the creation of a discrete but representative set of nodes covering the domain; and (ii) the definition of a connectivity level that reduces the number of bars in the ground structure. This connectivity level must be individually defined for each problem [4], [5].

Another practical problem inherent in GSM is the overlapping of bars. In order to prevent numerical instability in the optimization process, the initial ground structure should not contain overlapping bars and must favor smaller bars against larger ones [4], [6].

Zegard and Paulino developed the software called GRAND (*GRound structure ANalysis and Design*) [6] and GRAND3 [7], which implement the ground structure method. Their both implementations receive as input a polygon mesh, the restriction zones, and the design constraints (nodes with prescribed boundary conditions and loads). Zhang et al. [4] proposed two ground structure generation approaches: Macroelement and Macropatch. Those approaches have almost the same input as GRAND, except for not explicitly including restriction zones.

Nevertheless, for complex cases, generating a polygon mesh can be more expensive than generating the ground structure itself. The main contribution of this work[1] is the proposal of a new method for efficient ground structure generation that does not require a polygon mesh as input: the input for the proposed method is a simple domain contour description (lines in 2D and triangles in 3D) and the design constraints.

To test the generated ground structure for complex domains, this work also implements a method for the linear programming problem to solve the topology optimization. Then, the achieved topology optimization results, based on our generation method and our optimization implementation, are compared with benchmark examples available int the literature.

II. Ground Structure Generation

The proposed method to generate dense ground structure follows the step presented in the subsequent subsections.

A. Grid

The initial step of the proposed ground structure generation method is to create a regular grid covering the entire domain. The regular grid was chosen because it is simpler and faster to query node and bar locations, speeding up all the process. The grid cell size will influence the mesh quality; a more refined grid increases the precision to compute the distance field necessary in future steps, but decreases the computational efficiency.

The suggested cell size l is defined as $l = \bar{e}\tau$, where $\bar{e}$ is the average contour edge length and τ is a problem-dependent factor value.

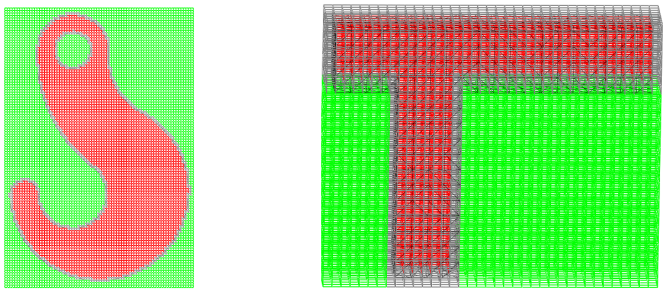
B. Cells Classification

After the grid creation, the grid cells are classified in three types: *boundary*, *inside* or *outside* the domain. We use a flood fill [8] technique for this classification. Initially, all the cells

[1]This work relates to a M.Sc. dissertation.

intersecting the domain contour are classified as *boundary*. It is necessary a distinct approach for each dimension type.

In a 2D grid, it is implemented the Amanatides and Woo [9] algorithm for grid traversal to discover the cells that collide with the domain contour. For the 3D, the Akenine-Möller algorithm [10] is used to determine all the cells that collide with the triangles of the domain contour. After the cell boundary classification, the other cells are classified using the flood fill algorithm. A classification result is illustrated in Fig. 1.



(a) 2D (b) 3D

Fig. 1. Example of cell classification. Green cells are classified as outside the domain, gray cells are on the boundary and the red cells are classified as inside the domain.

C. Node Generation

For the domain discretization, the input nodes (nodes on the domain contour) are used as seeds to create nodes inside the domain. For this, it is computed the domain distance field (distance from the contour), and its gradient is used to guide the propagation direction for the new nodes. The propagation step is based on information of the contour edges adjacent to the seed nodes.

