A Curvature and Density-based Generative Representation of Shapes

Z. Ye,[1] (iD) N. Umetani,[2] T. Igarashi[2] and T. Hoffmann[1]

[1]TU Munich, Germany
tim.hoffmann@ma.tum.de
[2]The University of Tokyo, Japan
n.umetani@gmail.com, takeo@acm.org

Abstract
This paper introduces a generative model for 3D surfaces based on a representation of shapes with mean curvature and metric, which are invariant under rigid transformation. Hence, compared with existing 3D machine learning frameworks, our model substantially reduces the influence of translation and rotation. In addition, the local structure of shapes will be more precisely captured, since the curvature is explicitly encoded in our model. Specifically, every surface is first conformally mapped to a canonical domain, such as a unit disk or a unit sphere. Then, it is represented by two functions: the mean curvature half-density and the vertex density, over this canonical domain. Assuming that input shapes follow a certain distribution in a latent space, we use the variational autoencoder to learn the latent space representation. After the learning, we can generate variations of shapes by randomly sampling the distribution in the latent space. Surfaces with triangular meshes can be reconstructed from the generated data by applying isotropic remeshing and spin transformation, which is given by Dirac equation. We demonstrate the effectiveness of our model on datasets of man-made and biological shapes and compare the results with other methods.

**Keywords:** digital geometry processing, modeling, geometric modeling, modeling

**CCS Concepts:** • Computing methodologies → Learning latent representations; Mesh geometry models

1. Introduction

While the convolutional neural network (CNN) has achieved significant success in 2D image processing, more and more attention has recently been drawn to applying the technique to the domain of 3D shapes. Unlike 2D images, which are typically represented by a multidimensional tensor, the representation of 3D shapes is usually unstructured; hence, the CNN is not directly applicable. Thus, the main challenge is how to create a suitable representation for 3D shapes which can take advantage of the state-of-art machine learning frameworks. Several such representations based on point clouds [FSG17, ADMG18, GFK*18], volumetric data [TDB17, WSLT18, WLG*17] and meshes [BHMK*18] have been proposed with different applications. However, all these representations are built on the positional data such as the coordinates of points, vertices or voxels.

In this paper, we propose a 3D deep generative model based on mean curvature and metric, which in discrete case are expressed by two functions that are invariant under Euclidean motion. It has the following advantages against the existing models:

Firstly, our model preserves more detailed structure in case that the curvature plays a critical role, especially when the surface is highly folded and convoluted like the cortical surfaces in Figure 1. The CNN is known to be good at capturing not only the global features but also the local fine structure of data. Its effectiveness, however, relies on a proper distance function defined on the space of features. For example, the Euclidean distance between two vectors is a straightforward option. As the result, the bumpy circle (inset) will tend to be deformed through the neural network to the round circle, which is more regular and is close to the bumpy one under the measurement by Euclidean distance. In contrast, we adopt curvature representation and subsequently the distance between curvatures, by which two circles are clearly distinguishable, hence the small hills will be safely preserved. Secondly, our model is less affected by rigid transformation and uniform scaling. Thanks to the invariant quantities that constitute our representation and the CNN on sphere (see Section 4.2 for detailed discussion), we provide a
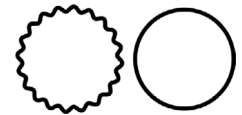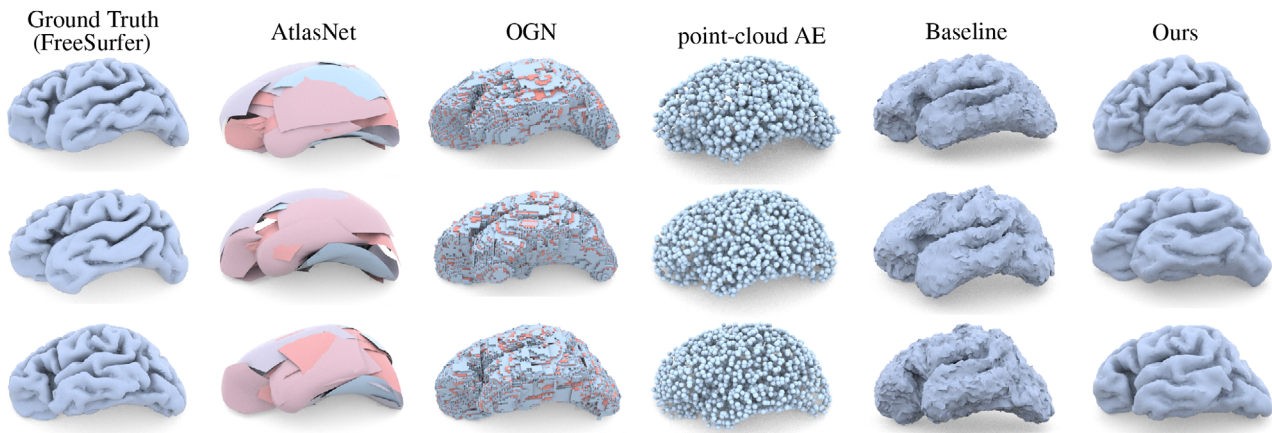
Figure 1: *Brain autoencoder. We build a curvature-to-curvature autoencoder and compare to the models based on point clouds, the AtlasNet [GFK*18] (point clouds to surface) and the point-cloud AE [ADMG18] (point clouds to point clouds), the voxel-based model OGN [TDB17] (IDs to voxels), and a mesh-based baseline model, which replaces the curvature in our model with vertex coordinates. All the neural networks, except for OGN, are trained on 1400 cortical surfaces and validated on 200 surfaces, which do not appear in the training set. Three of the predicted surfaces from the validation data are shown above. Although all the models can restore the brain structure in a large scale, only our model preserves the local fine structure. For more details see Section 4.3.*

simple and efficient way to handle the data without a consistent alignment.

The input shapes for our model are required to be surfaces with consistent simply-connected topology, e.g. the disk-like surface or the spherical surface. We first map the input surface to a canonical domain such as a sphere, where mean curvature and vertex density are extracted and recorded as the input data for the neural network. For generative models like VAE, the output is a variant of the input so it has the same form as the input. To reconstruct the shape, we first create a conformal parameterization by randomly sampling the points with respect to the generated density function and applying the isotropic remeshing. Then, we deform the mesh gradually towards the target shape with the prescribed mean curvature (see the attached videos).

A curvature-to-shape reconstruction algorithm with high accuracy is critical for generating plausible shapes. We follow the basic idea in [CPS13] and [YDT*18]. The deformation between the domain and target shapes is given by the solution of the Dirac equation. We propose a modified equation with a larger solution space and it results in the reconstruction comparatively closer to the target shape. Furthermore, one might be concerned about the stability of curvature-based methods, since tiny errors in curvature might accumulate across the surface and significantly affect the final reconstruction. Indeed, in our case, previous methods fail to locally scale the shape correctly at regions with large curvature. In fact, it is hard to directly manipulate the local scaling with the Dirac equation. Therefore we design a new algorithm inspired by Chern et al. [CPS15] to calibrate the area scaling factor. This compensates for the shortcoming of the Dirac equation and significantly stabilize the reconstruction.

We evaluate our reconstruction algorithm on several shapes, showing that our method outperforms previous methods visually

and quantitatively. In addition to some preliminary applications such as shape remeshing, interpolation and clustering, we demonstrate randomly generated shapes from various datasets and compare to other 3D generative models.

In summary, the contribution of this paper is (1) an improved algorithm for shape reconstruction from curvature with area calibration and (2) a 3D shape deep learning framework based on curvature.

2. Related work

2.1. Which invariant quantities determine an immersed surface in $\mathbb{R}^3$?

It is well-known that an immersed surface in $\mathbb{R}^3$ is determined up to a Euclidean motion by its first and second fundamental forms. However, their representation depends on the choice of coordinate. Hence, in order to consistently represent 3D shapes based on the two fundamental forms, an identical triangulation for all shapes, which is not always possible, is required.

Other options are point-wise shape descriptors such as the heat kernel signature [SOG09] and the wave kernel signature [ASC11]. Indeed, they have been employed in discriminative models for 3D shape classification and segmentation [BMM*15]. But they can hardly be used for generative models, because it is unclear whether these shape descriptors completely determine the shapes or how to reconstruct shapes from them.

The idea of this paper comes originally from Bonnet [Bon67]. In fact, except for some very special cases, an immersed surface is completely determined by conformal structure, regular homotopy class and mean curvature half-density, which is a scale-independent variant of the mean curvature [Kam98]. The exceptions, called the Bonnet immersions, includes minimal surfaces, constant mean
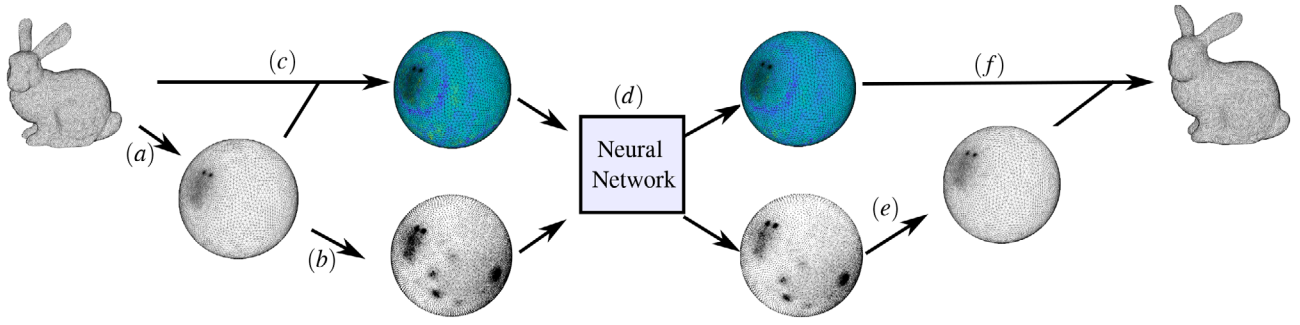
Figure 2: *The pipeline of our model for generating variant shapes: (a) the conformal parameterization (Section 3.1), (b) the density function extraction (Section 3.2), (c) the mean curvature half-density extraction (Section 3.2), (d) learning and generating (Section 3.3), (e) the isotropic remeshing (Section 3.4), (f) solving the Dirac equation and applying the spin transformation (Section 3.5).*

curvature surfaces and Bonnet pairs. In our case the regular homotopy class is unnecessary, since we only consider the simply-connected surfaces which have only one unique regular homotopy class [Pin85]. In summary, generic simply-connected immersed surfaces are uniquely determined by the conformal structure and the mean curvature half-density.

