Convolutional Neural Networks on 3D Surfaces Using Parallel Frames

HAO PAN, Microsoft Research Asia
SHILIN LIU, University of Science and Technology of China and Microsoft Research Asia
YANG LIU and XIN TONG, Microsoft Research Asia

We extend Convolutional Neural Networks (CNNs) on flat and regular domains (e.g. 2D images) to curved 2D manifolds embedded in 3D Euclidean space that are discretized as irregular surface meshes and widely used to represent geometric data in Computer Vision and Graphics. We define surface convolution on tangent spaces of a surface domain, where the convolution has two desirable properties: 1) the distortion of surface domain signals is locally minimal when being projected to the tangent space, and 2) the translation equi-variance property holds locally, by aligning tangent spaces for neighboring points with the canonical torsion-free parallel transport that preserves tangent space metric. To implement such a convolution, we rely on a parallel N-direction frame field on the surface that minimizes the field variation and therefore is as compatible as possible to and approximates the parallel transport. On the tangent spaces equipped with parallel frames, the computation of surface convolution becomes standard routine. The tangential frames have N rotational symmetry that must be disambiguated, which we resolve by duplicating the surface domain to construct its covering space induced by the parallel frames and grouping the feature maps into N sets accordingly; each surface convolution is computed on the N branches of the cover space with their respective feature maps while the kernel weights are shared.

To handle the irregular data points of a discretized surface mesh while being able to share trainable kernel weights, we make the convolution semi-discrete, i.e. the convolution kernels are smooth polynomial functions, and their convolution with discrete surface data points becomes discrete sampling and weighted summation. In addition, pooling and unpooling operations for surface CNNs on a mesh are computed along the mesh hierarchy built through simplification.

The presented surface-based CNNs allow us to do effective deep learning on surface meshes using network structures very similar to those for flat and regular domains. In particular, we show that for various tasks, including classification, segmentation and non-rigid registration, surface CNNs using only raw input signals achieve superior performances than other neural network models using sophisticated pre-computed input features, and enable a simple non-rigid human-body registration procedure by regressing to rest-pose positions directly.

CCS Concepts: • **Computing methodologies → Neural networks**;

Additional Key Words and Phrases: Curved Surface, Convolutional Neural Networks, Parallel Frames, Cover Space

1 INTRODUCTION

Convolutional Neural Networks (CNNs) have been widely used in diverse application fields for advanced statistical learning tasks, as they show great capability in modeling the latent complex behaviors of large scale datasets. In the field of geometric modeling and processing, recent works also apply CNNs to various tasks for large improvements over traditional methods, including 3D recognition, segmentation, correspondence, registration, etc. that are generally regarded as difficult due to the large number of factors and high-level semantics involved.

Geometric data can be represented in different forms. When the data is encoded by volumetric grids that regularly sample the 3D space, CNNs are straightforward to deploy. However, volumetric grids can be memory consuming and inflexible, especially for capturing fine-level geometric details. For representation efficiency, 3D objects and scenes are frequently encoded by their curved boundary surfaces, which are discretized as triangle meshes consisting of irregularly sampled 3D vertices. The curved surfaces and their irregular mesh representations, however, prohibit straightforward application of standard CNNs on flat domain with regular sampling grids, thus hindering deep learning on geometric datasets. To solve this problem, many methods have been developed.

The foundations for effective CNNs on machine learning tasks include utilizing raw input signals directly and sharing weight among different local regions for convolutions [Goodfellow et al. 2016; Lecun et al. 1998]. In particular, on flat image domains weight sharing is equivalent to the translation equi-variance property of convolutions. Unfortunately, previous works for CNNs on curved domains do not meet the fundamental requirements completely. For example, an approach for handling surface domains is to convert the 3D irregular geometric data into 2D regular images. Su et al. [2015] project the 3D shapes into many viewing planes, apply standard image-based CNNs on the projections, and finally pool the outputs from multiple views for predicting 3D shape category. Maron et al. [2017] globally parameterize a 3D surface of sphere topology into a flat 2D domain with toric topology where standard CNNs are applied. The problem with view-based projection or global parameterization is that the mappings from curved 3D surfaces to flat images occlude or distort the input signals in an unpredictable way and reduce the overall effectiveness. To avoid the problems with global projection or parameterization, there are works that define convolutions on localized geodesic patches [Boscaini et al. 2016; Masci et al. 2015; Monti et al. 2017] or patches projected on tangent spaces [Tatarchenko et al. 2018]. However, the convolutions on each patch are done individually without coordination, which loses the fundamental property of translation equi-variance that enables the effective sharing of trained convolution kernels.

In this paper, we propose a framework to extend standard CNNs to curved irregular surface domains, where the surface domain is mapped locally in a linearly optimal approach to avoid unnecessary distortion, and the effectiveness of standard CNNs is preserved by the local translation equi-variance of newly defined surface convolutions that enables meaningful weight sharing. In particular, the surface-based CNN framework works with localized surface patches (e.g. 1-ring neighbors of a mesh vertex) that are linearly approximated by being projected to the tangent plane of the central vertex, over which the surface convolution akin to standard convolution on flat domain is applied. On the tangent planes of nearby surface points, to enable meaningful weight sharing, we rely on the parallel transport of the canonical Levi-Civita connection that preserves tangent space metric to translate feature maps and convolution kernels, so that the convolution equi-variance property holds locally. The combined benefit of minimized distortion and surface domain translation equi-variance leads to outstanding performances on various deep learning tasks (Sec. 5).

To compute the surface convolutions efficiently, we use a point-wise tangential N-direction frame field that is constructed numerically for minimal variation and approximates the parallel transport. With canonical coordinates derived from the frame field equipping the tangent space, it becomes almost trivial to define surface convolutions on it. However, because the point-wise frame has N-th order rotational symmetry and unlike an image the surface domain has no default orientation, we disambiguate the symmetry by constructing the frame field induced N-cover space of the surface, on which coordinates for nearby points are properly aligned and the surface convolutions are evaluated.

In addition, to handle the irregular sampling points of a discretized surface mesh, we use a semi-discrete operation for surface convolution: the kernel function is restricted to continuous cubic polynomial functions, and the convolution with discrete points becomes a sampling of the continuous polynomial with the irregular points followed by their summation weighted by each point's share of surface area. Finally, supporting operations to form complete CNN structures for various tasks are developed. In particular, pooling and unpooling operations are naturally defined on hierarchical surface meshes that are efficiently constructed by simplification.

Overall, the surface CNNs thus defined resemble standard CNNs for which a lot of research on efficient designs and structures has been published and can be leveraged accordingly. The supporting structures of a parallel frame field and mesh hierarchy are efficient to build using existing techniques. The outstanding performances of surface CNNs taking raw input signals only are demonstrated and compared with previous approaches, on tasks of non-rigid shape classification, segmentation, and shape matching.

2 RELATED WORK

For geometric processing and modeling tasks, while there are many works on using 3D CNNs with data in volumetric or point cloud representations, we focus on reviewing CNNs and deep learning methods dealing with curved surface domains.

Manifold CNNs. A series of works extend standard CNNs to curved 2D manifold domains by applying convolution operations on localized geodesic patches, but differ more or less in their specific ways of convolution computation. Masci et al. [2015] parameterize each geodesic patch in polar coordinates, upon which the convolution operation is computed by rotating the kernel function for a set of discrete angles and convolving it with input features; to achieve independence of rotation angle sampling, the convolved features for different angles are further pooled for output. With such an approach, it is arguably hard to capture anisotropic signals. Thus later, Boscaini et al. [2016] proposed anisotropic CNNs that extend the geodesic convolutions by aligning the convolution kernels to frames of principal curvature direction; CNNs formed by such operators show improved performance on various tasks, including building correspondences between almost isometrically deformed shapes. A similar work by [Xu et al. 2017] uses directional convolutions aligned with principal curvature directions on surface patches of fixed face numbers which approximate geodesic patches, and applies them to non-rigid segmentation tasks. MoNet [Monti et al. 2017] further extends the previous approaches by modeling the convolution kernel as a mixture of Gaussians whose bases and coefficients are fully trainable rather than functions of fixed parameterizations, and obtains further improved performance.

