

Extending editing capabilities of subdivision schemes by refinement of point-normal pairs

Evgeny Lipovetsky^{*†,‡}

Nira Dyn[§]

July 13, 2021

Abstract

In this paper we extend the 2D circle average of [11] to a 3D binary average of point-normal pairs, and study its properties. We modify classical surface-generating linear subdivision schemes with this average obtaining surface-generating schemes refining point-normal pairs. The modified schemes give the possibility to generate more geometries by editing the initial normals. For the case of input data consisting of a mesh only, we present a method for computing "naive" initial normals from the initial mesh. The performance of several modified schemes is compared to their linear variants, when operating on the same initial mesh, and examples of the editing capabilities of the modified schemes are given. In addition we provide a link to our repository, where we store the initial and refined mesh files, and the implementation code. Several videos, demonstrating the editing capabilities of the initial normals are provided in our Youtube channel.

Keywords: surface-generating subdivision refining 3D point-normal pairs, 3D circle average, surface design.

1 Introduction

In a previous paper [11] we introduced a weighted binary average of two 2D point-normal pairs (PNPs), termed *circle average*, and defined subdivision schemes based on it, which refine PNPs in 2D, and generate curves. This paper presents a method for extending the 2D circle average to 3D, a method applicable to any 2D weighted binary average. The extension of the 2D circle average to 3D preserves several important properties of the 2D circle average, and even extends some of them.

With the 3D circle average we modify classical linear surface-generating subdivision schemes refining points, to surface-generating schemes refining PNPs. Our methodology in modifying linear schemes, is the same as that in the 2D case: we first write the refinement rules of a converging linear scheme in terms of

^{*}Corresponding author

[†]The contribution of Evgeny Lipovetsky is part of his Ph.D. research conducted at Tel-Aviv University.

[‡]evgenyl@post.tau.ac.il, School of Computer Sciences, Tel-Aviv Univ., Israel

[§]niradyn@post.tau.ac.il, School of Mathematical Sciences, Tel-Aviv Univ., Israel

repeated weighted linear binary averages, and then replace each weighted linear binary average by the 3D circle average with the same weight. As in the case of the modified 2D schemes [11], our new 3D schemes also enrich the variety of geometries that can be generated, just by editing the initial normals.

1.1 Contribution

With subdivision schemes refining points, the only way to alter the geometry generated from a given initial mesh is to change the location of the vertices of the mesh. The schemes we design in this work refine point-normal pairs. Such a scheme can generate a richer variety of geometries by editing also the initial normals.

With our approach, we can modify any convergent linear scheme refining points into a scheme refining PNPs. We can enrich any subdivision-based software system by establishing new editing tools.

1.2 Outline

Sections 3, 4 present the extension to 3D of the 2D circle average, and asserts properties of the 3D circle average, in particular the Consistency Property, which makes the weighted binary operation an average. In Section 5 we explain our methodology for modifying linear subdivision schemes refining points to schemes refining PNPs. We give a method (algorithm) to write a weighted linear average of several elements in terms of repeated weighted linear binary averages. We also present a method for defining "naive" normals from a mesh when initial normals are not given. In Section 6 four modified surface-generating schemes are discussed. Their performance on two initial meshes is compared with that of their linear counterparts. The editing capability of the modified schemes is demonstrated by examples presented in three videos and a figure of three snapshots from one of the videos. A link to the videos is given. A short discussion of the implementation consists of Section 7. The paper ends in Section 8 with Conclusions and Future Work.

2 Related work

Classical surface-generating linear subdivision schemes refine points given at the vertices of a mesh [18]. In recent years more involved data at the vertices of a mesh are refined by subdivision schemes, such as by Hermite schemes (see e.g. [16]), and by manifold-valued schemes (see e.g. [19]). The information in 3D PNPs is less than that in 3D data refined by Hermite schemes. In the latter case the data is "linear" and is refined by a linear scheme, while the unit normals of a surface is a nonlinear functional applied to the surface (it is the direction of the cross-product of two tangents). Indeed the circle average and the modified schemes based on it are nonlinear.

Previous works on subdivision schemes that refine PNPs are based on a binary operation between two PNPs which is used to define the insertion rule in an interpolatory scheme [3], [1].

There are various papers using the same methodology as ours in adapting linear refining-points subdivision schemes to schemes refining other types of

geometric objects. For example: manifold-valued data (see e.g. [8]), sets in \mathbb{R}^d (see e.g. [9]), and nets of functions (see e.g. [4]).

3 The circle average

In this section we introduce an extension of the 2D *circle average* - a weighted binary average of two 2D point-normal pairs (PNPs) - to an average of two 3D PNPs. First we recall the definition of the 2D circle average.

3.1 The circle average in 2D

Given two PNPs in 2D, each consisting of a point and a normal unit vector, $P_0 = (p_0, n_0)$, $P_1 = (p_1, n_1)$, and a real weight $\omega \in [0, 1]$, the circle average produces a new PNP $P_\omega = P_0 \odot_\omega P_1 = (p_\omega, n_\omega)$.

