DualSDF: Semantic Shape Manipulation using a Two-Level Representation

Zekun Hao, Hadar Averbuch-Elor, Noah Snavely, Serge Belongie
Cornell Tech, Cornell University

Figure 1: DualSDF represents shapes using two levels of granularity, allowing users to manipulate high resolution shapes (odd rows) with high-level concepts through manipulating a proxy primitive-based shape (even rows). Simple editing operations on individual primitives (colored in blue) are propagated to the other primitives and the fine-grained model in a semantically meaningful manner. Above, we illustrate how an existing shape (inside the red box) can be modified semantically by adjusting the radius of a primitive (fuselage diameter on the airplane) or the distance between two primitives (wheelbase of a car).

Abstract

We are seeing a Cambrian explosion of 3D shape representations for use in machine learning. Some representations seek high expressive power in capturing high-resolution detail. Other approaches seek to represent shapes as compositions of simple parts, which are intuitive for people to understand and easy to edit and manipulate. However, it is difficult to achieve both fidelity and interpretability in the same representation. We propose DualSDF, a representation expressing shapes at two levels of granularity, one capturing fine details and the other representing an abstracted proxy shape using simple and semantically consistent shape primitives. To achieve a tight coupling between the two representations, we use a variational objective over a shared latent space. Our two-level model gives rise to a new shape manipulation technique in which a user can interactively manipulate the coarse proxy shape and see the changes instantly mirrored in the high-resolution shape. Moreover, our model actively augments and guides the manipulation towards producing semantically meaningful shapes, making complex manipulations possible with minimal user input.

1. Introduction

There has been increasing interest in leveraging the power of neural networks to learn expressive shape representations for high-fidelity generative 3D modeling [4, 20, 52, 38, 34]. At the same time, other research has explored parsimonious representations of shape as compositions of primitives [50, 11] or other simple, abstracted elements [19, 10]. Such shape decompositions are more intuitive than a global, high-dimensional representation, and more suitable for tasks such as shape editing. Unfortunately, it is difficult to achieve both fidelity and interpretability in a single representation.

In this work, we propose a generative two-level model that simultaneously represents 3D shapes using two levels of granularity, one for capturing fine-grained detail and the other encoding a coarse structural decomposition. The two levels are tightly coupled via a shared latent space, wherein a single latent code vector decodes to two representations of the same underlying shape. An appealing consequence
is that modifications to one representation can be readily propagated to the other via the shared code (as shown in Figure 1 and Figure 2).

The shared latent space is learned with a variational auto-decoder (VAD) [53]. This approach not only imposes a Gaussian prior on the latent space, which enables sampling, but also encourages a compact latent space suitable for interpolation and optimization-based manipulation. Furthermore, as we empirically demonstrate, compared to an auto-encoder or auto-decoder, our model enforces a tighter coupling between different representations, even for novel shapes.

Another key insight is that implicit surface representations, particularly signed distance fields (SDFs) [38, 34, 9], are an effective substrate for both levels of granularity. Our coarse-level representation is based on the union of simple primitives, which yield efficient SDF formulations. Our fine-scale model represents SDFs with deep networks and is capable of capturing high-resolution detail [38]. In addition to other desirable properties of implicit shape formulations, expressing both representations under a unified framework allows for simpler implementation and evaluation.

We show that our two-level approach offers the benefits of simplicity and interpretability without compromising fidelity. We demonstrate our approach through a novel shape manipulation application, where a shape can be manipulated in the proxy primitive-based representation by editing individual primitives. These editions are simultaneously reflected to the high-resolution shape in a semantically meaningful way via the shared latent code. Moreover, minimal user input is needed to achieve complex shape manipulation. Under our optimization-based manipulation scheme, sparse edits on a subset of primitives can be propagated to the rest of the primitives while maintaining the shape on the manifold of likely shapes. Such an approach to manipulation is much more intuitive than a direct editing of the high-resolution mesh using deformation tools. A user can simply drag individual primitives in 3D to edit the shape (e.g. Figure 2) while observing the rest of the primitives and the high resolution shape change accordingly at an interactive rate.

Last, we introduce two novel metrics for evaluating the manipulation performance of our model: cross-representation consistency and primitive-based semantic consistency. These metrics provide insights on how well the two representations agree with each other as well as how consistent the primitives are across different shapes. Code is available at https://github.com/zekunhao1995/DualSDF.