2.2. Quaternions, Dirac-type operators and spin transformation

Now, we sketch the idea how to construct a surface from the mean curvature half-density. Roughly speaking, for every point on the surface we rotate its infinitesimal neighbourhood with a quaternion. Recall that a quaternion is a 4-dimensional vector $q = a + bi + cj + dj$ with the multiplicative structure:

$$i^2 = j^2 = k^2 = -1,$$

$$ij = -ji = k, \quad jk = -kj = i, \quad ki = -ik = j.$$

We always identify vectors in $\mathbb{R}^3$ as pure imaginary quaternions

$$(x, y, z) \mapsto xi + yj + zk.$$

Any quaternion can be written as $q = |q|(\cos\frac{\theta}{2} + \sin\frac{\theta}{2}u)$, where $\theta \in [0, 2\pi)$ and $u \in \mathbb{R}^3 \subset \mathbb{H}$. It is well known that q gives a scale rotation in $\mathbb{R}^3$ with scaling factor $|q|^2$, rotation angle θ and rotation axis u. The rotation is given by

$$R_q(v) = \overline{q} \cdot v \cdot q.$$

The explicit construction of shapes from mean curvature half-density and conformal structure is called spin transformation. Suppose given an immersion of a surface $f : M \to \mathbb{R}^3$ and a quaternion-valued function on the surface $\phi : M \to \mathbb{H}$, which is understood as a continuously varying rotation at each point. We scale and rotate every tangent plane by

$$d\tilde{f} = \overline{\phi} \cdot df \cdot \phi. \tag{1}$$

However, there is no guarantee that these rotated tangent planes will again form a surface. For simply connected surface, $d\tilde{f}$ is again the tangent plane of an immersion of surface if and only if it is closed:

$$dd\tilde{f} = 0$$

It turns out to be equivalent to the Dirac equation [KPP98]

$$D_f\phi = \rho\phi, \tag{2}$$

where the Dirac operator is defined by

$$D_f\phi = -\frac{df \wedge d\phi}{|df|^2}, \tag{3}$$

and $\rho : M \to \mathbb{R}$ is a real-valued function. Therefore, any solution of the Equation (2) will induce a new immersion $\tilde{f} : M \to \mathbb{R}^3$ by $\tilde{f} = \int_M d\tilde{f}$. Moreover, the mean curvature $\tilde{H}$ of $\tilde{f}$ is given by

$$\tilde{H}|d\tilde{f}| = H|df| + \rho|df|, \tag{4}$$

where H is the mean curvature of the original surface f. Observe that, due to the scaling factor $|d\tilde{f}|$ in (4), one cannot fully control the mean curvature $\tilde{H}$. However, by introducing a variant notion, namely the mean curvature half-density:

$$h := H|df|, \tag{5}$$

the Equation (4) turns to

$$\tilde{h} = h + \rho|df|. \tag{6}$$

This means that the mean curvature half-density $\tilde{h}$ can be precisely realized as long as the solution ϕ for Equation (2) exists.

Crane et al. [CPS11] first discretize the Equation (3) and show applications in computer graphics, such as curvature painting. The following works are, e.g. Crane et al. [CPS13] use the spin transformation for surface fairing. Liu et al. [LJC17] construct a continuous spectrum of operators between the square of the Dirac operator and the Laplace–Beltrami operator. These operators are utilized to enhance surface matching and segmentation problems. Ye et al. [YDT*18] create a framework, which consistently discretized the extrinsic Dirac operator and an intrinsic Dirac operator. In this paper, we improve the reconstruction based on [CPS11, YDT*18] by solving an equation with a larger solution space and introducing an area calibration (see Section 3.5).

2.3. Deep generative models for 3D shapes

Various representations of surfaces have been proposed for 3D shape generation, e.g. models based on volumetric representation

mean curvature
half-density

1.8

−1.2
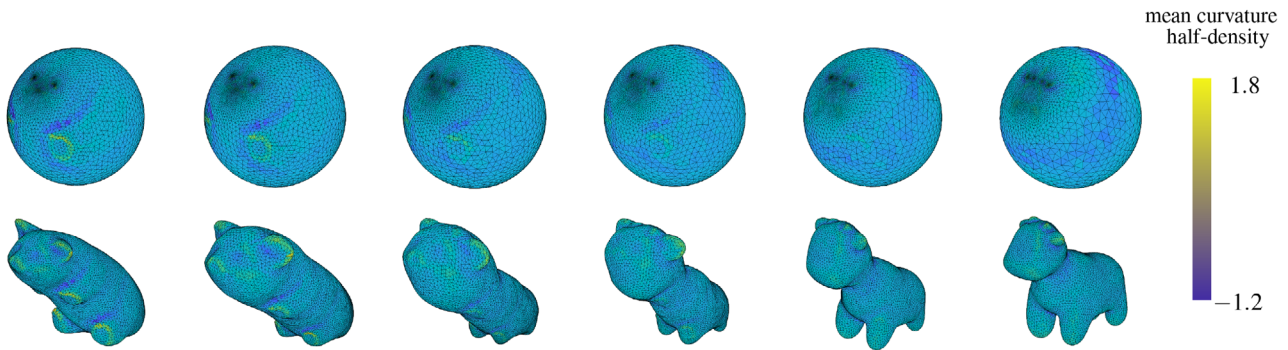
Figure 3: *The spherical conformal parameterizations of two animals are aligned by a Möbius transformation with three landmark points. Then, they are packed into tensors with dimension $320 \times 32 \times 32 \times 2$. This figure shows a linear interpolation between the curvature representation of two shapes and the resulting shape reconstruction from the curvature representation.*

[WZX*16, TDB17, SM17, WLG*17, WSLT18], or point clouds representation [FSG17, NW17, ADMG18]. These methods are particularly applicable for the dataset with inconsistent topology. However, without knowing the mesh structure it is hard to capture the fine structure of certain highly complicated surfaces (see Figure 1).

Our model is closer to the following works, which take the mesh structure into account. Ben-Hamu et al. [BHMK*18] propose a representation based on multiple charts, which conformally map different parts of shapes to a domain. Since features over each chart are normalized separately, the fine structure will be better preserved than with a single chart. However, while the creation of such charts requires a sparse correspondence, reconstruction of shapes from the charts needs a template shape, which amounts to a dense correspondence. In order to find such correspondence, one has to introduce a time-consuming workflow beforehand. Groueix et al. [GFK*18] learns a parameterization of shapes with multiple embedded charts. Hence one does not have to manually create the charts. However, the generated charts do not always perfectly fit with each other, nor do they preserve as much details as the ones in [BHMK*18]. Umetani [Ume17] develops a depth map representation with a cube as the domain. This representation works well for close-to-convex shapes like cars, but would be difficult to be applied on highly curved and non-convex shapes. Kostrikov et al. [KJP*18] use the same Dirac operator as ours. But they merely replaced the Laplace–Beltrami operator in the neural network with the Dirac operator, thus the real power of the Dirac operator, namely its connection to conformal transformation, is not exploited.

3. Method

The main pipeline of our model is depicted in Figure 2. In the sequel, we will explain the detailed methods for encoding shapes with curvature and vertex density in Sections 3.1 and 3.2, building a neural network based on our representation in Section 3.3 and reconstruction of shapes in Sections 3.4 and 3.5.

Encoding the conformal structure. In discrete case, how to encode a shape in the scheme of the Bonnet problem (Section 2.1)? While the mean curvature half-density can be represented by a vertex-based or face-based function, it is not straightforward to pack

conformal   add the curvature   reconstruction
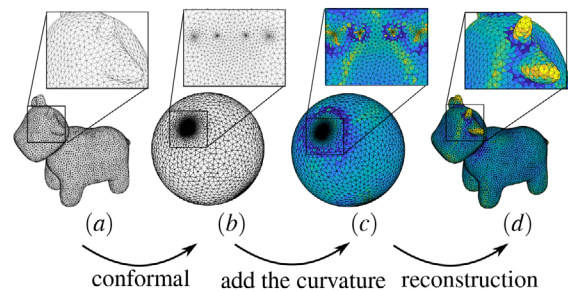
Figure 4: *[YDT*18] shows that a simply-connected surface in $\mathbb{R}^3$ can be faithfully reconstructed from its conformal parameterization by prescribing the mean curvature half-density.*

the conformal structure in a form that is suitable for machine learning pipeline. For example, we can recover the shape of a cow from its spherical conformal parameterization ((b) in Figure 4) by prescribing the function of mean curvature half-density ((c) in Figure 4). But it is not clear how to represent a spherical mesh that is conformal equivalent to a given shape purely by scalar functions. One might consider the notion of discrete conformal equivalence for triangular meshes by length cross-ratio on edges ([SSP08]). But it is unclear how to transfer the length cross-ratio across different meshes.

Recall that the conformal structure is the set of metrics modulo the equivalence relation $g \sim e^{2u}g$, i.e. two metrics are identified if they only differ by a scaling at each point. Therefore, instead of encoding the conformal structure, we encode the metric of shapes. In general, the space of all metrics still does not have an efficient form of representation, thus we focus on a smaller subset, i.e. the isotropic meshing. Since the conformal map is locally isotropic, i.e. it takes an isotropic mesh to a close-to-isotropic mesh (see the zoom-in in Figure 4), and we know that the isotropic meshing is usually generated by the centroidal Voronoi tessellation (CVT) with respect to a density function [ADVDI03], this density function can be utilized as an approximation of a metric. Therefore, at the beginning of our pipeline all the input shapes are isotropically remeshed (like (a) in Figure 4). Then, we successively take the following procedures.

