## Appendix I: Proofs of Geometric Properties of Barycentric Dual

**Property 1:** Referring to Fig. 1a (same as Fig. 2a in the paper), each of the 6 resulting triangles of the BCS has the same area, namely 1/6 of the area of triangle $ABC$.

**Proof:** To start, observe that triangles $ABQ$ and $ACQ$ have the same area, since their bases $\overline{BQ}$ and $\overline{CQ}$ are of the same length, and they also have the same height (a segment coming from $A$ and perpendicular to $\overline{BC}$). By the same argument, triangles $BOQ$ and $COQ$ have the same area. Now we have $ABO = ABQ - BOQ$ and similarly $ACO = ACQ - COQ$, and thus $ABO = ACO$. Now we also have $APO = BPO = 1/2(ABO)$, and similarly $ARO = CRO = 1/2(ACO)$, but as seen $ABO = ACO$, and thus $ARO = CRO = 1/2(ABO) = APO = BPO$. In the same way we can conclude that the 6 resulting triangles have the same area (i.e., 1/6 of the area of triangle $ABC$).

**Property 2:** Each of the 4 vertices of a blue tetrahedron has the *same* volume weight, i.e., 1/4 of the volume of the blue tetrahedron.

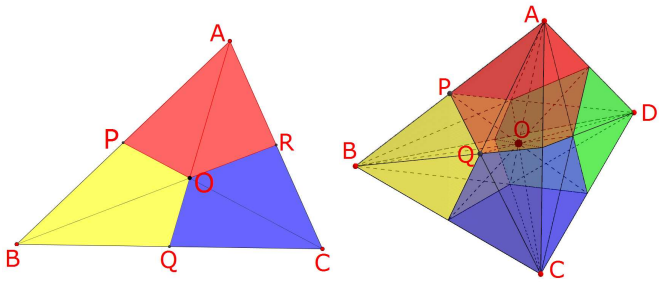
**Proof:** Recall that the BCS of a tetrahedron $S$ subdivides it into 24 tetrahedra/simplices of the same volume. Also, each vertex of $S$ has 6 such simplices incident to it (e.g., in Fig. 1b, $A$ has two such simplices coming from each of the faces $ABC, ABD, ACD$; one such simplex from $ABC$ is $APQO$) and thus gets the same contribution of the volume weight.

**Property 3:** Each of the 6 vertices of a red octahedron has the *same* volume weight, i.e., 1/6 of the volume of the red octahedron.

**Proof:** Refer to Fig. 2. The octahedron $ABCDEF$ has $O$ as the barycenter. In the BCS, there are 8 simplices incident to $A$ (darker red in Fig. 2); two of them are incident to the face $ABC$. Let $R$ be the center of the face $ABC$, and $P, Q, U$ the midpoints of edges $\overline{AC}, \overline{AB}$ and $\overline{BC}$. Recall that the BCS subdivides the face $ABC$ into 6 triangles of equal area (Property 1). In addition, each such triangle $T$, together with $O$, forms a simplex of the *same volume*, since the height from base $T$ is the same, i.e., the distance from $O$ to the face $ABC$. Therefore, the 6 simplices $APRO, AQRO$ (incident to $A$), $BQRO, BURO$ (incident to $B$), $CPRO, CURO$ (incident to $C$) all have the same volume, and each of $A, B, C$ gets two of them to contribute to the volume weight — in other words, $A, B, C$ each gets the *same volume contribution* from $ABC$. Note that faces $ABC$ and $DEF$ are *congruent and symmetric* in octahedron $ABCDEF$, and thus the 6 simplices from $ABC$ (as shown above) and the 6 simplices from $DEF$ (formed in the same way) are all of the same volume (since the distances from $O$ to $DEF$ and to $ABC$ are the same due to symmetry). Therefore, each of $D, E, F$ also gets the *same volume contribution* as each of $A, B, C$ from the *symmetric pair* $(ABC, DEF)$. In fact, there are four symmetric pairs of faces: $(ABC, DEF), (ABF, DEC), (AEF, DBC)$, and $(ACE, DFB)$. From $(ABC, DEF)$, each of the 6 vertices of the red octahedron $ABCDEF$ gets the *same* volume contribution, say $V_1$, and similarly from $(ABF, DEC)$, each of the 6 vertices gets the *same* volume contribution, say $V_2$, etc., and thus at the end each of the 6 vertices has the same volume weight, i.e., 1/6 of the volume of $ABCDEF$.

