The Visual Computer (2021) 37:2447–2460
https://doi.org/10.1007/s00371-021-02189-0

ORIGINAL ARTICLE

Tetrahedra of varying density and their applications

Dennis R. Bukenberger[1] · **Hendrik P. A. Lensch**[1]

Accepted: 1 June 2021 / Published online: 5 July 2021

Abstract
We propose concepts to utilize basic mathematical principles for computing the exact mass properties of objects with varying densities. For objects given as 3D triangle meshes, the method is analytically accurate and at the same time faster than any established approximation method. Our concept is based on tetrahedra as underlying primitives, which allows for the object's actual mesh surface to be incorporated in the computation. The density within a tetrahedron is allowed to vary linearly, i.e., arbitrary density fields can be approximated by specifying the density at all vertices of a tetrahedral mesh. Involved integrals are formulated in closed form and can be evaluated by simple, easily parallelized, vector-matrix multiplications. The ability to compute exact masses and centroids for objects of varying density enables novel or more exact solutions to several interesting problems: besides the accurate analysis of objects under given density fields, this includes the synthesis of parameterized density functions for the make-it-stand challenge or manufacturing of objects with controlled rotational inertia. In addition, based on the tetrahedralization of Voronoi cells we introduce a precise method to solve $L_{2|\infty}$ Lloyd relaxations by exact integration of the Chebyshev norm. In the context of additive manufacturing research, objects of varying density are a prominent topic. However, current state-of-the-art algorithms are still based on voxelizations, which produce rather crude approximations of masses and mass centers of 3D objects. Many existing frameworks will benefit by replacing approximations with fast and exact calculations.

Keywords Volumetric models · Shape analysis · 3D printing

1 Introduction

When it comes to estimating masses or computing gravitational centers for objects of varying density almost everywhere discretized approximations are used. Often their accuracy is not even questioned. Surprisingly, it is neither a very complex problem nor bound to numerical trade-offs between computation cost and accuracy. As used in finite element methods (FEM) [23,24], our techniques are based on tetrahedra as the underlying geometry primitive. Density fields are accurately represented by specifying the density at the four vertices. The analytically derived closed-form integrals for mass properties are expressed with simple matrix-vector multiplications. In Sect. 2, we derive the concept, demonstrate its versatile applicability in Sect. 3 and conclude in Sect. 4 with numerical comparisons and a quantitative evaluation.

1.1 Contributions

In our paper, we briefly review the mathematical basis that allows for accurate solutions of mass and mass-center integrals of objects with varying density. It is **simple** as computations for tetrahedra with constant density are easily extended towards varying density. It is **precise** due to the use of a tet-mesh itself as input and therefore avoiding aliasing bias from discretization, as it is common in state-of-the-art applications using axis-aligned voxelizations. It is **fast** as computations can be implemented as matrix-vector multiplications, suitable for vectorized or GPU execution. It is **versatile** because the framework is not limited to density and can be extended to integrate arbitrary linear properties over a volume.

Our application framework, introduced in Sect. 3, can be summarized with the following main contributions:

- **Arbitrary Objects** The solutions, known for tetrahedra, can be straightforward generalized to arbitrary polyhe-

✉ Dennis R. Bukenberger
dennis.bukenberger@uni-tuebingen.de

[1] University Tübingen, Tübingen, Germany

dra. We can accurately determine mass properties for any tetrahedralized input object in specified density fields.
- **Optimizing Density** We can also invert the problem and optimize a parameterized density field for an object and given mass properties.
- **Approaching nonlinearity** Arbitrary nonlinear functions can be approximated in a *Taylor*-like piecewise linear fashion, limited in accuracy only by the tetrahedralization's resolution.
- **Additive manufacturing** Combining the former two contributions, we eventually propose to 3D-print objects of nonlinear density with optimized mass properties for balance or rotation-aligned inertia momentum.
- **Expressing energy functions** As the method is not bound to only physical characteristics, we extend a Lloyd relaxation procedure based on L_2 Voronoi cells, using the concept to replace the cells energy integral with an accurate closed form solution for an L_∞ objective function.

1.2 Related work

Over the last few years, 3D printing not only attracted the do-it-yourself hobbyist community but also gained popularity in various industrial applications. Nowadays, additive manufacturing processes go way beyond stacked layers of plastic and support a wide range of multiple or mixed materials, even including metals. Its widespread use, e.g., in the medical [31] or automotive [11] industry, keep this a relevant research topic.

Hence the general interest of the computer graphics community for analyzing and processing 3D geometry, research in this field also spawned state-of-the-art algorithms aiming at 3D manufacturing. The procedure introduced by Prévost et al. [22] allows 3D models to be balanced in a specific position by shifting the object's center of gravity over a safe area on which the object eventually stands. Optimized weight distribution is achieved by carving out the object's interior and deformations of the hull, if necessary. Advancements of this technique optimize objects to have rotation-symmetric weight distributions and allow them to spin-like toy-tops [1]. Multistable balancing states are accomplished by using movable masses [21]. However, established techniques for mass property optimization are still based on approximations of the actual volume and mass distribution using quantized voxelizations. The approach by Musialski et al. [18] utilizes offset surfaces for shape and mass property optimization but also relies on binary material distribution. Even publications specialized on varying density for manufacturing [10,29] discretize their density field with marching cubes [15] or octrees [16] combined with dithering techniques.

Known methods for computing exact mass properties of polyhedral bodies [17] are restricted to constant density. Like ours, they are based on integrals over the volume and surface of an object. A later revision [6] made the concept feasible for implementation. The varying density of polyhedral bodies, however, was first studied in the field of geophysics and concluded with the focus on gravitational fields [5,7] but not general mass properties. Our approach to computing accurate masses and mass centers under varying density relies on a tetrahedral decomposition of the input object. *TetGen* [26] is a tetrahedralization tool for polyhedral manifolds based on a Voronoi/Delaunay tessellation. Most recently, *TetWild* [9] introduced another fast and robust way to tetrahedralize any given 3D triangle soup, providing many adjustable parameters to the user. Our results for examples and applications are based on the outputs of both tools but often specifically on *TetGen* since it is able to preserve the original input surface. However, any other tet-meshing pipeline will generate equally suitable input as well.

As mentioned with *TetGen*, tetrahedralization is closely related to Voronoi and Delaunay graphs. In our Sect. 3.5, we introduce a novel approach on Lloyd relaxations (based on Voronoi tessellations) using the L_∞ norm. Ray et al. proposed to compute meshless Voronoi [25] and restricted power diagrams [2] on the GPU, however, both only in the common L_2 space. It definitely is a promising task to explore combinations of their diagrams and our take on L_∞ relaxations.

2 Concept

Our goal is to approach mass properties for polyhedral manifolds of varying densities with analytical tools. In order to compute the mass, the center of mass, or other related quantities in a field of varying density that is bounded by a triangle mesh we use closed-form solutions for a tetrahedralization, induced by the given mesh surface. These kinds of approaches are admittedly standard in FEM but so far rarely have been used in computer graphics. Therefore, this section briefly summarizes all important formulas and introduces the geometric concept that allows for computing these quantities exactly for linearly varying density fields inside a tetrahedron. Detailed derivations of the resulting equations are featured in "Appendix A".

2.1 Problem statement

Mesh data structures are the straightforward and, therefore, most common way to store and represent 3D manifolds. Under constant density, mass properties like the center of mass or an inertia momentum can be easily computed directly on a mesh using the divergence theorem. With varying density, however, a volume bounded by arbitrarily shaped polyhedra cannot be directly integrated. A common fallback solution is to approximate the object shape by decomposi-

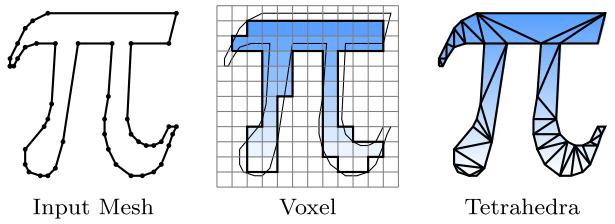

Fig. 1 2D example: Voxelizations are common approximations to determine mass properties under varying density. Tetrahedra allow for accurate representations using precise analytic results

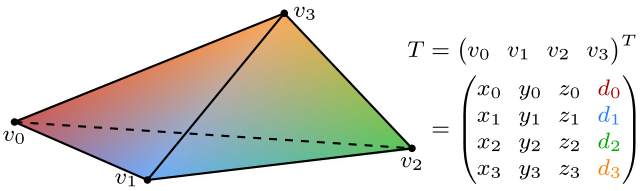

Fig. 2 A general tetrahedron T with varying density, defined by its four vertices v_i. In addition to their geometrical dimensions (x, y, z), each vertex is attributed with a fourth density dimension d

tion into feasible volume entities, usually voxels. However, as illustrated with the 2D example in Fig. 1, the success and precision of the approximation will always be limited by the chosen resolution (for space and values), even hierarchical concepts can only reduce sampling artifacts but not fully avoid them. In contrast, our solution for the computation under varying density is based on the simple idea of an alternative volume representation, namely the tetrahedron. As illustrated on the right in Fig. 1 with a trivially triangulated 2D shape, every 3D shape with a polygonal surface can be decomposed using tetrahedra. With tetrahedra, the mesh's true shape can be used in all computations and, therefore, corresponding results are free of discretization and aliasing bias.