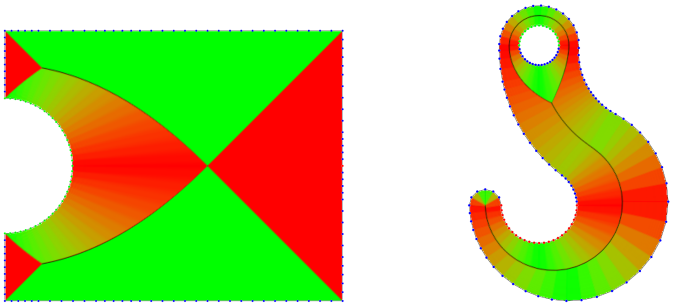
Distance Field

The generated distance field is a representation where, at each point within the domain, it is known the distance from that point to the closest point on the contour [11]. In addition to the distance value, it is possible to infer other properties, such as the direction to the border by the field gradient. This property is used in this work.

With the grid created previously, we compute a discrete distance field with signal. An important property of a signed distance field d is that $||\nabla d|| = 1$ at almost everywhere (the exceptions are points without a unique closest point on the contour) [11].

The gradient at a given point p is orthogonal to the isoline (or isosurface in 3D) passing through p [11]. This information is used in this work, since, with the gradient, it is possible to "walk" inside the domain following the contour shape, starting in the border and going to the medial axis direction (region where the gradient does not have a defined value since the point is equidistant to at least two points on the contour).

To illustrate, Fig. 2 shows some results of the gradient obtained from computed distance fields: the red color represents the x component and the green color, the y component. The black lines indicate the medial axis. Similar results are also obtained for 3D domains.



(a) Cantilever with circular support (b) Hook

Fig. 2. Visualization of the distance field gradient.

Domain Discretization

To discretize the domain in a way that the generated internal nodes are aligned to the domain contour, we use the distance field gradient as the direction of propagation. For each seed node (contour node), nodes are created inward the domain following the distance field gradient. The step of propagation is adjusted as necessary according to the curvature of the domain contour. For concave vertices, the step is decreased along the propagation; for convex vertices, it is increased.

We set this adaptive step in the follow manner. Let us consider a seed vertex $\mathbf{v}$ that will generate the internal nodes $\{\mathbf{v}^1, \mathbf{v}^2, \mathbf{v}^3, ...\}$. Consider the set of vertices $\mathbf{p}_i$ adjacent to $\mathbf{v}$. We first compute the smallest adjacent edge lenght, $l_{min} = \min_i(\|\mathbf{v} - \mathbf{p}_i\|)$. Then, we take one virtual step for $\mathbf{v}$ and all $\mathbf{p}_i$, along the gradient direction inward the domain, using l_{min} as the step, obtaining $\mathbf{v}'$ and $\mathbf{p}'_i$, respectively. The adaptive step factor, γ, is then given by:

$$\gamma = \frac{\sum_i(\|\mathbf{v}' - \mathbf{p}'_i\|)}{\sum_i(\|\mathbf{v} - \mathbf{p}_i\|)}$$

Fig. 3 illustrates this computation. In summary, γ captures the change on average adjacent edge lengths as if the entire contour was propagated inward the domain.

The first step of real propagation is then set as $s_0 = \gamma\, l_{min}$. Generated internal nodes and subsequent steps are given by:

$$\mathbf{v}^k = \mathbf{v}^{k-1} + s_{k-1}\, \nabla d_{\mathbf{v}^{k-1}}$$
$$s_k = \gamma\, s_{k-1}$$

where $\mathbf{v}^0 = \mathbf{v}$, and $\nabla d_{\mathbf{v}^{k-1}}$ represents the distance field gradient at $\mathbf{v}^{k-1}$.

The propagation of a seed node $\mathbf{v}$ is interrupted when one of the following conditions happens:

- $s_k \leq 0.2 s_0$, because too close points do not add quality to the optimized result.
- The new created node is on the medial axis, because the gradient is not defined.

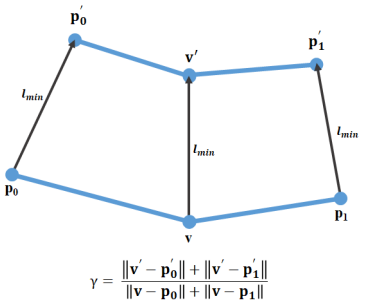
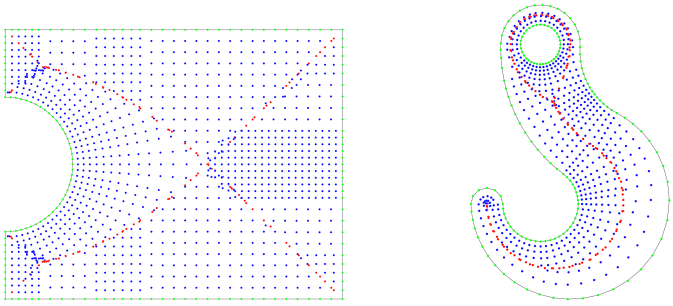


Fig. 3. Adaptive step computation for seed node propagation inward the domain.