Compared with these methods, our surface convolution is locally similar to the standard convolution, as it is evaluated on flat tangent spaces with common Cartesian coordinates. In addition, our framework properly handles the alignment of neighboring tangent spaces and convolutions by parallel transport, thus enabling the translation equi-variance property; in contrast, these previous works do not preserve this property because their convolutions lack the notion of translation and are evaluated individually for each patch. As such, their trained convolution kernels cannot be efficiently shared among different surface points. These differences lead to superior performance for our surface-based CNNs than the previous manifold CNNs (Sec. 5).

Global parameterization. Sinha et al. [2016] uses the geometry image parameterization to convert a 3D surface of sphere topology to the planar domain, upon which standard CNNs are used for shape recognition tasks. However, as noted by Maron et al. [2017], such a parameterization is not seamless across the cuts, for a surface there are large variations of the associated feasible parameterizations, and there is no translation equi-variance with the induced convolution operation. As an improvement, Maron et al. [2017] parameterize a surface of sphere topology to planar domain by first constructing a 4-cover of the surface defined by a cut of three points, and then conformally mapping the 4-cover to the flat domain with toric topology, where standard convolutions are applied with awareness of its topology. With such a construction, the authors show that the convolution is conformally translation equi-variant over the original surface. While it is obvious that this approach can only handle sphere-like surfaces, there is also the geometric problem of unavoidable distortions introduced by the conformal mapping that varies with cuts and 3D models, which distort input signals unevenly and unpredictably, thus degrading CNN performance. In comparison, our approach handles surfaces of arbitrary topology, and by construction the projection onto tangent plane is a linearly optimal mapping that preserves the original signals with minimal distortion. In addition, the local translation equi-variance enabled by using parallel frames holds for the 3D surface with its true metric that is not scaled spatial-variantly as in the conformal case. Experiments on a human body segmentation task show that our network outperforms the global parameterization approach (Sec. 5).

Tangent convolution. For surfaces represented by point clouds connected with k-nearest neighbors, Tatarchenko et al. [2018] define convolutions on tangent spaces of the surface points, which is similar to our framework. But similar to [Boscaini et al. 2016; Monti et al. 2017; Xu et al. 2017], they use the principal directions as coordinate frames of tangent planes, and do not consider the alignment of frames and features between neighboring vertices. In comparison, a major difference of our work is using parallel frames and sectioned feature maps to enable translation equi-variant convolutions on tangent planes of surfaces. Indeed, we have validated the impact of weight sharing enabled by using parallel frames (Sec. 5), and found our framework has much better results than a simplified construction as in [Tatarchenko et al. 2018].

Surface Network. Kostrikov et al. [2018] define deep neural networks on surfaces, where each trainable layer first applies the Laplacian or Dirac operator to input features of all vertices, and then transform the result with a trainable linear mapping. Such operators are different from the localized convolution operations of a CNN. It is not clear how the Surface Network performs on common tasks of classification, segmentation, and registration.

3 SURFACE CONVOLUTIONAL NEURAL NETWORKS

For a surface $\mathcal{M}$ embedded in $\mathbf{R}^3$, the tangent plane $T_p\mathcal{M}$ provides a linearly optimal approximation of the local surface geometry around point $p \in \mathcal{M}$, where it is possible to directly apply the usual convolutions for flat domains like images. However, the major problem with naively applying convolutions on tangent planes of a surface is the uncoordinated convolution for different surface points, which prevents effective weight sharing of the trainable convolution kernels as fundamental factors for CNN effectiveness [Goodfellow et al. 2016; Lecun et al. 1998]. To address this problem, our surface-based CNN framework starts from the fundamental property of translation equi-variance, and realizes it by the construction of parallel frames and sectioned feature maps on cover spaces, which leads to weight sharing and therefore effective CNNs for deep learning on surfaces.

3.1 Convolution on surface

To properly share the 2D convolutions on tangent planes of surface points, we start from the desired fundamental property, i.e. translation equi-variance, of standard convolutions, because translation effectively builds a way to connect the convolutions for different surface points. To be specific, between two nearby surface points $x, y \in \mathcal{M}$, we define translation equi-variance on surface as

$$T_{x,y}\left(W_x \circ F_x\right) = W_y \circ T_{x,y}(F_x),$$

where F_x is the feature map of the surface projected to the tangent plane of x, W_x the convolution kernel, $T_{x,y}$ the transport of tangent space from x to y, and $\circ$ denotes the convolution operation.

Transporting tangent spaces on a curved surface is well studied in differential geometry through the devices of connections and their associated parallel transports [Lee 1997]. In particular, the unique Levi-Civita connection is canonical in that it preserves metric and introduces zero torsion for translating nearby tangent spaces, therefore naturally fitting for convolutions on tangent spaces in a way similar to flat domain.

To enable efficient numerical computation that approximates the desirable properties of Levi-Civita connection, we build a tangential parallel frame field that quantizes the continuous orientations of tangent spaces and is as compatible to the connection as possible, for both defining tangent space convolutions and translating them while preserving the equi-variance property.

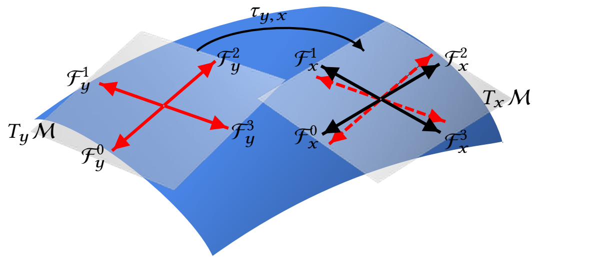

Fig. 1. Tangential 4-direction frames translated by parallel transport $\tau_{y,x}$ from y to x. A smooth frame field minimizes the difference between each pair of such frames. As a result, for every choice of coordinate axis $\mathcal{F}_x^i$ in $T_x\mathcal{M}$, the smooth frame field induces the corresponding axis $\mathcal{F}_y^i$ in $T_y\mathcal{M}$.

Parallel tangential frames. We construct point-wise coordinate frames of tangent planes, by first computing a parallel (or smooth) frame field over the surface, and then converting them to coordinate frames by choosing the bases.

Given a unit tangent vector $\mathbf{u}_x \in T_x\mathcal{M}$, consider rotating it by $k\frac{2\pi}{N}$, $k = 0, \cdots, N-1$ around the normal vector, so we have a rotationally symmetric N-direction frame $\{\mathbf{u}_x^i | i = 1, \cdots, N\}$ in the tangent space. By choosing each of the frame axis as the x-axis of coordinate frame, we effectively quantize all possible tangent space orientations into N bins.

We use a smooth field of N-direction frames over the surface to approximate Levi-Civita connection. To be specific, with the connection's parallel transport $\tau : T_x\mathcal{M} \to T_y\mathcal{M}$, the frames for two nearby points can be compared after the two tangent spaces are translated to coincidence (see Fig. 1 for an example with $N = 4$). We define the frame difference as the smallest distance between any two vectors of the two frames, i.e. $\nabla\mathbf{u} = \min_{i,j} \|\mathbf{u}_x^i, \tau_{y,x}(\mathbf{u}_y^j)\|$. A *smooth* N-direction field minimizes the field variation that integrates all frame differences over the surface:

$$\min \int_{\mathcal{M}} \|\nabla\mathbf{u}\|^2 \mathrm{d}\sigma.$$

Therefore, for two nearby frames of a smooth direction field, their matching induces a tangent space transformation that is as close to the Levi-Civita connection as possible, with an upper bound for its angular error from the ground truth connection equal to half the quantization bin width, i.e. $\frac{\pi}{N}$.