## Appendix II: Contour Spectrum: Accurate and Simplified Formula

*Contour spectrum* [1] gives, for each scalar-field tetrahedral cell, the accurate function that maps each isovalue to its isosurface area within the cell. This is a B-spline function defined over the scalar-value range of the cell.




(a) (b)

Fig. 1: Examples of barycentric subdivisions (BCS) for simplices: (a) 2D case; (b) 3D case. (This is Fig. 2 in the paper.)

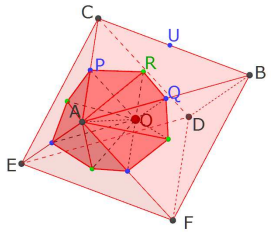



Fig. 2: Barycentric subdivision for the internal octahedron; only the 8 simplices incident to $A$ are shown (in darker red).

Consider the tetrahedron in Fig. 3(left), where the scalar values at vertices are sorted as $\mathscr{F}(v_1) \le \mathscr{F}(v_2) \le \mathscr{F}(v_3) \le \mathscr{F}(v_4)$. The shaded areas are isosurfaces. The area of isosurface with isovalue $\mathscr{F}(v_2)$ and the one with isovalue $\mathscr{F}(v_3)$ are denoted as $L(\mathscr{F}(v_2))$ and $L(\mathscr{F}(v_3))$, which can be easily computed. We then compute the control polygon $C'PD'$ followed by the quadratic spline function.

The overall spline function consists of three B-spline basis functions, each defined with the recurrence below:

$$N_{i,0}(t) = \begin{cases} 1 & \text{if } t_i \le t \le t_{i+1} \\ 0 & \text{otherwise} \end{cases}$$
$$N_{i,k}(t) = N_{i,k-1}(t)\frac{t - t_i}{t_{i+k} - t_i} + N_{i+1,k-1}(t)\frac{t_{i+k+1} - t}{t_{i+k+1} - t_{i+1}} \quad \text{for } k \ge 1. \tag{1}$$

In our case, $k = 2$ and $t_0, \ldots, t_5$ are as shown in Fig. 3. Let $B_1 = N_{0,2}$, $B_2 = N_{1,2}$, and $B_3 = N_{2,2}$. The control points are $C'(x_1, y_1)$, $P(x_2, y_2)$, $D'(x_3, y_3)$ where their actual coordinates are as given in Fig. 3. The overall function $f(t)$ is obtained by blending the three basis functions as follows:

$$f(t) = \begin{cases} B_1(t) \cdot y_1 & t \in [t_0, t_1] \\ B_1(t) \cdot y_1 + B_2(t) \cdot y_2 & t \in [t_1, t_2] \\ B_1(t) \cdot y_1 + B_2(t) \cdot y_2 + B_3(t) \cdot y_3 & t \in [t_2, t_3] \\ B_2(t) \cdot y_2 + B_3(t) \cdot y_3 & t \in [t_3, t_4] \\ B_3(t) \cdot y_3 & t \in [t_4, t_5] \end{cases} \tag{2}$$

**Remark.** As mentioned, the formula given in the original paper [1] was slightly imprecise. Not much detail was given there, and it seems to imply using $BPE$ as the control polygon rather than $C'PD'$. Note that $B$ and $E$ both have $y$-values $= 0$, i.e., $y_1 = y_3 = 0$ (Fig. 3). This results in $f(t) = 0$ for $t \in [t_0, t_1]$ or $t \in [t_4, t_5]$ (see Equation (2)); this is not correct since $f(t) > 0$ for $t \in [t_0, t_5]$ except for the endpoints where $f(t_0) = f(t_5) = 0$ (see the top figures in Fig. 3, in particular the function value $f(t)$ with $t \in (t_0, t_1)$).