- The distance field gradient turns to the opposite direction, because the propagation crossed the medial axis.

Fig. 4 shows the result obtained. Green nodes are the seeds; blue nodes are propagated node far from the medial axis regions; red nodes are propagated nodes in the medial axis region. Note that the generated nodes follow the contour shape, a condition necessary to obtain good optimization results. We can also note that some nodes are too close to each other, especially near the medial axis region, and it is necessary to reduce this concentration.



(a) Cantilever with circular support (b) Hook

Fig. 4. Initial domain discretization. The green nodes are the seed nodes, the blue nodes are those which are in the region where the gradient exists and the red nodes are in the region of the medial axis.

Nodes Removal

For a given generated node $\mathbf{v}^k$, we define its *order* as being the value of k. To reduce node concentration, a "zone of influence" is defined for each node. The radius of this zone is set to $r_{\mathbf{v}^k} = \beta\, s_{k-1}$, with $\beta < 1$. We then process the generated nodes in increasing order value; for each visited node $\mathbf{v}^k$, we identify all other nodes inside its zone of influence. From those, we remove all nodes in the medial axis region (red nodes in Fig. 4) and all nodes which order is greater than k. Fig. 5 illustrates this procedure.

When processing the nodes on the medial axis regions (*red nodes*), we employ a similar approach. However, no *blue node* is removed because a *red node*, even inside its zone of influence. Also, in the end, the vertex $\mathbf{v}^k$ is displaced by the influence of the removed nodes $\mathbf{p}_i^j$. The new vertex position is

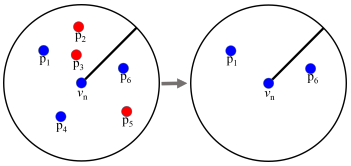
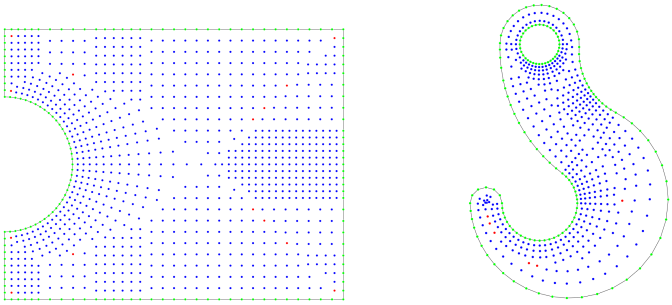


Fig. 5. Example of *red* and *blue* node removals: in this case, it is assumed that the $\mathbf{p}_1$ and $\mathbf{p}_6$ orders are less than the $\mathbf{v}_n$ order, so they are not removed.

given by a weight average position considering the removed nodes and the current node, using the inverse of the order as weight:

$$\mathbf{v}^k = \frac{\frac{\mathbf{v}^k}{k} + \sum_i \frac{\mathbf{p}_i^{j_i}}{j_i}}{\frac{1}{k} + \sum_i \frac{1}{j_i}}$$

Fig. 6 illustrates some results achieved after node removal for different domains. It is possible to see that a good discretization was generated, following the shape defined by the boundary. In addition, the nodes also fill well the entire domain.



(a) Cantilever with circular support (b) Hook

Fig. 6. Final domain discretization. The green nodes are the seed nodes, the blue nodes are those which are in the region where the gradient exists and the red nodes are in the region of the medial axis.

