An immersed boundary approach for the numerical analysis of objects represented by oriented point clouds

László Kudela*[1], Stefan Kollmannsberger[1], and Ernst Rank[1,2]

[1]Chair for Computation in Engineering, Technical University of Munich, Arcisstr. 21, 80333 München, Germany
[2]Institute for Advanced Study, Technische Universität München, Germany

Abstract

This contribution presents a method aiming at the numerical analysis of solids whose boundaries are represented by oriented point clouds. In contrast to standard finite elements that require a boundary-conforming discretization of the domain of interest, our approach works directly on the point cloud representation of the geometry. This is achieved by combining the inside-outside information that is inferred from the members of the point cloud with a high order immersed boundary technique. This allows for avoiding the challenging task of surface fitting and mesh generation, simplifying the image-based analysis pipeline drastically. We demonstrate by a numerical example how the proposed method can be applied in the context of linear elastostatic analysis of solids.

Keywords: image-based finite element analysis, point clouds, finite cell method

1 Introduction

A core challenge in the context of image-based finite element analysis revolves around the question of how to derive an analysis-suitable finite element model from the point cloud data that describes the shape of the domain of interest. Examples include the structural analysis of statues [1], historical structures [2, 3, 4] and the coupling of finite element computations to *in vitro* measurements of biological tissues [5]. In order to construct a finite-element mesh that resolves the boundaries of the geometry, the usual approach is to process the point cloud data through a multi-step pipeline that results in a mesh of boundary-conforming finite elements. Generally, the main steps of such cloud-to-analysis pipelines can be characterized as follows:

1. *Geometry recovery*
   A geometric model is derived from the point cloud information using geometric segmentation and surface fitting techniques.

2. *Mesh generation*
   Once the geometric representation of the object is recovered, the model is subdivided into a set of boundary-conforming finite elements.

3. *Finite Element Analysis*
   The mesh from the previous step together with the necessary material parameters and boundary conditions is handed over to a finite element solver.

*laszlo.kudela@tum.de, Corresponding Author

These steps are difficult to automate, as a solution which is tailored to a specific class of geometries is usually not directly applicable on other types of objects. Moreover, because the pipeline requires the interplay of many techniques from computational science and engineering, the analyst performing the above steps needs to be experienced with a wide variety of softwares and has to be aware of their respective pitfalls. For example, even if the geometry of the object is available through a CAD model, the preparation of an analysis-suitable finite element mesh may require a great amount of human interaction and can take up to 80% of the total analysis time [6].

In recent years, research efforts aiming at avoiding the difficult task of mesh generation brought forth many approaches, such as the Finite Cell Method (FCM) introduced in [7]. The FCM relies on the combination of approaches well-known in computational mechanics: immersed boundary (IB) methods [8] and high-order finite elements (p-FEM) [9]. While initially suffering from low accuracy and high computational costs, IB methods have seen a complete revival in recent years. New numerical technologies addressing issues related to discretization [10], stability [11, 12], boundary conditions [13] and numerical integration [14, 15, 16] allowed for the application of IB approaches in non-trivial fields. Examples include geometrically non-linear problems [17], plasticity [18], simulation of biomechanical structures [19], flow problems [20] and contact simulation [21].

Instead of generating a boundary-conforming discretization, FCM extends the physical domain of interest by a so-called fictitious domain, such that their union forms a simple bounding box that can be meshed easily. To keep the consistency with the original problem, the material parameters in the fictitious domain are penalized by a small factor α. The introduction of α shifts the analysis effort from mesh generation to numerical integration. The biggest advantage of the FCM lies in high convergence rates with almost no meshing costs.

In its simplest implementation, the only information that FCM needs from a geometric model is inside-outside state: given a point in space, does this point lie in the physical or the fictitious part of the domain? A wide variety of geometric representations is able to provide such *point membership tests* and have been shown to work well in combination with the FCM, ranging from simple shapes to models as complex as metal foams.

In this contribution, the Finite Cell Method is combined with geometries that are represented by oriented point clouds. The members of the point cloud and the vectors associated to them provide enough information for point membership tests, allowing for simulating objects directly on their cloud representation. This way, the tedious tasks of recovering a geometric model and generating a boundary conforming mesh can be avoided. This allows for significant simplifications in the cloud-to-analysis pipeline.