Given the frame field, there are N rotationally symmetric ways to define a canonical 2D coordinate frame at each point by assigning each frame axis as the x-axis; we denote the set of coordinate frames as $\{\mathcal{F}^i | i = 1, \cdots, N\}$. For two neighboring surface points x, y, as long as we utilize two consistent frames $\mathcal{F}_x, \mathcal{F}_y$ which are compatible with parallel transport (i.e. closest after translation, Fig. 1) as coordinate frames, equi-variance under translation that approximates the canonical connection is trivially achieved by defining the transport operator $T_{x,y}$ as the identity mapping and the convolution kernels $W_x \equiv W_y$ as in the flat $\mathbf{R}^2$ domain.

However, the reliance on pairwise frame consistency necessitates the grouping of feature maps and convolution operations over the surface, as discussed below.

Remark. While it is obvious that a larger N allows for tighter upper bound and better approximation of the true connection, it also comes with more computational cost for network training and

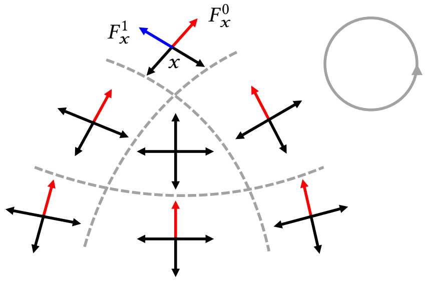

Fig. 2. A singularity of 4-direction field. After traveling a circular path around the central singular point, the frame at x has an angle defect of $\frac{\pi}{2}$ in this case, as the red axis would end up matching with the blue axis. It is therefore possible that the grouped feature maps F_x^0 corresponding to the red axis be blended with F_x^1 for the blue axis along the path.

evaluation. Sec. 5.2 provides discussion and numerical data about choosing the frame field symmetry order, where we find $N = 4$ to be a balanced choice. In the illustrations and experiments shown in the paper, we used $N = 4$ unless otherwise specified.

Grouped convolutions. For each surface point, we divide the feature map into N groups of equal size, denoted as $\{F^i | i = 1, \cdots, N\}$, that are associated with the coordinate frame $\{\mathcal{F}^i\}$. As such, the base surface $\mathcal{M}$, the coordinate frames and their corresponding grouped features form a fiber bundle whose fiber is a set of cardinality N. The consistent matching of neighboring frames induces N different sections, each of the form $\sigma^i : x \longrightarrow (x, \mathcal{F}_x^i, F_x^i)$, and satisfying that for a neighboring point $y \sim x$, $\sigma^i(y) = (y, \mathcal{F}_y^j, F_y^j)$ such that $\mathcal{F}_x^i, \mathcal{F}_y^j$ are consistent frames that are compatible with parallel transport. Finally, the grouped convolution is defined on each section of the fiber bundle:

$$F_{out,x}^i = W_x \circ_{\mathcal{F}_x^i} F_{in,x}^i,$$

where F_{in}^i is the i-th group of feature map before convolution, F_{out}^i after convolution, W_x the convolution kernel centered at x, and $\circ_{\mathcal{F}_x^i}$ the convolution operator defined in the tangent space equipped with coordinate frame $\mathcal{F}_x^i$.

Note that the same trainable convolution kernel W_x is shared by N different feature groups. Depending on applications, the grouped features will be pooled before the final output of a surface CNN (Sec. 3.2).

Remark. Another way of seeing how feature maps and convolutions are grouped is through branched covering. The base surface with the rotational symmetric frames define a branched N-cover of the base surface [Felix et al. 2007]. It is easy to see that the set of grouped feature maps actually are normal feature maps over the branched covering space, where the grouped convolutions are normal tangent space convolutions applied on the local patches of the covering space.

Singularities of frame field. As noted and discussed in many works (see [Vaxman et al. 2016] for a latest survey), a smooth frame field has singular points which generally correspond to holonomy of the Levi-Civita connection and whose singularity index sum is bounded to the domain topology due to Poincaré-Hopf theorem. In our surface convolution construction, the singular points do not pose particular difficulty, because the convolution operations in groups are defined locally based on the consistency of each nearby frame to the central frame. However, it should be noted that it is around the singular points that features of one group can blend into another group (see Fig. 2). Without proof, we conjecture that the blending as it happens in accordance to the curvature of the underlying surface is meaningful for the ultimate deep learning tasks.

3.2 Surface CNN structure

Given the sections of tangential frames and grouped feature maps $(\mathbf{x}, \mathcal{F}_x^i, F_x^i)$, $i = 1, \cdots, N$, there is no canonical ordering for them. However, when we evaluate a CNN for a surface domain, we expect the CNN output to be independent of the ordering of sections. Therefore, we construct surface CNNs with the following general structure to ensure the independence of section orders (Fig. 3):

(1) duplicate the input feature map into N groups,
(2) apply layers of grouped convolution with trainable weights shared among groups, and layers of pooling, unpooling and other operations that respect the sections,
(3) reduce the grouped feature maps into one feature map, which is the final output for testing, or evaluated against labeling data for training.

Examples of such surface CNN networks are shown in Figs. 4, 5, 6, 7, 9, where we denote grouped convolution on surface as GConv to distinguish it from standard convolutions. Next we elaborate on the other operations used in a surface CNN that work on the sections and grouped features.

Feature duplication and reduction. Duplicating input features defined for each surface point is straightforward, as it simply copies the features into N groups.

Reducing the grouped feature maps is generally in the form of max or average pooling of the grouped feature maps, i.e.

$$f_x = \text{Reduce}\{f_x^i\}, \quad i \in \{1, \cdots, N\}$$

where $f_x^i \in \mathbf{R}^k$ is the grouped feature vector of size k at surface point x (thus in total x has kN features), $f_x \in \mathbf{R}^k$ is the reduced feature vector, and Reduce$\{\cdot\}$ is the operation of taking maximum or average of all groups, for each component of the feature vector. By default, we use the maximum reduction; see examples in Figs. 6, 9. However, we can also make the reduction to be implicitly learned by standard 1x1 convolutions (Figs. 4, 7). Note that in these network illustrations we show the total feature size of a surface point for each tensor, which is the sum of all grouped feature maps.

Pooling and unpooling. Pooling and unpooling operations that transfer features between domains with different level-of-details in a hierarchy are also in a grouped manner. In particular, for a coarse level point y and its associated fine level points x, the pooling operation is

$$f_y^i = \text{Pool}\{f_x^j\},$$

where the j-th feature group of x is aligned with i-th feature group of y, i.e. the frame $\mathcal{F}_x^j$ out of $\mathcal{F}_x$ is closest to $\mathcal{F}_y^i$ by rotation. The Pool$\{\cdot\}$ operation takes component-wise average or maximum out of such matched feature vectors of the fine-level points. Unpooling operation is just the inverse of this process.