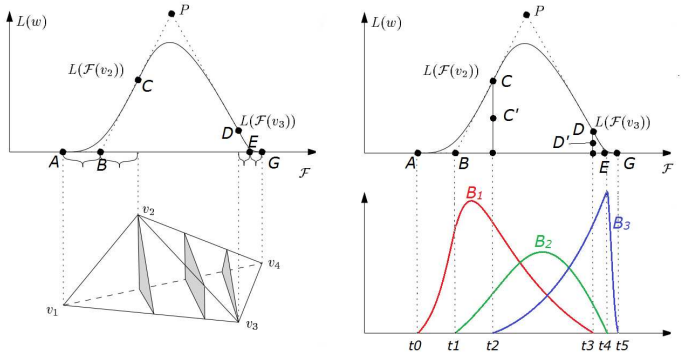



Fig. 3: Computing the B-spline function for a single tetrahedral cell. $A = (\mathscr{F}(v_1), 0); B = (\frac{\mathscr{F}(v_1)+\mathscr{F}(v_2)}{2}, 0)$; $C = (\mathscr{F}(v_2), L(\mathscr{F}(v_2)))$; $C' = (\mathscr{F}(v_2), \frac{L(\mathscr{F}(v_2))}{2})$; $D = (\mathscr{F}(v_3), L(\mathscr{F}(v_3)))$; $D' = (\mathscr{F}(v_3), \frac{L(\mathscr{F}(v_3))}{2})$; $E = (\frac{\mathscr{F}(v_3)+\mathscr{F}(v_4)}{2}, 0)$; $G = (\mathscr{F}(v_4), 0)$; $P = (\frac{\mathscr{F}(v_2)+\mathscr{F}(v_3)}{2}, Lp)$.

**Simplification.**

We observe that as we move the isosurface triangle from isovalue $\mathscr{F}(v_1) = t_0$ to $\mathscr{F}(v_2) = t_2$, the isovalue $t$ interpolates linearly and the triangle side length varies linearly (see Fig. 3(left)), and thus the triangle area varies quadratically from 0 to $2y_1$. Therefore we have $f(t) = (\frac{t-t_0}{t_2-t_0})^2 \cdot 2y_1$ for $t \in [t_0, t_2]$. Similar situation occurs when we move from $t_5$ to $t_3$. Overall we can simplify $f(t)$ as follows:

$$f(t) = \begin{cases} (\frac{t-t_0}{t_2-t_0})^2 \cdot 2y_1 & t \in [t_0, t_2] \\ B_1(t) \cdot y_1 + B_2(t) \cdot y_2 + B_3(t) \cdot y_3 & t \in [t_2, t_3] \\ (\frac{t_5-t}{t_5-t_3})^2 \cdot 2y_3 & t \in [t_3, t_5] \end{cases} \quad (3)$$

After the simplification, the number of intervals on which $f(t)$ is defined is reduced from 5 to 3, and thus our work of computation/integration is reduced.

**Appendix III: The Neighborhood Box Size**

Recall from Sec. 3 that we have a user-defined parameter $t$ for the neighborhood box size. Observe that too small $t$ could fail to reflect the salient features in the neighborhood, and too large $t$ could "average out" features with non-features. Here we discuss how to choose a reasonable $t$.

We suggest to compute an estimated average edge length in the mesh for $t$. For regular grids, this is 1. For tetrahedral meshes, recall that the volume of a tetrahedron with vertices $v_0$, $v_1$, $v_2$, $v_3$ is

$$\frac{1}{6}|(v_1 - v_0) \cdot ((v_2 - v_0) \times (v_3 - v_0))|,$$

namely, it is 1/6 of the volume of any parallelepiped that shares three converging edges with the tetrahedron. Therefore, we compute an average edge length $l_1$ in terms of the volume as $l_1 = \sqrt[3]{\frac{\text{Total Volume}}{n} \cdot 6}$, where Total Volume is the sum of the volumes of the $n$ tetrahedral cells in the mesh. For each cell we also compute its longest edge. We then take the average, $l_2$, of such longest edges from all cells. Finally, we choose $t = \min\{l_1, l_2\}$.

Usually we have $l_1 < l_2$ (and $t = l_1$) for most datasets. But for some datasets the embedded objects are extremely small compared to the whole volume size. The cell sizes in the surrounding space are very large, while the mesh resolution near the object is extremely high and thus the cell edges are very small. In this case we have $l_2 < l_1$ and $t = l_2$.