3.1. Conformal parameterization

We map all the shapes to a canonical domain, e.g. the unit disk for disk-like surfaces and the unit sphere for spherical surfaces. The resulting disk-like or spherical meshes are called the conformal parameterization. However, these maps are not unique but differ by a conformal automorphism of the domain. To deal with the ambiguity one may choose from the following approaches depending on the application:

Landmark alignment. We know that the conformal automorphism of S^2, i.e. the Möbius transformation, is fully determined by three distinguished points and the conformal automorphism of a disk is determined by one point and one rotation. Hence we choose two landmark points for disk-like surfaces and three landmark points for closed surfaces and align these landmarks via a conformal mapping. One example is shown in Figure 12.

Landmark-free alignment. For example, [BCK18] proposed a canonical Möbius transformation such that the mass centre is aligned with the sphere centre. Then, we register two spherical meshes of centred Möbius transformations by searching for an optimal rotation.

Without any alignment at all. This will result in a larger shape latent space and consequently poses higher demands on the capacity of neural network, because, for example a rotation of shapes might also cause a rotation of curvature function. However, our model is particularly good at capturing this uncertainty (see the discussion in Section 4.2).

Specifically, there are many available algorithms for conformal parameterization for disk-like and spherical surfaces, e.g. [GWC*04, CPS13, CL15, CLL15, SC17, YDT*18]. In fact, we did not observe significant differences between these algorithms in our experiments.

3.2. Making the representation

In order to build the neural network, we need some fixed meshes for canonical domains. In particular, we use the standard 256×256 grids for the disk. For the spherical domain, we obtain a spherical mesh by iteratively applying the 1-to-4 subdivision and normalization on an icosahedron.

Then, we interpolate the following two functions from the conformal parameterization of shapes to the domain with inverse distance weight.

Mean curvature half-density. The mean curvature half-density h is a face-based function given by [YDT*18]

$$h_i = \frac{\sum_j |e_{ij}| \tan \theta_{ij}/2}{2\sqrt{A_i}}, \tag{7}$$

where the sum runs over all the edges e_{ij} of the face T_i, θ_{ij} are bending angles at the edge e_{ij} and A_i are the face area.

Figure 5: *Downsampling layers based on the subdivision structure of the spherical meshes. The tensors in the previous layer, which are corresponding to a common triangle in the next layer, are merged to the tensor associated with the father triangle. These downsampling layers respect the spatial relations among the triangles.*

Vertex density function. We estimate the density function $\mathfrak{d}$ by the reciprocal of vertex area, $\mathfrak{d}_i := 1/\tilde{A}_i$, where $\tilde{A}_i$ is the vertex area of the conformal parameterization. We do not normalize the density $\mathfrak{d}$, since the integral of the piecewise constant function $\int_U \mathfrak{d}\, dA = \sum_i \mathfrak{d}_i \tilde{A}_i = i$ is equal to the number of points located in the area U. At the step of reconstruction, this gives us the information about how many points should be sampled. In the experiment, we observe that the logarithmic density $\tilde{\mathfrak{d}} := \log \mathfrak{d}$ is more evenly distributed. Therefore, the logarithmic density $\tilde{\mathfrak{d}}$ is instead recorded on the domain.

3.3. Building CNNs over meshes

Since the disk-like surface is represented like a 2D image with two channels, any classical CNNs can be directly applied. Hence, we will focus on the case of spherical surfaces.

Convolution layers. Many CNNs on arbitrary graphs or surfaces have been proposed in recent works, see [BZSL14, KW17, DBV16, MBM*17, BMM*15, MBBV15, MGA*17] and the survey [BBL*17]. We opt for a mesh-based CNN based on small tangent patches [TPKZ18].

Each face of the domain is assigned with a tangent plane, identified with $\mathbb{R}^2$, at the barycenter. Let l be a positive number such that the projection of the triangular face lies entirely in the patch $[-l, l] \times [-l, l]$ on the tangent plane. This projection π gives a local coordinate system of the points in the pre-image $\pi^{-1}([-l, l] \times [-l, l]) \subset S^2$. Hence, the functions restricted in this region can be interpolated to some grids on the patch. The distortion caused by the projection is neglectable when the size of the patches is small. We choose a fixed length l such that all the triangular faces on the domain are projected inside the corresponding patches. The convolution is the ordinary $2D$ convolution within each patch with the shared filter weights across different patches.
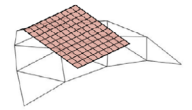
Downsampling and upsampling layers. Like the MaxPooling and UpPooling layers for classical CNNs, we need the same sort of operations for mesh domain to decrease and increase the spatial dimension of neural network. One can first apply the ordinary $2D$ pooling layers within each patch. Furthermore, since our spherical domain is constructed by subdividing an icosahedron, it is naturally endowed with a hierarchical structure (Figure 5), which gives rise to downsampling and upsampling layers between spherical meshes with different refinements.
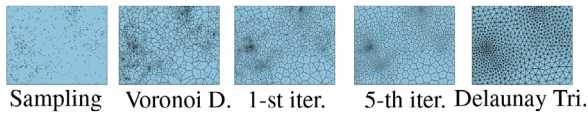
Figure 6: *Centroidal Voronoi Tessellation. In order to obtain an isotropic meshing with respect to a given density, we first sample a point set according to the density and repeatedly apply the Lloyd's relaxation. Observe that the point set becomes more and more isotropic as the iteration goes.*
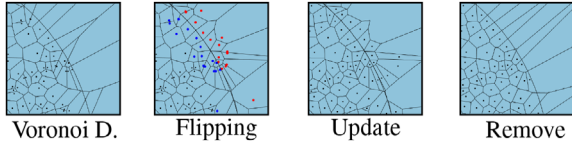
Figure 7: *Constraint CVT. To avoid dealing with unbounded Voronoi cells, we flip the points, which are close to the boundary, such that the cells close to the boundary are all bounded.*

The detailed architectures of our CNNs are depicted in the appendix.

3.4. Reconstruction of conformal parameterization

In order to construct a conformal parameterization from a given vertex density function $\mathfrak{d}$, we first randomly sample n_i points in every faces of the domain, where $n_i = \mathfrak{d}_i \tilde{A}_i$ and $\tilde{A}_i$ is the face area. Next, an isotopic meshing is constructed as follows.

Centroidal Voronoi tessellation. The isotropic meshing is usually made by CVT [DFG99]. Given a set of points $\{v_i\}$ in a metric space, particularly $\mathbb{R}^2$ or S^2. The Voronoi region V_i corresponding to v_i is defined by

$$V_i = \{x \mid |x - v_i| \le |x - v_j|, \ j \ne i\}, \tag{8}$$

which are polygons (see Appendix A.1 for the formula for computing the weighted centroid of polygons). Given a density function $\mathfrak{d}$, the centroid v_i^* of the polygon V_i is given by

$$v^* = \frac{\int_V y \mathfrak{d}(y) dy}{\int_V \mathfrak{d}(y) dy}. \tag{9}$$

We call a point set $\{v_i\}$ the weighted CVT if $v_i = v_i^*$ holds true for all i.

In this paper we use Lloyd relaxation to compute the CVT. Given a point set $\{v_i\}$, we iteratively update the point v_i with the corresponding centroid v_i^* until it converges (see Figure 6):

1. Randomly sample the points with respect to the density $\mathfrak{d}$ (defined in Section 3.2).
2. Create the Voronoi diagram. For the disk case, we have to be a bit careful that the Voronoi cells close to the boundary are mostly unbounded. Hence we reflect the points close to the boundary, so that all the Voronoi cells inside or close to the unit disk are bounded.

3. Compute the weighted centroids of the (bounded) Voronoi cells and, for the disk case, remove the points lying outside the disk (see Figure 7).

Then, a Delaunay triangulation is constructed by taking the dual of the Voronoi diagram. Generally, this triangulation does not perfectly fit the disk at the boundary, but it does not significantly affect the global appearance of shapes.

3.5. Surface reconstruction

Now, we are ready to reconstruct the surface from a conformal parameterization with prescribed mean curvature half-density. In the following we first demonstrate an improved reconstruction method which is a slight modification of [YDT*18] and then introduce a new procedure of area calibration, which would be particularly effective when the area scaling is not accurately restored by the previous method.

Dirac energy. In practice, the exact solution of the Dirac equation (2) can hardly be obtained, so we actually search for the solution $\phi : M \to \mathbb{H}$ such that:

$$(D_f - \rho - \sigma)\phi = 0$$

for a very small real number σ, which actually amounts to the eigenvalue problem

$$(D_f - \rho)\phi = \lambda\phi,$$

where λ is the eigenvalue with the smallest magnitude [CPS11].

In discrete case, $D_f - \rho$ is a $|F| \times |F|$ quaternion-valued matrix [CPS11], or in practice, a $4|F| \times 4|F|$ real-valued matrix such that any quaternion $q = a + bi + cj + dk$ is represented by a 4×4 real-valued block matrix:

$$\begin{pmatrix} a & -b & -c & d \\ b & a & -d & c \\ c & d & a & -b \\ d & -c & b & a \end{pmatrix}.$$

We briefly introduce the discretization of the matrix $D_f - \rho$ and refer the reader to [CPS11, YDT*18] for more details. Let $e_{ij} \in \text{Im}(\mathbb{H})$ be the oriented edge embedded in the quaternion space and $\mathbf{H}_{ij} := \frac{1}{2}|e_{ij}| \tan \frac{\theta_{ij}}{2}$ be the integrated mean curvature at the edge e_{ij}, where θ_{ij} is the bending angle between the face i and j. The matrix of the Dirac operator is a $4|F| \times 4|F|$ matrix D_f given by ([YDT*18])

$$(D_f \phi)_i = \frac{1}{2} E_{ij} \cdot \phi_j - \mathbf{H}_i \phi_i,$$

where $E_{ij} := 2\mathbf{H}_{ij} + e_{ij}$ and $\mathbf{H}_i = \sum_j \mathbf{H}_{ij}$. The discrete form of ρ is a $4|F| \times 4|F|$ diagonal matrix P with the discrete mean curvature half-density (7) as the diagonal. Instead of building the target shape in one step, we slowly flow the initial shape to the target for the purpose of stability. Hence, we build the matrix $\hat{D}(t) = D_f - tP$, where $t \in [0, 1]$ is a step length parameter.