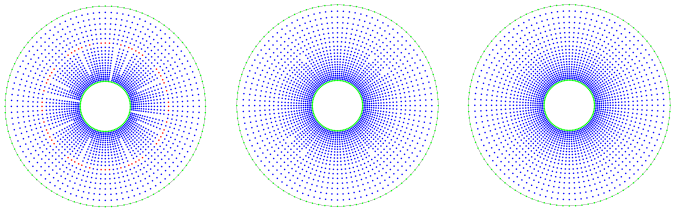
Parameter Discussion

During the node generation process, we use two parameters:

- Parameter τ: as a factor used to set cell grid size. By varying this value, the quality of the distance field is affected.
- Parameter β: as a factor used to set the zone of influence for node removals. By varying this value, the number of removed nodes is affected.

The quality of the domain discretization is directly linked to the grid cell size. The more refined the grid, the better the achieved quality. Fig. 7 shows the results obtained for τ with values 1.0, 0.5 and 0.25. The value of τ influences the quality of the distance field and, consequently, the accuracy for gradient computation. Tests performed with different models showed that a value of τ less than 0.2 does not change the final achieved result, only increasing the computational cost.

Regarding the value of $beta$, it is difficult to justify a single appropriate value for all models; tests have shown that the value $\beta = 0.8$ was satisfactory for all tested examples.



(a) $\tau = 1.0$ (b) $\tau = 0.5$ (c) $\tau = 0.25$

Fig. 7. Result of cell size variation in node generation.

D. Bar Generation

With the generated nodes, the next step is to connect them through bars to generate the ground structure. For this, we borrow the heuristic used for node removal and also define a zone of influence for each node. However, rather than removing the nodes within the zone of influence, bars are created between these nodes.

The radius of this zone is similarly set to $r_{\mathbf{v}^k} = \sigma\, s_{k-1}$, but now with $\sigma > 1$. Different from GRAND [6], [7], which uses a topological adjacency for the definition of mesh density, this work proposes a geometric concept for the density choice, based on the value of σ.

The candidate bars associated with the $\mathbf{v}^k$ node are defined as being all bars originated at $\mathbf{v}^k$ entirely contained within its zone of influence. Initially, all the candidate bars of all nodes are added to a set. Then, the set is sorted in increasing order of bar length, and each bar, from the shortest to the longest, is considered to be added to the ground structure. Once a bar is added, it will never be removed. A given bar is added to the GS according to the following conditions:

- A bar can only be added if it is entirely contained in the domain.
- A bar can only be added if it is not collinear, within a given tolerance, to other any other bar already in the solution.

The collinearity test is performed as in GRAND [6], [7]. A maximum θ_{tol} tolerance angle is defined. The angle of two bars sharing a node must not exceed this tolerance value. For example, in Fig. 8, the dashed bar will only be added if $cos(\beta_1) < cos(\theta_{tol})$ and $cos(\beta_2) < cos(\theta_{tol})$.

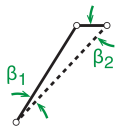
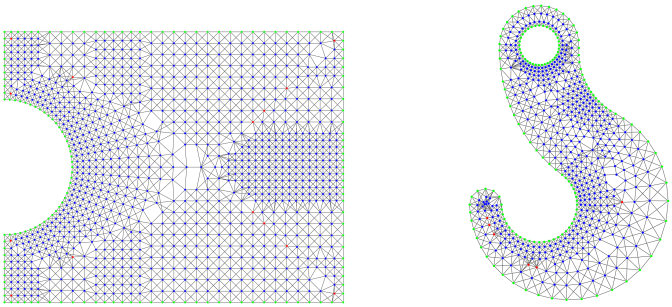


Fig. 8. Collinearity test (from [6]). If $cos(\beta_2) > cos(\theta_{tol})$ or $cos(\beta_1) > cos(\theta_{tol})$, then the dashed bar won't be added.

Fig. 9 shows one achieved ground structure using the connectivity radius factor set to $\sigma = 1.6$. Despite a small value, such as 1.6, not being adequate for the quality of the optimization, it is shown here as an example for ease of visualization of the generated ground structure.



(a) Cantilever with circular support (b) Hook

Fig. 9. Result of generating bars for the radius of connectivity 1.6.