2.2 Geometry integration

Computing mass properties for the general tetrahedron T, specified in Fig. 2, is trivial for constant density: For example with $d_i = 1$, the mass is equal to the tet-volume and the center of mass is equal to its centroid. However, as the density attributes at each vertex can be individually specified, expressing a linear density field inside the tetrahedron, the computation of mass and mass center changes. Instead, as in FEM [23,24], we utilize a simple basis case in a linear density field, for which the integration is solved analytically. A linear combination of four base cases (one per vertex) already gives the desired properties for a general tetrahedron.

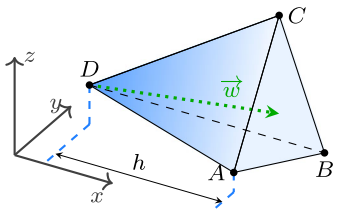

Fig. 3 The basis case for an *integrable* tetrahedron with $d_D = 1$ and $d_{A,B,C} = 0$. The density gradient over the extent of h is visualized with fading blue. $\vec{w}$ indicates the center of mass vector

2.3 Mass properties for arbitrary tetrahedra

Mass Due to linearity, the mass for a tetrahedron with four different density values at its vertices can be simply expressed as the mean of these values times the volume, formulated in Eq. 1.

$$\mathbf{M}_T = \mathbf{V}_T \frac{d_A + d_B + d_C + d_D}{4} \tag{1}$$

Center of Mass As expressed in Eq. 2, the mass center computes as a weighted sum of vertex positions and their normalized density values. An extended derivation for the combination of the four base cases to this general form is included in Eq. 12.

$$\mathbf{C}_T = \frac{1}{5}\left(A + B + C + D + \frac{A \cdot d_A + B \cdot d_B + C \cdot d_C + D \cdot d_D}{d_A + d_B + d_C + d_D}\right) \tag{2}$$

3 Application

With the presented approach, one can now calculate the mass and further mass properties of tet-meshes with arbitrary density fields efficiently and exactly. The power of the approach will be demonstrated in three different application scenarios: For example, as a fast and accurate replacement for widespread voxel-based approximations of arbitrary objects' mass properties. Further, it can be used to optimize the density distribution inside an object to obtain a given center of mass or a stable rotation axis, potentially even with nonlinear density fields. By introducing a closed-form solution, our concept even allows us to formulate an objective function in a volumetric Lloyd relaxation process, which was so far not analytically feasible.

3.1 Mass properties of arbitrary objects

With the techniques, introduced in Sect. 2, to compute mass and center of mass for general tetrahedra, we can generalize this concept further and approach arbitrary polyhedral manifolds: Objects are partitioned into tetrahedra, mass properties are determined individually, and results eventually recombined. Any tetrahedral mesh is suitable as input for our

method; if the model is not already available as tetrahedral mesh, it can easily be generated using freely available tools like *TetGen* [26] or *TetWild* [9]. "Appendix C" proves the concept to be invariant of the actual tetrahedralization.

$$\begin{aligned} \mathbf{M_O} &= \sum_{T_i \in \mathbf{O}} \mathbf{M}_{T_i} \\ \mathbf{C_O} &= \frac{1}{\mathbf{M_O}} \sum_{T_i \in \mathbf{O}} \mathbf{M}_{T_i} \mathbf{C}_{T_i} \end{aligned} \tag{3}$$

For an object $\mathbf{O}$ and a given density-field, one can now compute the accurate mass $\mathbf{M}_{T_i}$ and center of mass $\mathbf{C}_{T_i}$ for all tetrahedra $T_i \in \mathbf{O}$ using Eqs. 1 and 2, respectively. These calculations can be executed very efficiently, using fast matrix-vector multiplications. As formulated in Eq. 3, the object's overall mass is obtained by simple summation and the center of mass as mass-weighted dot-product. Further, one may extend the derivation, as described by Tonon [28], for the inertia tensor $\Theta_{\mathbf{T_i}}$ of a general tetrahedron with specified density values of the individual vertices. Following the rules for rigid bodies and the parallel axis theorem, one can derive further mass properties, e.g., the moment of inertia as formulated with the inertia tensor $\Theta_{\mathbf{O}}$ in Eq. 4, where $\mathbf{I}_3$ is the 3×3 identity matrix and $\otimes$ the outer product.

$$\begin{aligned} \hat{\mathbf{C}}_{T_i} &= \mathbf{C}_{T_i} - \mathbf{C_O} \\ \Theta_{\mathbf{O}} &= \sum_{T_i \in \mathbf{O}} \Theta_{\mathbf{T_i}} + \mathbf{M}_{T_i} \left(|\hat{\mathbf{C}}_{T_i}|^2 \mathbf{I}_3 - \hat{\mathbf{C}}_{T_i} \otimes \hat{\mathbf{C}}_{T_i} \right) \end{aligned} \tag{4}$$

3.2 Optimizing density fields

Now, that an object's center of mass can be determined for a given density field, one can invert the problem and fit a density field to an object where the mass properties are given. As an exemplary use case, we approached the make-it-stand challenge described by Prévost et al. [22] to balance objects in a given pose. The center of mass has to be within certain boundaries of a projected surface polygon on which the object is supposed to be balanced. However, a solution to this problem is limited by the following constraints: **(i)** Negative mass is reasonable only in theoretical fields of physics, so we limit our model to the realm of positive density for now. **(ii)** Zero density is a special case that can be modeled with our concept, e.g., with $d_i = 0$. **(iii)** The shape of an object together with constraints (**i**) and (**ii**) will put some limits on the achievable location of an object's center of mass, e.g., it simply cannot pass a certain point.

Rather than optimizing the per-vertex density directly, let's first consider a simplified density field as illustrated in Fig. 4. We utilize two planes that separate volumes of constant minimum $d_{\min}$ and maximum density $d_{\max}$ respectively, sandwiching a slice of volume of width r with linearly

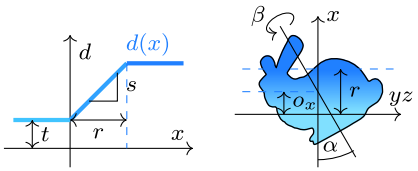

Fig. 4 Left: The parameterization of a simplified density field $d(x)$ with parameters r, s and t. Right: Embedding of the object, with angles α, β and offset o_x

growing density $\in [d_{\min}, d_{\max}]$. To simplify many computation steps we fix the density field to be axis aligned, i.e., the planes are parallel to the yz-plane. As accommodation for this fixed orientation of the field, the optimization needs to rotate and to translate the object accordingly instead. To realize the arbitrary location of the bisection-planes in the density function, we have to prepare our input mesh by intersecting some of the tetrahedra, see "Appendix B" for details. The energy to be minimized is formulated as the Euclidean distance $E = |\mathcal{C} - \mathbf{C}_O|$ between a target point $\mathcal{C}$ and the object's current center of mass $\mathbf{C}_O$ when embedded in the density field, obviously with respect to the object's rotated and translated state.

The optimization energy is smooth and in some sense probably differentiable but deriving gradients is left for future work. In our experiments, we used Powell's method [20] to minimize the objective function.

Results Figure 5 compares our balanced objects to cross-sections of Prévost et al. Their proposed method found a solution to make the three spheres stand, by carving out the voxelized interior and deforming the object. To move the mass center of the *Spheres* into the balance region, the top sphere is shrunk and the bottom sphere is enlarged. For the *MrHumpty* figure to stand upside down, the belly is enlarged and half the interior carved out to compensate for the off-axis legs. In our results, the objects remain untouched as they are only embedded in an optimized density field. As our output geometry incorporates the input, error measures like the Hausdorff distance are simply 0.

Our first experiment meets the same conditions as Prévost et al. where the center of gravity only has to be on the central vertical axis of the bottom sphere so that the object is in balance. For our next experiment we chose the center of mass to also be located centered in the bottom sphere, but 10% of the sphere's radius below its horizontal equator-line. Due to this low center of gravity, the *standing* spheres would roll into this position on their own (Table 1).