2 The Finite Cell Method Combined with Oriented Point Clouds

In the following, the essential ideas of the Finite Cell Method for steady linear elastic problems are discussed. For further details, see [7, 22].

2.1 A Brief Overview of FCM

As mentioned in the introduction, the FCM aims at circumventing the problem of mesh generation by extending the boundaries of the physical domain of interest Ω_{phy} by a fictitious part Ω_{fict}. The union of these two parts $\Omega_{\text{phy}} \cup \Omega_{\text{fict}}$ forms the embedding domain $\Omega_{\cup}$. This possesses a simple, box-like geometry that can be meshed easily. The concept is depicted in Figure 1.

The derivation of FCM is based on the principle of virtual work [23]:

$$\delta W(\mathbf{u}, \delta\mathbf{u}) = \int_{\Omega} \boldsymbol{\sigma} : (\nabla_{\text{sym}} \delta\mathbf{u})\, dV - \int_{\Omega_{\text{phy}}} \delta\mathbf{u} \cdot \mathbf{b} dV - \int_{\Gamma_{\text{N}}} \delta\mathbf{u} \cdot \mathbf{t} dA = 0, \tag{1}$$

Where $\boldsymbol{\sigma}, \boldsymbol{b}, \boldsymbol{u}, \delta\boldsymbol{u}$ and ∇_{sym} denote the Cauchy stress tensor, the body forces, the displacement

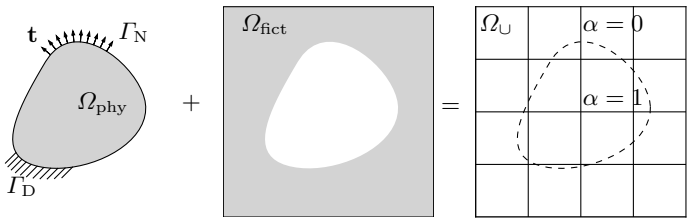

Figure 1: *The core concept of the FCM. The physical domain Ω_{phy} is extended by the fictitious domain Ω_{fict}. Their union, the embedding domain $\Omega_{\cup}$ can be meshed easily. The influence of the fictitious domain is penalized by the scaling factor α.*

vector, the test function and the symmetric part of the gradient, respectively. The traction vector $\boldsymbol{t}$ specifies the Neumann boundary conditions on Γ_{N}. Stresses and strains are related through the constitutive tensor $\mathbf{C}$:

$$\boldsymbol{\sigma} = \alpha \mathbf{C} : \boldsymbol{\varepsilon}, \tag{2}$$

where α is an indicator function defined as:

$$\alpha(\boldsymbol{x}) = \begin{cases} 1 & \forall \boldsymbol{x} \in \Omega_{phy} \\ 10^{-q} & \forall \boldsymbol{x} \in \Omega_{fict}. \end{cases} \tag{3}$$

In practice, the value of q is chosen between 6 and 12.

Homogeneous Neumann boundary conditions are automatically satisfied by the formulation. Non-homogeneous Neumann boundary conditions can be realized by evaluating the contour integral over Γ_{N} in Equation 1. Dirichlet boundary conditions are generally formulated in the weak sense, e.g. using the penalty method or Nitsche's method [13].

The unknown quantities $\delta\mathbf{u}$ and $\mathbf{u}$ are discretized by a linear combination of N_i shape functions with unknown coefficients $\mathbf{u}_i$:

$$\mathbf{u} = \sum_i \mathrm{N}_i \mathbf{u}_i \ ; \ \delta\mathbf{u} = \sum_i \mathrm{N}_i \delta\mathbf{u}_i, \tag{4}$$

leading to the discrete finite cell representation:

$$\mathbf{K}\mathbf{u} = \mathbf{f}. \tag{5}$$

In the standard version of FCM, integrated Legendre polynomials known from high-order finite elements are employed as shape functions [22].

The stiffness matrix $\mathbf{K}$ results from a proper assembly of the element stiffness matrices:

$$\mathbf{k}^e = \int_{\Omega^{\mathrm{e}}} \left[\mathbf{L}\mathbf{N}^e\right]^T C^\alpha \left[\mathbf{L}\mathbf{N}^e\right] d\Omega^{\mathrm{e}}, \tag{6}$$

where $\mathbf{L}$ is the standard strain-displacement operator, $\mathbf{N}^e$ is the matrix of shape functions associated to the element and $C^\alpha = \alpha C$ is the constitutive matrix.