III. Optimization

A. Formulation

The ground structure formulation used in this work is based on plastic analysis. The goal of this optimization is to minimize the truss-structure volume that satisfies the force equilibrium equations [6]. The optimization only modifies the cross-sectional area of each bar and, thus minimizing the final volume, without changing the node positions. The formulation is [3], [7], [12]:

$$\begin{aligned} \min_{\boldsymbol{a}} \quad & V = \boldsymbol{l}^T \boldsymbol{a} \\ \text{s.t.} \quad & \boldsymbol{B}^T \boldsymbol{n} = \boldsymbol{f} \\ & -\sigma_C a_i \leq n_i \leq \sigma_T a_i,\ i = 1, 2 \ldots N_b \end{aligned} \tag{1}$$

where:

- N_b is the number of bars in the mesh.
- V is the truss-structure volume.
- σ_C e σ_T are the limits of tension in compression and traction.
- a_i, l_i, σ_i, f_i e n_i are the cross-sectional area, length, stress, external force, and internal (axial) force of the i^{th} truss member.
- $\boldsymbol{B}^T$ is the nodal equilibrium matrix, built from the directional cosines of the members.

$\boldsymbol{B}^T$ is a matrix with size $N_{dof} \times N_b$, which:

- N_n is the number of nodes in the ground structure.
- N_{sup} is the number of fixed (supported) degrees of freedom.
- N_{dof} is the number of degrees of freedom, where: $N_{dof} = 2N_n - N_{sup}$, for 2D, or $N_{dof} = 3N_n - N_{sup}$, for 3D.

In order to transform the Equation 1 in a linear programming problem (i.e. the inequalities turns into equalities), it is added slack variables s_i^+ and s_i^- [3], [7], [12] and the final formulation becomes [5], [6], [13], [14]:

$$
\begin{aligned}
\min_{s_i^+, s_i^+} \quad & V^* = \frac{V}{\sigma_T} = \underset{1\times 2N_b}{\begin{bmatrix} \boldsymbol{l}^T & k\boldsymbol{l}^T \end{bmatrix}} \underset{2N_b\times 1}{\begin{bmatrix} \boldsymbol{s}^+ \\ \boldsymbol{s}^- \end{bmatrix}} \\
\text{s.t.} \quad & \underset{N_{dof}\times 2N_b}{\begin{bmatrix} \boldsymbol{B}^T & -\boldsymbol{B}^T \end{bmatrix}} \underset{2N_b\times 1}{\begin{bmatrix} \boldsymbol{s}^+ \\ \boldsymbol{s}^- \end{bmatrix}} = \underset{N_{dof}\times 1}{\boldsymbol{f}} \\
& a_i = \frac{s_i^+}{\sigma_T} + \frac{s_i^-}{\sigma_C}; n_i = s_i^+ - s_i^- \\
& s_i^+, s_i^- \geq 0
\end{aligned} \tag{2}
$$

B. Implementation

To solve the linear programming problem of Equation 2, it was implemented the Primal-Dual Interior Point Method (IPM) with the Predictor-Corrector algorithm [15]. It is used the Eigen [16] library to store the sparse matrix, the PARDISO [17] library to solve the linear system in the IPM, and the TopSim [18] framework to achieve large scale.

IV. Results

In this section, the results obtained using the proposed ground structure generation and the implemented optimization are presented. Initially, we do a verification considering well-known cases, and then we consider other more complex examples.

As in GRAND, in order to visualize the optimization final result, a *cutoff* value is defined for displaying the bars. Thus, only bars with cross-sectional area $\frac{a_i}{max(a)} > cutoff$ are displayed, setting the value of *cutoff* to 0.002 for all 2D models and to 0.005 for all 3D cases. Also, all the results were obtained setting $\tau = 0.2$ and $\beta = 0.8$, which are the parameters described in Section II.

A. Verification

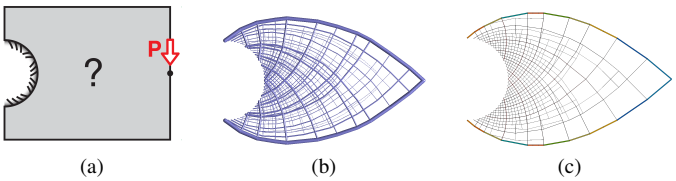
All verification examples are compared to GRAND, comparing the final structure volume and the obtained truss topology. In all cases, it is tried to achieve similar number of bars in the ground structure.