Our optimization managed to define density fields for which the object's center of mass is exactly on the specified axis or target point respectively. The results have regions of constant minimum and maximum density with a tilted and shifted gradient between them. Due to the symmetry of the

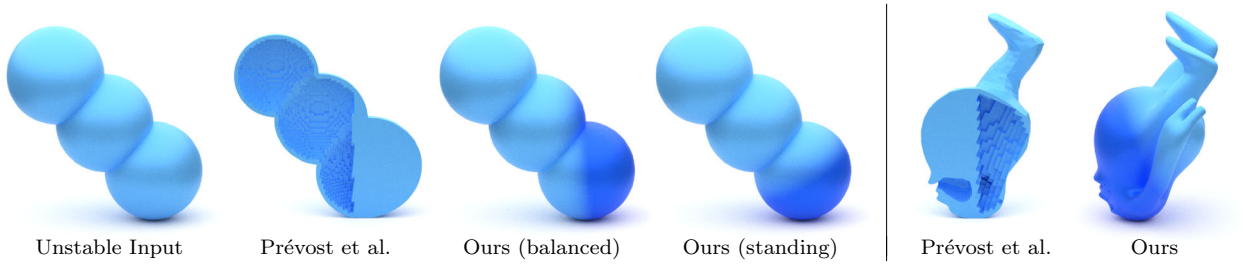

Fig. 5 An unstable input of three spheres and a figure which is supposed to stand upside down. Prévost et al. [22] managed to balance the objects by deforming them, caving out the interior and shifting the center of mass over a defined safe-region. Our *balanced* version of the spheres can also be balanced on a small flattened face, the *standing* version will roll into this position on its own, due to its low center of gravity. Varying density is sufficient to balance the objects, deformation is not required. As reference for Table 2, the spheres are scaled to have a diameter of 1 and MrHumpty has a hand-to-hand width of 2

Table 1 Parameters of the density field, shown in Fig. 4, to be determined by the optimization

α, β	Angles for tilt and rotation of the object
o_x	Object center offset on the x-axis
r	Width of the density range
s	Steepness of the density gradient
t	Constant density offset

Table 2 The parameters for the density function, specified in Fig. 4 and Table 1, optimized for balancing the objects, shown in Fig. 5 with $\beta = 0$ and $t = 1$

	α	o_x	r	s
Spheres (b)	−1.666670	−0.278929	0.404357	13.418110
Spheres (s)	−0.513042	−0.453811	0.565776	26.173806
MrHumpty	1.104747	0.244073	0.562282	2.614978

objects, the angle β is zero. To approach somewhat reasonable manufacturing limits, we set $t = 1$ ($\hat{=} d_{\min}$). The other found parameters are given in Table 2. Results of this comparison should be seen as a theoretical proof of concept, as this rather unconstrained optimization leads to quite high values for the gradient steepness s. Density differences of this multitude are ill-suited for current single-material manufacturing techniques. Additive multi-material techniques, on the other hand, could approximate smooth gradients like this, e.g., using dithering.

Fig. 6 Examples of nonlinear density fields, sampled at vertex positions. Tetrahedralizations created with *TetWild* on default settings for the Bunny and with a smaller edge-length for the Femur

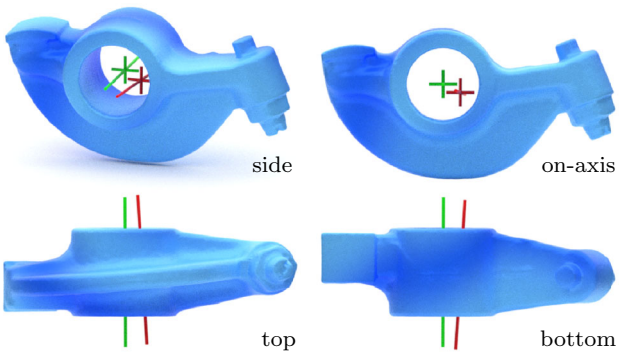

Fig. 7 The rockerarm is a prominent example for an asymmetric object with rotary mount. Due to imbalance, the native center of mass (red) is not located on the rotation axis. With optimized density, our center of mass (green) is located on the rotation axis and the principle inertia momentum axis is parallel to the rotation axis

3.3 Optimizing nonlinear fields

Section 2.3 introduces our concept for density fields with a generalization to define geometry-independent density values per vertex. This allows for the approximation of arbitrary nonlinear fields, as illustrated with the examples in Fig. 6: The Bunny is embedded in a spherical sinusoidal density function, the density in the Femur decreases from surface to core with a Gaussian slope. Tet-mesh vertices become 3D sampling positions for the 3D density field, however, gradients within each tetrahedron are still linear. Nevertheless, this piecewise linear *Taylor* approximation of a nonlinear field is C_0 continuous everywhere (C_∞ within a tetrahedron). The accuracy of this representation is only limited by the resolution of the tetrahedral mesh, which can be specified in common tetrahedralization tools.

Advanced applications, specifying more than a single center of mass, may require density fields, more sophisticated than linear gradients. An approach related to the make-it-stand concept proposed the challenge to make objects spinnable [1] by moving mass centers to a specified rotation axis. This is not only a desirable criterion for toy tops or yo-yos but is also of great value in any mechanical process involving rotating movements to reduce the wear and tear of involved components. Engineering such mechanical components often comes with tight constraints on available space and does not allow for arbitrary placement of counterweights. Figure 7 illustrates an example with the *rockerarm* object, which is to be mounted on rotary bearings. With constant density, the native center of gravity and inertia tensor are off-axis due to the obvious asymmetry of the object.

$$d(v) = \sin(|p - v| \cdot k) + 1 \tag{5}$$

For this object, the optimized density field results in a center of mass located on the rotation axis along with a parallel principle inertia momentum axis. The density field is parameterized with the nonlinear density function $d(v)$ (Eq. 5), where v is a tet-mesh vertex, p a 3D coordinate and k a scalar factor.

Results Optimizing for a specific center of mass, as in Sect. 3.2, is not trivial but possible, dependent on given constraints. Additionally fitting a principle inertia momentum axis, however, can be challenging as the density distribution for a certain center of mass may be in conflict with the optimal density for the momentum axis. A field parameterization with more degrees of freedom than ours (Eq. 5) might be more suitable for optimization but unreasonable for practical results. Our results are shown in Fig. 7 with an optimized center of mass (green). The density parameters are:

$$p = \begin{pmatrix} 0.509475 \\ -0.699066 \\ 1.328176 \end{pmatrix} \quad \text{and} \quad k = 7.853375$$

The principle inertia momentum axis was met with accuracy of $< 1°$, the center of mass is actually precisely located on the specified target rotation axis.

3.4 3D printing of varying density

For some objects, we demonstrate both synthetic results as well as 3D printouts. One has to mention that 3D printing hardware for varying density is still in an early development state [8,13,19] and the range of available densities is limited. On recent *Prusa* FDM printers, however, it is possible to alter the extrusion rate while printing. The first step is to obtain the G-code for an object with a regular infill pattern. In order to approximate the optimized density field, we modify the line thickness to vary along printed segments by accordingly adjusting the relative extrusion rate in the G-code slice by slice.

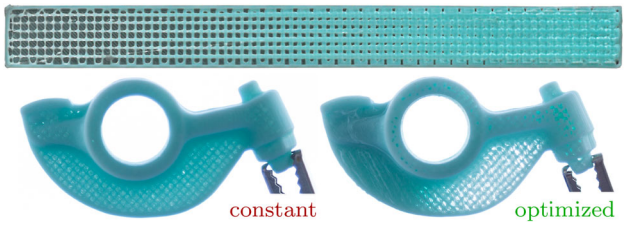

Fig. 8 Modifying the extrusion rate allows for printing density gradients. The bar on top is a cross-section of an object 10×10×100 mm in size, scanned with a photocopier. Pictures of the rockerarms (62.9×33×17.6 mm) were taken in front of a lightsource to highlight the different density distributions. See video for live demo

Figure 8 shows a 3D printed example of a simple bar with increasing density. The varying amount of printed material alters the translucency of the object. The 3D printed rockeram of Fig. 7 with optimized density for an on-axis center of mass and a parallel principle inertia momentum axis leads to significantly smoother spinning as can be seen in the accompanying video.

3.5 Lloyd relaxation with the L_∞ norm

The last proposed application scenario makes use of the closed-form integral solution of measures on volumes of varying density to approach Lloyd relaxations under non-standard norms. All calculations are done on the actual shape of the Voronoi cells avoiding any voxelization, which reduces artifacts and speeds up the computation.

Lloyd's algorithm [14] is an iterative optimization procedure that is proven to converge to Centroidal-Voronoi-Tessellations (CVT) under the L_2 norm [4]. The iteration alternates two steps: **I.** Compute a Voronoi diagram for a given set of points. **II.** Reposition each point to the centroid of its Voronoi cell. This can also be formulated as an optimization task, minimizing the diagram's global energy function.