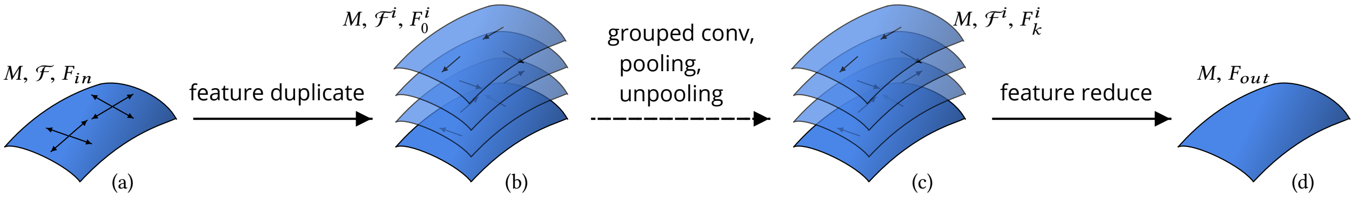
M, $\mathcal{F}$, F_{in}
feature duplicate
M, $\mathcal{F}^i$, F_0^i
grouped conv, pooling, unpooling
M, $\mathcal{F}^i$, F_k^i
feature reduce
M, F_{out}
(a)
(b)
(c)
(d)

Fig. 3. The structure of a surface CNN. From left to right: (a) the input surface M, its corresponding frame field $\mathcal{F}$, and input feature map F_{in}. (b) the feature map is duplicated N times and assigned to the N sections $(M, \mathcal{F}^i, F_0^i)$, $i = 1, \cdots, N$. (c) grouped convolution, pooling and unpooling operations are applied to the individual sections to get the k-th feature map. (d) the feature map on all sections is reduced to the output feature map F_{out} on the surface, which is used as final prediction or for training loss computation.

Grouped 1x1 convolution. Grouped 1x1 convolution is simply to map each group of the input feature map to an output map, thus preserving the group structure. Note the trainable convolution kernel is shared among all groups, as in the normal surface convolution. See Fig. 9 for an example.

4 IMPLEMENTATION

In this section, we present the concrete procedures to implement a surface CNN network. The differences from standard CNNs on images are mainly about constructing supporting data, and computing the grouped convolution, pooling and unpooling operations.

4.1 Parallel frame computation

Given a 3D surface mesh, the smooth frame field that approximates parallel transport of tangent spaces for neighboring points can be efficiently constructed.

In our implementation, we adopt the complex number based approach [Diamanti et al. 2014; Knöppel et al. 2013] to encode the N-direction fields. In particular, we identify the tangent plane T_pM at point p with the complex plane, and a set of unit length vectors $\{u \cdot e^{ik\frac{2\pi}{N}} | k = 0, \cdots, N-1\} \subset \mathbb{C}$ that form a symmetric N-direction frame can be conveniently encoded by their common N-th order power $v = u^N \in \mathbb{C}$. To compute a smooth frame field that is compatible with the Levi-Civita connection, we solve the following optimization problem:

$$\min_{\{v_i\}} \sum_{i\sim j} \|v_i - t_{ji}v_j\|^2 + \lambda \sum_i w_i \|v_i - v_i^0\|^2,$$

where i, j are neighboring vertices on the surface mesh, $t_{ji} \in \mathbb{C}$ is built from the discrete Levi-Civita connection that rotates the tangent plane of j to identify with that of i [Knöppel et al. 2013], λ is the weight for the second curvature direction alignment term, $w_i = \tanh(|k_{max} - k_{min}|)$ measures the anisotropy at i-th vertex using its maximum and minimum principle curvature values k_{max}, k_{min}, and v_i^0 is computed from the maximum curvature direction at the vertex. The first term is a simple discretization of the Dirichlet energy of the frame field that measures its variation. The second term encourages alignment of the frame field to strong anisotropic directions of the surface.

There are alternative formulations for computing the smooth frame field. For example, in [Knöppel et al. 2013] the optimization objective can contain the Dirichlet energy only, which is solved with a unit length constraint $\|v\| = 1$ as a minimum Eigenvalue problem. On the other extreme, we may discard the Dirichlet energy (or field smoothness) and use the frames of principal curvature directions directly. As validated in Sec. 5.1, we find that the trade-off between frame field smoothness and alignment to strong anisotropic surface features does not have obvious impact in a reasonable rage, while the extreme choices of smoothness only and curvature frames only have suboptimal performances. Therefore we use the above formulation with $\lambda = 0.01$ for all tasks shown in this paper.

While the problem is a simple quadratic optimization that can be solved efficiently, in practice we have further accelerated its computation through a multi-scale process. In particular, we build a hierarchy of surface meshes with different level-of-details for a given surface domain (Sec. 4.3); based on the surface hierarchy, we first compute the frame field with very few variables on the coarsest level surface mesh, and then copy it to the next level finer mesh as initialization, where the frame field can be solved with an iterative linear solver like Conjugate Gradient for very few iterations to convergence. The process goes on to the most detailed level.

4.2 Semi-discrete convolution for irregular data

The surface convolution takes features of spatially irregular data points as input, because the surfaces are represented by irregular meshes. On the other hand, the convolution kernels must be encoded in a uniform manner so that they can be shared for different locations on the surface mesh. To mitigate this conflict, we use a semi-discrete convolution operation.

The convolution kernel, denoted as $k_x^{i,j}(y) : y \in T_xM \to \mathbf{R}$ when evaluated at point x, belongs to the class of smooth polynomial functions of two variables. Convolution with features $f(y)$ of discrete and irregular points y in the local patch $P(x)$ then becomes a weighted sampling of the function

$$f^i(x) = \frac{1}{\sum w_y} \sum_j \sum_{y \in P(x)} w_y k_x^{i,j}(y) f^j(y),$$

where $f^i(x)$ is the i-th output feature of x, $f^j(y)$ the j-th input feature of y, w_y the vertex weight, which we computed as the 1-ring triangle area of y in the surface mesh. Next we give details about the computation of this operation.

Local patch. Similar to the standard convolution on regular image grids, the surface convolution works on a local patch of the domain. We define the local patch $P(x)$ to be the 1-ring neighboring vertices $y \sim x$ and x, although larger patches can be used, similar to the notion of larger kernel size in standard convolutions.

Local coordinate frame. For each surface point x, there are N rotationally symmetric frame directions $\mathcal{F}_x^i$, $i = 1, \cdots, N$ (Sec. 3.1).

Under each frame $\mathcal{F}_x^i = (\mathbf{e}_0, \mathbf{e}_1, \mathbf{n}_x)^T$ represented in its orthogonal bases ($\mathbf{n}_x$ is the surface normal), the vertex $y \in P(x)$ has local 3D coordinates $\frac{1}{r}\mathcal{F}_x^i(\mathbf{y}-\mathbf{x})$, whose first two coordinates in the tangent plane are used for evaluating the convolution kernels. Here r is the local patch radius for normalization, computed simply as the maximum of all projected neighbor offsets in the tangent plane.

Convolution kernel function. We use cubic polynomials of two variables as the convolution kernel functions. Formally, they are $k(u,v) = (1, u, v, u^2, uv, v^2, u^3, u^2v, uv^2, v^3)^T\mathbf{w}$, where $\mathbf{w} \in \mathbf{R}^{10}$ is the vector of trainable parameters. Note the size 10 is comparable to the size 9 of commonly used 3×3 convolution kernels on regular grids. We have tried with polynomials of other degrees, but find that cubic polynomial achieves a good balance between computational complexity and modeling capacity.

Input patch data. It is desirable that the input features for the surface CNNs include general low-level geometric quantities only while the CNNs output sophisticated signals that are task dependent. Similar to [Tatarchenko et al. 2018], for the input to surface CNNs, we include the normal vectors $\mathcal{F}_x^i\mathbf{n}_y$ and height values $\frac{1}{r}\mathbf{n}_x\cdot(\mathbf{y}-\mathbf{x})$ of all patch vertices for each vertex's local patch. Note these quantities do not vary under rigid transformations of the surface. In practice, we have used these input features along with a 1-channel constant input for all tasks shown in this paper. Since these patch vertex data are associated to patch vertices and have size unequal to the number of surface vertices, so we do not visualize them in network illustrations in Figs. 4,5,6,7,9.