We observe that, even though this face-based Dirac operator gives the exact solution, it is not numerically stable, because its solution space is often too large (technically, some solutions that give the

edge-constraint normals far from the actual face normal will result in unwanted transformations). On the other hand, while the vertex-based operators in [CPS11, YDT*18] works well in many cases, they are not able to faithfully recover the high curvature regions on the surface, because their solution spaces are too limited. To have a balance between these two approaches we propose the following regularized energy based on the face-based operator:

$$E_D(t) = \hat{D}^T(t) \cdot \hat{D}(t) + cR,$$

where c is a positive coefficient and R is the $4|F| \times 4|F|$ regularization matrix such that

$$R = \sum_{ij} |e^*_{ij}|(\phi_i - \phi_j)^2,$$

where the sum runs over all adjacent faces i and j. Note that the weights with the dual edge length are used in [CKPS18]. To have finer control of the regularizer, one can decompose R into four components and set different weights as in [CKPS18], but we did not see that this will make any obvious difference in our setting. Empirically, the coefficient c is set to be $0.001 \max_{ij} |e_{ij}|$.

By the min–max principle, solving the generalized eigenvalue problem

$$E_D(t)\phi = \lambda M\phi,$$

where M is the mass matrix, is actually equivalent to minimizing the energy

$$\min E_D, \text{ s.t. } |\phi| = 1,$$

with the metric defined by $|\phi|^2 := \phi^T \cdot M \cdot \phi$.

Finally, the edges are constructed by the spin transformation

$$e_{ij} \mapsto \text{Im}(\overline{\phi_i} \cdot E_{ij} \cdot \phi_j),$$

the position of vertices v_i are recovered by solving the Poisson equation (see Section 3 of [SA07] or Section 5.6 of [CPS11]). In the attached videos, we prescribe the mean curvature half-density of two shapes (red) on their conformal parameterization (blue) and it shows deformation from the sphere to the original shapes.

Area calibration. Even though the Dirac operator with regularization term improves the accuracy of reconstruction, we observe that some area distortion is still visible, especially at the region with really high curvature. To overcome this problem, we make the reconstruction algorithm be aware of the area scaling factor. Chern et al. [CPS15] prescribe a volumetric scaling factor e^u and obtains the close-to-conformal volumetric deformation by minimizing an energy E_u depending on u. While the energy E_u in [CPS15] is specifically designed for 3D volumetric meshes, an analogy for 2D surfaces still holds in smooth case:
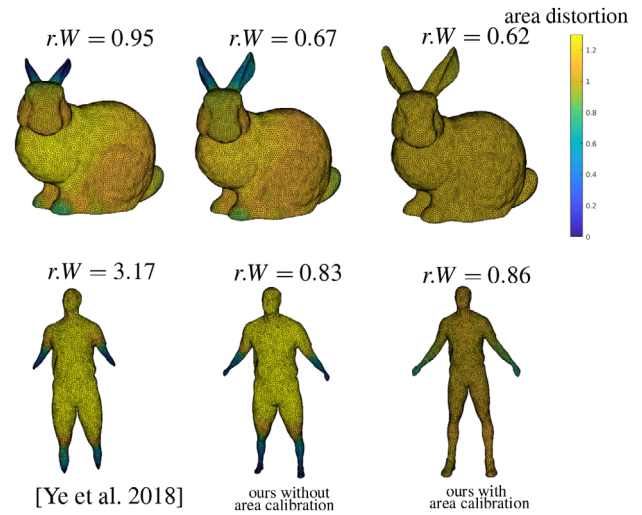
Figure 8: *Reconstruction of shapes from their conformal parameterization. While the Willmore energy is defined by $W = \sum_i h_i^2$, we define the relative Willmore energy between two meshes with identical connectivity by $r.W := \sum_i((h_1)_i - (h_2)_i)^2$, which measures how close the mean curvature half-density of two meshes are. This experiment shows that our method substantially improves the accuracy of curvature reconstruction. Furthermore, the area distortion, which usually appears in the regions with high curvature, gets much reduced by the area calibration. Note that, in contrast to [CPS15], we only encode the expected scaling factor in the energy $|\omega|^2$ and the factual scaling factor $|\phi|^4$ is determined by the optimizer.*

Theorem 1. *Let $f : M \to \mathbb{R}^3 \subset \mathbb{H}$ be an isometric immersion and $h : M \to \mathbb{R}$ be any function. The quaternion gradient is defined by*

$$\text{grad}_f h = df(\text{grad}\, h).$$

The spin transformation $d\tilde{f} := \overline{\phi} \cdot df \cdot \phi$ with $D_f \phi = 0$ is closing if

$$d\phi\phi^{-1} = -\frac{1}{2}Gdf, \qquad (10)$$

where $G := \text{grad}_f u$ is the gradient of the logarithmic factor $e^u := |\phi|^2$.

Proof. See Appendix A.2. $\square$

Therefore, given a spin transformation induced from ϕ with the area factor $u = \log|\phi|$, the quaternion-valued 1-form

$$\omega := d\phi + \frac{1}{2}Gdf\phi$$

vanishes. In practice, we minimize the energy $E_u := |\omega|^2$, where the metric for quaternion-valued 1-form is defined by

$$\langle \omega, \eta \rangle := \int_M \overline{\omega} \wedge (*\eta). \qquad (11)$$

In discrete case, minimizing the energy E_u again amounts to solving a generalized eigenvalue problem for a $4|F| \times 4|F|$ matrix (see

| $\|V\| = 2442$ | $\|V\| = 3738$ | $\|V\| = 5018$ | $\|V\| = 10001$ |
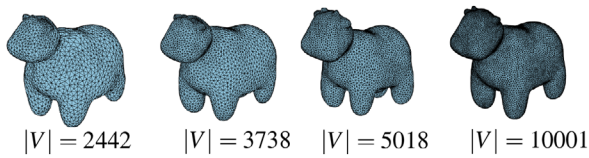
Figure 9: *Remeshing. Given an original shape with $\|V\| = 5000$, the density is modified by multiplying with 0.25, 0.75, 1 and 2. The mean curvature half-density changes accordingly such that the mean curvature is preserved.*

Section A.4). To avoid introducing the scaling factor as one more function in our representation and subsequently increasing the data size, we first apply the isotropic remeshing with approximate equalized face area [FAKG10] for all shapes. In this case the logarithmic factor u should be set to $u_i = \log(1/\sqrt{|\tilde{A}_i|})$, where $\tilde{A}_i$ is the face area of the conformal parameterization.
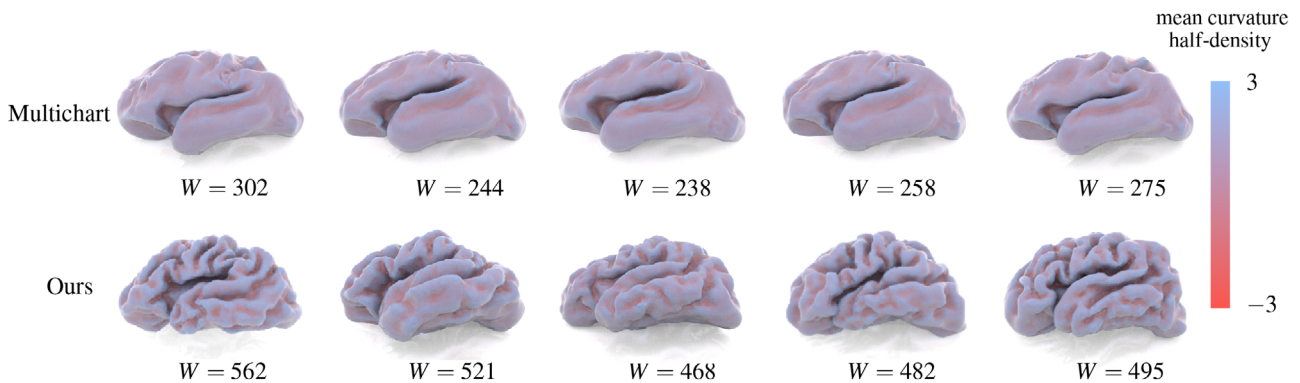
In summary, we first minimize the energy E_D with a small step length several times until the mean curvature half-density converges to the prescribed one. Then, we minimize the energy E_u once to get the correct area scaling factor.

4. Results

We use the Matlab package gptoolbox [J*18] for data pre-processing and Tensorflow [AAB*15] for building the neural networks on meshes. All the neural networks are trained and evaluated with the GPU GeForce GTX 1080 with 8GB memory.

4.1. Preliminary applications

We first present some simple applications that are unrelated to machine learning.

In smooth case, the mean curvature half-density changes co-variantly $h \mapsto m \cdot h$ under the parameterization scaling $x \mapsto m \cdot x$, $m \in \mathbb{R}$. Analogously, in discrete case, one can adjust the parameterization by scaling the vertex density, i.e. multiplies the density $\mathfrak{d}$ with a constant number, $\mathfrak{d} \mapsto m\mathfrak{d}$. In order to preserve the shape, one has to adjust the mean curvature half-density by $h \mapsto \frac{h}{\sqrt{m}}$. The shapes reconstructed from the modified representation are actually remeshings with approximately $m\|V\|$ vertices, where $\|V\|$ is the number of vertices of the original mesh. Figure 9 shows that our method will preserve the smooth features on the shape. However, the regions of high curvature tend to be smoothed with declining vertex number.

Shape interpolation. We visualize the interpolation of our curvature-based representation. Figure 3 shows the shapes reconstructed from a linear interpolation of two animals, whose conformal parameterizations are matched by a Möbius transformation that aligns 3 chosen landmark points. In addition, one can interpolate the latent space representation of a trained autoencoder (see Section 4.3). Figure 11 shows two latent space bi-linear interpolations of cars.
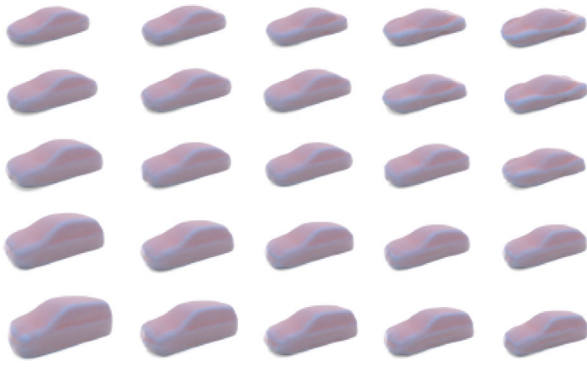
Random generation of disk-like and spherical shapes. We test our model for disk-like surfaces on a dataset of anatomical shapes provided by [BLC*11]. In particular, we choose the shapes of teeth, which is one of three types of bone in this dataset. To create the representation, we first take an intermediate conformal map, which maps the teeth to the unit disk by the algorithm from [CL15].