The point p_ω is on an auxiliary arc $\widehat{P_0 P_1}$, at arc distance $\omega\theta$ from p_0 , where θ is the angle between n_0 and n_1 . The normal n_ω is the geodesic average of n_0 and n_1 . For the definition of $\widehat{P_0 P_1}$ and for more details consult [11].

3.2 Construction of the circle average in 3D

All the objects mentioned in the rest of the paper, specifically points and vectors, are in 3D, if not stated otherwise.

First, we introduce some notation. For two vectors u, v , with $u \times v \neq 0$, $z(u, v)$ denotes the normalized vector in direction $u \times v$. Note that

$$z(\alpha u + \beta v, \gamma u + \delta v) = z(u, v), \text{ for } \alpha, \beta, \gamma, \delta \in \mathbb{R}, \text{ s.t. } \alpha^2 + \beta^2 > 0, \gamma^2 + \delta^2 > 0. \quad (1)$$

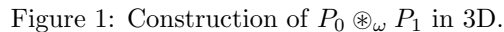
For a point p and a vector n , let $\Pi(p, n)$ be the plane which passes through the point p and has the normal n .

For $P_0 = (p_0, n_0)$ and $P_1 = (p_1, n_1)$, two PNPs to be averaged, we consider the two parallel planes $\Pi_0 = \Pi(p_0, z(n_0, n_1))$, $\Pi_1 = \Pi(p_1, z(n_0, n_1))$. The length of the projection of $[p_0, p_1]$ on $z(n_0, n_1)$, which is the distance between Π_0 and Π_1 , is denoted by h . We define Π_ω to be the plane parallel to Π_0, Π_1 , which is at distance ωh from Π_0 towards Π_1 . We say that $P = (p, n)$ belongs to a plane Π , $P \in \Pi$, if both p and n are in Π .

Let $P_\omega = P_0 \otimes_\omega P_1$ denote the circle average in 3D. The construction of $P_\omega = (p_\omega, n_\omega)$ is done by the following procedure. (See Fig. 1 for an example.)

- (i) Project p_1 on Π_0 to obtain p_1^* . Note that $P_1^* = (p_1^*, n_1) \in \Pi_0$, and that also $P_0 \in \Pi_0$.
- (ii) Compute the 2D circle average $P_0 \odot_\omega P_1^*$ in a local coordinate system in Π_0 , and convert the 2D result to a PNP $P_\omega^* = (p_\omega^*, n_\omega)$ in 3D.
- (iii) Project p_ω^* on Π_ω . Obtain $P_\omega = (p_\omega, n_\omega)$.

Note that if $P_0, P_1 \in \Pi_0$ then the 3D average reduces to the 2D average. As in the 2D case, the construction is not defined when $\theta = \pi$. Although, when $\theta = 0$, the direction z is not defined, the 3D average can be obtained by continuity as shown in Section 4.3.



4 Properties of the 3D circle average

4.1 Consistency

$$(P_0 \circledast_t P_1) \circledast_k (P_0 \circledast_s P_1) = P_0 \circledast_{\omega^*} P_1, \quad \omega^* = ks + (1-k)t. \quad (2)$$

4.2 Helix trace

4

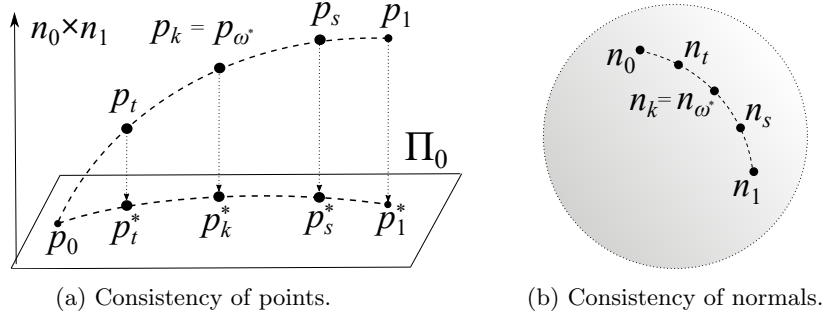


Figure 2: Consistency property.

Note that z and $\widehat{P_0 P_1^*}$ are the same for the circle average with weights t, s, k .

4.3 Limit cases of the circle average

The investigation of several limit cases of the circle average requires the next lemma. The proof of this lemma is straightforward but we give it for the convenience of the reader.