4.3 Surface hierarchy and pooling/unpooling

Hierarchical structures on the surface are needed for quickly changing the receptive fields of convolutions through intermediate pooling and unpooling operations. We construct a hierarchy of surface meshes with different resolutions and level-of-details using the Quadratic Error Metric mesh simplification method [Garland and Heckbert 1997], although other methods [Hoppe 1996] can also be used. During the simplification process, we record the nesting relationship of a vertex v_p in the coarse mesh and the vertices $\{v_c\}$ in the fine mesh that are merged to form v_p; when doing pooling operation, grouped features of $\{v_c\}$ are aggregated to features of v_p, using maximization, averaging, etc., and the inverse is done for unpooling (Sec. 3.2).

Example networks using multiple level-of-detailed surfaces are shown in Fig. 6, 7, 9. We usually make the vertex numbers of different levels follow the pattern $N_{k+1} = N_k/r$, where N_k is the k-th level vertex number, and r is generally in the range $[2, 4]$.

5 DISCUSSION AND RESULTS

In this section, we first discuss how the parallel frames enable the effectiveness of our surface-based CNNs through two major ablation studies: in Sec. 5.1, we test the performances of surface-based CNNs with or without parallel frames and feature grouping; in Sec. 5.2 we discuss the impact of frame symmetry order on learning accuracy and computational cost. Then we present several applications of surface-based CNNs, including classification, segmentation and matching, of non-rigid deformable shapes in particular, for which CNNs designed for surfaces inherently have desirable properties of being invariant to rigid transformations and robust to non-rigid deformations. The results show that compared with previous methods which frequently rely on sophisticated precomputed shape features, our surface CNNs using raw input signals achieve outstanding performances on these tasks.

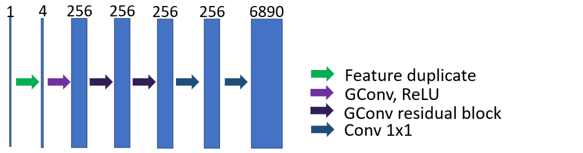

Fig. 4. The network structure used for human body registration through a classification of mesh vertices into 6890 classes. Each box represents a feature map of shape $N \times C \times V$, where $N = 1$ is batch size, C the size of feature channels (given by numbers aside the boxes), and V the number of surface vertices. The input feature map is a constant one for all vertices. The convolution through residual block contains two sequential residual blocks, with each block made by two convolutions that keep the input feature size. All convolution operations except the last one are followed with batch normalization and ReLU. The number of surface vertices is 6890 in this case.

5.1 Ablation test on frames

In this section, we first present the evaluation task, i.e. non-rigid human body registration through per-vertex classification, with which we carry out the experiments. Then we study how the core constructions of the presented surface CNNs, i.e. parallel frames and grouped convolutions, and different weight configurations for smooth frame field computation, affect performance.

Registration by classification task. The non-rigid registration problem we consider here is to match a template surface to an input shape through non-rigid deformations. In particular, here the registration is achieved by classifying each input mesh vertex into its target vertex on the template mesh by a convolutional network, as done in previous works [Boscaini et al. 2016; Fey et al. 2018; Masci et al. 2015].

We use a simple surface CNN with field symmetry order $N = 4$ that is made by a sequence of convolutions in the same level-of-detail for this task (Fig. 4). To train the network, we use the published part of FAUST dataset [Bogo et al. 2014] with 100 human bodies belonging to 10 subjects, each having 10 poses; for each shape, we make the network classify each vertex of the registered mesh into its ground truth template vertex label. 80 shapes are used for training, and 20 for testing.

The network with default configurations is trained on a single GPU for 400 epochs, using Adam solver [Kingma and Ba 2014] and a fixed learning rate 10^{-4}. We achieved 99.4% accuracy on the test set, slightly better than the best of previous results, 99.2% by [Fey et al. 2018]. On the other hand, our network is inherently invariant to rigid transformations of the input shapes, while the volumetric convolution approach by [Fey et al. 2018] is not and potentially needs expensive data augmentation to be resilient against rigid transformations.

In the following paragraphs, we discuss the performances of different configurations that add components of our surface CNN construction one-by-one onto a baseline model.

	SF	0.01	0.1	1	CF	CF-NG
Accuracy	0.985	0.994	0.994	0.989	0.963	0.942

Table 1. Testing accuracy of different frame alignment choices, on the FAUST non-rigid registration task by classification. SF means smoothness only, i.e. the frames are computed as the minimizer of the Dirichlet energy without alignment to other guidance. CF, curvature frames only, means the frames are simply the principle curvature directions. The numbers in-between are used as the curvature direction alignment weight λ for computing the smooth frame field (Sec. 4.1). CF-NG stands for curvature frames and no feature grouping, which is similar to frameworks of previous works [Boscaini et al. 2016; Monti et al. 2017; Tatarchenko et al. 2018; Xu et al. 2017].

Baseline model using principal curvature frames. When using principal curvature frames as coordinate frames without grouped feature maps, we have the construction similar to [Boscaini et al. 2016; Monti et al. 2017; Tatarchenko et al. 2018; Xu et al. 2017], with differences in how the numerical convolution with irregular vertices is defined: kernels in the shape of geodesic patches are used in [Boscaini et al. 2016], nearest neighbor sampling of a regular grid kernel is used in [Tatarchenko et al. 2018], and the semi-discrete convolution with polynomial kernels in our framework. To focus on the impact of the core constructions of parallel frames and feature grouping, for the baseline model we apply the same semi-discrete convolutions, but use the un-smoothed principal curvature frames and no feature grouping, to solve the human body registration task through per-vertex classification.

Using the network structure shown in Fig. 4 without feature duplication layer for this baseline, the trainable convolution kernel parameters are actually 16 times of those using feature grouping. However, the accuracy for the baseline configuration is 94.2% (Table 1, “CF-NG”), which is considerably worse than other configurations to be discussed.

Curvature frames with feature grouping. As a modification of the baseline model, we use feature grouping for principal curvature frames, with feature groups for neighboring vertices connected when their corresponding coordinate frames are closest after parallel transport. As shown in Table 1, “CF”, the result accuracy of using curvature frames with feature grouping is 96.3%, much higher than without for the baseline model, although the latter has 15 times more trainable parameters. Through this comparison, we see that feature grouping that resolves frame symmetry ambiguity is critical for performance improvement. On the other hand, the result accuracy of curvature frames with feature grouping is still significantly lower than other configurations using parallel frames as discussed next.

Frame field smoothness and alignment to anisotropy. For full constructions with parallel frames and grouped features, we test how different balances of field smoothness and alignment to strong anisotropy of the surfaces affect performances. We generate four different sets of frames for the registration task, using $\lambda = 0, 0.01, 0.1, 1$ respectively. The testing accuracies are reported in the first four columns of Table 1, where “SF”, meaning smooth frames without curvature direction alignment, corresponds to $\lambda = 0$. From the results, we see that while all smooth field configurations have superior performances than the curvature frames with or without feature grouping, a mild alignment to strong surface anisotropic directions is helpful in achieving the best performances. Based on the results, we have used $\lambda = 0.01$ for all tasks shown in other parts of the paper.