In the context of FCM, the above integral is usually evaluated by means of specially constructed quadrature rules to account for the discontinuous integrand due to the scaling factor α. The most popular method is based on composed Gaussian quadrature rules combined with a recursive subdivision of the elements cut by the boundary of the physical domain. In this process, every intersected element is subdivided into equal subcells, until a pre-defined depth is reached. Quadrature points

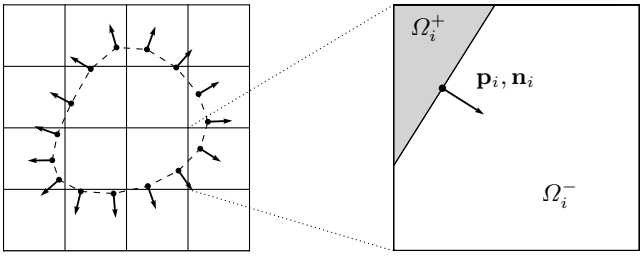

Figure 2: *Point membership classification on oriented point clouds. The domain is represented by a set of points* $\mathbf{p}_i$ *and associated normals* $\mathbf{n}_i$*. Every such pair locally separates the space along a hyperplane into two half-spaces:* Ω_i^- *and* Ω_i^+*.*

are then distributed on the domains of the leaf cells of this integration mesh.

To compute the value of C^α for a given quadrature point, the indicator function in Equation 3 needs to be evaluated. This requires the geometric model that represents Ω_{phy} to provide point-membership tests: given a quadrature point, does this point belong to Ω_{phy} or not? Many geometric representations are able to answer such inside-outside queries and have been successfully applied in combination with the FCM. Examples include voxel models from CT-scans [24], constructive solid geometries [25], boundary representations [15] and STL descriptions [26].

2.2 Point Membership Tests on Oriented Point Clouds

In the context of point cloud based simulations, the domain Ω_{phy} is represented by a set of sample points $\mathbf{p}_i$ and their associated normal vectors $\mathbf{n}_i$. Assuming that no outliers are present, the set of pairs $S = \{\mathbf{p}_i, \mathbf{n}_i\}$ constitute a discrete sampling of the boundary $\partial\Omega_{\text{phy}}$ of the domain.

Each element in S defines a hyperplane that separates the space in two half spaces: the open half-space Ω_i^- lying on the side of the hyperplane where the direction vector $\mathbf{n}_i$ points, and the closed half-space Ω_i^+ lying on the other side. This concept is depicted in Figure 2. For every $\mathbf{x} \in \Omega_i^+$, the following holds:

$$(\mathbf{p}_i - \mathbf{x}) \cdot \mathbf{n}_i \geq 0. \tag{7}$$

Therefore, to determine whether a quadrature point $\mathbf{q}$ lies inside or outside the domain, it suffices to find the $\mathbf{p}_i$ and the associated $\mathbf{n}_i$ in S that lies closest to $\mathbf{q}$ and evaluate the scalar product of Equation 7. The algorithm requires an efficient nearest neighbor query. In our examples, we use the k-d tree implementation from the Point Cloud Library [27]. The point membership classification method is summarized in Algorithm 1.

Algorithm 1: Point membership test for oriented point clouds

```
function isPointInside (q, S) ;
  Input  : Quadrature point q and oriented point cloud S = {p_i, n_i}
  Output: Boolean true if q lies inside the domain represented by S, false otherwise
p_i, n_i = getClosestPointInCloud(q, S);
v = p_i − q;
d = v · n_i;
if d ≥ 0 then
  | return true;
end
return false;
```