Lemma 4.1. The intersection point x_ω of $[p_0, p_1]$ and Π_ω is given by

$$x_\omega = (1 - \omega)p_0 + \omega p_1. \quad (3)$$

Proof. Figure 3 illustrates the proof. Let $\widehat{p}_0, \widehat{p}_1$ be the projections of p_0, p_1 on Π_ω , and let $\widehat{\Pi} = \Pi(p_0, \widehat{n})$ where \widehat{n} is the normalized vector $\overrightarrow{p_0 p_1} \times \overrightarrow{p_0 \widehat{p}_1}$. Note that $\widehat{\Pi}$ contains the segments $[p_0, p_1]$ and $[\widehat{p}_0, \widehat{p}_1]$. Since $[\widehat{p}_0, \widehat{p}_1]$ is in Π_ω then $[\widehat{p}_0, \widehat{p}_1]$ is contained in the intersection line of $\widehat{\Pi}$ and Π_ω . Moreover, by the definition of $\widehat{p}_0, \widehat{p}_1$, $[\widehat{p}_0, \widehat{p}_1]$ contains all the projections of points of $[p_0, p_1]$ on Π_ω , in particular $x_\omega \in [\widehat{p}_0, \widehat{p}_1]$. Thus the two triangles $\triangle p_0 \widehat{p}_0 x_\omega$ and $\triangle p_1 \widehat{p}_1 x_\omega$ are in $\widehat{\Pi}$. These triangles are similar, having equal angles. Therefor $\frac{|p_0 x_\omega|}{|p_1 x_\omega|} = \frac{|p_0 \widehat{p}_0|}{|p_1 \widehat{p}_1|} = \frac{\omega}{1-\omega}$, which proves (3). \square

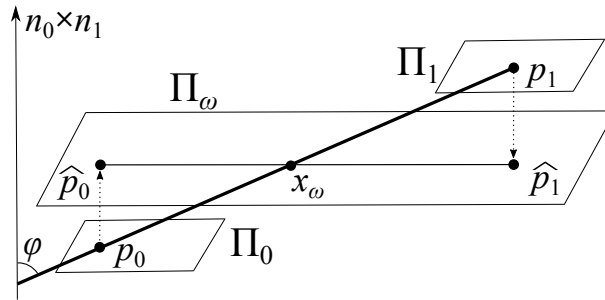


Figure 3: The setup of Lemma 4.1. The intersection point of $[p_0, p_1]$ and Π_ω is the linear average of p_0 and p_1 with weight ω .

Two basic properties of the 2D circle average are that it is not defined for $\theta = \pi$, and that its point tends to the 2D linear average when $\theta \rightarrow 0$. A similar

behavior holds in 3D. Furthermore, in the 3D case, there are two parameters, θ and the angle φ between $n_0 \times n_1$ and $\overrightarrow{p_0 p_1}$ (see Figure 3). Note that $\theta \in [0, \pi]$ and $\varphi \in [0, \pi]$.

Next we show that the point of $P_0 \odot_\omega P_1$ tends to $x_\omega = (1 - \omega)p_0 + \omega p_1$ as either $\theta \rightarrow 0$ or $\varphi \rightarrow 0$ (or $\varphi \rightarrow \pi$).

First we analyze the case $\varphi \rightarrow 0$ or $\varphi \rightarrow \pi$ for fixed $\theta \geq 0$.

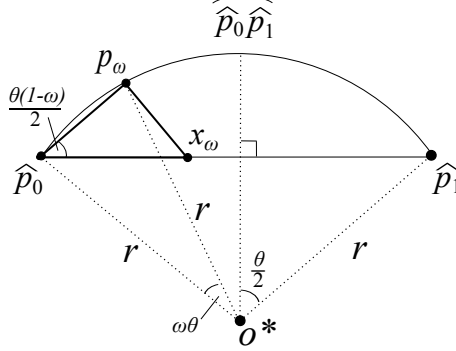


Figure 4: The triangle $\triangle \widehat{p}_0 p_\omega x_\omega$.

Using the geometry as depicted in Figure 4, we get $|\widehat{p}_0 p_\omega| = |\widehat{p}_0 \widehat{p}_1| \frac{\sin(\frac{\theta\omega}{2})}{\sin(\frac{\theta}{2})}$.

We express $|p_\omega x_\omega|$ by the cosine theorem in the triangle $\triangle \widehat{p}_0 p_\omega x_\omega \subset \Pi_\omega$,

$$\begin{aligned} |p_\omega x_\omega|^2 &= (|\widehat{p}_0 \widehat{p}_1| \omega)^2 + \left(|\widehat{p}_0 \widehat{p}_1| \frac{\sin(\frac{\theta\omega}{2})}{\sin(\frac{\theta}{2})} \right)^2 \\ &\quad - 2|\widehat{p}_0 \widehat{p}_1|^2 \omega \frac{\sin(\frac{\theta\omega}{2})}{\sin(\frac{\theta}{2})} \cos\left(\frac{\theta(1-\omega)}{2}\right). \end{aligned} \quad (4)$$

Since $|\widehat{p}_0 \widehat{p}_1| = |p_0 p_1| \sin \varphi$, all the terms in the right side of (4) tend to zero, independently of the value of θ . Thus, $p_\omega \rightarrow x_\omega$, as $\varphi \rightarrow 0$ (or $\varphi \rightarrow \pi$).