Since the native L_2 cells are all convex, the computation of new centroids is quite simple. However, in many meshing applications, L_p or even L_∞ are more desirable [12], due to their more rectangular or cubical cell shapes. For L_p norms ($p > 2$), the Voronoi cells are no longer always convex and the diagram becomes very impractical to handle or even generate since there is (to the best of our knowledge) no software library that is able to compute L_p or L_∞ Voronoi tessellations.

In meshing applications [12,27] the diagram itself is actually not relevant, but only the site positions are of interest [25]. We propose a way to compute Lloyd relaxed site positions with the L_∞ metric, also called the *Chebyshev* distance: First, cell geometry and topology are borrowed from an L_2

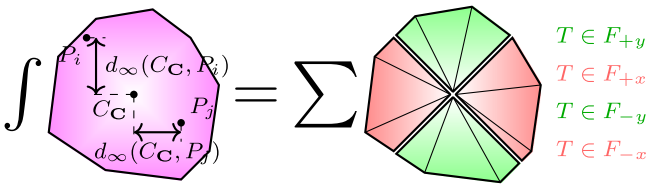

Fig. 9 2D visualization of the equivalent energy terms of Eq. 7 with an integral over the *Chebyshev* distances of all points in a cell or the sum over its tetrahedral fragments with density gradients

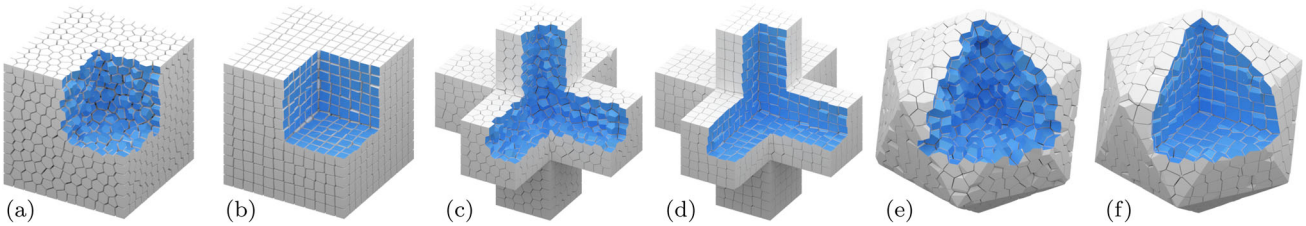

Fig. 10 Results after 50 Lloyd relaxations: A unit-cube on the left and two Clipped-Voronoi-Diagrams [30] in the middle and on the right (Cutouts in blue). Examples **a**, **c** and **e** used the L_2 norm to reposition their sites in each step, the examples **b**, **d** and **f** show results using the L_∞ norm, generating close to cube-like cells. Despite different relaxation norms, all results are visualized as L_2 tessellations

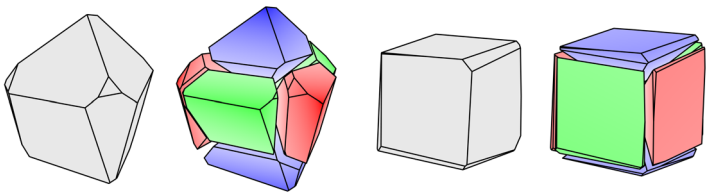

Fig. 11 Voronoi cells under the L_∞ norm before and after the relaxation. Saturation visualizes the L_∞ energy, increasing in each dimension. A centroid in a cubical cell minimizes this energy

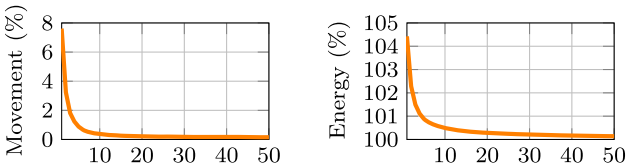

Fig. 12 L_∞ relaxation plots of the cube from Fig. 10b over 50 iterations. Left: Average movement of all cell centers in one iteration, given in percent of the optimal cell's diagonal. Right: Average L_∞ energy of all cells, given in percent of an optimal cell's L_∞ energy, which is why the result converges to 100%

tessellated diagram, which comes with the convenience of convex-only cell shapes. Then we use our method and compute a cell's mass, reinterpreted as the energy which is to be minimized by a new cell center.

The Energy Term For the goal to minimize the L_∞ energy within a cell, let us briefly recapitulate how the *Chebyshev* distance d_∞ is defined. As formulated in Eq. 6, the distance between two points p and q is the maximum of their absolute differences over all dimensions, in the 3D case x, y and z:

$$d_\infty(p, q) = \max_{k \in [x,y,z]} |q_k - p_k| \tag{6}$$

$$E_\mathbf{C} = \int_{P \in \mathbf{C}} \mathrm{d}_\infty(C_\mathbf{C}, P) = \sum_{i \in [\pm x, \pm y, \pm z]} \sum_{T \in F_i} \mathbf{M}_T^i \tag{7}$$

Equation 7 formulates the energy $E_\mathbf{C}$ of a cell $\mathbf{C}$ as the total *Chebyshev* distance of all points $P \in \mathbf{C}$ to the cell's centroid $C_\mathbf{C}$. However, there are infinitely many points $P \in \mathbf{C}$, so the energy function can only be evaluated with a nontrivial integral over the cell volume.

As illustrated in Fig. 9 (in 2D), this integral becomes feasible as a finite sum of analytical solutions. To achieve this, a cell is split into six fragments F_k ($k \in [\pm x, \pm y, \pm z]$), as illustrated in Fig. 11. This effectively separates all points P within the cell with respect to their maximum difference-dimension (*Chebyshev*). Due to the separation into the six fragments, the $d_\infty(C_\mathbf{C}, P)$ distance dimension conveniently coincides with the corresponding geometric dimension k, i.e., the distance linearly increases along one of the coordinate axes. As the hull of a Voronoi cell might be complex, the six fragments are tetrahedralized using a trivial triangulation of their hull faces and the cell center itself. The inner sum in Eq. 7 accumulates masses $\mathbf{M}_T^k$ of all tetrahedra T in a

Table 3 A practical performance comparison to octree approximations, where our method generated the reference mass properties

Input			*TetGen*		Ours (tet)	Octree (depth 4)				Octree (depth 8)			
Object	$\#v_{\text{in}}$	$\#f_{\text{in}}$	$\#t$	time (s)	time (s)	$\#n_4$	time (s)	ϵ_C (%)	ϵ_M (%)	$\#n_8$	$time(s)$	ϵ_C (%)	ϵ_M (%)
buddha	99370	198736	364531	3.0942	*0.1982*	2047	**5.8286**	0.436240	0.811702	730647	**637.780**	0.002739	0.043214
casting	5096	10224	15827	0.1200	*0.0064*	1719	**1.3123**	4.153140	12.116408	911551	**480.849**	0.712045	0.983078
fandisk	2877	5750	8556	0.0678	*0.0035*	1505	**0.9744**	1.059797	0.550461	433740	**217.282**	0.032861	0.087246
fertility	241607	483226	848832	7.9802	*0.4155*	1854	**10.3933**	0.338710	1.361950	695297	**996.629**	0.011783	0.121899
maxplanck	49132	98260	164562	1.3608	*0.0790*	1759	**4.9667**	0.195222	0.045735	561597	**446.385**	0.022529	0.076491
nefertiti	1009118	2018232	3301743	32.8039	*1.7506*	1677	**56.4184**	0.485230	0.636516	570914	**2756.926**	0.000983	0.008526
rockerarm	10044	20088	31131	0.2560	*0.0119*	2148	**1.7444**	0.404197	3.106854	818778	**469.041**	0.072802	0.130769
trefoil	10240	20480	37473	0.3160	*0.0244*	2221	**1.9554**	0.095244	0.459989	825107	**448.259**	0.000791	0.135191

Input objects are listed with number of vertices and faces. They are uniformly scaled to a unit-height of 1 with density linearly increasing from bottom ($d_{\text{min}} = 0$) to top ($d_{\text{max}} = 1$). Tet-inputs are listed with number of tets, time to generate with *TetGen* using the `-Y` option. Octree results are from two different depths, listing the number of nodes, the computation time (build + traversal) and two error measures (Eq. 8)

fragment F_k as defined in Eq. 1. The *Chebyshev* distance is simply encoded as the density dimension along the coordinate axes for our computation. The outer sum accumulates the density- (or *Chebyshev* distance-) weighted volumes of the six fragments, resulting in the cell's L_∞ energy. The cell center is finally repositioned to minimize the computed L_∞ energy using the *L-BFGS-B* algorithm [3,32] for bound constrained minimization.