2D

Cantilever with circular support: This example approximates a Michell's analytical solution [2], [6]. Table I shows that both volume values converge to the same value from the analytical solution, which is 16.0944 [6]. Fig. 10 shows the obtained topology. Note that both methods delivered symmetric structures as desired. In all figures illustrating achieved optimized structures, we have opted for displaying our results using a color scale that indicates bar volumes, blue being large values and red small ones.

	Level/Radius Connectivity	Nodes	Bars	Volume
GRAND	7	1069	75900	16.2192
Proposed Method	7.5	1083	76928	16.1836

TABLE I
CANTILEVER WITH CIRCULAR SUPPORT COMPARISON

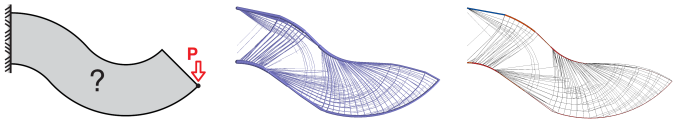


(a) (b) (c)

Fig. 10. Cantilever with circular support. (a) Domain definition (from [6]) (b) Solution from GRAND (c) Solution from Proposed Method.

	Level/Radius Connectivity	Nodes	Bars	Volume
GRAND	7	1045	84706	91.4121
Proposed Method	8.3	920	84488	91.4419

TABLE II
SERPENTINE COMPARISON



(a) (b) (c)

Fig. 11. Serpentine. (a) Domain definition (from [6]) (b) Solution from GRAND (c) Solution from Proposed Method.

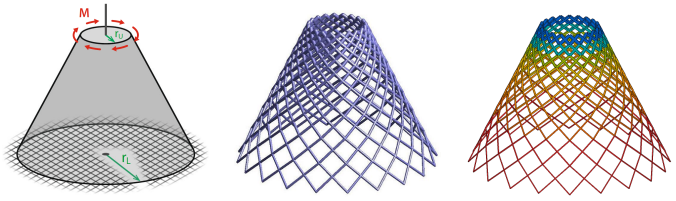
Serpentine: Fig. 11 shows very similar structure, with the same patterns, with the final volumes shown in Table II.

3D

Torsion cone: The analytical solution for the torsion cone volume is $v = 16.8076$ [7]. Table III shows that both solutions are converging to this optimal volume. Both methods resulted in the structure, are shown in Fig. 12.

	Level/Radius Connectivity	Nodes	Bars	Volume
GRAND	3	1010	115789	17.0310
Proposed Method	3.3	1346	117716	17.0310

TABLE III
TORSION CONE COMPARISON



(a) (b) (c)

Fig. 12. Torsion cone. (a) Domain definition (from [7]) (b) Solution from GRAND (c) Solution from Proposed Method.

Torsion cylinder: The optimal volume for this example is 36.6667. Table IV shows that both solutions are converging to the optimal volume and Fig. 13 presents identical topologies.

	Level/Radius Connectivity	Nodes	Bars	Volume
GRAND	3	1308	152795	37.2637
Proposed Method	4.1	1169	155586	37.2637

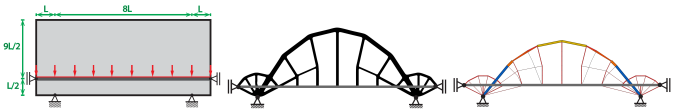
TABLE IV
TORSION CYLINDER COMPARISON

(a) (b) (c)

Fig. 13. Torsion cylinder. (a) Domain definition (from [7]) (b) Solution from GRAND (c) Solution from Proposed Method.

B. Complex Examples

Bridge: A more realistic engineering model is presented here, aiming to build an arch bridge. The result provided by the proposed model is very similar to the analytical solution, with the topologies being shown in Fig. 14.



(a) (b) (c)

Fig. 14. Bridge. (a) Domain definition (from [4]) (b) Analytical solution [4] (c) Solution from Proposed Method.

Tower: This example presents a conceptual tower, showing another field that this method is able to attend. Fig. 15 shows the final result for the tower, in which both GRAND and the proposed method achieved similar results.

(a) (b) (c)

Fig. 15. Tower. (a) Domain definition (from [7]) (b) Solution from GRAND (c) Solution from Proposed Method.