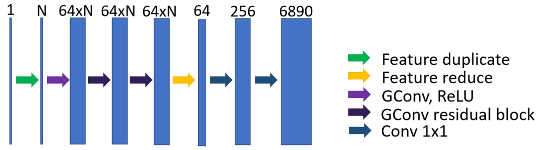

Fig. 5. A modified network from Fig. 4 for testing different frame field symmetry orders (denoted by N). Compared with the network in Fig. 4, the size of each grouped feature map for surface convolution layers remains 64, but there is an additional feature reduction layer before the 1×1 convolutions, which effectively reduces the first 1×1 conv layer’s trainable parameters to 64×256.

Symmetry	1	2	4	6	8
Accuracy	0.964	0.983	0.990	0.991	0.992
Time(ms)	47	85	142	207	275

Table 2. Testing accuracy of different frame field symmetry orders, on the FAUST non-rigid registration task by classification. The prediction time is measured on an NVidia GeForce Titan X GPU.

5.2 Ablation test on frame symmetry order

As discussed in Sec. 3.1, when the rotational symmetry order N of the frame field gets larger, the smooth field has reduced variation and is closer to the parallel transport, which provides a better approximation to Levi-Civita connection but leads to larger computational cost due to the greater number of sections (branched covers) and their corresponding grouped feature maps.

We tested the different choices of field symmetry orders on the FAUST non-rigid registration by classification task, where we modified the network structure to make sure each feature map group has the same size (i.e. 64), so that for different symmetry orders the amount of trainable parameters of each surface convolution layer remains the same ($64 \times 64 \times 10$), while the other parts of the network has exactly the same number of trainable parameters. The modified network structure is shown in Fig. 5; it differs from Fig. 4 by the final feature reduction layer before the 1×1 convolutions. The network has fewer trainable parameters than Fig. 4 for the first 1×1 convolution, i.e. 64×256 versus 256×256, and thus slightly reduced accuracy as reported next.

The performances of different symmetry orders are shown in Table. 2. From that we can see that the choice of $N = 4$ strikes a balance between accuracy and computational overhead: for $N < 4$ the accuracy is notably lower due to the larger deviation from parallel transport, and for $N > 4$ the computational cost which is roughly proportional to N is much higher. However, depending on the accuracy targets and computational budgets of different applications, the field symmetry order can be chosen adaptively.

5.3 Classification

We test on the SHREC’15 non-rigid shape classification challenge [Lian et al. 2015]. The published dataset has 1200 3D shapes in the form of surface meshes that belong to 50 categories. Some objects in one category are deformed versions of a common original object.

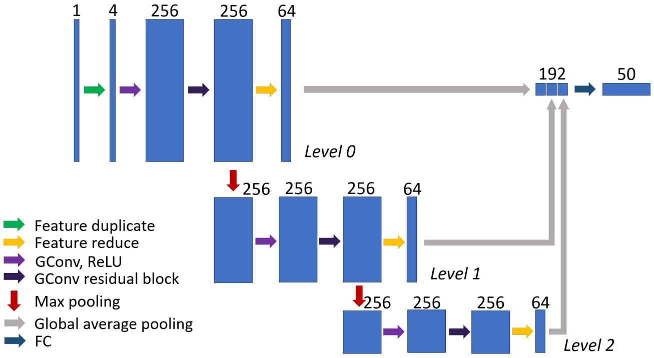

Fig. 6. Illustration of the network structure used for SHREC'15 non-rigid shape classification task. See caption of Fig. 4 for detailed explanation. Global average pooling is a standard average pooling over all vertices. For this dataset there are around 10k, 1700, 300 vertices for the three level-of-details respectively.

We use a three-level network for this task (Fig. 6). Following [Qi et al. 2017], we evaluate the network by five-fold cross validation. In each experiment, the network is trained for 95 epochs on four GPUs, using the Adam solver with batch size one and fixed learning rate 4×10^{-4}.

The testing accuracy for five experiments are 1, 0.996, 1, 0.988, 1, with an average 99.7%. Note our surface CNN achieves the near perfect accuracy while taking raw input signals (Sec. 4.2). In comparison, on the same task, PointNet++ [Qi et al. 2017] reports 60.18% for raw input signals, and 96.09% for using an ensemble of intrinsic shape features as input.

5.4 Segmentation

We test on the human body segmentation task published by [Maron et al. 2017]. The dataset contains 381 meshes of labeled human poses from various sources for training, and 18 meshes for testing. Because the meshes from various sources have very different resolutions and sizes, we normalize and remesh each mesh into 20 different meshes having 15k±750 vertices; ground truth segmentation labels are then assigned to the mesh vertices by projecting the mesh vertices to closest faces of the given dataset. The remeshing also serves as data augmentation, as the surface CNN directly takes the discrete meshes as input; as a comparison, in the global parameterization approach of [Maron et al. 2017], data augmentation is achieved by cutting a mesh in different ways which lead to differing conformal parameterizations to the toric domain.

The network structure used for this segmentation task is shown in Fig. 7. It has four level-of-details, with around 15k, 5100, 1750, 600 vertices for the four levels respectively. The network is trained for one epoch, using Adam solver on 2 GPUs with a fixed learning rate of 2×10^{-4}.

To obtain the predicted per-face segmentation labels, we sample points for each face of a test mesh and project the points onto closest vertices of our remeshed models, whose labels are used to vote for the face label of the original test mesh. Results of network predictions on test samples are shown in Fig. 8 along with ground truth labeling. Note that the ground truth labeling for different samples are not always consistent, which hinders the possibility of achieving very high accuracy. Still the segmentation accuracy on the test set, defined as the area of correctly labeled faces over all faces, is 90.4%, higher than the best accuracy of 88% by [Maron et al. 2017] which uses precomputed intrinsic features as input, while our network uses raw input signals. We argue that the improvement is due to our minimized distortion of signals by doing convolutions on localized surface patches rather than the global mapping where large distortion is unavoidable.

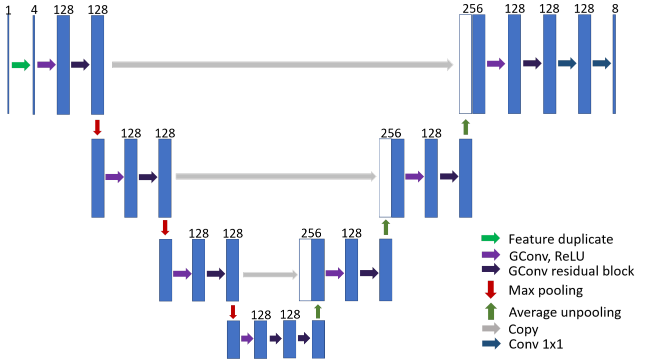

Fig. 7. The network structure used for human body segmentation task. See caption of Fig. 4 for detailed explanation. There are around 15k, 5100, 1750, 600 vertices for the four level-of-details.

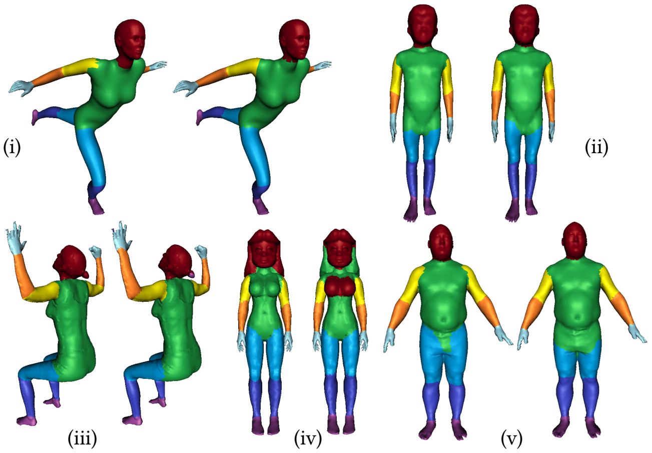

Fig. 8. Results of human body segmentation on the dataset from [Maron et al. 2017]. For each pair, the left one is the ground-truth labeling, and the right one is the predicted segmentation. Note in (i)(ii) that region boundaries of our results are sometimes more regular than ground truth labeling. (iv) shows a major failure case, since in training data there is no such models whose hair is glued to torso. (v) shows an inconsistent and noisy GT labeling, while our prediction is still reasonable.