As for curvilinear grids, we use the same formula, except that $l_1 = \sqrt[3]{\text{Average Cell Volume}}$, where for each hexahedral cell we decompose it into 5 tetrahedra, compute and sum their volumes globally to get the global total volume of the mesh, and then divide by the number of hexahedral cells to get the Average Cell Volume.

**Appendix IV: Details on the KD-tree Operations**

We discuss how to build the KD-tree $T$ and perform range queries on $T$. Initially the root of $T$ is associated with the entire volume and contains all mesh vertices. We subdivide the current node $u$ and its domain, by an axis-aligned plane through a median coordinate of the current cut-dimension (but avoid hitting any vertex), so that each side of the cut has (roughly) the same number of vertices and the two resulting sub-domains become the two children of $u$, which are continued to be cut recursively until the current node has a constant number of vertices; such vertices are then stored in the current node (leaf). In the process, the cut-dimension cyclically rotates among the $x$-, $y$-, and $z$-dimensions at consecutive levels. Given an axis-aligned query box $Q$, we can perform a range query on $T$ to find all vertices contained in $Q$. The search starts from the root. If $Q$ does not intersect the cutting plane of the current node, then the search goes recursively to the child that $Q$ lies in; if $Q$ intersects the cutting plane, then the search goes on both children recursively. At the end, $Q$ reaches several leaves and we test $Q$ against the vertices stored in these leaves.

**Appendix V: Complexity Analysis of Our Algorithm**

We analyze the running time and space complexity of the algorithm *cell sampling with sweeping* in Sec. 3.1.C. Let $N$ be the number of mesh vertices (which is also the number of neighborhood boxes), $cN$ the number of mesh cells, $S$ the number of sample points per cell, $t$ the neighborhood box size, $M$ the average number of potentially intersecting cells for each neighborhood box ($M$ is proportional to $t^3$; the average number of sample points per neighborhood box is $MS$), and $B$ the number of histogram bins. Also, let $\alpha \in [0,1]$ be the average fraction of the fully contained cells per neighborhood box, and $1-\alpha$ the average fraction of the potentially partially intersected cells per neighborhood box.

The initial sorting before sweeping takes $O(N \log N)$ time. Each KD-tree query takes $O(\log N + K)$ time to report $K$ vertices[1], where the KD-tree also takes $O(N \log N)$ time to build and uses $O(N)$ space. For each of the $cN$ cells, we use the KD-tree to find the $M/c$ potentially intersecting neighborhood boxes (discussed next) on an average ($O(\log N + M/c)$ time per cell, with $K = M/c$), apply Contour Spectrum at the current cell ($O(B)$ time each), generate $S$ sample points in the current cell ($O(S)$ time each), and add contribution to the histograms of the $M/c$ boxes. (The $M/c$ is an estimated average number: $N$ vertices correspond to $cN$ cells in the mesh, and each neighborhood box (per vertex) corresponds to $M$ potentially intersecting cells. So on an average, each cell corresponds to $(NM)/(cN) = M/c$ potentially intersecting neighborhood boxes.) The overall running time is $O(cN(\log N + M/c + B + S) + NF(B))$, where $F(B)$ is the time

1. The worst-case query time for the KD-tree is $O(N^{1-1/d} + K)$ in $d$ dimensions [2] (here $d = 3$). However, such worst-case bound is obtained by making the query box to span the whole $xy$-plane but with a very small $z$-span, so that we must visit $\Theta(N^{2/3})$ nodes but find nothing to report. When the query box is relatively small and has a reasonable aspect ratio, as in our case, the query time is $O(\log N + K)$ [2].

to compute the statistic function value from a histogram with $B$ bins. (For both entropy and standard deviation $F(B)$ is $O(B)$.)