In case $\theta \rightarrow 0$ and $0 < \varphi < \pi$, the right side of (4) is zero because

$$\lim_{\theta \rightarrow 0} \frac{\sin(\frac{\theta\omega}{2})}{\sin(\frac{\theta}{2})} = \omega. \quad (5)$$

4.4 Preservation of special geometries

In [11] it is shown that if P_0, P_1 are sampled from a circle then $P_0 \odot_\omega P_1 = (p_\omega, n_\omega)$ corresponds to a point on this circle with n_ω the normal of the circle at p_ω . We say that the 2D circle average "preserves circles". This property extends in the case of the 3D circle average to "preservation of spheres and cylinders". Indeed, any two PNPs sampled from a sphere are also samples from the big circle C , determined by the two points and the center of the sphere. Thus, the circle average of the two PNPs is the 2D circle average of the two PNPs sampled from C , implying that the 3D circle average "preserves" spheres. See Figure 5a for an example.

Next, consider the case that the two PNPs P_0, P_1 are sampled from a cylinder of the form $x = \cos t, y = \sin t, z = t, 0 \leq t \leq \pi$. Note that $z(n_0, n_1)$ is the axis

of the cylinder, and that $H(P_0, P_1)$ is on the cylinder, and its projection on Π_0 is $\widehat{P_0 P_1^*}$. Due to the "circle preservation" of the 2D circle average, $P_0 \odot_\omega P_1^* = (p_\omega^*, n_\omega)$ corresponds to a PNP sampled from $\widehat{P_0 P_1^*}$. By (iii) of the construction of the 3D circle average, p_ω is on $H(P_0, P_1)$ and the normal to $H(P_0, P_1)$ at this point is n_ω . Thus the 3D circle average "preserves" cylinders. See Figure 5b for an example.

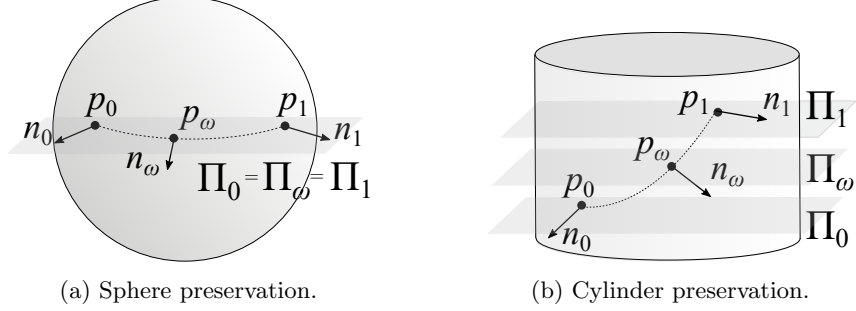


Figure 5: Preservation of special geometries.

5 Modified subdivision schemes

We consider subdivision schemes refining point-normal pairs, which are obtained from converging linear subdivision schemes. To obtain these schemes we express the linear schemes in terms of repeated, weighted, linear, binary averages of points and replace these averages by the 3D circle average. In this paper we term the so obtained schemes "Modified schemes".

We first explain how we rewrite the refinement rules of any linear converging subdivision scheme in terms of repeated, weighted, linear, binary averages. Then we mention four classical surface-generating linear schemes to be modified. The performance of these schemes and their modifications is tested in the next section. Finally we provide a method which computes initial normals, if normals are not given as input.

5.1 Repeated binary averaging

Here we propose a method for rewriting a weighted linear average of several points in terms of repeated binary averages. Consider computing a point q as a weighted average of points $\{p_i\}_{i=0}^{k-1}$, i.e.

$$q = \alpha_0 p_0 + \alpha_1 p_1 + \dots + \alpha_{k-1} p_{k-1}, \quad (6)$$

where $\alpha_i \in \mathbb{R}, \alpha_i \neq 0, i = 0, \dots, k-1$, and

$$\sum_{i=0}^{k-1} \alpha_i = 1. \quad (7)$$

We rewrite (6) as

$$q = (\alpha_0 + \alpha_1) \left(\frac{\alpha_0}{\alpha_0 + \alpha_1} p_0 + \frac{\alpha_1}{\alpha_0 + \alpha_1} p_1 \right) + \alpha_2 p_2 + \dots + \alpha_{k-1} p_{k-1}, \quad (8)$$

reducing by one the number of elements in the outer sum and obtaining a binary average as the first term. Note that in (8) q is a linear average of $k - 1$ points while in (6) q is a linear average of k points. We repeat this step $k - 2$ times and obtain the weighted average (6) written as a sequence of $k - 1$ repeated binary averages.

To avoid division by zero in this process, we have to guarantee that

$$\sum_{i=0}^{\ell} \alpha_i \neq 0, \quad \ell = 1, \dots, k - 1.$$

We reorder the terms in (6) such that all positive α_i precede all the negative ones. With this reordering, (7) guarantees that each partial sum $\sum_{i=0}^{\ell} \alpha_i$ is positive.

It seems that a challenge is to find an order of summands in (6) that performs best. Yet our experiments indicate that the performance of a modified scheme is almost independent of the order of the summands.