Several landmark points are available in [BLC*11], hence we choose two landmark points u_i, v_i for every shape M_i. We know that the conformal automorphisms of the unit disk have the form

$$f(z) = e^{i\theta} \frac{z - a}{1 - \overline{a}z},$$

where $\theta \in \mathbb{R}$ and $a \in \mathbb{C}$. Set $a = u_i$ and θ such that $f(v_i) \in \mathbb{R}$. Clearly, this uniquely determined map $f_{a,\theta}$ satisfies $f(u_i) = 0$ and $f(v_i) \in \mathbb{R}$. Fixing a reference shape M_0, for any shape M_i we apply the alignment map $f_0^{-1} \circ f_i$ for every shapes.

Figure 10: *The randomly generated cortical surfaces by Multi-chart GAN [BHMK*18] and the VAE based on our representation. Our representation has dimension $320 \times 32 \times 32 \times 2 = 655360$, which has the same magnitude as the data size of Multi-chart, i.e. $16 \times 64 \times 64 \times 3 = 196608$. However, we only require 3 landmark points for alignment, while the Multi-chart needs a dense correspondence for surface reconstruction. The surfaces are labeled by the mean curvature half-density. Note that, the training data mostly have the Willmore energy from 900 to 1000. Although the generated surfaces from our model have been smoothed to a certain extent (partly due to a well-known limitation of VAE), our model apparently preserves more fine structures than the position-based model.*

Figure 11: *Latent space interpolation. We choose four examples in the car dataset and interpolate their mean values in the latent space of VAE. The left lower triangle is a bilinear interpolation of a van, a car and an SUV. The right upper triangle is a bilinear interpolation of a van, a car and a race car.*

All the aligned disk meshes are then mapped to the square via the Schwarz-Christoffel mapping. The functions are interpolated on the 256×256 grid using the `scatteredInterpolant` function in Matlab.

For spherical surfaces we take the dataset of 1240 cars from ShapeNet [CFG*15]. All the shapes are converted into genus-0 surfaces by Umetani [Ume17]. Then we create the aligned conformal parameterization by the canonical Möbius transformation [BCK18]. The canonical domain with is obtained by subdividing the icosahedron twice so it has $20 \times 4^2 = 320$ faces. Each face is assigned with a 32×32 grid. Hence, each shape is represented by a $320 \times 32 \times 32 \times 2$-dimensional tensors.

The randomly generated teeth and cars are shown in the appendix as well as their curvature representation.

4.2. Generation of unaligned data

Discussion of local invariance. We call two functions f_1 and f_2 local invariant if they have the same function value but only differ by a transformation g of domain, i.e. $f_1 = f_2 \circ g$. Traditional CNNs are able to capture the translational features such as (a) of inset. Hence one would expect the CNNs for 3D shapes with the similar properties like local invariance under translation, rotation or even scaling. How-

ever, 3D generative models based on position, such as point cloud and mesh, will not have such properties due to the varied function value of coordinates (see (b)). This makes it more difficult for CNNs to extract meaningful information. The voxel-based models are local invariant, but they are not applicable for data with high resolution due to the high cost of memory and computation. Some multi-resolution representations, e.g. octree [TDB17, WSLT18], are designed to overcome this problem, but the local invariant property
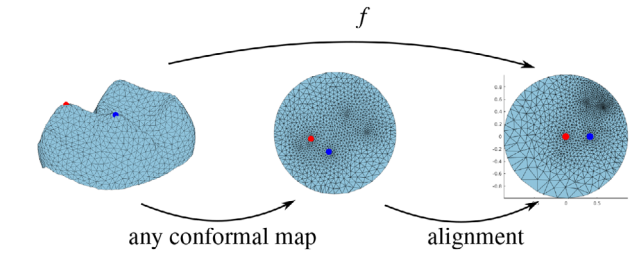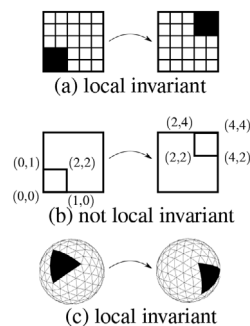
Figure 12: *For disk-like surfaces, given two landmark points there is a unique conformal map which maps the first point (red) to zero and maps the second one (blue) to the x-axis.*
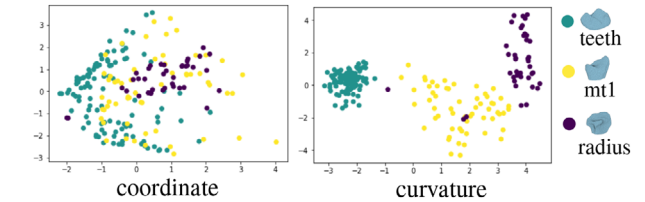

Figure 13: *Latent space visualization. The dataset is composed of three different types of anatomical surfaces. We project the latent space representation on a 2-dimensional space by PCA. Though all the shapes are packed without alignment, the three types of bones are clearly separated in the latent space. In contrast, the model based on the coordinate failed to learn the structure of the bones, so their distribution in the latent space is not well separated.*

does not hold anymore. In contrast, our model (sketched by (c)), together with the CNN on the sphere, provides an efficient way to learn the 3D data without a certain alignment. We verify our argument with the following two examples.

Learning unaligned anatomical data. We merge three different anatomical models in [BLC*11] and create the representations without any alignment methods. Insect shows the randomly generated bones of different types. Com-

pared with Figure 16 the bones get smoothed due to the expanded shape space. However, we show that our model is still capable to extract the meaningful information from the ambiguity by visualizing the latent space distribution (Figure 13). We compare the result with a baseline model that has the same network architecture but operates on the coordinate functions.
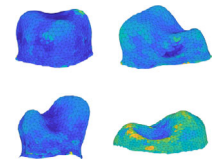
Generation of transformed cars. In this experiment we would like to see whether the 3D generative models are able to correctly predict shapes with various transformations. The dataset is created by randomly translating, rotating and scaling a single shape of car in the cube of size $[-1, 1] \times [-1, 1] \times [-1, 1]$. We train autoencoders based on different models on 900 training data and test them on 100 validation data. The comparison shows that our method produces more accurate predictions than others (see Figure 15). Since only our model considers the mesh structure of shapes, to make a fair comparison, we evaluate the results with Chamfer dis-
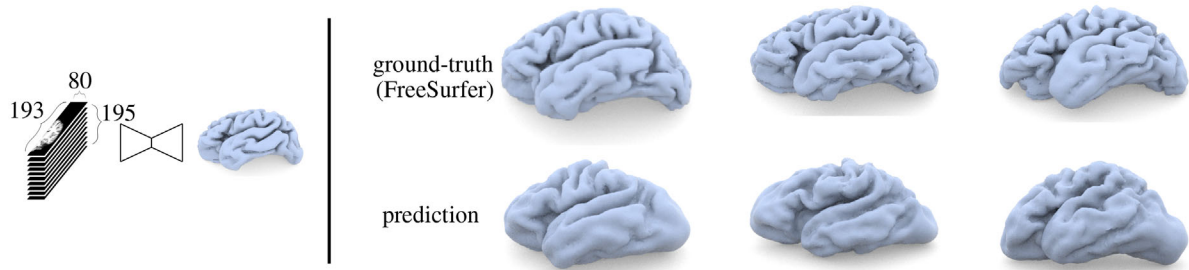
Figure 14: *Volume-Curvature autoencoder. The input is the MRI volumetric data from [MFC*10]. Since only the left hemisphere is generated, we align the volume using FreeSurfer [RRF10] and chop the volume properly such that the dimension is $193 \times 80 \times 195$. The encoder is shown in Figure 3 and the decoder is the same as Figure 1.*

tance which only depends on the underlying point clouds. Note that, as a trade-off, our representation loses the information of translation and scaling. Thus we first normalize the shapes reconstructed from our model and then calculate the Chamfer distance to the ground truth.

4.3. Cortical surface generation

To show that our model is particularly good at preserving the fine structure, we perform the experiment on human cortical surfaces, which are highly folded with a lot of "hills" and "valleys". A dataset of cortical surfaces are available on the Open Access Series of Imaging Studies (OASIS) [MFC*10]. The MRI images are converted to genus-0 surfaces via the open-source reconstruction software FreeSurfer (http://surfer.nmr.mgh.harvard.edu/).

We first compare our model to three other state-of-art autoencoders for 3D shapes. Figure 1 shows that, although all models succeed in characterizing the shapes in a large scale, our model preserves much more small features, e.g. the curvature, than the others.

Training details. Our model and the baseline model are trained with 200 epochs for around 5 h. The point-cloud AE [ADMG18] with 2048 points for each data and AtlasNet [GFK*18] with 2500 points for each data are both trained with 500 epochs for approximately 4 h. Although the point-cloud based models above have smaller data size than ours, the training of their neural networks already exhausted our GPU memory. The OGN, with the octree representation of $128 \times 128 \times 128$-dimensional voxels, is trained with 4000 epochs with 5 h. While other models produce the shapes instantly after training, it takes 2 min with our method to reconstruct a mesh with 10000 vertices from curvature.

Next, we compare the cortical surfaces randomly generated by our VAE to the ones by Multi-chart GAN [BHMK*18] (Figure 10). While both mesh-based models generate significantly more faithful results than other types of representation in Figure 1, the "hills" and "valleys" are much more visible with our model. Moreover, we only choose 3 landmark points on each shape to align the conformal parameterization, while it requires 21 landmark points to create 16 charts as in [BHMK*18], and even a template shape, which amounts to a dense correspondence, to reconstruct the final shapes.