2. Related Work

Generative 3D modeling. Prior to the Deep Learning era, 3D modeling of a shape collection was typically performed on a mesh representation. Many methods focus specifically on human models [2, 40, 17], and aim at modeling deformations of a template model. The main limitation of most mesh-based representations, modern ones included, is that they are limited to meshes sharing the same connectivity [32, 49]. Recently, Gao et al. [18] proposed a technique to generate structured deformable meshes of a shape collection, which overcomes the same-connectivity constraint. However, part annotations are needed for training their model.


Point clouds are also widely used in representing 3D shapes due to their simplicity. Following the pioneering work of Fan et al. [13], many common generative models have been applied to point clouds, including generative adversarial networks [1, 31], adversarial autoencoders [54], flow-based models [52] and autoregressive models [48]. However, as point clouds do not describe the shape topology, such techniques can produce only relatively coarse geometry. Furthermore, compared to primitive based representations, they are less expressive and require considerably more points to represent shapes at a similar level of detail, making them less suitable for user interaction.

Implicit representations have recently shown great promise for generative 3D modeling [38, 34, 9]. These methods model shapes as isosurfaces of functions. Generally, models within this category predict the condition of sampled 3D locations with respect to the watertight shape surface.
with simple transformations, thus their method cannot guide (e.g., inside/outside). Unlike explicit surface representations and point cloud representations, shapes are modeled as volumes instead of thin shells. Such models have been successfully applied to a variety of applications including shape generation, completion, and single-view reconstruction. As demonstrated in prior work, they are capable of representing shapes with high level of detail.

3D modeling with primitive shapes. Reconstructing surfaces using simple primitives has long found application in reverse engineering [5], and more generally in the computer vision and graphics communities [41, 6, 44]. Among other use cases, prior work has demonstrated their usefulness for reconstructing scanned [16] or incomplete [43] point clouds.

Several primitive types have been proposed for modeling 3D shapes using neural networks, including cuboids [50, 46], superquadrics [39], anisotropic 3D Gaussian balls [19], and convex polytopes [10]. Deprelle et al. [11] learn which primitives best approximate a shape collection.

Hybrid and hierarchical representations. Hybrid representations benefit from the complementary nature of different representations. There are prior works that assume a shared latent space across different representations and combine voxel-based, image-based, and point-based representations for various discriminative tasks, including 3D classification and segmentation [25, 47, 36]. However, none of them has addressed the problem of shape generation and manipulation.

Some previous works learn representations in several different resolutions, which has become the standard in computer vision [14, 24, 8, 23]. Many recent image-generation methods also operate hierarchically, where fine-grained results are conditioned on coarser level outputs [21, 12, 55, 26, 27, 28]. While these works primarily utilize multi-level approaches to improve performance, our work focuses on another important yet under-explored problem: semantic shape manipulation.

Shape manipulation. Shape manipulation was traditionally utilized for character animation [33, 30], where the model is first rigged to a skeleton and then a transformation is assigned to each skeleton bone in order to deform the shape. One could consider our coarse proxy as a skeleton of the shape, allowing for a simple manipulation of the high resolution model. Tulsiani et al. [50] present a learning-based technique for abstract shape modeling, fitting 3D primitives to a given shape. They demonstrate a shape manipulation application that is similar in spirit to the one we propose. However, unlike our method, the coupling between their primitive representation and the input shape is hand-designed with simple transformations, thus their method cannot guide the manipulation towards producing semantically meaningful shapes. Similar problems have also been studied in the image domain, where a image is manipulated semantically.

Figure 3: Learning a primitive-based representation of a single target shape. We optimize the parameters of the set of geometric elements (boxes colored with blue stripes) by minimizing the loss between the predicted and ground truth signed distance values on each sampled points.

3. Method

We first describe our shape representation in Sections 3.1 and 3.2. In Section 3.3, we describe how to learn a shared latent space over an entire collection of shapes and over multiple representations, while maintaining a tight coupling between representations. In Section 3.4, we describe our approach for shape manipulation using the proposed framework.

3.1. Coarse Primitive-based Shape Representation

In this section, we describe our approach for approximating a 3D shape with a finite number of simple shape primitives such as spheres, rods, boxes, etc. First, we need to define a metric that measures how well the primitive-based representation approximates the ground truth. Following Tulsiani et al. [50], we measure the difference of the signed distance fields between the target shape and the primitive-based representation.