5.2 The linear schemes to be modified

In this paper we modify four classical surface-generating linear subdivision schemes: Catmull-Clark (CC) [2], Kobbelt 4-point (K4) [10], Butterfly (BY) [6], and Loop (LP) [15]. The modification of all these schemes is done by the method of Section 5.1. We denote a modified scheme by adding "M" before the acronym of the corresponding linear scheme.

5.3 Naive normals

Here we present a method for determining initial normals at the vertices of a given control mesh, when the normals are not given as input.

Consider neighboring faces $\{f_i\}_{i=0}^{k-1}$ and neighboring vertices $\{v_i\}_{i=0}^{k-1}$ of a given vertex p in a control mesh (see Figure 6). Denote by a_i the normalized vector $\overrightarrow{pv_i} \times \overrightarrow{pv_{i+1}}$. The unit vector a_i defines a normal related to f_i . Let γ_i be the angle $\angle v_i p v_{i+1}$, and let $\gamma = \sum_{i=0}^{k-1} \gamma_i$. We suggest the normalized vector

$$n = \frac{a}{\|a\|} \quad \text{with } a = \sum_{i=0}^{k-1} \frac{\gamma_i}{\gamma} a_i, \quad (9)$$

as the "naive normal" at the vertex p . This method is similar to the one discussed in Section 3.5 in [17].

6 Performance of the modified schemes

In our examples we consider input meshes that consists of either quadrilateral faces or triangular faces. We accept meshes with irregular vertices (of valency

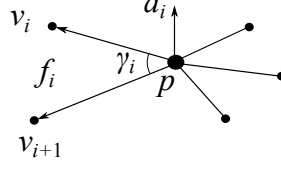


Figure 6: Determining a naive normal at p from its neighborhood in the mesh.

$\neq 4$ in quadrilateral meshes, and $\neq 6$ in triangular meshes). Our implementation is limited to meshes of exactly one type of faces. The initial data consists of the vertices of a mesh with a naive normal attached to each vertex.

6.1 Comparison methodology

We apply the linear schemes of Section 5.2 and their modifications on two example meshes with naive normals. We compare the performance of a linear scheme with its modified scheme by inspecting the generated surfaces and by measuring estimates of dihedral angles and curvatures. First we explain how we compute these estimates using the notation of Section 5.3. These estimates indicate deviation from C^1 smoothness, when considering the magnitude of the dihedral angles and from C^2 smoothness when considering the magnitude of the local changes in the curvatures.

6.1.1 Estimating dihedral angles

In the case of a triangular mesh we compute the dihedral angle corresponding to an edge as the angle between the normals of the neighboring triangles to the edge.

In the case of a quadrilateral mesh we estimate the dihedral angle at the midpoint of an edge e in the mesh by the following steps:

1. Compute the directions ℓ, r connecting the midpoint of e with the midpoints of the opposite edges of e in the neighboring faces to the left and to the right of e , respectively.
2. Compute $n_\ell = \ell \times \vec{e}$ and $n_r = r \times \vec{e}$, where \vec{e} is the edge direction.
3. Compute the angle between n_ℓ and n_r .

In the following we refer to this angle as the dihedral angle of the edge e .

6.1.2 Measuring discrete curvatures and its local changes

For every vertex p in a mesh M , we compute its discrete curvature K_p by

$$K_p = \frac{(2\pi - \sum_{i=0}^{k-1} \gamma_i)}{\mathcal{A}_p}, \quad \text{with } \mathcal{A}_p = \frac{1}{6} \sum_{i=0}^{k-1} |pv_i| |pv_{i+1}| \sin \gamma_i. \quad (10)$$

Here $v_k = v_0$, and \mathcal{A}_p stands for the area of the barycentric cell around p (See e.g. [17].) In the following we refer to K_p as the curvature at p .

To estimate the magnitude of the local changes in the curvatures, we first evaluate the curvature at all vertices of the mesh, and then estimate the local changes of the curvatures at p as

$$\zeta_p = \left| \max_{v \in V_p} \{K_v\} - \min_{v \in V_p} \{K_v\} \right|, \quad (11)$$

where V_p consists of p and all the adjacent vertices to p in the mesh.

6.2 Results

In studying the performance of the modified schemes, we consider only the generated meshes and ignore the generated normals. Although we do not have a convergence proof for the modified schemes investigated in this paper, our tests indicate that the generated meshes converge to a surface. The convergence of the normals is guaranteed by general results about convergence of manifold-valued subdivision schemes, based on geodesic averages (see e.g. [7]). We do not display the limit of the normals because, as in the 2D case [11], they are not the normals of the limit surface. Yet, as demonstrated in Table 2 of Section 6.3, the closer are the initial normals to the naive normals of the initial mesh, the closer are the limit normals to the normals of the limit surface.