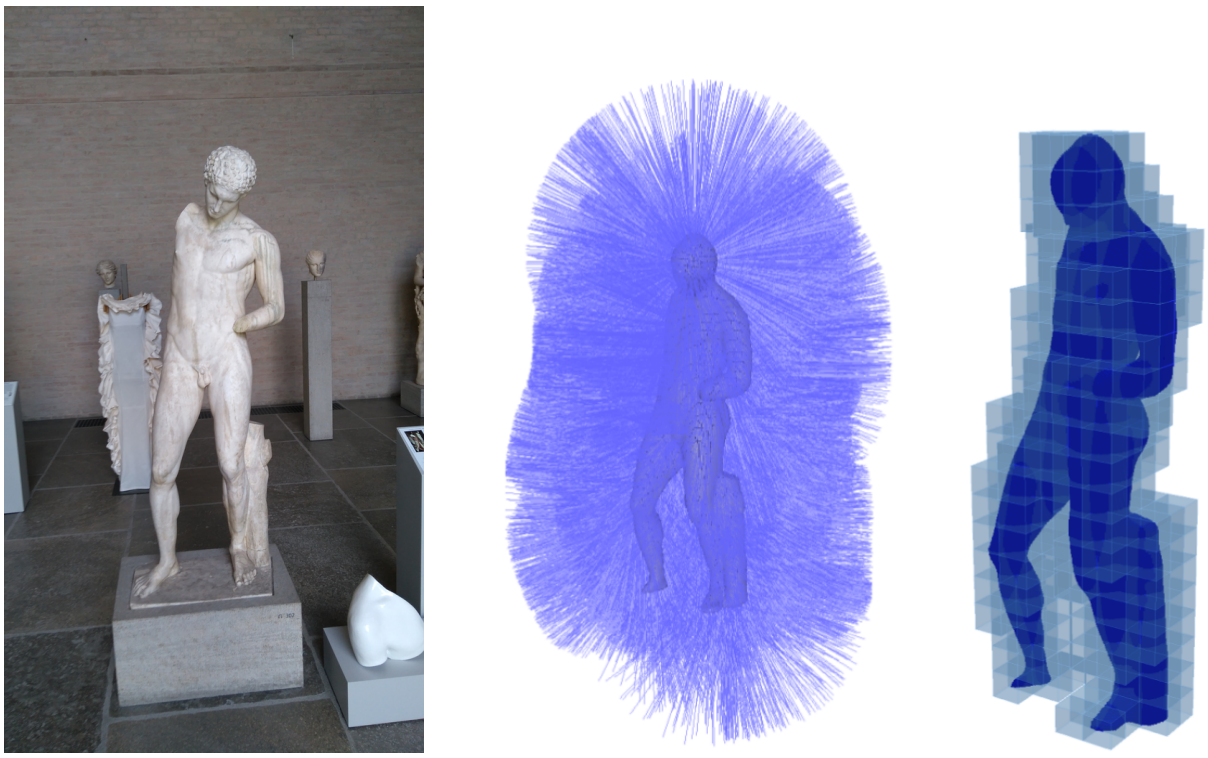

Figure 3: Linear elastic example: an input image, the resulting cloud with normal vectors and the structure embedded into a mesh of finite cells

3 Numerical Example

To demonstrate how the combination of the FCM and the proposed algorithm can be applied to solve linear elastostatic problems, a statue from the museum "Glypthotek" in Munich was recorded by a simple cell phone camera from 36 different views. The input images were processed by the popular structure-from-motion toolbox VisualSFM [28], and the multi-view reconstruction algorithm of [29]. The images and the resulting point cloud are depicted in Figure 3. The cloud was embedded in a regular mesh of 325 finite cells that were using integrated Legendre polynomials of order $p = 5$ for the discretization of the displacement field. The structure was loaded under its own weight and homogeneous Dirichlet (no displacement) boundary conditions were prescribed on the two feet. The value of α was chosen as 10^{-6}. The resulting field of von Mises stresses on the deformed structure is plotted in Figure 4.

4 Conclusion and Outlook

This contribution presented a method aiming at the simulation of objects represented by oriented point clouds. The technique is based on the Finite Cell Method, which, in its simplest implementation, only requires inside-outside information from the geometric model of interest. It was shown that oriented point clouds—if no outliers are present—are able to provide such point membership tests. As demonstrated by a numerical example, this allows for computations in the context of linear elastostatics, without the need for a boundary conforming finite element mesh or the reconstruction of a geometric model. This way, a seamless connection between photogrammetric shape measurements and high order numerical simulations can be established. In an upcoming paper we will report on an extension to more complex boundary conditions and focus on the robustness of the method, in particular on cases, when point clouds include outliers or yield incomplete information of some parts of the surface.