Results Although cell energies are only optimized on an individual basis, the relaxation iteration also leads to a global decrease of the diagram's energy, analogously to the L_2 case. With our reformulation of the objective function, the second part of the Lloyd relaxation (repositioning of cell centers) becomes feasible for the L_∞ norm. The initialization of each iteration is still based on a computable L_2 Voronoi tessellation, which turned out to be sufficient as the relaxation still converges. Considering the shape of an L_2 Voronoi cell while optimizing the centers for the L_∞ energy, this optimization is not a full L_∞ relaxation but a convenient alternative. If an L_∞ tessellation was available, the relaxation would probably converge even faster and would also allow for individually oriented cells. Nevertheless, considering the alternatives, e.g., labeling underlying high-resolution voxel grids, it is an improvement in terms of both accuracy and performance.

The plots in Fig. 12 show convergence results of the cube (Fig. 10b) throughout 50 Lloyd relaxation steps. The *Movement* plot shows the average distance traveled by all sites (cell centroids) in the Voronoi diagram during each relaxation step. This distance is given in percent of an optimal cubical cell's diagonal. The average cell-energy, shown in the *Energy* plot, is computed for each cell as illustrated in Fig. 11 by separating a cell into six fragments and accumulating the density-weighted volume. It is expressed in percent of an optimal cubical cell's L_∞ energy. Therefore, the convergence towards 100% indicates that our cells approach the anticipated optimal cubical cell shape.

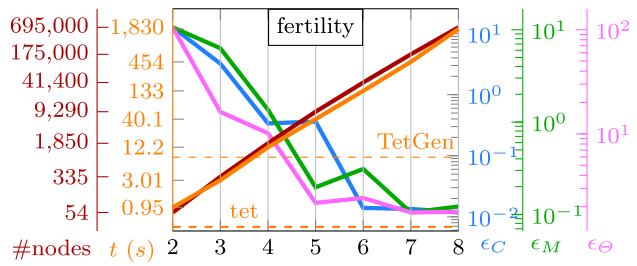

Fig. 13 By increasing the depth of an octree (# levels on the x-axis), the approximations converge against our analytic results. The plot shows the time in seconds to build and traverse the tree, the number of nodes in the tree, the mass center error ϵ_C, the mass error ϵ_M and inertia tensor error ϵ_Θ. Dashed lines show timings for our computation based on tets and the time it took for TetGen [26] to tetrahedralize the input

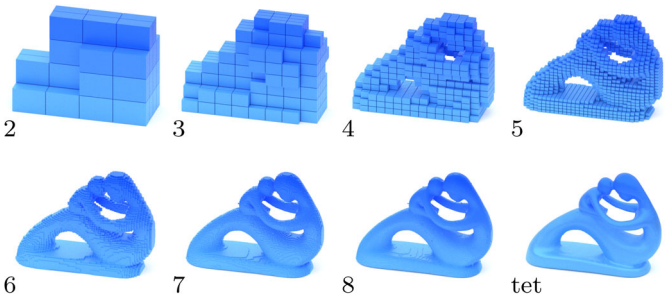

Fig. 14 Visualization of the octrees used in Fig. 13. With increasing depth (2-8) of the tree, the approximations for mass properties converge against our analytic results based on the real mesh

4 Discussion

This section presents the results of the proposed application scenarios for our concept. Benefits over traditional methods in terms of performance and accuracy are quantitatively discussed with numerical results. After an outlook on potential extensions and future work, we conclude with a roundup of our main contributions.

4.1 Results

As mentioned in Sect. 1.2, voxelizations or octrees are currently the most common method to approximate mass properties for objects in fields of varying density. Table 3 documents comparisons of our exact tetrahedron-based method to octrees of different depths. Our results provide the ground truth reference, to which the octree approximations are compared. Timings for the octrees include the build-up phase and traversal to compute the results. The most demanding parts in the build-up are in/out-tests, to decide if a cell is to be split again. To be comparable, we included the time to create the tet-mesh inputs for our method from basic surface meshes. Delaunay tetrahedralizations are computed with *TetGen* [26], using the `-Y` option which preserves the source mesh, so that our method and the octree have the exact same input. Both, octree and our method, are implemented in *Python* using vectorized *NumPy* arrays where possible for optimized efficiency. Although our method is well suited to be implemented in parallelized GPU code, all timings are measured on a single CPU core. The measurements show that, not only compared to the very deep but also for the small octree of only 4 levels, our method is multitudes faster even including the input tetrahedralization.

Timings and performance aside, the probably most valuable takeaway is the analytical accuracy of the results. Our method establishes actual ground truth results for mass properties under varying densities. Figure 13 plots the mass-property-errors of octree approximations (Fig. 14) converging against our results, as we increase their depth and therefore accuracy. Featured errors of mass ϵ_M, center of mass ϵ_C, and the inertia tensor ϵ_Θ are specified in Eq. 8.

$$\begin{aligned} \epsilon_C &= |C_{\text{octree}} - C_{\text{tet}}| \\ \epsilon_M &= \frac{|M_{\text{octree}} - M_{\text{tet}}|}{M_{\text{tet}}} \\ \epsilon_\Theta &= \sum_{i \in [x,y,z]} \frac{|\Theta^i_{\text{octree}} - \Theta^i_{\text{tet}}|}{|\Theta^i_{\text{tet}}|} \end{aligned} \tag{8}$$

In theory, a voxel grid of infinite resolution or an octree of infinite depth would give correct and unbiased mass property results. We use this capacity to show that octree results converge against our analytic results by increasing their depth and accuracy. The error plots do not converge monotonically due to aliasing artifacts, for some lower levels the approximations are just more accurate by chance.

4.2 Conclusion

We propose novel application scenarios for object's mass properties under varying densities. Easy to use analytical solutions make approximations obsolete, which are still common in recent state-of-the-art applications [1,10,21,22,29]. Our concept is fast, lightweight, easy to implement, and suitable for vectorized or parallelized frameworks. We demonstrate possible use cases where our method can be utilized straightforward: Masses, mass centers, and inertia tensors of arbitrary manifolds in given density fields are computed accurately. We formulate an optimization to determine a parameterized density field for an object and specified mass properties like a center of gravity or inertia tensor. Our proposed modification of the Lloyd relaxation is a novel $L_{2|\infty}$ hybrid that allows us to imitate real L_∞ relaxations, which is a leap forward compared to the existing approximative alternatives. While our approach may find direct application in established research topics as meshing, spatial tessellation and simulation [9,25], we also see great potential in the young scientific field of additive manufacturing and hope to inspire many further research [10,29,31].

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Human and animal rights No animals were harmed during the development of this publication.

A In-depth derivations

This appendix completes the derivations of the closed-form equations for mass and center of mass calculations used in Sect. 2.

$$\begin{aligned} M_T(D) &= \int_0^{h_D} \overline{ABC} \cdot \left(\frac{r}{h_D}\right)^2 \cdot \left(1 - \frac{r}{h_D}\right) \mathrm{d}r \\ &= \overline{ABC} \int_0^{h_D} \frac{r^2}{h_D^2} \cdot \left(1 - \frac{r}{h_D}\right) \mathrm{d}r \\ &= \overline{ABC} \cdot \frac{h_D}{12} = \underbrace{\overline{ABC} \cdot \frac{h_D}{3}}_{\mathbf{V}_T} \cdot \frac{1}{4} \end{aligned} \tag{9}$$

Figure 3 illustrates an exemplary basis case with density $d_D = 1$ and 0 at the three other vertices. Since the density is normalized, it is not surprising that the density integral over the volume in Eq. 9 results in a quarter of the tetrahedron's volume $\mathbf{V}_T$. Thus, when combined for a general tetrahedron embedded in an arbitrary density field, the mass $\mathbf{M}_T$ computes as the tetrahedron volume times the mean of all four density values, as formulated in Eq. 10.

$$\begin{aligned} \mathbf{M}_T &= d_A \cdot M_T(A) + d_B \cdot M_T(B) \\ &\quad + d_C \cdot M_T(C) + d_D \cdot M_T(D) \\ &= d_A \cdot \mathbf{V}_T \cdot \frac{1}{4} + d_B \cdot \mathbf{V}_T \cdot \frac{1}{4} \\ &\quad + d_C \cdot \mathbf{V}_T \cdot \frac{1}{4} + d_D \cdot \mathbf{V}_T \cdot \frac{1}{4} \\ &= \mathbf{V}_T \frac{d_A + d_B + d_C + d_D}{4} \end{aligned} \tag{10}$$

Equation 11 formulates the volume integration of the mass center $C_T(D)$ for the shown base case, using a scaled center vector $\vec{w}$.