Two input meshes are studied in this section demonstrating typical performance of a modified scheme in case of naive initial normals. One is referred as "Tower" and one is referred as "Tube". The Tower mesh is taken both in its quadrilateral and triangular form. For each example several iterations of one linear subdivision scheme and its modified variant are executed. The resulting surfaces are depicted in Figures 7, 8, 9. The colors indicate the curvature of the final mesh, if not mentioned otherwise. The colors yellow to red (cyan to blue) indicate positive (negative) values. All the examples are provided as mesh files in our online repository, as explained in Section 7. Our observations regarding these examples are based on these files. See Table 1 for typical numerical comparisons. We compute the maximal dihedral angle, ψ , and the maximal ζ_p , ζ^* , for each final mesh.

scheme	LP	MLP	CC	MCC	K4	MK4
ψ	14.78°	11.04°	11.02°	8.68°	37.11°	27.98°
ζ^*	0.034	0.031	0.019	0.025	1.231	0.846

Table 1: Numerical comparisons for the Tower mesh.

The numerical comparisons indicate that ζ^* of meshes produced by modified C^2 schemes (MCC, MLP) are of the same order as ζ^* of their linear counterparts, while for C^1 schemes, ζ^* of the modified schemes are significantly smaller. Also, the decrease rate of the maximal estimated dihedral angle from one refinement level to its next level is faster in case of the modified schemes.

Remark 6.1. For P_0, P_1 with $\theta = 0$ the circle average of the points is their linear average with the same weight (as shown in Subsection 4.3), and the normal is the normal of each of the two PNPs. Thus we conjecture that, if a modified scheme converges, then its smoothness equals that of the corresponding linear scheme.

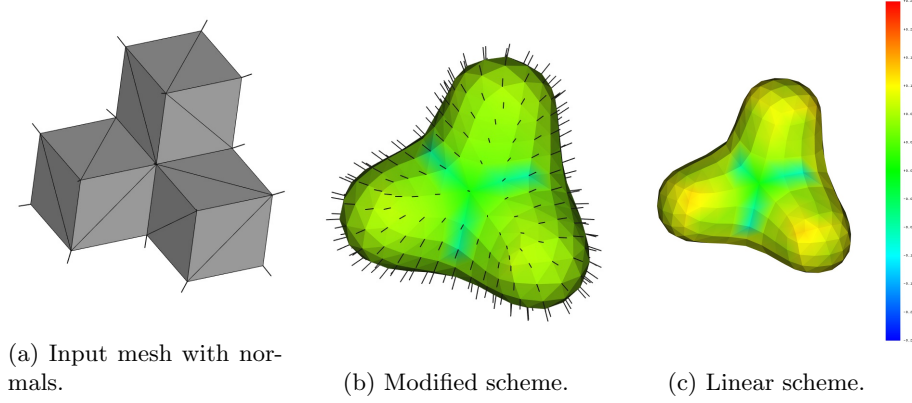


Figure 7: Triangulated Tower mesh and meshes generated by MLP and LP after 2 iterations.

Colors indicate magnitudes of curvature in the range $[-0.25, 0.25]$.

6.2.1 Approximating schemes (CC, LP)

The results of the linear LP scheme and the MLP scheme, applied to the triangulated Tower, are depicted in Figure 7. The results of the CC and the MCC schemes, applied to the quadrilateral Tower, look very similar, and are barely distinguishable from the LP results.

The meshes produced by MCC and MLP after four iterations appear to be "blown up" versions of the meshes generated by the CC and LP schemes respectively. The "blown up" meshes are not contained in the convex hull of the initial mesh, as do the linear variants.

6.2.2 Interpolating schemes (K4, BY)

The results in Figure 8 were obtained by applications of the linear and the modified K4 schemes to the tower mesh. The MK4 scheme produces meshes with smoother discrete curvature. However, the result of the linear variant follows the input control mesh more accurately, as is demonstrated in Figure 8. A similar behavior is observed in the BY/MBY case, when applied to the triangulated Tower.

It is shown in [5] that the linear 4-point scheme produces a self intersecting curve for an input polygon with edges of significantly different lengths. We observe the same artifact in the case of 3D meshes refined by the linear K4 and BY schemes. This artifact is not surprising, since both schemes are generalizations of the 4-point scheme.

An example of such performance by the BY scheme is depicted in Figure 9. The self intersections can be observed in the actual 3D models given in our Github repository (see Section 7). Also in Figure 9(c) this self intersection is indicated by the red color, which is assigned to the "inner side of the surface". On the other hand, there are no self intersections in the surfaces generated by the modified schemes (see Figure 9(b)).