A signed distance field specifies, for every point \( p = (p_x, p_y, p_z) \), the distance from that point to the nearest surface, where the sign encodes whether the point is inside (negative) or outside (positive) the shape. Representing basic geometric shapes with distance fields is particularly appealing, as many of them have simple SDF formulations. Furthermore, Boolean operation across multiple shapes can be achieved using simple operators over the SDFs of individual shapes. Therefore, complex shapes can be represented in a straightforward manner as a union of simple primitives.

More precisely, we denote a set of \( N \) basic shape primitives by tuples:

\[
\{(C^i, \alpha^i) | i = 1, ..., N\}
\]  

where \( C^i \) describes the primitive type and \( \alpha^i \in \mathbb{R}^{k^i} \) describes the attributes of the primitives. The dimensionality...
\(k^i\) denotes the degree of freedom for primitive \(i\), which vary across different choices of primitives. The signed distance function of a single element \(i\) can thus be written as follows:

\[ d_C (\mathbf{p}, \mathbf{a}^i) = \text{SDF}_C (\mathbf{p}, \mathbf{a}^i). \quad (2) \]

An example of a simple geometric primitive is a sphere, which can be represented with \(k^\text{sphere} = 4\) degrees of freedoms, i.e., \(\mathbf{a}^\text{sphere} = [c, r]\), where \(c = (c_x, c_y, c_z)\) describe its center and \(r\) is the radius. The signed distance function of the sphere takes the following form:

\[ d_{\text{sphere}} (\mathbf{p}, \mathbf{a}^\text{sphere}) = ||\mathbf{p} - \mathbf{c}||_2 - r. \quad (3) \]

For simplicity, we adopt spheres as our basic primitive type. However, as we later illustrate in Section 4, our framework is directly applicable to other primitive types.

To approximate the signed distance function of an arbitrarily complex shape, we construct the signed distance function of the union of the geometric elements (spheres in our case):

\[ \mathbf{a} = [\mathbf{a}^1, ..., \mathbf{a}^N], \quad (4) \]

\[ d_C (\mathbf{p}, \mathbf{a}) = \min_{1 \leq i \leq N} d_C (\mathbf{p}, \mathbf{a}^i). \quad (5) \]

Alternatively, smooth minimum functions like LogSumExp can be used in place of the (hard) minimum function to get a smooth transition over the interface between geometric elements. We refer the readers to Frisken et al. [15] for an in-depth explanation of signed distance fields and their Boolean operations.

To train the primitive-based model, given a target shape \(x\) (usually in the form of a mesh), we sample pairs of 3D points \(\mathbf{p}_i\) and their corresponding ground truth signed distance values \(s_i = \text{SDF}_x (\mathbf{p}_i)\). \(\mathbf{a}\) can be learned by minimizing the difference between predicted and real signed distance values:

\[ \hat{\mathbf{a}} = \arg \min_{\mathbf{a}} \sum_i L_{\text{SDF}} (d_C (\mathbf{p}_i, \hat{\mathbf{a}}), s_i). \quad (6) \]

Figure 3 shows the full structure of our primitive-based model.

### 3.2. High Resolution Shape Representation

We adopt DeepSDF [38] for our fine-scale shape representation. Similar to the coarse-scale representation, the shapes are modeled with SDFs. However, instead of constraining the shape to be within the family of shapes that can be constructed by simple primitives, we directly learn the signed distance function with a neural network \(g_\phi\):

\[ g_\phi (\mathbf{p}) \approx \text{SDF}_x (\mathbf{p}). \quad (7) \]

Just like the coarse representation, its zero iso-surface w.r.t. \(\mathbf{p}\) implicitly defines the surface of the shape, and can be retrieved efficiently with ray-marching algorithms. The training of the fine-scale SDF model follows the same procedure as the coarse-scale model, described in Section 3.1.

### 3.3. Learning a Tightly Coupled Latent Space

We learn a two-level shape representation over an entire class of shapes \(\{x_j \mid j = 1, ..., M\}\) by using two representation models that share the same latent code \(z_j\) (Figure 4 left).