$$\begin{aligned} C_T(D) &= D + \frac{1}{M_T(D)} \int_0^{h_D} \overline{ABC}(r) \cdot \left(1 - \frac{r}{h_D}\right) \cdot \vec{w}(r)\, \mathrm{d}r \\ &= D + \frac{1}{M_T(D)} \int_0^{h_D} \overline{ABC} \cdot \left(\frac{r}{h_D}\right)^2 \cdot \left(1 - \frac{r}{h_D}\right) \cdot \vec{w} \cdot \frac{r}{h_D}\, \mathrm{d}r \\ &= D + \frac{1}{\overline{ABC} \cdot \frac{h_D}{12}} \overline{ABC} \cdot \vec{w} \int_0^{h_D} \left(\frac{r}{h_D}\right)^3 \cdot \left(1 - \frac{r}{h_D}\right) \mathrm{d}r \\ &= D + \vec{w} \cdot \frac{12}{h_D} \cdot \frac{h_D}{20} = D + \vec{w} \cdot \frac{3}{5} \\ &= D + \left(\frac{A+B+C}{3} - D\right) \cdot \frac{3}{5} \\ &= \frac{1}{5}(A + B + C + D + D) \end{aligned} \tag{11}$$

The center of mass $\mathbf{C}_T$ in Equation 12 for the general tetrahedron computes as the combination of the individual base case mass centers, weighted and normalized by the individual density values at the four vertices, respectively.

$$\begin{aligned} \mathbf{C}_T &= \frac{d_A \cdot B_T(A) + d_B \cdot B_T(B) + d_C \cdot B_T(C) + d_D \cdot B_T(D)}{d_A + d_B + d_C + d_D} \\ &= \begin{pmatrix} d_A \\ d_B \\ d_C \\ d_D \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{5} \cdot (A + A + B + C + D) \\ \frac{1}{5} \cdot (A + B + B + C + D) \\ \frac{1}{5} \cdot (A + B + C + C + D) \\ \frac{1}{5} \cdot (A + B + C + D + D) \end{pmatrix} \cdot \frac{1}{d_A + d_B + d_C + d_D} \\ &= \begin{pmatrix} d_A \\ d_B \\ d_C \\ d_D \end{pmatrix} \cdot \left(\begin{pmatrix} A+B+C+D \\ A+B+C+D \\ A+B+C+D \\ A+B+C+D \end{pmatrix} + \begin{pmatrix} A \\ B \\ C \\ D \end{pmatrix} \right) \cdot \frac{1}{5 \cdot \sum d_i} \\ &= \left(\sum d_i \cdot (A + B + C + D) + \begin{pmatrix} d_A \\ d_B \\ d_C \\ d_D \end{pmatrix} \cdot \begin{pmatrix} A \\ B \\ C \\ D \end{pmatrix} \right) \cdot \frac{1}{5 \cdot \sum d_i} \\ &= \frac{1}{5} \left(A + B + C + D + \frac{A \cdot d_A + B \cdot d_B + C \cdot d_C + D \cdot d_D}{d_A + d_B + d_C + d_D} \right) \end{aligned} \tag{12}$$

B Tet-split

To model the density function described in Sect. 3.2, the object is bisected using split-planes. As the input is already tetrahedralized, there might be tetrahedra that are only partially in one or the other region. This potentially violates the constraints of Sect. 3.2, e.g., if a tetrahedron would cross the 0-density limit. Therefore, such tetrahedra with vertices on both sides of a split-plane have to be cut. Figure 15 illustrates the four possible cases, how a plane may intersect a tetrahedron. The splits result in geometry that is always further tetrahedralizable, hence triangular prisms, quad-based pyramids or trivial tetrahedra. Affected tetrahedra in the input are easily identified by checking if they match one of these cases.

After the cut, corresponding tetrahedra are simply replaced by the subdivided geometry.

C Proof of concept

This appendix aims to demonstrate the validity of our approach, especially proving mass properties of polyhedra to be invariant of the used tetrahedralization. Therefore, we assemble the simple scenario shown in Fig. 16: An axis-aligned box is (w.l.o.g.) centered on the x-axis with the $\langle AEHD\rangle$ quad face parallel to the yz-plane and the $\langle BFGC\rangle$ quad at a distance h to the origin at x_0. The density gradually increases over the x dimension dependent on the density function $d(x) = s \cdot x + t$. Box-related equations are denoted with an overset box-symbol $\square$, the tetrahedra equivalents are marked with a triangle-symbol $\triangle$.

C.1 Setup

For the constructed box scenario, it is fairly easy to determine its mass and mass center via integration as formulated in Eqs. 13 and 14, respectively. The cross-section-area of the box is formally expressed as a function $\Phi_B(x)$ over the integration domain, which is constant and, for simplicity, assumed to be 1.

$$\begin{aligned}\overset{\square}{\mathbf{M}}_B &= \int_{x_0}^{x_0+h} \Phi_B(x) \cdot (sx+t)\,\mathrm{d}x \\ &= \int_{x_0}^{x_0+h} sx + t\,\mathrm{d}x \\ &= \frac{h}{2}(hs + 2(sx_0 + t)) \end{aligned} \tag{13}$$

$$\begin{aligned}\overset{\square}{\mathbf{C}}_B &= \frac{1}{\mathbf{M}_B}\int_{x_0}^{x_0+h} \Phi_B(x)\cdot(sx+t)\cdot\overrightarrow{w}\cdot\left(\frac{x-x_0}{h}\right)\mathrm{d}x \\ &= \overrightarrow{w}\,\frac{1}{\frac{h}{2}(hs+2(sx_0+t))}\int_{x_0}^{x_0+h}(sx+t)\left(\frac{x-x_0}{h}\right)\mathrm{d}x \\ &= \overrightarrow{w}\,\frac{2}{h(hs+2(sx_0+t))}\frac{h}{6}(2hs+3(sx_0+t)) \\ &= \overrightarrow{w}\,\frac{2hs+3(sx_0+t)}{3hs+6(sx_0+t)} \end{aligned} \tag{14}$$

C.2 Mass

To demonstrate equality of our approach to the box-solution, we first gather all the tetrahedra-related components. Equation 15 lists the independent masses of the tetrahedra, using the assumption of the box's cross-section to have an area of 1. The volume of the individual tetrahedra in this configuration compute as $\mathbf{V}_{rgcm} = \frac{1}{6}$ and $\mathbf{V}_y = \frac{1}{3}$. They scale linearly if

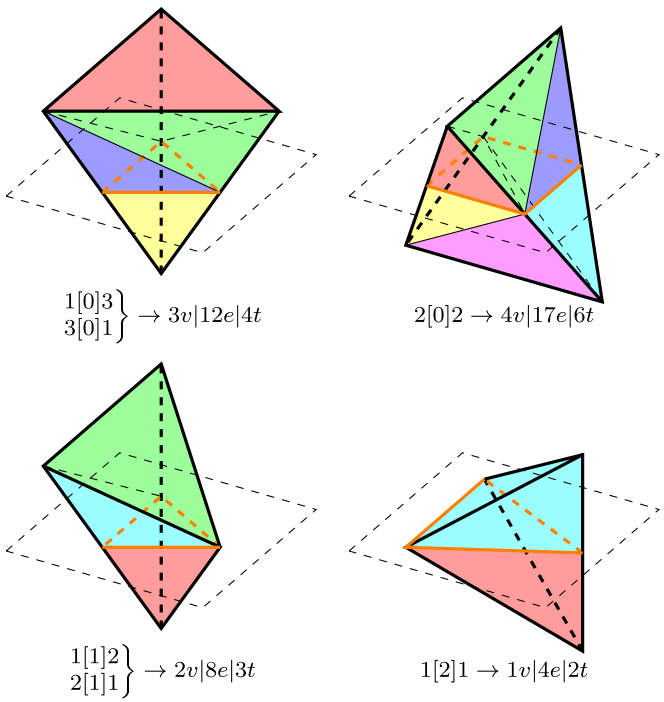

Fig. 15 Up to symmetry or ration, there are four cases, how tetrahedra are split by a plane. Resulting geometry is again tetrahedralizable. The left-hand numbers account for vertices separated by, or lying within the plane (in brackets). Right-hand numbers list the number of *new* vertices, edges and tets created by the split

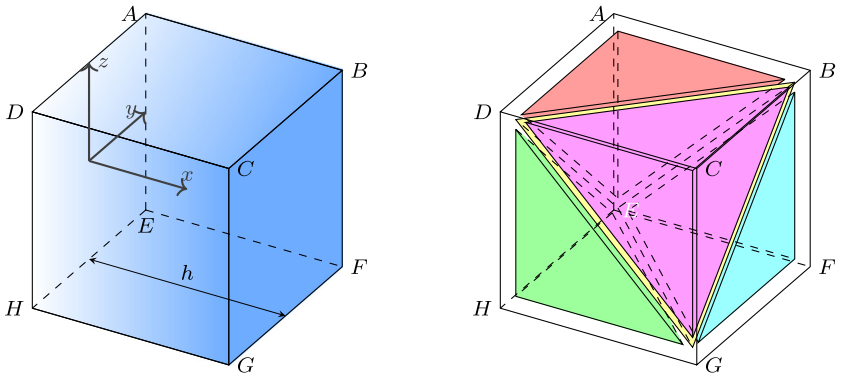

Fig. 16 To prove our concept, we construct an axis-aligned box and simple but general tetrahedralization. The five tetrahedra are assigned to the three integrable base-classes introduced in Fig. 17. Colors **r**ed, **g**reen, **c**yan, **m**agenta and **y**ellow correspond to the indices used in upcoming equations to refer to the individual tetrahedra

the box elongates along the x-axis, thus include the factor h. Further, we utilize the simplifications that $d_0 = sx_0 + t$ and $d_h = s(x_0 + h) + t$.