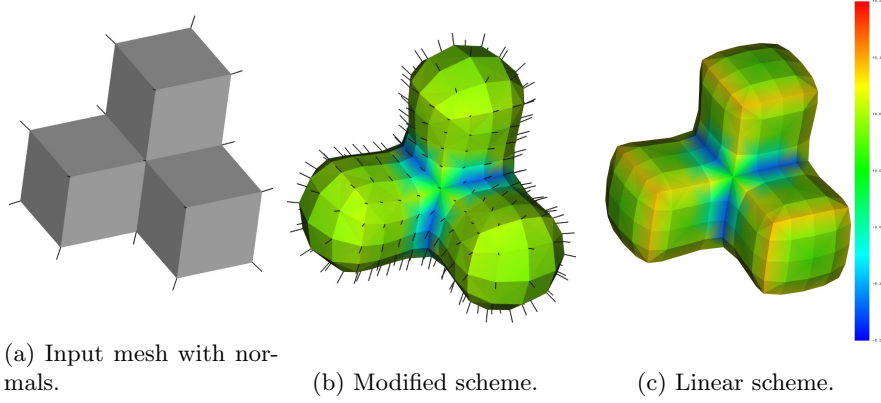


Figure 8: Tower mesh and meshes generated by MK4 and K4 after 2 iterations. Colors indicate magnitudes of curvature in the range $[-0.25, 0.25]$.

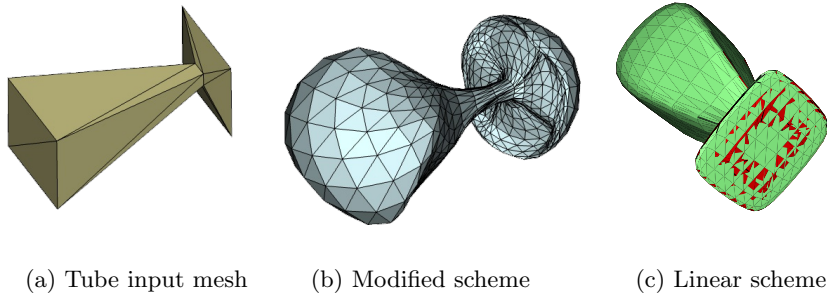


Figure 9: Triangulated Tube mesh and meshes generated by MBY and BY after 3 iterations. Colors emphasize self intersection, or the lack of it.

6.3 Demonstration of the editing capabilities of the modified schemes

Three videos demonstrating the variety of geometries obtainable with modified schemes are on our Youtube channel at [12]. The videos show geometry morphing processes when the initial mesh is kept and the initial normals are rotated. We obtain a sequence of eleven sets of initial normals, starting from all normals equal to some normal n^* , and arriving in ten steps at the naive normals of the mesh. The initial normal at a mesh point in case $i, i = 0, \dots, 10$, is the weighted geodesic average between n^* and the naive normal at that point, with weight $\mu_i = i/10$. Refining four times each set of initial data by the same modified subdivision scheme, a sequence of geometries is obtained. These geometries are combined into a morphing video, demonstrating changes from a surface generated by the corresponding linear scheme (in the case when all initial normals are equal to n^* , see Section 4.3), to the surface generated from the naive normals. Three snapshots of one of the videos are given in Figure 10. Note the changes

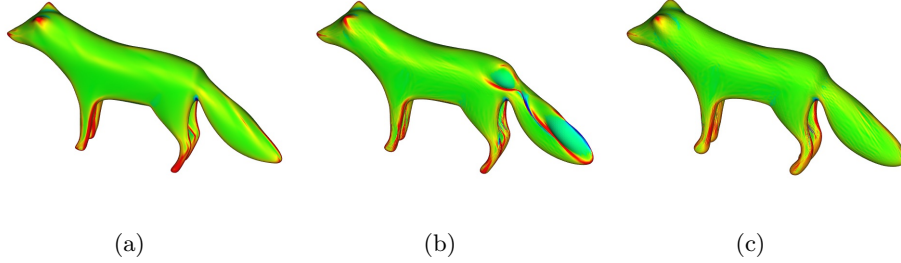


Figure 10: Surfaces generated by MLP, starting from the same "fox" mesh with different initial normals. Colors indicate magnitudes in the range $[-0.5, 0.5]$ of curvature at the vertices of the final refined mesh, generated by four iterations.

in the curvatures due to the changes in the initial normals.

We compute a numerical quantity which estimates the "distance" between the limit normals and the normals of the limit surface, for the different initial normals in cases $i = 0, 1, \dots, 10$. For every PNP in a final mesh, we compute the angle between the calculated normal of the PNP and the naive normal in the final mesh at the point of the PNP (approximating the normal of the limit surface there).

Table 2 contains averages of these angles for the weights μ_i , denoted by $\xi_i, i = 0, \dots, 10$. Note, that ξ_i decrease monotonically with i .

μ_i	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
ξ_i	85°	77°	71°	64°	56°	48°	38°	29°	22°	16°	12°

Table 2: Averages of the angles between the generated normals at the final refinement level and the corresponding naive normals of the final mesh.

Table 2 indicates that the closer are the initial normals to the naive normals, the closer are the limit normals to the normals of the limit surface.

Another observation is that one can setup initial normals such that a modified scheme computes a surface with unexpected geometry. See Figure 10b, for an example. Observe that in Figure 10 the main changes are in the tail of the fox.