For representing multiple shapes with the primitive based coarse-scale representation, we reparameterize \(\mathbf{a}\) with a neural network \(f_\theta\):

\[ \mathbf{a}_j = f_\theta (z_j), \quad (8) \]

where \(f_\theta\) is shared across all shapes. Likewise, for the fine-scale representation, we condition the neural network \(g_\phi\) on the latent code \(z_j\):

\[ g_\phi (z_j, \mathbf{p}) \approx \text{SDF}_x (\mathbf{p}). \quad (9) \]

To ensure that the manipulation made on one representation has the same effect on other representations, we would like to learn a shared latent space where every feasible latent vector is mapped to the same shape in both representations (see Figure 2 for an illustrative example). Furthermore, we also expect the latent space to be compact, so that latent code interpolation and optimization become less likely to “fall off the manifold.” Thus we utilize the variational autoencoder (VAE) framework [53] which enforces a strong regularization on the latent space by representing the latent vector of each individual shape \(z_j\) with the parameters of its approximate posterior distributions \((\mu_j, \sigma_j)\), similar to a VAE [29].

In the language of probability, we select the family of Gaussian distributions with diagonal covariance matrix as the approximate posterior of \(z\) given shape \(x_j\):

\[ q (z | x_j) := \mathcal{N} (z; \mu_j, \sigma_j^2 \cdot \mathbf{I}). \quad (10) \]

We apply the reparameterization trick [29], sampling \(\epsilon \sim \mathcal{N} (0, \mathbf{I})\) and setting \(z_j = \mu_j + \sigma_j \odot \epsilon\) to allow direct optimization of the distribution parameters \(\mu_j, \sigma_j\) via gradient descent.

During training, we maximize the lower bound of the marginal likelihood (ELBO) over the whole dataset, which is the sum over the lower bound of each individual shape \(x\) presented below:

\[ \log p_{\theta, \phi} (x) \geq \mathbb{E}_{z \sim q(z|x)} [\log p_{\theta, \phi} (x|z)] - D_{KL} (q(z|x)||p(z)). \quad (11) \]

Here the learnable parameters are \(\theta, \phi\), as well as the variational parameters \(\{\mu_j, \sigma_j\}_{j = 1, ..., M}\) that parameterize \(q(z|x)\). Since we would like the two representations to be tightly coupled, i.e., to both assign high probability density to a shape \(x\) given its latent code \(z_j \sim q(z|x = x_j)\), we model the first term of Eq. 11 using a mixture model:

\[ p_{\theta, \phi} (x|z) := \frac{p_{\theta} (x|z) + p_{\phi} (x|z)}{2}. \quad (12) \]
Here $p_0(x|z)$ and $p_\phi(x|z)$ are the posterior distributions of coarse and fine representations, implied by the signed distance function loss $L_{\text{SDF}}$ and its sampling strategies. Following Park et al. [38], we assume they take the form of:

$$\log p_0(x|z) = -\lambda_1 \int p(p) L_{\text{SDF}}(d_c(p, f_0(z)), \text{SDF}_x(p)) dp,$$

(13)

$$\log p_\phi(x|z) = -\lambda_2 \int p(p) L_{\text{SDF}}(g_\phi(z, p), \text{SDF}_x(p)) dp.$$

(14)

Eq. 13 and 14 can be approximated via Monte Carlo method, where $p$ is sampled randomly from the 3D space following a specific rule $p(p)$.

The benefits of using a VAD objective are two-fold: First, it encourages the model to learn a smooth and densely packed latent space. A similar effect has been leveraged in a related technique called conditioning augmentation [55]. This not only benefits optimization-based manipulation, but also improves coupling on novel shapes (shapes not seen during training). Secondly, being able to model the lower bound of the likelihood of every shape provides us with a way of regularizing the manipulation process by actively guiding the user away from unlikely results (Section 3.4). Detailed experiment and analysis on the effect of VAD are presented in Section 4.

3.4. Interactive Shape Manipulation

Our two-level model enables users to perform modifications on the primitive-based representation in an interactive manner while simultaneously mirroring the effect of the modifications onto the high-resolution representation. Additionally, our model is able to augment and regularize the user input in order to avoid generating unrealistic shapes. This form of manipulation is extremely useful, as it is generally hard for users to directly edit the mesh of a 3D shape. Even for a minor change, many accompanying (and time-consuming) changes are required to obtain a reasonable result.

In contrast, shape manipulation is much more intuitive for users with our model. To start with, we encode a user-provided shape into the latent space by optimizing the variational parameters w.r.t. the same VAD objective used during training. Alternatively, we can also start with a randomly sampled shape. Users can then efficiently modify the high-resolution shape by manipulating the shape primitives that represents parts of the shapes.