At last, we try to create an autoencoder that converts the 3D MRI images of brain to cortical surfaces. In this case, the encoder consists of several 3D convolutional layers (see Figure 3) and the decoder is the same as the ones in previous experiments. Figure 14 shows that our model is able to predict the cortical surface from the MRI volume to a certain extent, but the accuracy is not yet optimal, because the neural network failed to capture the spatial correspondence between the volumetric data and the spherical data. We leave the construction of a finer 3D-to-2D autoencoder to future work.

5. Limitations and future work

First, currently it is difficult to model the shapes like long tubes, such as arms and legs of human, because the conformal parameterization of such shapes always has extremely large area distortion. The information easily gets lost while being transferred from such regions to the canonical domain (inset), unless one uses a domain with extremely high resolution. A solution might be a multi-resolution data structure, such as [GKS02, WSLT18]. Then it is desirable to design a structure of neural network that is specifically adapted to such multi-resolutional data structures.
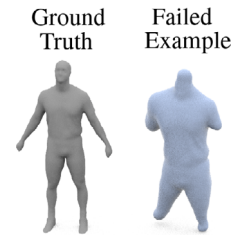
Second, to make our model fully rotational invariant rather than just local invariant, one might combine our representation with the equivariant neural networks by Cohen et al. [CGKW18], so that the alignment procedure can be completely removed. Then it would be interesting to develop a corresponding decoder network.

6. Conclusion

We propose a novel intrinsic representation of 3D surfaces based on mean curvature and metric. A 3D generative model is built based on this representation and it manifests better performance than other models in capturing the fine structure and the symmetry of the ambient space.

Acknowledgement

Appendix: A

A.1. Compute the weighted centroid of polygons

The weighted centroid of a polygon is given by

$$v^* = \frac{\int_V y\eth(y)dy}{\int_V \eth(y)dy}.$$

A Voronoi cell is naturally decomposed in several triangles, of which we first compute the weighted centroid.

Denote the density on the vertex v_i by $\eth_i$ and we assume that the density is linearly interpolated on every triangle. The denominator of v^* is called the weighted area, which is given by $\mathfrak{A}_i = \frac{\eth(v_1)+\eth(v_2)+\eth(v_3)}{3}A_i$, where A_i is the triangle area.
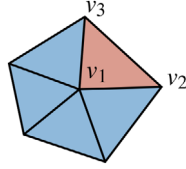
Integrating the linear function on the triangle i, we obtain

$$v_i^* = \frac{(2\eth_1 + \eth_2 + \eth_3)v_1 + (\eth_1 + 2\eth_2 + \eth_3)v_2 + (\eth_1 + \eth_2 + 2\eth_3)v_3}{4(\eth_1 + \eth_2 + \eth_3)}.$$

Then, the centroid of the polygon is the weighted sum

$$v^* = \frac{\sum_i v_i^* \cdot \mathfrak{A}_i}{\sum_i \mathfrak{A}_i}.$$

A.2. Proof of Theorem. 1 (Closing condition for prescribing the area factor)

Proof. Let (x, y) be a conformal coordinate of the immersion $f : M \to \mathbb{R}^3$. The left-hand side of (10) is actually
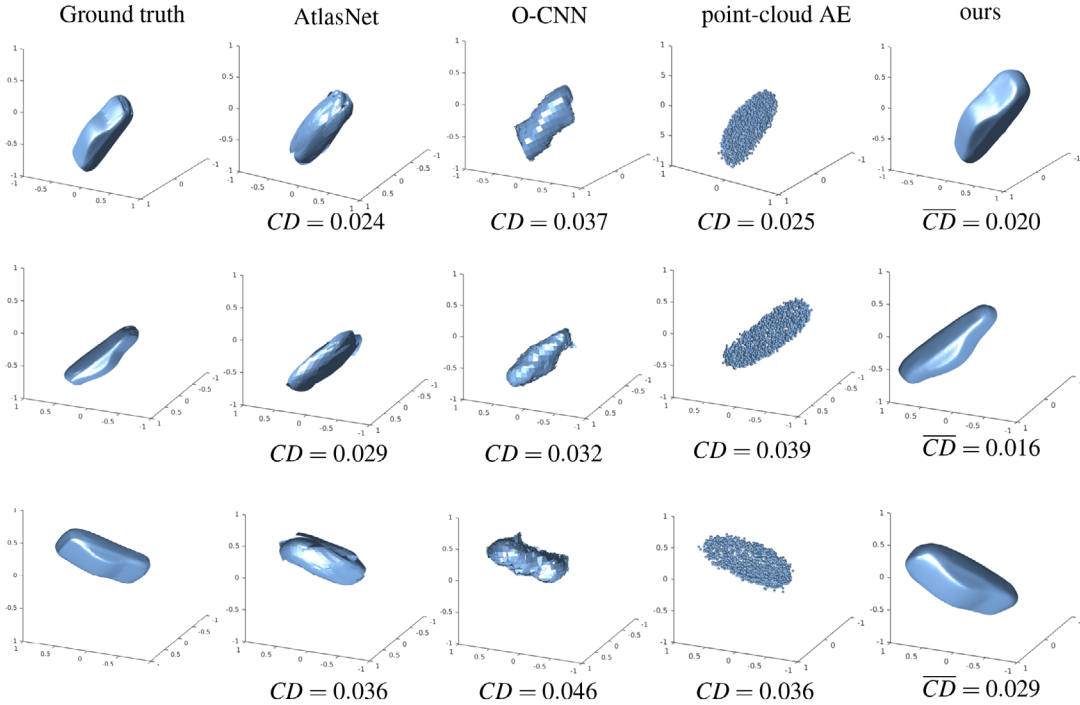
$$\phi_x \cdot \phi^{-1}dx + \phi_y \cdot \phi^{-1}dy,$$

while the right hand side reads

$$-\frac{1}{2}(-u_x f_x^{-1} - u_y f_y^{-1}) \cdot (f_x dx + f_y dy)$$

$$= \frac{1}{2}((u_x + u_y f_y^{-1} f_x)dx + (u_y + u_x f_x^{-1} f_y)dy)$$

$$= \frac{1}{2}((u_x + u_y n)dx + (u_y - u_x n)dy). \tag{A.1}$$

The equation (10) implies that

$$\phi_x \cdot \phi^{-1} = \frac{1}{2}(u_x + u_y n),$$

	Ground truth	AtlasNet	O-CNN	point-cloud AE	ours
		$CD = 0.024$	$CD = 0.037$	$CD = 0.025$	$\overline{CD} = 0.020$
		$CD = 0.029$	$CD = 0.032$	$CD = 0.039$	$\overline{CD} = 0.016$
		$CD = 0.036$	$CD = 0.046$	$CD = 0.036$	$\overline{CD} = 0.029$

Figure A.1: *Autoencoder for transformed cars. We transform a shape of car by applying random translation, scaling and rotation. We demonstrate our results with other models based on the point clouds, namely the point-cloud AE [ADMG18] and the AtlasNet [GFK*18], the one based on voxels, namely the O-CNN [WSLT18]. Other methods, though were shown to achieve satisfying results on the aligned dataset, do not correctly capture the symmetry of various transformations. In contrast, our model succeeds in producing convincing transformed shapes. We evaluate the results by measuring the Chamfer distance $\overline{CD}$. However, since our model loses the information of translation and scaling, we have to first normalize the volume of the results with a centered position (unnormalized shapes are shown above). In the end we compute the Chamfer distance of the normalized outputs $\overline{CP}$.*
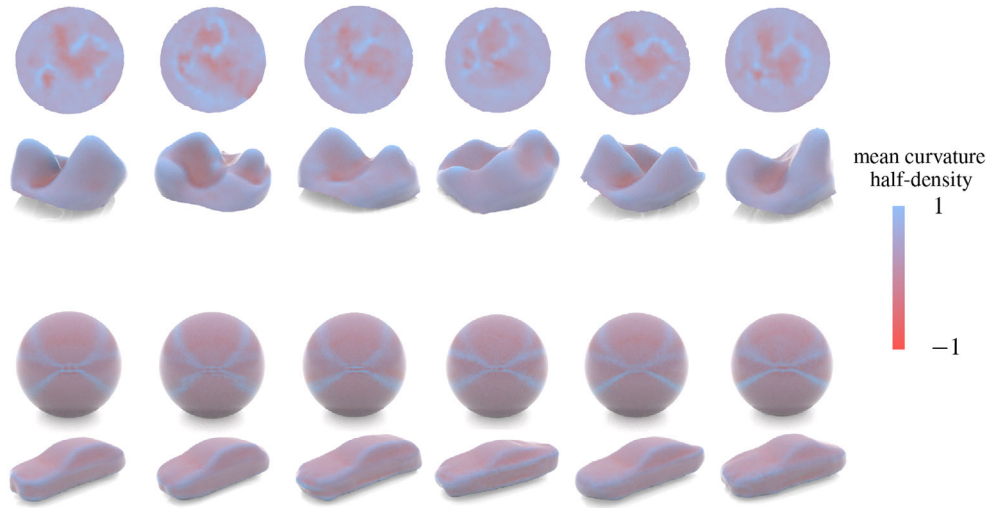
Figure A.2: *Randomly generated teeth and cars via the variational autoencoder. The first and third rows show the isotropic meshings, which are induced from the generated density function, with the generated mean curvature half-density. The second and fourth rows show the resulting reconstruction. The architectures of neural networks are modified from the traditional autoencoders in Table B.1 and B.2 to variational autoencoder.*

$$\phi_y \cdot \phi^{-1} = \frac{1}{2}(u_y - u_x n).$$

Substituting the equations above into the Dirac operator (2) in local form, we obtain

$$D_f \phi = f_x \phi_y - f_y \phi_x$$

$$= \frac{1}{2} f_x (u_y - u_x n) - \frac{1}{2} f_y (u_x + u_y n)$$

$$= 0,$$

by $f_x \cdot n = -f_y$ and $f_y \cdot n = -f_x$.   □

A.3. Finite element method for quaternion gradient

To obtain the discrete formula of the energy $|\omega^2|$, we first derive the formula of the quaternion gradient in discrete case.