As for the space, $O(N)$ is needed for the KD-tree, and we also need the space for the currently active neighborhood boxes. On an average, $O(N^{2/3}(M/c)^{1/3})$ active neighborhood boxes are close to the sweeping plane[2], each needing $O(B)$ space for the $B$ histogram bins. Therefore the total space is $O(N^{2/3}(M/c)^{1/3}B+N)$ on an average. Typically $N$ is the dominating term and thus the overall space is $O(N)$. Note that if we do not use sweeping, then we can avoid the initial sorting time but the overall running time is still asymptotically the same. However, $O(NB)$ space would be needed for all $N$ histograms ($O(B)$ bins each), and thus the overall space would be $O(NB)$. Therefore, using sweeping reduces the space from $O(NB)$ to $O(N)$. This $O(N)$ space can be viewed as optimal since we need to use $O(N)$ space to store the statistical values (e.g., entropy values) of all vertices.

**Appendix VI: Additional Images**

In Fig. 4 we show the results of direct volume rendering on the local entropy/standard deviation (SD) as a scalar field for additional datasets; see Sec. 4.3 of the paper for more details.

## REFERENCES


[1] C. Bajaj, V. Pascucci, and D. Schikore. The contour spectrum. In *Proc. IEEE Visualization Conference (Vis '97)*, pp. 167–173, 1997.

[2] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 3rd edition, 2008.

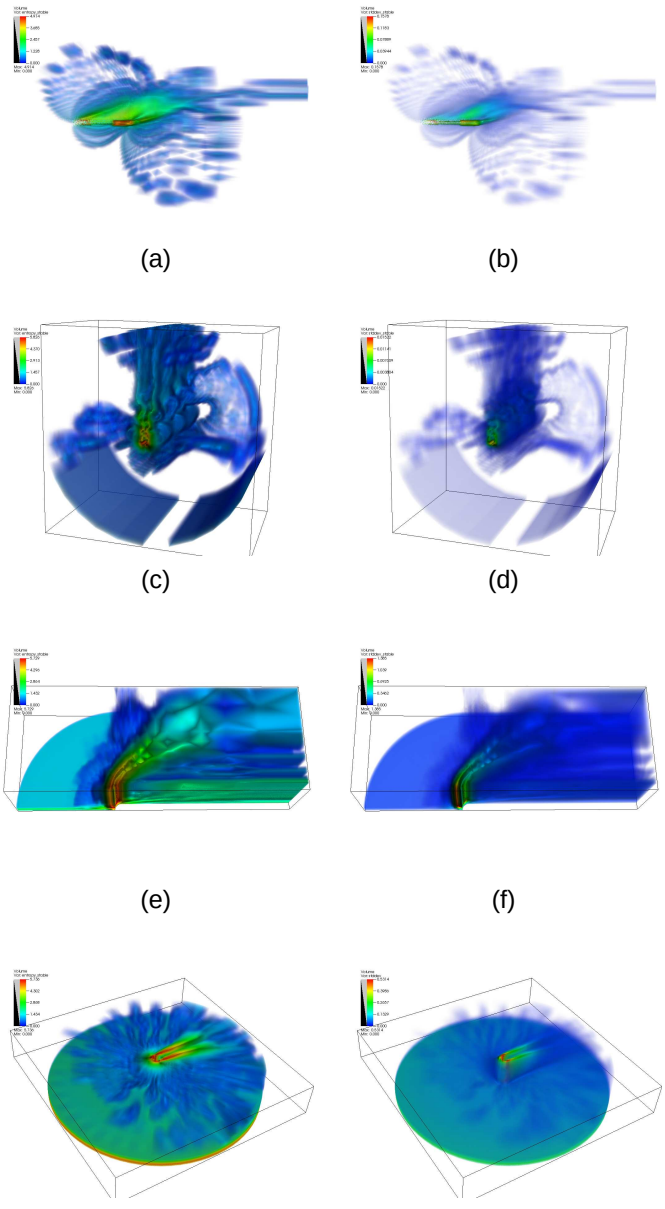




Fig. 4: Direct volume rendering on the local entropy/standard deviation (SD) as a scalar field: **(a) (b)**: delta, **(c) (d)**: Tpost, **(e) (f)**: blunt, **(g) (h)**: post. The left column is for local entropy field, and the right column is for local SD field.

2. The active region is given by the sweeping plane with thickness being the cell thickness; on an average, the former covers $N^{2/3}$ vertices (neighborhood boxes), and the latter corresponds to $(M/c)^{1/3}$ potentially intersecting neighborhood boxes.