Our model supports any manipulation operation that can be expressed as minimizing an objective function over primitive attributes $\alpha$, such as increasing the radius of a sphere, moving a primitive one unit further towards the $z$ axis, or increasing the distance between two primitives, as well as a combination of them. The manipulation operation can be either dense, which involves all the attributes, or sparse, which only involves a subset of attributes or primitives. In the case of sparse manipulations, our model can automatically adjust the value of the unconstrained attributes in order to produce a more convincing result. For example, when a user makes one of the legs of a chair longer, the model automatically adjusts the rest of the legs, resulting a valid chair.

To reiterate, $\alpha$ contains the location as well as the primitive-specific attributes for each primitive. We use gradient descent to minimize the given objective function by optimizing the $z$:

$$z = \arg \min_z \left( L_{\text{MAN}}(f_0(z)) + L_{\text{REG}}(z) \right),$$

(15)
\[ L_{\text{REG}}(z) = \gamma \max \left( \| z \|_2^2, \beta \right). \]  

Note that \( L_{\text{REG}} \) is the optimization objective of the specific manipulation operation. For example, the objective of moving a single sphere \( i \) (parameterized by \( \alpha^i = [c_i, r_i] \)) to a new position \( \hat{c} \) is as follows:

\[ L_{\text{MAN}}^\text{Move}(\alpha) = \| c_i - \hat{c} \|_2 \]  

The attributes that are not constrained by the objective, including the position of other spheres, as well as the radii of all the spheres, are allowed to adjust freely during the optimization.

The latent code \( z \) is initialized as the expectation of \( q(z|x) \), where \( x \) is the shape to be modified. An appropriate choice of \( \gamma \) and \( \beta \) in the regularization term ensures a likely \( z \) under the Gaussian prior, which empirically leads to a more plausible shape. Multiple different manipulation steps can be executed consecutively to achieve complex or interactive manipulations. The optimization process is illustrated in Figure 4 (right).

Another important prerequisite for a successful shape manipulation framework is that every individual primitive should stay approximately at the same region of the shape throughout the entire class of shapes. As we later show in Section 4, primitives retain their semantic meanings well across all the shapes.

Our model is also advantageous in terms of speed. The coarse model can run at an interactive rate, which is crucial in providing users with immediate feedback. The high-resolution model is capable of dynamically adjusting the trade-off between quality and speed by using different rendering resolution and different number of ray-marching iterations. High quality result can be rendered only as needed, once the user is satisfied with the manipulated result.

4. Experiments

We demonstrate the shape representation power of our model as well as its potential for shape manipulation with various experiments.

We first show that our model is capable of representing shapes in high quality, comparing it with various state-of-the-art methods on the ShapeNet dataset [7], using a set of standard quality metrics.

To demonstrate the suitability of our model in the context of shape manipulation, we separately evaluate two aspects: First, we evaluate how tightly the two levels of representations are coupled by sampling novel shapes from the latent space and evaluating the volumetric intersect-over-union (IoU) between the two representations. As all of the manipulations are first performed on the primitive-based representation and then propagated to high-resolution representation through the latent code, a tight coupling is a crucial indicator for faithful shape manipulation. Second, we evaluate how well each primitive retains its semantic meaning across different shapes with a semantic consistency score. A semantically consistent primitive stays associated to the same part of the object across all the objects, which enables intuitive shape manipulation. We complement the quantitative evaluation by presenting a diversified collection of shapes manipulated with our method, demonstrating the flexibility of our manipulation framework and the fidelity of the result.

Data preparation. We normalize each individual shape to be inside a unit sphere. To sample signed distance values from mesh, we implemented a custom CUDA kernel for calculating the minimum distance from a point to the mesh surface. To determine the inside/outside of each point (and thus its sign), we use a ray stabbing method [37], which is robust to non-watertight meshes and meshes with internal structures and it does not require any pre-processing. For training the high-resolution representation, we use the same sampling strategy used in Park et al. [38]. For training the primitive-based representation, we sample points uniformly inside a unit sphere centered at the origin.

Shape reconstruction. We report reconstruction results for known and unknown shapes (i.e., shapes belonging to the train and test sets) in Table 1. Following prior work, we report several metrics: Chamfer distance (mean and median), EMD and mesh accuracy [45].

For unknown shapes, we compare our reconstruction performance against two variants of AtlasNet [20] (one generating surfaces from a sphere parameterization and one from 25 square patches) and DeepSDF [38], which we adopt for our fine-scale representation. As the table illustrates, our reconstruction performance is comparable to state-of-the-art techniques. As suggested in Park et al. [38], the use of a VAD objective trades reconstruction performance for a smoother latent space.