Let $h : M \to \mathbb{R}$ be any function. We know that the gradient is defined by $\operatorname{grad} u := (du)^\sharp$, where $\sharp : T^*M \to TM$ is called raising indices defined by

$$\langle \omega^\sharp, v \rangle = \omega(v), \quad \text{for any } v \in TM$$

In a triangle i in quaternion space with the oriented edges $a, b, c \in \mathbb{H}$, we choose a coordinate (x, y) system (inset). Assuming that h is a linear function with the value h_1, h_2, h_3 at the vertices, write dh in local form as:

$$dh = (h_2 - h_1)dx + (h_3 - h_1)dy$$

Since $\langle (dx)^\sharp, \partial x \rangle = 1$ and $\langle (dx)^\sharp, \partial y \rangle = 0$, $df((dx)^\sharp)$ is perpendicular to b and has the length $\frac{1}{|c|\sin\theta} = \frac{|b|}{2A}$, where A is the area of the
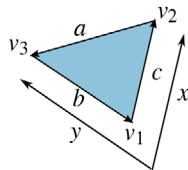
triangle. Thus $df(dx^\sharp) = \frac{n \cdot b}{2A}$ and, by the same argument, we have $df(dy^\sharp) = \frac{n \cdot c}{2A}$.

Therefore,

$$\operatorname{grad}_f h = \frac{n}{2A}(ah_1 + bh_2 + ch_3)$$

A.4. The energy of quaternion 1-form

We discretize the energy

$$E_u = |\omega|^2 = |d\phi + \frac{1}{2} Gdf\phi|^2$$

in the scheme of finite element method. In the local coordinate system above, the metric and its inverse read:

$$g = \begin{pmatrix} |c|^2 & -\langle c, b \rangle \\ -\langle c, b \rangle & b^2 \end{pmatrix}, \quad g^{-1} = \frac{1}{2A} \begin{pmatrix} b^2 & \langle c, b \rangle \\ \langle c, b \rangle & |c|^2 \end{pmatrix}.$$

With $\omega = \omega_x dx + \omega_y dy$, (11) becomes

$$\int (|\omega_x|^2 |b|^2 + \langle c, b \rangle (\overline{\omega_x}\omega_y + \overline{\omega_y}\omega_x) + |\omega_y|^2 |c|^2) dx \wedge dy.$$

Now, we work out the formula $\omega = d\phi + \frac{1}{2} Gdf\phi$ in one triangle:

$$\omega = \left((\phi_2 - \phi_1) + \frac{1}{2} G \cdot c ((1 - x - y)\phi_1 + x\phi_2 + y\phi_3) \right) dx$$

$$+ \left((\phi_3 - \phi_1) - \frac{1}{2} G \cdot b ((1 - x - y)\phi_1 + x\phi_2 + y\phi_3) \right) dy$$

where

$$G \cdot c = u_1 - u_2 + \frac{n}{2A}(-\langle a, c \rangle u_1 - \langle b, c \rangle u_2 - |c|^2 u_3)$$

$$G \cdot b = -u_1 + u_3 + \frac{n}{2A}(-\langle a, b \rangle u_1 - |b|^2 u_2 - \langle c, b \rangle u_3)$$

The energy E_u is a $|V| \times |V|$ quaternion-valued matrix. With a tedious calculation the entries related to the triangle are given by

$$|\omega|_{11}^2 = \frac{1}{2}|a|^2 - \frac{1}{6}(|a|^2 u_1 + \langle b, a \rangle u_2 + \langle c, a \rangle u_3) + \frac{1}{6}|G|^2 A^2,$$

$$|\omega|_{23}^2 = \frac{1}{2}\langle b, c \rangle + \frac{|G|^2 A^2}{12}$$

$$+ \frac{1}{12}((4An + |a|^2)u_1 - (a \cdot b)u_2 - (c \cdot a)u_3)$$

where

$$|G|^2 = \frac{1}{4A^2}(a^2 u_3^2 + b^2 u_2^2 + c^2 u_1^2$$

$$+ 2\langle a, b \rangle u_1 u_2 + 2\langle b, c \rangle u_2 u_3 + 2\langle c, a \rangle u_3 u_1).$$

Appendix B: Architectures

Table B1: *The architecture for spherical surfaces.*

Encoder		
layers	input	output
Conv2D (4 × 4)	320 × 32 × 32 × 2	320 × 32 × 32 × 4
BatchNormalization		
LeakyReLu		
Conv2D (4 × 4)	320 × 32 × 32 × 4	320 × 16 × 16 × 8
BatchNormalization		
LeakyReLu		
Conv2D (4 × 4)	320 × 16 × 16 × 8	320 × 8 × 8 × 16
BatchNormalization		
LeakyReLu		
Conv2D (4 × 4)	320 × 8 × 8 × 16	320 × 4 × 4 × 32
BatchNormalization		
LeakyReLu		
Conv2D (4 × 4)	320 × 4 × 4 × 32	320 × 2 × 2 × 64
BatchNormalization		
LeakyReLu		
Conv2D (4 × 4)	320 × 2 × 2 × 64	320 × 1 × 1 × 128
BatchNormalization		
LeakyReLu		
Reshape	320 × 1 × 1 × 128	80 × 512
FC	80 × 512	80 × 256
BatchNormalization		
LeakyReLu		
Reshape	80 × 256	20 × 1024
FC	20 × 1024	20 × 512
BatchNormalization		
LeakyReLu		
FC	20 × 512	200

Decoder		
layers	input	output
FC	200	20480
BatchNormalization		
LeakyReLu		

Table B1: *(Continued).*

Decoder		
Reshape	20480	20 × 1024
FC	20 × 1024	20 × 2048
BatchNormalization		
LeakyReLu		
Reshape	20 × 2048	80 × 512
FC	80 × 512	80 × 1024
BatchNormalization		
LeakyReLu		
Reshape	80 × 1024	320 × 2 × 2 × 64
Deconv2D (4 × 4)	320 × 2 × 2 × 64	320 × 4 × 4 × 32
BatchNormalization		
LeakyReLu		
Deconv2D (4 × 4)	320 × 4 × 4 × 32	320 × 8 × 8 × 16
BatchNormalization		
LeakyReLu		
Deconv2D (4 × 4)	320 × 8 × 8 × 16	320 × 16 × 16 × 8
BatchNormalization		
LeakyReLu		
Deconv2D (4 × 4)	320 × 16 × 16 × 8	320 × 32 × 32 × 4
BatchNormalization		
LeakyReLu		
Deconv2D (4 × 4)	320 × 32 × 32 × 4	320 × 32 × 32 × 2

Table B2: *The architecture for disk-like surfaces.*

Encoder		
layers	input	output
Conv2D (4 × 4)	256 × 256 × 2	128 × 128 × 4
BatchNormalization		
LeakyReLu		
Conv2D (4 × 4)	128 × 128 × 4	64 × 64 × 8
BatchNormalization		
LeakyReLu		
Conv2D (4 × 4)	64 × 64 × 8	32 × 32 × 16
BatchNormalization		
LeakyReLu		
Conv2D (4 × 4)	32 × 32 × 16	16 × 16 × 32
BatchNormalization		
LeakyReLu		
FC	16 × 16 × 32	100

Decoder		
layers	input	output
FC	100	8192
BatchNormalization		
LeakyReLu		
Reshape	8192	16 × 16 × 32
Deconv2D (4 × 4)	16 × 16 × 32	32 × 132 × 16
BatchNormalization		
LeakyReLu		
Deconv2D (4 × 4)	32 × 32 × 16	64 × 64 × 8

(Continues)

(Continues)

Table B2: *(Continued).*

Decoder		
BatchNormalization		
LeakyReLu		
Deconv2D (4×4)	$64 \times 64 \times 8$	$128 \times 128 \times 4$
BatchNormalization		
LeakyReLu		
Deconv2D (4×4)	$128 \times 128 \times 4$	$256 \times 256 \times 2$

Table B3: *The architecture for volumetric data.*

Encoder		
layers	input	output
Conv3D ($4 \times 4 \times 4$)	$193 \times 80 \times 195 \times 1$	$97 \times 40 \times 98 \times 4$
BatchNormalization		
LeakyReLu		
Conv3D ($4 \times 4 \times 4$)	$97 \times 40 \times 98 \times 4$	$49 \times 20 \times 49 \times 8$
BatchNormalization		
LeakyReLu		
Conv3D ($4 \times 4 \times 4$)	$49 \times 20 \times 49 \times 8$	$25 \times 10 \times 25 \times 16$
BatchNormalization		
LeakyReLu		
Conv3D ($4 \times 4 \times 4$)	$25 \times 10 \times 25 \times 16$	$13 \times 5 \times 13 \times 32$
BatchNormalization		
LeakyReLu		
Conv3D ($4 \times 4 \times 4$)	$213 \times 5 \times 13 \times 32$	$7 \times 3 \times 7 \times 64$
BatchNormalization		
LeakyReLu		
FC	$7 \times 3 \times 7 \times 64$	200

References

[AAB*15] ABADI M., AGARWAL A., BARHAM P., BREVDO E., CHEN Z., CITRO C., CORRADO G. S., DAVIS A., DEAN J., DEVIN M., GHEMAWAT S., GOODFELLOW I., HARP A., IRVING G., ISARD M., JIA Y., JOZEFOWICZ R., KAISER L., KUDLUR M., LEVENBERG J., MANÉ D., MONGA R., MOORE S., MURRAY D., OLAH C., SCHUSTER M., SHLENS J., STEINER B., SUTSKEVER I., TALWAR K., TUCKER P., VANHOUCKE V., VASUDEVAN V., VIÉGAS F., VINYALS O., WARDEN P., WATTENBERG M., WICKE M., YU Y., ZHENG X.: TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. URL: https://www.tensorflow.org/. Accessed: 2019-01-01

[ADMG18] ACHLIOPTAS P., DIAMANTI O., MITLIAGKAS I., GUIBAS L. J.: Learning representations and generative models for 3D point clouds. In *Proceedings of the 35th International Conference on Machine Learning 80* (2018), 40–49.

[ADVDI03] ALLIEZ P., DE VERDIERE E. C., DEVILLERS O., ISENBURG M.: Isotropic surface remeshing. In *2003 Shape Modeling International*. IEEE, New York, 2003, pp. 49–58.

[ASC11] AUBRY M., SCHLICKEWEI U., CREMERS D.: The wave kernel signature: A quantum mechanical approach to shape analysis. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference*. (IEEE, New York, 2011, pp. 1626–1633.