V. CONCLUSION

In this work, it was presented and discussed the pipeline of structural topology optimization of bars, from the ground structure generation to its optimization. We proposed a new method for the ground structure generation that does not depend on the existence of a discrete model as input, in contrast to previous works. From the domain contour, domain-aligned nodes are created. For this, it was proposed to use the distance field of the model, using the direction of the distance field gradient to guide the internal node creation. Once this is done, potential bars are ordered by their size and added to the solution when possible, favoring smaller bars. A collision check and collinearity test prevent the insertion of invalid bars.

To test the generated ground structures, we implemented a topological optimization method using the plastic formulation. All the examples showed coherent results, both in the final obtained structure volume and in the quality of the resulting topology, showing the effectiveness of the proposed method.

ACKNOWLEDGMENT

The authors are grateful to CNPq grant 132853/2015-9 and to Instituto Tecgraf/PUC-Rio support.

REFERENCES

[1] W. S. Dorn, "Automatic design of optimal structures," *Journal de mecanique*, vol. 3, pp. 25–52, 1964.
[2] A. Michell, "M. The limits of economy of material in frame structure [J]," *Philosophical Magazine*, vol. 8, no. 6, pp. 589–597, 1904.
[3] W. Hemp, *Optimum structures*, ser. Oxford engineering science series. Clarendon Press, 1973.
[4] X. Zhang, S. Maheshwari, A. S. Ramos Jr., and G. H. Paulino, "Macroelement and Macropatch Approaches to Structural Topology Optimization Using the Ground Structure Method," *Journal of Structural Engineering*, vol. 142, no. 11, 2016.
[5] T. Sokół, "A 99 line code for discretized Michell truss optimization written in Mathematica," *Structural and Multidisciplinary Optimization*, vol. 43, no. 2, pp. 181–190, 2011.
[6] T. Zegard and G. H. Paulino, "GRAND — Ground structure based topology optimization for arbitrary 2D domains using MATLAB," *Structural and Multidisciplinary Optimization*, vol. 50, no. 5, pp. 861–882, 2014.
[7] ——, "GRAND3 — Ground structure based topology optimization for arbitrary 3D domains using MATLAB," *Structural and Multidisciplinary Optimization*, vol. 52, no. 6, pp. 1161–1184, 2015.
[8] M. Levoy, "Area Flooding Algorithms," in *SIGGRAPH '81 Two-Dimensional Computer Animation course notes*. Dallas, Texas: ACM, 1981, pp. 6–12, Com correções feitas em 1982.
[9] J. Amanatides and A. Woo, "A Fast Voxel Traversal Algorithm for Ray Tracing," in *Eurographics '87*, 1987, pp. 3–10.
[10] T. Akenine-Möller, "Fast 3D Triangle-box Overlap Testing," in *ACM SIGGRAPH '05 Courses*. New York, NY, USA: ACM, 2005.
[11] M. W. Jones, J. A. Baerentzen, and M. Sramek, "3D distance fields: a survey of techniques and applications," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 4, pp. 581–599, July 2006.
[12] M. Ohsaki, *Optimization of Finite Dimensional Structures*. CRC Press, 2016.
[13] W. Achtziger, "On simultaneous optimization of truss geometry and topology," *Structural and Multidisciplinary Optimization*, vol. 33, no. 4, pp. 285–304, 2007.
[14] M. Gilbert and A. Tyas, "Layout optimization of large-scale pin-jointed frames," *Engineering Computations*, vol. 20, no. 8, pp. 1044–1064, 2003.
[15] S. Mehrotra, "On the Implementation of a Primal-Dual Interior Point Method," *SIAM Journal on Optimization*, vol. 2, no. 4, pp. 575–601, 1992.
[16] G. Guennebaud, B. Jacob *et al.*, "Eigen v3," http://eigen.tuxfamily.org, 2010.
[17] O. Schenk and K. Gärtner, "Solving unsymmetric sparse systems of linear equations with PARDISO," *Future Generation Computer Systems*, vol. 20, no. 3, pp. 475 – 487, 2004, selected numerical algorithms.
[18] L. S. Duarte, "TopSim: A plugin-based framework for large-scale numerical analysis," D.Sc. in Computer Science, Departamento de Informática, PUC-Rio, 2016.