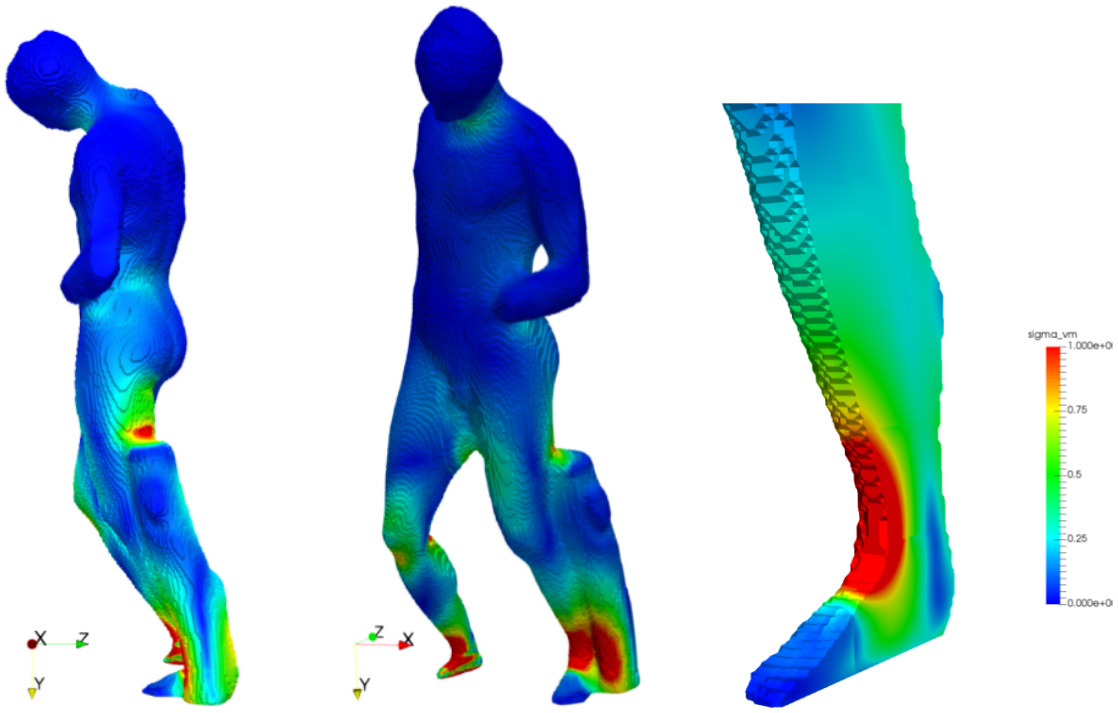

Figure 4: Linear elastic example: the resulting von Mises stresses over the structure with a cross-sectional view on the left foot

References

[1] A. Borri and A. Grazini, "Diagnostic analysis of the lesions and stability of Michelangelo's David," *Journal of Cultural Heritage*, vol. 7, no. 4, pp. 273–285, 2006.

[2] B. Riveiro, J. Caamaño, P. Arias, and E. Sanz, "Photogrammetric 3D modelling and mechanical analysis of masonry arches: An approach based on a discontinuous model of voussoirs," *Automation in Construction*, vol. 20, no. 4, pp. 380–388, 2011.

[3] U. Almac, I. P. Pekmezci, and M. Ahunbay, "Numerical Analysis of Historic Structural Elements Using 3D Point Cloud Data," *The Open Construction and Building Technology Journal*, vol. 10, no. 1, 2016.

[4] S. G. Barsanti, G. Guidi, and L. De Luca, "Segmentation of 3D Models For Cultural Heritage Structural Analysis-Some Critical Issues.," *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences*, vol. 4, 2017.

[5] L. Kudela, F. Frischmann, O. E. Yossef, S. Kollmannsberger, Z. Yosibash, and E. Rank, "Image-based mesh generation of tubular geometries under circular motion in refractive environments," *Machine Vision and Applications*, Apr 2018.

[6] J. A. Cottrell, T. J. Hughes, and Y. Bazilevs, *Isogeometric analysis: toward integration of CAD and FEA*. John Wiley & Sons, 2009.

[7] J. Parvizian, A. Düster, and E. Rank, "Finite cell method," *Computational Mechanics*, vol. 41, no. 1, pp. 121–133, 2007.

[8] C. S. Peskin, "The immersed boundary method," *Acta numerica*, vol. 11, pp. 479–517, 2002.

[9] B. Szabó, A. Düster, and E. Rank, "The p-version of the finite element method," *Encyclopedia of computational mechanics*, 2004.

[10] D. Schillinger, M. Ruess, N. Zander, Y. Bazilevs, A. Düster, and E. Rank, "Small and large deformation analysis with the p-and B-spline versions of the finite cell method," *Computational Mechanics*, vol. 50, no. 4, pp. 445–478, 2012.