[BBL*17] BRONSTEIN M. M., BRUNA J., LECUN Y., SZLAM A., VANDERGHEYNST P.: Geometric deep learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine 34*, 4 (July 2017), 18–42.

[BCK18] BADEN A., CRANE K., KAZHDAN M.: Möbius registration. *Computer Graphics Forum 37*, 5 (2018), 211–220.

[BHMK*18] BEN-HAMU H., MARON H., KEZURER I., AVINERI G., LIPMAN Y.: Multi-chart generative surface modeling. In *SIGGRAPH Asia 2018 Technical Papers on - SIGGRAPH Asia '18)*. ACM Press, New York, 2018.

[BLC*11] BOYER D. M., LIPMAN Y., CLAIR E. S., PUENTE J., PATEL B. A., FUNKHOUSER T., JERNVALL J., DAUBECHIES I.: Algorithms to automatically quantify the geometric similarity of anatomical surfaces. *Proceedings of the National Academy of Sciences 108*, 45 (Oct 2011), 18221–18226.

[BMM*15] BOSCAINI D., MASCI J., MELZI S., BRONSTEIN M. M., CASTELLANI U., VANDERGHEYNST P.: Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks. *Computer Graphics Forum 34*, 5 (2015), 13–23.

[Bon67] BONNET O.: Memoire sur la theorie des surfaces applicables. *JEC Polyt 42* (1967), 27–29.

[BZSL14] BRUNA J., ZAREMBA W., SZLAM A., LECUN Y.: Spectral networks and deep locally connected networks on graphs. In *Proceedings of ICLR, 2014* (2014).

[CFG*15] CHANG A. X., FUNKHOUSER T., GUIBAS L., HANRAHAN P., HUANG Q., LI Z., SAVARESE S., SAVVA M., SONG S., SU H., XIAO J., YI L., YU F.: ShapeNet: An Information-Rich 3D Model Repository. *arXiv preprint arXiv:1512.03012*. Tech. Rep. arXiv:1512.03012 [cs.GR], 2015.

[CGKW18] COHEN T. S., GEIGER M., KÖHLER J., WELLING M.: Spherical CNNs. In *International Conference on Learning Representations (ICLR)* (2018). https://github.com/jonas-koehler/s2cnn

[CKPS18] CHERN A., KNÖPPEL F., PINKALL U., SCHRÖDER P.: Shape from metric. In *ACM Transactions on Graphics 37*, 4 (2018), 63:1–17.

[CL15] CHOI P. T., LUI L. M.: Fast disk conformal parameterization of simply-connected open surfaces. *Journal of Scientific Computing 65*, 3 (Feb 2015), 1065–1090.

[CLL15] CHOI P. T., LAM K. C., LUI L. M.: FLASH: Fast landmark aligned spherical harmonic parameterization for genus-0 closed brain surfaces. *SIAM Journal on Imaging Sciences 8*, 1 (Jan 2015), 67–94.

[CPS11] CRANE K., PINKALL U., SCHRÖDER P.: Spin transformations of discrete surfaces. In *ACM Transactions on Graphics 30*, 4 (2011), 104:1–10.

[CPS13] CRANE K., PINKALL U., SCHRÖDER P.: Robust fairing via conformal curvature flow. In *ACM Transactions on Graphics 32*, 4 (2013), 61:1–10.

[CPS15] CHERN A., PINKALL U., SCHRÖDER P.: Close-to-conformal deformations of volumes. In *ACM Transactions on Graphics 34*, 4 (July 2015), 56:1–13.

[DBV16] DEFFERRARD M., BRESSON X., VANDERGHEYNST P.: Convolutional neural networks on graphs with fast localized spectral filtering. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*. NIPS 16, Curran Associates Inc., Red Hook, NY, 2016, pp. 3844–3852.

[DFG99] DU Q., FABER V., GUNZBURGER M.: Centroidal voronoi tessellations: Applications and algorithms. *SIAM Review 41*, 4 (1999), 637–676.

[FAKG10] FUHRMANN S., ACKERMANN J., KALBE T., GOESELE M.: Direct resampling for isotropic surface remeshing. In *VMV* (2010), Citeseer, pp. 9–16.

[FSG17] FAN H., SU H., GUIBAS L. J.: A point set generation network for 3d object reconstruction from a single image. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), 2463–2471.

[GFK*18] GROUEIX T., FISHER M., KIM V. G., RUSSELL B., AUBRY M.: AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2018).

[GKS02] GRINSPUN E., KRYSL P., SCHRÖDER P.: CHARMS: a simple framework for adaptive simulation. *ACM Transactions on Graphics 21*, 3 (July 2002).

[GWC*04] GU X., WANG Y., CHAN T., THOMPSON P., YAU S.-T.: Genus zero surface conformal mapping and its application to brain surface mapping. *IEEE Transactions on Medical Imaging 23*, 8 (Aug 2004), 949–958.

[J*18] JACOBSON A., et al.: gptoolbox: Geometry processing toolbox, 2018. http://github.com/alecjacobson/gptoolbox. Accessed: 2019-01-01

[Kam98] KAMBEROV G.: Prescribing mean curvature: existence and uniqueness problems. *Electronic Research Announcements of the American Mathematical Society 4*, 2 (1998), 4–11.

[KJP*18] KOSTRIKOV I., JIANG Z., PANOZZO D., ZORIN D., BRUNA J.: Surface networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2018).

[KPP98] KAMBEROV G., PEDIT F., PINKALL U.: Bonnet pairs and isothermic surfaces. *Duke Mathematical Journal 92*, 3 (04 1998), 637–644.

[KW17] KIPF T. N., WELLING M.: Semi-supervised classification with graph convolutional networks. In *Proceedings of ICLR, 2017* (2017).

[LJC17] LIU D., JACOBSON A., CRANE K.: A dirac operator for extrinsic shape analysis. *Computer Graphics Forum (SGP) 36*, 5 (2017).

[MBBV15] MASCI J., BOSCAINI D., BRONSTEIN M. M., VANDERGHEYNST P.: Geodesic convolutional neural networks on riemannian manifolds. In *Proc. of the IEEE International Conference on Computer Vision (ICCV) Workshops* (2015), pp. 37–45.

[MBM*17] MONTI F., BOSCAINI D., MASCI J., RODOLA E., SVOBODA J., BRONSTEIN M. M.: Geometric deep learning on graphs and manifolds using mixture model CNNs. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (Jul 2017), IEEE.

[MFC*10] MARCUS D. S., FOTENOS A. F., CSERNANSKY J. G., MORRIS J. C., BUCKNER R. L.: Open access series of imaging studies: longitudinal mri data in nondemented and demented older adults. *Journal of Cognitive Neuroscience 22*, 12 (2010), 2677–2684.

[MGA*17] MARON H., GALUN M., AIGERMAN N., TROPE M., DYM N., YUMER E., KIM V. G., LIPMAN Y.: Convolutional neural networks on surfaces via seamless toric covers. *SIGGRAPH* (2017).

[NW17] NASH C., WILLIAMS C. K. I.: The shape variational autoencoder: A deep generative model of part-segmented 3D objects. *Computer Graphics Forum 36*, 5 (aug 2017), 1–12.

[Pin85] PINKALL U.: Regular homotopy classes of immersed surfaces. *Topology 24*, 4 (1985), 421–434.

[RRF10] REUTER M., ROSAS H. D., FISCHL B.: Highly accurate inverse consistent registration: a robust approach. *Neuroimage 53*, 4 (2010), 1181–1196.

[SA07] SORKINE O., ALEXA M.: As-rigid-as-possible surface modeling. In *Symposium on Geometry Processing* (2007), vol. *4*, pp. 109–116.

[SC17] SAWHNEY R., CRANE K.: Boundary first flattening. *ACM Transactions on Graphics 37*, 1 (Dec. 2017), 5:1–5:14.

[SM17] SMITH E. J., MEGER D.: Improved Adversarial Systems for 3D Object Generation and Reconstruction. In *Proceedings of the 1st Annual Conference on Robot Learning 78*, (2017), 87–96.

[SOG09] SUN J., OVSJANIKOV M., GUIBAS L.: A concise and provably informative multi-scale signature based on heat diffusion. In *Computer Graphics Forum* (2009), vol. *28*, Wiley Online Library, pp. 1383–1392.

[SSP08] SPRINGBORN B., SCHRÖDER P., PINKALL U.: Conformal equivalence of triangle meshes. In *ACM Transactions on Graphics (TOG)* (2008), vol. *27*, ACM, p. 77.

[TDB17] TATARCHENKO M., DOSOVITSKIY A., BROX T.: Octree Generating Networks: Efficient Convolutional Architectures for High-Resolution 3D Outputs. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (2017), 2088–2096.

[TPKZ18] TATARCHENKO M., PARK J., KOLTUN V., ZHOU Q.-Y.: Tangent convolutions for dense prediction in 3d. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2018).

[Ume17] UMETANI N.: Exploring generative 3D shapes using autoencoder networks. In *SIGGRAPH Asia 2017 Technical Briefs on - SA '17* (2017), ACM Press, New York.

[WLG*17] WANG P.-S., LIU Y., GUO Y.-X., SUN C.-Y., TONG X.: O-cnn: octree-based convolutional neural networks for 3D shape analysis. *ACM Transactions on Graphics 36* (2017), 72:1–72:11.

[WSLT18] WANG P.-S., SUN C.-Y., LIU Y., TONG X.: Adaptive o-CNN. In *SIGGRAPH Asia 2018 Technical Papers on - SIGGRAPH Asia '18* (2018), ACM Press, New York.

[WZX*16] WU J., ZHANG C., XUE T., FREEMAN W. T., TENENBAUM J. B.: Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In *Advances in Neural Information Processing Systems* (2016), pp. 82–90.

[YDT*18] YE Z., DIAMANTI O., TANG C., GUIBAS L., HOFFMANN T.: A unified discrete framework for intrinsic and extrinsic dirac operators for geometry processing. *Computer Graphics Forum 37*, 5 (2018), 93–106.

Supporting Information

Additional supporting information may be found online in the Supporting Information section at the end of the article.

Data Video S1

Data Video S2