5.5 Non-rigid registration by fitting template embedding

In this section we present an application that is aimed at resolving the non-rigid registration problem using an approach different from the per-vertex classification (Sec. 5.1). The new approach is proposed because in real applications we notice registration by classification has severe limitations: to classify each vertex to 6k classes for example is not scalable when there are many input vertices, and

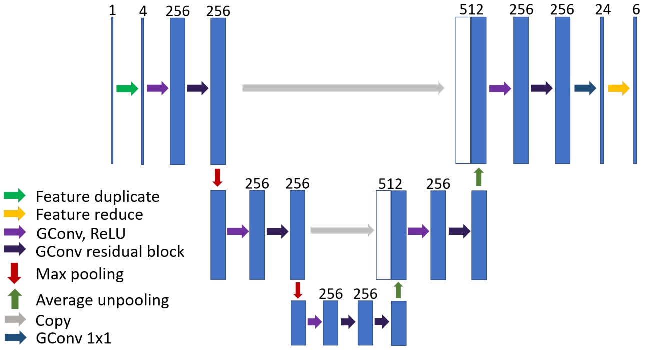

Fig. 9. The regression network structure used for human body regression task. See caption of Fig. 4 for detailed explanation. The number of surface vertices are around 10k, 3.2k, 1k for the three level-of-details respectively.

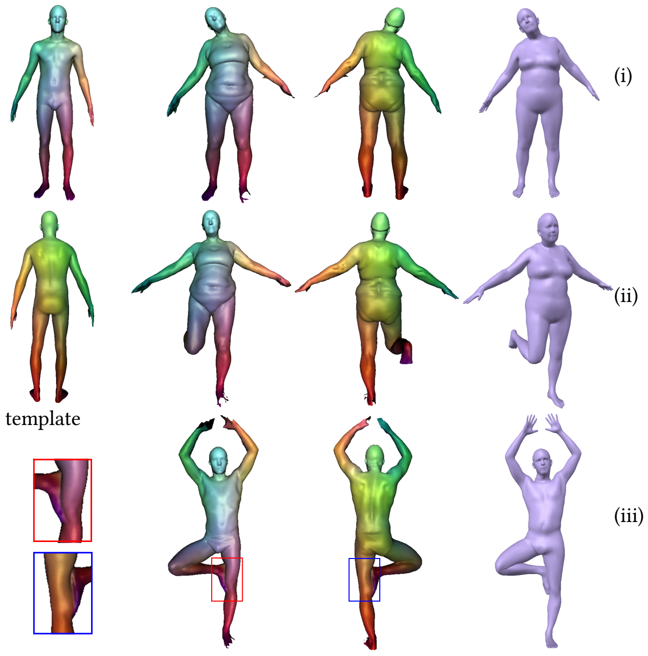

Fig. 10. Results of non-rigid human body registration through regression of the template embedding coordinates. The first column shows the template mesh color-coded by its point coordinates. Rows of (i) to (iii) are three different poses of two subjects, where each row shows the front and back view of the input scan mesh color-coded by coordinates of its regressed surface in the embedding space of the template, and the SMPL model fitted to the input scan using the correspondence built for the template. Defects with input scans are most obvious at the incomplete hands and feet. The input scan mesh of (iii) has its right foot glued to leg, which our network still distinguishes clearly (see the color difference in the zoomed regions).

the classification error does not measure at all how far away a misclassified vertex is from ground-truth. Thus here we present a new and simple method for non-rigid registration that uses a surface CNN for direct regression of the template embedding in $\mathbf{R}^3$.

As shown in Fig. 9, the network structure for point-wise regression is a standard encoder-decoder. For each vertex of an input raw scan mesh, the output contains the position and normal vectors of the corresponding point on the canonical template mesh. With the correspondence we do an inverse search of target points for each template vertex; the target points drive fitting the template mesh to the raw scan by providing correspondence for the skinned human body deformation model SMPL [Loper et al. 2015].

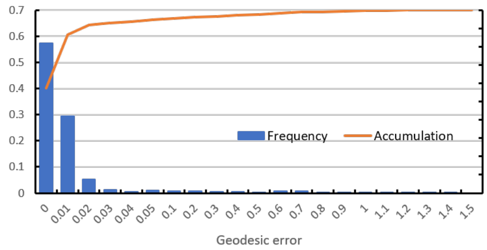

Fig. 11. Distribution of geodesic errors for non-rigid human body registration. The per-vertex error is the geodesic distance between the predicted point position and ground truth on the template surface, normalized by square root of surface area. The bar chart shows vertex distribution frequency for different ranges of geodesic errors, where a majority concentrates in the range under 0.03. The curve plots accumulation of the distribution, where it is shown the percentage of correspondences under 0.03 is actually 91.8%.

The network is trained on the FAUST dataset. For each real scan of the dataset there is a registered template mesh. Since the raw scans are very dense meshes, we have remeshed each raw scan to 10 simpler meshes with the number of vertices around 10k; the remeshing also provides data augmentation for network training. To build the training data, we project a vertex of the real scan to the closest point on the registered template mesh, and take its position and normal vectors on the rest pose template as the supervising regression target. The training loss is

$$
\begin{aligned}
L =& \frac{1}{V}\sum_{i}^{V}\left(\|\mathbf{p}_i - \mathbf{p}_i^0\|_1 + \frac{w_{reg}}{V_i}\sum_{j\sim i}\|\mathbf{p}_i - \mathbf{p}_j\|_1\right)\\
&+ \frac{w_n}{V}\sum_{i}^{V}\left(\|\mathbf{n}_i - \mathbf{n}_i^0\|_1 + \frac{w_{reg}}{V_i}\sum_{j\sim i}\|\mathbf{n}_i - \mathbf{n}_j\|_1\right)\\
&+ \frac{w_{con}}{E}\sum_{i\sim j}|\mathbf{n}_i\cdot(\mathbf{p}_i - \mathbf{p}_j)|,
\end{aligned}
$$

where V is the number of vertices of the raw scan mesh, $\mathbf{p}$ the regressed vertex position, $\mathbf{p}^0$ the target position, $\mathbf{n}$ the regressed vertex normal, $\mathbf{n}^0$ the target normal, $w_n = 0.1$ to normalize different scales between position and normal in the dataset, $w_{reg} = 0.2$ the weight for Laplacian regularization terms of position and normal, $w_{con} = 20$ the weight for normal and position consistency, V_i the number of neighboring vertices of the i-th vertex, and E the number of directed mesh edges. We use l_1 norm for these losses because there are noisy vertices in the raw scans which do not have valid target points on the template surface. We train the network for 300 epochs on 4 GPUs using Adam solver with a fixed learning 4×10^{-4}.

Geodesic errors of the network predictions on the test set are shown in Fig. 11. Following [Kim et al. 2011], the geodesic error for a surface point x with predicted position y and ground truth point y^* on the template surface $\mathcal{M}$ is computed as $\epsilon(x) = \frac{d_{\mathcal{M}}(y,y^*)}{\sqrt{|\mathcal{M}|}}$, where

$d_{\mathcal{M}}(\cdot,\cdot)$ computes the geodesic distance of two points projected onto the surface $\mathcal{M}$, and $|\mathcal{M}|$ is its area for normalization.