$$
\begin{aligned}
\mathbf{M}_r &= \mathbf{V}_r \frac{d_A + d_B + d_D + d_E}{4} &&= \frac{h}{24}(3d_0 + d_h) \\
\mathbf{M}_g &= \mathbf{V}_g \frac{d_D + d_E + d_G + d_H}{4} &&= \frac{h}{24}(3d_0 + d_h) \\
\mathbf{M}_c &= \mathbf{V}_c \frac{d_B + d_E + d_F + d_G}{4} &&= \frac{h}{24}(d_0 + 3d_h) \\
\mathbf{M}_m &= \mathbf{V}_m \frac{d_B + d_C + d_D + d_G}{4} &&= \frac{h}{24}(d_0 + 3d_h) \\
\mathbf{M}_y &= \mathbf{V}_y \frac{d_B + d_D + d_E + d_G}{4} &&= \frac{h}{24}4(d_0 + d_h) \qquad (15)
\end{aligned}
$$

$$
\begin{aligned}
\overset{\triangle}{\mathbf{M}}_B &= \mathbf{M}_r + \mathbf{M}_g + \mathbf{M}_c + \mathbf{M}_m + \mathbf{M}_y \\
&= \frac{h}{24}\left(2(3d_0 + d_h) + 2(d_0 + 3d_h) + 4(d_0 + d_h)\right) \\
&= \frac{h}{24}\left(12d_0 + 12d_h\right) \\
&= \frac{h}{2}\left(sx_0 + t + s(x_0 + h) + t\right) \\
&= \frac{h}{2}\left(hs + 2(sx_0 + t)\right) = \overset{\square}{\mathbf{M}}_B \qquad (16)
\end{aligned}
$$

□

Masses of the individual tetrahedra (Eq. 15) summed up (Eq. 16) prove equality to the integrated box-mass (Eq. 13).

Center of mass

In Eq. 14 the center of mass integral results as a scaled vector $\overrightarrow{w}$, parallel to the x-axis and scaled dependent on the density function. However, the center of mass for a general tetrahedron is formulated in Eq. 2 solely based on its vertices. To eventually express the equality of the box- and tetrahedralized solutions, we first reformulate the center of mass for tetrahedra in a similar $\overrightarrow{w}$ vector-dependent way. Therefore, we establish the three basic integrable tetrahedra cases shown in Fig. 17.

$$
\begin{aligned}
\mathbf{C}_r\mathbf{M}_r &= B\mathbf{M}_r + 3\overleftarrow{w}_r \cdot \overline{ADE}\frac{h}{3}\frac{hs + 5(sx_0 + t)}{20} \\
&= B\mathbf{M}_r + 3\overleftarrow{w}_r \cdot \frac{h}{120}(hs + 5(sx_0 + t)) \\
\mathbf{C}_g\mathbf{M}_g &= G\mathbf{M}_g + 3\overleftarrow{w}_g \cdot \overline{DHE}\frac{h}{3}\frac{hs + 5(sx_0 + t)}{20} \\
&= G\mathbf{M}_g + 3\overleftarrow{w}_g \cdot \frac{h}{120}(hs + 5(sx_0 + t)) \\
\mathbf{C}_c\mathbf{M}_c &= E\mathbf{M}_c + 3\overrightarrow{w}_c \cdot \overline{BFG}\frac{h}{3}\frac{4hs + 5(sx_0 + t)}{20} \\
&= E\mathbf{M}_c + 3\overrightarrow{w}_c \cdot \frac{h}{120}(4hs + 5(sx_0 + t)) \\
\mathbf{C}_m\mathbf{M}_m &= D\mathbf{M}_m + 3\overrightarrow{w}_m \cdot \overline{BCG}\frac{h}{3}\frac{4hs + 5(sx_0 + t)}{20} \\
&= D\mathbf{M}_m + 3\overrightarrow{w}_m \cdot \frac{h}{120}(4hs + 5(sx_0 + t)) \\
\mathbf{C}_y\mathbf{M}_y &= \frac{D + E}{2}\mathbf{M}_y + \widetilde{w}_y \cdot |u \times v|\frac{h}{6}\frac{3hs + 5(sx_0 + t)}{10} \\
&= \frac{D + E}{2}\mathbf{M}_y + \widetilde{w}_y \cdot \frac{4h}{120}(3hs + 5(sx_0 + t)) \qquad (17)
\end{aligned}
$$

We approach the center of mass analogously to the mass itself and first establish the component-wise results for the individual tetrahedra in Eq. 17, however, using the vector-based formulations expressed in Fig. 17.

$$
\begin{aligned}
\overset{\triangle}{\mathbf{C}}_B &= \frac{1}{\overset{\triangle}{\mathbf{M}}_B}\left(\mathbf{C}_r\mathbf{M}_r + \mathbf{C}_g\mathbf{M}_g + \mathbf{C}_c\mathbf{M}_c + \mathbf{C}_m\mathbf{M}_m + \mathbf{C}_y\mathbf{M}_y\right) \\
&= \frac{1}{\overset{\triangle}{\mathbf{M}}_B}\begin{pmatrix} \mathbf{M}_{rg}(B + G) + (3\overleftarrow{w}_r + 3\overleftarrow{w}_g) \cdot \frac{h}{120}(hs + 5d_0) \\ + \mathbf{M}_{cm}(E + D) + (3\overrightarrow{w}_c + 3\overrightarrow{w}_m) \cdot \frac{h}{120}(4hs + 5d_0) \\ + \quad \frac{D+E}{2}\mathbf{M}_y + \widetilde{w}_y \cdot \frac{4h}{120}(3hs + 5d_0) \end{pmatrix}
\end{aligned}
$$

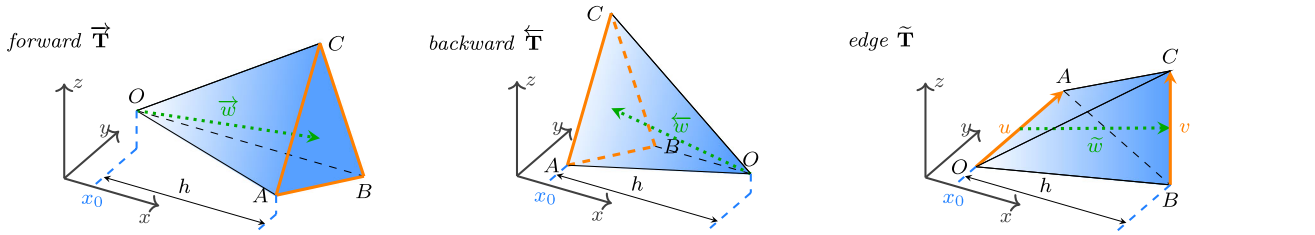

Fig. 17 The three basis classes of *integrable* tetrahedra, exemplary with the x-axis as the integration domain. The density gradient over the extent of h is visualized in blue. Faces and edges orthogonal to the integration dimension, are highlighted in orange. For them the density is constant. The green arrows indicate the center of mass vectors $\overrightarrow{w}$, $\overleftarrow{w}$ and $\widetilde{w}$. Equations are formulated using $\tau = \frac{x-x_0}{h}$

$$
\begin{aligned}
&= \frac{1}{\underset{\mathbf{M}_B}{\triangle}} \begin{pmatrix} \mathbf{M}_{rg} \cdot 2\overrightarrow{w} - 6\overrightarrow{w} \cdot \frac{h}{120}(hs + 5d_0) \\ + \quad \mathbf{M}_{cm}0 + 6\overrightarrow{w} \cdot \frac{h}{120}(4hs + 5d_0) \\ + \quad \frac{0}{2}\mathbf{M}_y + \overrightarrow{w} \cdot \frac{4h}{120}(3hs + 5d_0) \end{pmatrix} \\
&= \frac{1}{\underset{\mathbf{M}_B}{\triangle}} \overrightarrow{w} \begin{pmatrix} 2\mathbf{M}_{rg} - 6\frac{h}{120}(hs + 5d_0) \\ + \quad 6\frac{h}{120}(4hs + 5d_0) \\ + \quad 4\frac{h}{120}(3hs + 5d_0) \end{pmatrix} \\
&= \frac{1}{\underset{\mathbf{M}_B}{\triangle}} \overrightarrow{w} \begin{pmatrix} 5\frac{h}{60}(3d_0 + d_h) - 3\frac{h}{60}(hs + 5d_0) \\ + \quad 3\frac{h}{60}(4hs + 5d_0) \\ + \quad 2\frac{h}{60}(3hs + 5d_0) \end{pmatrix} \\
&= \overrightarrow{w}\,\frac{1}{\frac{h}{24}(12d_0 + 12d_h)}\,\frac{h}{60}\,(25d_0 + 5d_h + 15hs) \\
&= \overrightarrow{w}\,\frac{25d_0 + 5d_h + 15hs}{30d_0 + 30d_h} \\
&= \overrightarrow{w}\,\frac{25(sx_0 + t) + 5(s(x_0 + h) + t) + 15hs}{30(sx_0 + t) + 30(s(x_0 + h) + t)} \\
&= \overrightarrow{w}\,\frac{2hs + 3(sx_0 + t)}{3hs + 6(sx_0 + t)} = \overset{\square}{\mathbf{C}}_B
\end{aligned}
\tag{18}
$$