Effect of VAD objective on cross-representation consistency. We evaluate the consistency between fine and coarse shapes generated with our model by randomly sampling

<table>
<thead>
<tr>
<th>Shape</th>
<th>CD(^*)</th>
<th>CD(^D)</th>
<th>EMD</th>
<th>ACC</th>
<th>CD(^*)</th>
<th>CD(^D)</th>
<th>EMD</th>
<th>ACC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Airplane</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AtlasNet-Sph.</td>
<td>0.19</td>
<td>0.08</td>
<td>0.04</td>
<td>0.013</td>
<td>0.75</td>
<td>0.51</td>
<td>0.07</td>
<td>0.033</td>
</tr>
<tr>
<td>AtlasNet-25</td>
<td>0.22</td>
<td>0.07</td>
<td>0.04</td>
<td>0.013</td>
<td>0.37</td>
<td>0.28</td>
<td>0.06</td>
<td>0.018</td>
</tr>
<tr>
<td>DeepSDF</td>
<td>0.14</td>
<td>0.04</td>
<td>0.03</td>
<td>0.004</td>
<td>0.20</td>
<td>0.07</td>
<td>0.05</td>
<td>0.009</td>
</tr>
<tr>
<td>DualSDF</td>
<td>0.22</td>
<td>0.14</td>
<td>0.04</td>
<td>0.010</td>
<td>0.45</td>
<td>0.21</td>
<td>0.05</td>
<td>0.014</td>
</tr>
<tr>
<td>DualSDF (K)</td>
<td>0.19</td>
<td>0.13</td>
<td>0.04</td>
<td>0.009</td>
<td>0.65</td>
<td>0.19</td>
<td>0.05</td>
<td>0.012</td>
</tr>
</tbody>
</table>

Table 1: Reconstruction results on unknown shapes (top rows) and known (K) shapes (bottom row) for the Airplane and Chair collections. We report the mean and median of Chamfer distance (denoted by CD\(^*\) and CD\(^D\), respectively, multiplied by 10\(^3\)), EMD and mesh accuracy (ACC).
Figure 5: Measuring semantic consistency across the entire Chair collection. Above we illustrate the scores obtained on a few chair samples, where each primitive is colored according to the consistency score computed over the entire collection. Warmer colors correspond to higher scores (more consistent).

Figure 6: Shape correspondence via the coarse shape proxy. Above we demonstrate shape reconstructions from the Airplane dataset, with several primitives highlighted in unique colors. As the figure illustrates, the shape primitives are consistently mapped to the same regions. These correspondences can then be propagated to the fine-scale reconstructions.

<table>
<thead>
<tr>
<th>Intersection-over-union (IoU)</th>
<th>Airplane</th>
<th>Car</th>
<th>Chair</th>
<th>Bottle</th>
<th>Vase</th>
</tr>
</thead>
<tbody>
<tr>
<td>DualSDF (S)</td>
<td>0.52</td>
<td>0.76</td>
<td>0.50</td>
<td>0.68</td>
<td>0.44</td>
</tr>
<tr>
<td>w/o VAD (S†)</td>
<td>0.41</td>
<td>0.65</td>
<td>0.30</td>
<td>0.58</td>
<td>0.29</td>
</tr>
<tr>
<td>DualSDF (K)</td>
<td>0.56</td>
<td>0.70</td>
<td>0.53</td>
<td>0.69</td>
<td>0.54</td>
</tr>
<tr>
<td>w/o VAD (K)</td>
<td>0.53</td>
<td>0.70</td>
<td>0.53</td>
<td>0.69</td>
<td>0.55</td>
</tr>
</tbody>
</table>

Table 2: Cross-representation consistency evaluation. In the top rows, we measure the consistency of primitive based model and the high resolution model by randomly sampling (S) shapes from the latent space and calculating the intersection-over-union (IoU) of the two representations. We also report scores over known (K) shapes in the bottom rows. Note that due to the approximate nature of primitive based model, the numbers are only comparable with models trained under similar settings. †We train an additional VAE on top of the latent code to enable sampling.

Table 3: Semantic consistency evaluation. For each primitive index, we measure the fraction of shapes in each collection that agree with that primitive’s most commonly associated labels (i.e., the top-1, top-2 and top-3 most frequent labels). We report averages over all the primitives.