7 Implementation

We provide an implementation of the algorithms and comparison methods, developed in this work, in our Github repository at [13]. The implementation is in the Python language. The input and the refined mesh files of the examples studied in this paper are in that repository too.

8 Conclusions and future work

In this paper we design an extension of the 2D circle average to 3D. This is indeed an extension, since the 3D circle average coincides with the 2D circle average when the two averaged PNPs are in the same plane. We modify surface-generating linear subdivision schemes refining points to surface-generating schemes refining PNPs, using the 3D circle average. The modified schemes can generate a variety of new geometries from a given mesh, by editing the initial normals. These editing capabilities are demonstrated in Figure 10 and by three videos [12].

Our investigation of the performance of the modified schemes included more examples than those of Section 6.2. An overall observation is that the results of a modified scheme with naive normals appear to be smoother than those of the corresponding linear scheme in case the initial mesh is homogeneous (consists of edges with lengths of the same order of magnitude). Also, the results of the modified approximating schemes are "blown up" versions of their corresponding linear counterparts, and are not necessarily contained in the convex hull of the initial mesh.

Several research directions should be addressed in the future:

- To prove the convergence of the modified schemes.
- To analyze the smoothness of the modified schemes.

These two topics are addressed in [11],[14] for certain 2D schemes.

- How to support creases/sharp edges with the modified schemes?
- To design a new binary operation between two point-normal pairs such that the modified schemes with this operation generate limit normals which are the normals of the limit surface, a property not possessed by our modified schemes (see Section 6.2).

References

- [1] Mao Aihua, Luo Jie, Chen Jun, and Li Guiqing. A new fast normal-based interpolating subdivision scheme by cubic Bézier curves. *The Visual Computer*, 32(9):1085–1095, 2016.
- [2] Edwin Catmull and Jim Clark. Recursively generated b-spline surfaces on arbitrary topological meshes. *Computer Aided Design*, 10(6):355 – 355, 1978.
- [3] Pavel Chalmovianský and Bert Jüttler. A non-linear circle-preserving subdivision scheme. *Advances in Computational Mathematics*, 27:375–400, 2007.
- [4] Costanza Conti and Nira Dyn. Analysis of subdivision schemes for nets of functions by proximity and controllability. *Journal of Computational and Applied Math*, 236:461 – 475, 2011.

- [5] Nira Dyn, Michael S. Floater, and Kai Hormann. Four-point curve subdivision based on iterated chordal and centripetal parametrizations. *Computer Aided Geometric Design*, 26:279–286, 2009.
- [6] Nira Dyn, John A. Gregory, and David Levin. A butterfly subdivision scheme for surface interpolation with tension control. *ACM Transactions on Graphics*, 9:160 – 169, 1990.
- [7] Nira Dyn and Nir Sharon. A global approach to the refinement of manifold data. *Mathematics of Computation*, 1:1–2, 2016.
- [8] Nira Dyn and Nir Sharon. Manifold-valued subdivision schemes based on geodesic inductive averaging. *Journal of Computational and Applied Mathematics*, 311:5467, February 2017.
- [9] Shay Kels and Nira Dyn. Subdivision schemes of sets and the approximation of set-valued functions in the symmetric difference metric. *Foundations of Computational Mathematics*, 13(5):835–865, 2013.
- [10] Leif Kobbelt. Interpolatory subdivision on open quadrilateral nets with arbitrary topology. *Computer Graphics Forum*, 15:409 – 420, 1996.
- [11] Evgeny Lipovetsky and Nira Dyn. A weighted binary average of point-normal pairs with application to subdivision schemes. *Computer Aided Geometric Design*, 48:3648, November 2016.
- [12] Evgeny Lipovetsky and Nira Dyn. Demo video. https://www.youtube.com/channel/UCq_EHFcG1CguUY20f14XAva, 2018.
- [13] Evgeny Lipovetsky and Nira Dyn. Supplementary source code. <https://github.com/subdivision/CircAvg3D>, 2018.
- [14] Evgeny Lipovetsky and Nira Dyn. C^1 analysis of some 2D subdivision schemes refining point-normal pairs with the circle average. *Computer Aided Geometric Design*, 69:45–54, 2019.
- [15] Charles T. Loop. Smooth subdivision surfaces based on triangles. Master’s thesis, Department of Mathematics, University of Utah, 1987.
- [16] Jean-Louis Merrien and Tomas Sauer. From hermite to stationary subdivision schemes in one and several variables. *Advances in Computational Mathematics*, 36 (4):547 – 579, 2012.
- [17] Mark Meyer, Mathieu Desbrun, Peter Schroder, and Alan H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. *Visualization and mathematics*, 3 (2):52 – 58, 2002.
- [18] Jörg Peters and Ulrich Reif. *Subdivision Surfaces*, volume 3 of *Geometry and Computing*. Springer-Verlag Berlin Heidelberg, Berlin Heidelberg, 2008.
- [19] Johannes Wallner. On convergent interpolatory subdivision schemes in riemannian geometry. *Constructive Approximation*, 40 (3):473 – 486, 2014.