[11] F. de Prenter, C. Verhoosel, G. van Zwieten, and E. van Brummelen, "Condition number analysis and preconditioning of the finite cell method," *Computer Methods in Applied Mechanics and Engineering*, vol. 316, pp. 297–327, 2017.

[12] E. Burman, S. Claus, P. Hansbo, M. G. Larson, and A. Massing, "CutFEM: Discretizing geometry and partial differential equations," *International Journal for Numerical Methods in Engineering*, vol. 104, no. 7, pp. 472–501, 2015.

[13] M. Ruess, D. Schillinger, Y. Bazilevs, V. Varduhn, and E. Rank, "Weakly enforced essential boundary conditions for NURBS-embedded and trimmed NURBS geometries on the basis of the finite cell method," *International Journal for Numerical Methods in Engineering*, vol. 95, no. 10, pp. 811–846, 2013.

[14] T.-P. Fries and S. Omerović, "Higher-order accurate integration of implicit geometries," *International Journal for Numerical Methods in Engineering*, vol. 106, no. 5, pp. 323–371, 2016.

[15] L. Kudela, N. Zander, S. Kollmannsberger, and E. Rank, "Smart octrees: Accurately integrating discontinuous functions in 3D," *Computer Methods in Applied Mechanics and Engineering*, vol. 306, pp. 406–426, 2016.

[16] M. Joulaian, S. Hubrich, and A. Düster, "Numerical integration of discontinuities on arbitrary domains based on moment fitting," *Computational Mechanics*, vol. 57, no. 6, pp. 979–999, 2016.

[17] D. Schillinger, A. Düster, and E. Rank, "The hp-d-adaptive finite cell method for geometrically nonlinear problems of solid mechanics," *International Journal for Numerical Methods in Engineering*, vol. 89, no. 9, pp. 1171–1202, 2012.

[18] A. Abedian, J. Parvizian, A. Düster, and E. Rank, "The finite cell method for the J2 flow theory of plasticity," *Finite Elements in Analysis and Design*, vol. 69, pp. 37–47, 2013.

[19] M. Elhaddad, N. Zander, T. Bog, L. Kudela, S. Kollmannsberger, J. Kirschke, T. Baum, M. Ruess, and E. Rank, "Multi-level hp-finite cell method for embedded interface problems with application in biomechanics," *International Journal for Numerical Methods in Biomedical Engineering*, vol. 34, no. 4, p. e2951.

[20] F. Xu, D. Schillinger, D. Kamensky, V. Varduhn, C. Wang, and M.-C. Hsu, "The tetrahedral finite cell method for fluids: Immersogeometric analysis of turbulent flow around complex geometries," *Computers & Fluids*, vol. 141, pp. 135–154, 2016.

[21] T. Bog, N. Zander, S. Kollmannsberger, and E. Rank, "Weak imposition of frictionless contact constraints on automatically recovered high-order, embedded interfaces using the finite cell method," *Computational Mechanics*, vol. 61, no. 4, pp. 385–407, 2018.

[22] A. Düster, E. Rank, and B. Szabó, "The p-Version of the Finite Element and Finite Cell Methods," *Encyclopedia of Computational Mechanics Second Edition*, 2017.

[23] T. J. R. Hughes, *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Dover Publications, 2000.

[24] M. Ruess, D. Tal, N. Trabelsi, Z. Yosibash, and E. Rank, "The finite cell method for bone simulations: verification and validation," *Biomechanics and modeling in mechanobiology*, vol. 11, no. 3-4, pp. 425–437, 2012.

[25] B. Wassermann, S. Kollmannsberger, T. Bog, and E. Rank, "From geometric design to numerical analysis: A direct approach using the Finite Cell Method on Constructive Solid Geometry," *Computers & Mathematics with Applications*, vol. 74, no. 7, pp. 1703–1726, 2017.

[26] M. Elhaddad, N. Zander, S. Kollmannsberger, A. Shadavakhsh, V. Nübel, and E. Rank, "Finite cell method: High-order structural dynamics for complex geometries," *International Journal of Structural Stability and Dynamics*, p. 1540018, 2015.

[27] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE International Conference on Robotics and Automation (ICRA)*, (Shanghai, China), May 9-13 2011.

[28] C. Wu *et al.*, "VisualSFM: A visual structure from motion system,"

[29] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multiview stereopsis," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 8, pp. 1362–1376, 2010.