Visual results are shown in Fig. 10. The predictions by our network can be directly used for fitting the parametric SMPL model to the raw scans (since only a subset of SMPL model bases are publicly available, very fine details of raw scans cannot be fitted). We notice that unlike previous descriptor-based methods for non-rigid registration which use expensive post-processing like functional maps [Litany et al. 2017] or Markov Random Fields [Chen and Koltun 2015] to filter and aggregate the dense correspondences, our approach of directly regressing the spatial coordinates of template surface is simple and fast, requiring one network forward pass and one solving of the SMPL fitting model.

6 CONCLUSION

In this paper, we have presented an effective framework for defining CNNs for deep learning on curved 3D surfaces. The basic surface convolution operation extends the standard convolution on regular flat domains to curved and irregular surfaces, while preserving its fundamental properties of utilizing raw input signals and translation equi-variance for sharing weight. The extension is made possible by using parallel frames of surface tangent planes. In particular, the coordinates derived from the frames make it straightforward to define convolutions on tangent spaces, and the parallelism (or smoothness) of the frames makes them compatible with and well approximate the metric preserving Levi-Civita connection that defines the notion of translation on surfaces. To handle the rotational symmetry of N-direction frames, we use grouped convolutions on the individual sections (covers) induced by the parallel frames. Pooling, unpooling and other operations for the surface-based CNNs are also developed to support the grouped feature maps.

The surface CNNs and supporting data can be efficiently computed on meshes discretizing the surfaces. To handle the irregular sampling of surface meshes, we use a semi-discrete convolution operation where the trainable convolution kernel belongs to cubic polynomials and is convolved with discrete sampling vertices. We build the hierarchy of surface meshes through simplification, upon which both pooling and unpooling operations are computed and frame fields efficiently constructed.

We evaluated the surface CNNs on various tasks involving classification, segmentation and non-rigid registration. Results show that the surface CNNs have superior performance than previous methods, and are inherently invariant to rigid transformations and robust to non-rigid deformations.

Limitations and future work. The meshing of surfaces has significant influence on the surface-based CNNs. In particular, if the surface meshes are of very poor quality (e.g. containing many triangles of high-contrasting sizes and extreme aspect ratios), or they differ systematically between training and testing datasets, it would be difficult to learn meaningful and generalizable semi-discrete convolution kernels, therefore degrading the performance. In the future, we would like to improve the current semi-discrete convolution computation to make it more robust to the mesh sampling variations.

3D surfaces may also be represented as point clouds, which our framework should be able to handle with minor extension as long as locally the surface patches can be constructed, e.g. by k nearest neighbors. Note that the computation of smooth N-direction frame fields and point cloud hierarchy have efficient techniques that are in spirit similar to those for triangle meshes.

REFERENCES

Federica Bogo, Javier Romero, Matthew Loper, and Michael J. Black. 2014. FAUST: Dataset and evaluation for 3D mesh registration. In *CVPR*.

Davide Boscaini, Jonathan Masci, Emanuele Rodolà, and Michael Bronstein. 2016. Learning shape correspondence with anisotropic convolutional neural networks. In *NIPS*. 3189–3197.

Q. Chen and V. Koltun. 2015. Robust Nonrigid Registration by Convex Optimization. In *ICCV*. 2039–2047.

Olga Diamanti, Amir Vaxman, Daniele Panozzo, and Olga Sorkine-Hornung. 2014. Designing N-PolyVector Fields with Complex Polynomials. *Computer Graphics Forum* 33, 5 (Aug. 2014), 1–11.

Kälberer Felix, Nieser Matthias, and Polthier Konrad. 2007. QuadCover - Surface Parameterization using Branched Coverings. *Computer Graphics Forum* 26, 3 (2007), 375–384.

Matthias Fey, Jan Eric Lenssen, Frank Weichert, and Heinrich Müller. 2018. SplineCNN: Fast Geometric Deep Learning with Continuous B-Spline Kernels. In *CVPR*.

Michael Garland and Paul S. Heckbert. 1997. Surface Simplification Using Quadric Error Metrics. In *SIGGRAPH*. 209–216.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Hugues Hoppe. 1996. Progressive Meshes. In *SIGGRAPH*. 99–108.

Vladimir Kim, Yaron Lipman, and Thomas Funkhouser. 2011. Blended Intrinsic Maps. *ACM Trans. Graph. (SIGGRAPH)* 30, 4 (July 2011).

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. In *ICLR*.

Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. 2013. Globally Optimal Direction Fields. *ACM Trans. Graph. (SIGGRAPH)* 32, 4, Article 59 (2013), 10 pages.

Ilya Kostrikov, Zhongshi Jiang, Daniele Panozzo, Denis Zorin, and Burna Joan. 2018. Surface Networks. In *CVPR*.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (Nov 1998), 2278–2324.

John M. Lee. 1997. Connections. In *Riemannian Manifolds*. Springer New York, 47–64.

Z. Lian, J. Zhang, S. Choi, H. ElNaghy, J. El-Sana, T. Furuya, A. Giachetti, R. A. Guler, L. Lai, C. Li, H. Li, F. A. Limberger, R. Martin, R. U. Nakanishi, A. P. Neto, L. G. Nonato, R. Ohbuchi, K. Pevzner, D. Pickup, P. Rosin, A. Sharf, L. Sun, X. Sun, S. Tari, G. Unal, and R. C. Wilson. 2015. Non-rigid 3D Shape Retrieval. In *Eurographics Workshop on 3D Object Retrieval*. The Eurographics Association. http://www.icst.pku.edu.cn/zlian/shrec15-non-rigid/

Or Litany, Tal Remez, Emanuele Rodolà, Alex Bronstein, and Michael Bronstein. 2017. Deep Functional Maps: Structured Prediction for Dense Shape Correspondence. In *ICCV*.

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. 2015. SMPL: A Skinned Multi-Person Linear Model. *ACM Trans. Graph. (SIGGRAPH Asia)* 34, 6 (Oct. 2015), 248:1–248:16.

Haggai Maron, Meirav Galun, Noam Aigerman, Miri Trope, Nadav Dym, Ersin Yumer, Vladimir G. Kim, and Yaron Lipman. 2017. Convolutional Neural Networks on Surfaces via Seamless Toric Covers. *ACM Trans. Graph. (SIGGRAPH)* 36, 4, Article 71 (July 2017), 10 pages.

J. Masci, D. Boscaini, M. M. Bronstein, and P. Vandergheynst. 2015. Geodesic Convolutional Neural Networks on Riemannian Manifolds. In *ICCV*. 832–840.

Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodolà, Jan Svoboda, and Michael M. Bronstein. 2017. Geometric deep learning on graphs and manifolds using mixture model CNNs. In *CVPR*.

Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. 2017. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *NIPS*.

Ayan Sinha, Jing Bai, and Karthik Ramani. 2016. Deep Learning 3D Shape Surfaces Using Geometry Images. In *ECCV*. 223–240.

Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik G. Learned-Miller. 2015. Multi-view convolutional neural networks for 3d shape recognition. In *ICCV*.

Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, and Qian-Yi Zhou. 2018. Tangent Convolutions for Dense Prediction in 3D. In *CVPR*.

Amir Vaxman, Marcel Campen, Olga Diamanti, Daniele Panozzo, David Bommes, Klaus Hildebrandt, and Mirela Ben-Chen. 2016. Directional Field Synthesis, Design, and Processing. *Computer Graphics Forum* (2016).

H. Xu, M. Dong, and Z. Zhong. 2017. Directionally Convolutional Networks for 3D Shape Segmentation. In *ICCV*. 2717–2726.