□

In Eq. 18 we sum up individual results from Eq. 17 and compact the term. Therefore, we use the facts that $\overleftarrow{\mathbf{M}}_r = \overleftarrow{\mathbf{M}}_g$ and $\overrightarrow{\mathbf{M}}_c = \overrightarrow{\mathbf{M}}_m$, respectively. Furthermore, we can combine mass center w-vectors and express them with the axis-aligned box-vector $\overrightarrow{w}$. The solution again proves equality to the box's mass center derived in Eq. 14.

References

1. Bächer, M., Whiting, E., Bickel, B., Sorkine-Hornung, O.: Spin-it: optimizing moment of inertia for spinnable objects. ACM Trans. Graph. (TOG) **33**(4), 96 (2014)
2. Basselin, J., Alonso, L., Ray, N., Sokolov, D., Lefebvre, S., Lévy, B.: Restricted power diagrams on the GPU. In: Eurographics 2021 (2021)
3. Byrd, R.H., Lu, P., Nocedal, J., Zhu, C.: A limited memory algorithm for bound constrained optimization. SIAM J. Sci. Comput. **16**(5), 1190–1208 (1995)
4. Du, Q., Emelianenko, M., Ju, L.: Convergence of the lloyd algorithm for computing centroidal Voronoi tessellations. SIAM J. Numer. Anal. **44**(1), 102–119 (2006)
5. D'Urso, M.: Gravity effects of polyhedral bodies with linearly varying density. Celest. Mech. Dyn. Astron. **120**(4), 349–372 (2014)
6. Eberly, D.: Polyhedral mass properties (revisited). Tech. Rep, Geometric Tools, LLC (2002)
7. Hansen, R.: An analytical expression for the gravity field of a polyhedral body with linearly varying density. Geophysics **64**(1), 75–77 (1999)
8. Hornus, S., Kuipers, T., Devillers, O., Teillaud, M., Martínez, J., Glisse, M., Lazard, S., Lefebvre, S.: Variable-width contouring for additive manufacturing. ACM Trans. Graph. **39**(Siggraph'2020 Conference proceedings) (2020)
9. Hu, Y., Zhou, Q., Gao, X., Jacobson, A., Zorin, D., Panozzo, D.: Tetrahedral meshing in the wild. ACM Trans. Graph. (TOG) **37**(4), 60 (2018)
10. Kuipers, T., Wu, J., Wang, C.C.: Crossfill: foam structures with graded density for continuous material extrusion. Computer-Aided Des. **114**, 37–50 (2019)
11. Leal, R., Barreiros, F., Alves, L., Romeiro, F., Vasco, J., Santos, M., Marto, C.: Additive manufacturing tooling for the automotive industry. Int. J. Adv. Manuf. Technol. **92**(5–8), 1671–1676 (2017)
12. Lévy, B., Liu, Y.: L p centroidal Voronoi tessellation and its applications. ACM Trans. Graph. (TOG) **29**, 119 (2010)
13. Li, B., Fu, J., Feng, J., Shang, C., Lin, Z.: Review of heterogeneous material objects modeling in additive manufacturing. Vis. Comput. Ind. Biomed. Art **3**(1), 1–18 (2020)
14. Lloyd, S.: Least squares quantization in PCM. IEEE Trans. Inf. Theory **28**(2), 129–137 (1982)
15. Lorensen, W.E., Cline, H.E.: Marching cubes: a high resolution 3D surface construction algorithm. ACM Siggraph Comput. Graph. **21**, 163–169 (1987)
16. Meagher, D.: Geometric modeling using octree encoding. Comput. Graph. Image Process. **19**(2), 129–147 (1982)
17. Mirtich, B.: Fast and accurate computation of polyhedral mass properties. J. Graph. Tools **1**(2), 31–50 (1996)
18. Musialski, P., Auzinger, T., Birsak, M., Wimmer, M., Kobbelt, L.: Reduced-order shape optimization using offset surfaces. ACM Trans. Graph. **34**(4), 102–1 (2015)
19. Online-Sources: Recent Gradient Infill Developements (2020). https://ultimaker.com/en/resources/52670-infill https://www.sublimelayers.com/2019/05/dynamic-infill-density-in-new-

kisslicer.html https://www.cnckitchen.com/blog/gradient-infill-for-3d-prints
20. Powell, M.J.: An efficient method for finding the minimum of a function of several variables without calculating derivatives. Comput. J. **7**(2), 155–162 (1964)
21. Prévost, R., Bächer, M., Jarosz, W., Sorkine-Hornung, O.: Balancing 3D models with movable masses. In: VMV (2016)
22. Prévost, R., Whiting, E., Lefebvre, S., Sorkine-Hornung, O.: Make it stand: balancing shapes for 3D fabrication. ACM Trans. Graph. (TOG) **32**(4), 81 (2013)
23. Rathod, H., Nagaraja, K., Venkatesudu, B.: Numerical integration of some functions over an arbitrary linear tetrahedra in Euclidean three-dimensional space. Appl. Math. Comput. **191**(2), 397–409 (2007)
24. Rathod, H., Rao, H.G.: Integration of polynomials over an arbitrary tetrahedron in Euclidean three-dimensional space. Comput. Struct. **59**(1), 55–65 (1996)
25. Ray, N., Sokolov, D., Lefebvre, S., Lévy, B.: Meshless Voronoi on the GPU. ACM Trans. Graph. (TOG) **37**(6), 265 (2019)
26. Si, H.: Tetgen, a delaunay-based quality tetrahedral mesh generator. ACM Trans. Math. Softw. (TOMS) **41**(2), 11 (2015)
27. Sokolov, D., Ray, N., Untereiner, L., Lévy, B.: Hexahedral-dominant meshing. ACM Trans. Graph. (TOG) **35**(5), 157 (2016)
28. Tonon, F.: Explicit exact formulas for the 3-D tetrahedron inertia tensor in terms of its vertex coordinates. J. Math. Stat. **1**(1), 8–11 (2004)
29. Wu, J., Aage, N., Westermann, R., Sigmund, O.: Infill optimization for additive manufacturing-approaching bone-like porous structures. IEEE Trans. Vis. Comput. Graph. **24**(2), 1127–1140 (2017)
30. Yan, D.M., Wang, W., LéVy, B., Liu, Y.: Efficient computation of clipped Voronoi diagram for mesh generation. Computer-Aided Des. **45**(4), 843–852 (2013)
31. Yan, Q., Dong, H., Su, J., Han, J., Song, B., Wei, Q., Shi, Y.: A review of 3D printing technology for medical applications. Engineering **4**(5), 729–742 (2018)
32. Zhu, C., Byrd, R.H., Lu, P., Nocedal, J.: Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization. ACM Trans. Math. Softw. (TOMS) **23**(4), 550–560 (1997)

Dennis R. Bukenberger received his MSc in computer science from the University of Tübingen in 2016. He is currently working as a PhD student at the Computer Graphics Department at the University of Tübingen. His research interests include geometry processing, automated abstraction, 3D reconstruction, and meshing.

Hendrik P. A. Lensch holds the chair for computer graphics at the University of Tübingen. He received his diploma in computers science from the University of Erlangen in 1999 and his PhD from Saarland University in 2003. He was a visiting assistant professor at Stanford University, USA, and head of an independent research group at the MPI Informatik. From 2009 to 2011, he was a full professor at the Institute for Media Informatics at Ulm University, Germany. His research interests include 3D appearance acquisition, computational photography, machine learning, global illumination and image-based rendering, and massively parallel programming.