We conjecture that the improved consistency comes from the fact that, unlike the auto-decoder objective which only focuses on individual data points, the VAD objective actively explores the latent space during training.

**Semantic part-based abstraction.** We perform a quantitative evaluation on the PartNet dataset [35] to demonstrate that the semantic interpretation of the primitives in our model is consistent across different shapes. PartNet dataset contains part-labels of several levels of granularities. We train our model individually on Chair, Bottle and Vase collections, and evaluate the semantic consistency of the learned shape primitives using the 1000 labeled 3D points (per shape) provided by the dataset. We measure performance on the first level of the hierarchies, which contains 3-5 semantic labels per category. We would like to show that primitives are consistently mapped to the same semantic part of the shapes.
Figure 7: Learning with other primitive types. Our technique is directly applicable for other shapes which can be represented with SDFs. Above we demonstrate shapes represented with capsule primitives (cylinders with rounded ends), and their corresponding high-resolution representation.

across the entire shape collection. Thus, for each shape, we assign primitives with part labels according to their closest labeled 3D point. We calculate the semantic consistency score by measuring the fraction of shapes in the collection that agree with the most frequent labels.

In Figure 5 we illustrate the per-primitive semantic consistency scores on several samples from the Chair category. As the figure illustrates, some primitives have a clear semantic meaning (e.g., the legs of the chairs are consistently labelled as chair legs). Also unavoidably, some primitives have to “adjust” semantically to accommodate for the large variability within the shape collection, for instance, to generate chairs with and without arms. In Table 3 we report the average scores obtained on all the primitives (for each collection). We also report the fraction of shapes that agree with the top-2 and the top-3 labels. As the table illustrates, the semantic meanings of the primitives learned by our model are highly consistent among different shapes. This property allows the user to intuitively regard primitives as the proxies for shape parts.

Exploring other primitive types. While all of our results are illustrated on spherical shape primitives, our technique can directly incorporate other elementary shapes that can be represented with signed distance functions into the primitive-based representation. Figure 7 demonstrates a variant of our model that uses capsule primitives. We present the results with more primitive types in the supplementary material.

4.1. Applications

Our main application is shape manipulation using our coarse primitive-based representation as a proxy (see Section 3.4, Figures 1-2, and many more examples in the supplementary material). In the following we speculate on several other applications enabled by our two-level representation.

Shape interpolation. Similar to other generative models, our technique allows for a smooth interpolation between two real or generated shapes via interpolating the latent code. Furthermore, as an extension to our manipulation-through-optimization framework, our technique allows for controllable interpolation, where instead of interpolating the black box latent code, we interpolate the primitive attributes in the coarse representation via optimization. This enables selective interpolation. For example, the user can specify to only interpolate the height of one chair to the height of the other chair. Although this application is somewhat related to shape manipulation, there is one important distinction between the two: this application deals with two (or more) given shapes while shape manipulation deals with one shape only. In the supplementary material we demonstrate many interpolation results in both regular (latent space) and controllable (primitive attribute space) settings.

Shape correspondence. As our primitives are semantically meaningful, we can also perform shape correspondence between the high resolution shapes via the coarse shape proxy. To do so, we map every point on the surface of the high-resolution shape to its closest primitive shape. Figure 6 illustrates several corresponding regions over airplanes which are structurally different.

Real-time traversal and rendering. Previous work has shown that perception can be improved by arranging results by similarity [42]. As the shape primitives can be rendered in real-time, our two-level representation allows for a real-time smooth exploration of the generative shape space. Once the user would like to “zoom-in” to a shape of interest, the system can render the slower high resolution model.

5. Conclusions

In this work, we have presented DualSDF, a novel 3D shape representation which represents shapes in two levels of granularities. We have shown that our fine-scale representation is highly expressive and that our coarse-scale primitive based representation learns a semantic decomposition, which is effective for shape manipulation. We have demonstrated that the two representations are tightly coupled, and thus modifications on the coarse-scale representation can be faithfully propagated to the fine-scale representation. Technically, we have formulated our shared latent space model with the variational autodecoder framework, which regularizes the latent space for better generation, manipulation and coupling.

6. Acknowledgements

We would like to thank Abe Davis for his insightful feedback. This work was supported in part by grants from Facebook and by the generosity of Eric and Wendy Schmidt by recommendation of the Schmidt Futures program.
References


