Sketch Beautification: Learning Part Beautification and Structure Refinement for Sketches of Man-made Objects

Deng Yu, Manfred Lau*, Lin Gao, and Hongbo Fu*

Abstract—We present a novel freehand sketch beautification method, which takes as input a freely drawn sketch of a man-made object and automatically beautifies it both geometrically and structurally. Beautifying a sketch is challenging because of its highly abstract and heavily diverse drawing manner. Existing methods are usually confined to their limited training samples and thus cannot beautify freely drawn sketches with both geometric and structural variations. To address this challenge, we adopt a divide-and-combine strategy. Specifically, we first parse an input sketch into semantic components, beautify individual components by a learned part beautification module based on part-level implicit manifolds, and then reassemble the beautified components through a structure beautification module. With this strategy, our method can go beyond the training samples and handle novel freehand sketches. We demonstrate the effectiveness of our system with extensive experiments and a perceptual study.

Index Terms—sketch beautification, sketch implicit representation, sketch assembly.

1 Introduction

Sketching is a universal and intuitive tool for humans to render and interpret the visual world. Even if sketches are usually drawn in an imprecise and abstract format, human viewers can still implicitly beautify them and easily envision their underlying objects. But for machines, existing computer algorithms [1], [2] are struggling to make use of these freely drawn sketches directly, in particular, the sketches created for depicting man-made objects with diverse geometry and non-trivial topology. Although the beautification problem has been studied for decades, ranging from the primitives of sketched geometric objects [3] to strokes in handwritings [4], systematic analysis of beautifying sketches for man-made objects has rarely been studied. The key challenge here is how to instantiate poorly drawn conceptual geometries and refine imprecise structures simultaneously. Addressing the beautification problem of man-made object sketches can inspire and facilitate various downstream sketch-based applications such as sketch-based modeling [1], [5], [6], sketch-based retrieval [7], [8], and other sketch understanding tasks [9].

As sketches are widely utilized in manufacture designing [10], [11], [12], animation drafting [13], [14] and HCI (Human-Computer Interaction) [15], many algorithms have been presented to process freely drawn sketches targeting vectorization [16], rough sketch cleanup [14], and simplification [17], [18]. These methods are common in two places: first, they usually produce more local modifications on input sketches, and seldom consider or touch the global structures; also, the input of these methods is nearly ready-to-use sketches with perfect strokes that are straight, mutually parallel or curved with perpendicular angles. Hence, the aforementioned approaches are inapplicable to our task of beautifying the artifacts in the man-made object sketches with diverse local distortion and global inconsistency. As for the drawing assistance to the overall structures, prior approaches [19], [20], [21] are mainly designed in a heuristic and interactive way that provides a holistic scaffold or shadow guidance, and updates their guidance based on drawn strokes during the drawing process. Alternatively, Fišer et al. [22] proposed a rule-based stroke beautification approach, which, however, is still conditioned on the former strokes and confined to the stroke sequences. Therefore, these methods cannot be applied as post-processing to the existing hand-drawn sketches without the drawing order of individual strokes.

We observe that the traditional data-driven methods [14], [23] usually lack robustness and fail to produce satisfactory beautification results for freely drawn sketches of man-made objects, especially when the inputs are created with large variations. This is because their trained inference models are heavily confined to the limited training samples. We are largely inspired by CompoNet [24], which transcends the bound of the empirical distribution of the observed data and enriches the distribution diversity of generated samples by learning both novel part synthesis and plausible part compositions. Different from their generation task that tries to synthesize infinite outputs from finite inputs, our beautification task can be regarded as the reverse process that aims to beautify the infinite freehand inputs to the finite outputs.

* *Corresponding authors*

- *Deng Yu, Manfred Lau, and Hongbo Fu are with the School of Creative Media, City University of Hong Kong. E-mail: {deng.yu@my, manfred.lau@, hongbofu@}cityu.edu.hk*
- *Lin Gao is with the Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, and also with the University of Chinese Academy of Sciences, Beijing, China. E-mail: gaolin@ict.ac.cn*

Manuscript received xx xx, xx; revised xx xx, xx.

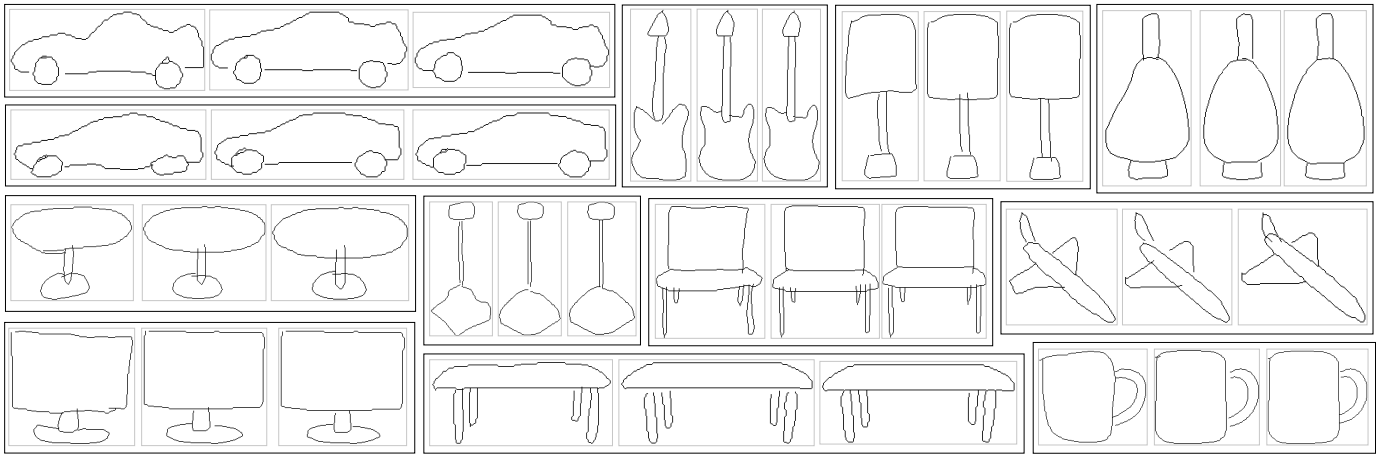

Fig. 1. We present a novel technique for beautifying freehand sketches of man-made objects. Each triplet contains an input sketch (left), the sketch after part beautification (middle), and the final result after structure refinement (right).

In this work, we treat a sketched object as a combination of part sketches and further transform the sketch beautification problem into two sub-problems: part sketch beautification and global structure beautification. Given a freehand input sketch, we expect to beautify the local geometry defects in the part beautification module and address the overall structure issues in the structure beautification module. To better instantiate the user's conceptual sketches, we make several strict constraints for this task. First, during the process of part beautification, one should follow the user's original drawing intention as much as possible. In our algorithm, we regard this constraint as: (i) The endpoints of the strokes in the beautified result should be as close as possible to those in the user's original sketch; (ii) The curvature of the strokes in the beautified result should be as close as possible to its beautification reference (see Figure 7). Second, during the process of structure beautification, one should maintain the user's original structure as much as possible. Therefore, directly replacing the user's input part sketch with a better part sketch retrieved from the database is not considered in our method. In addition, we do not aim for large transformations and structural alterations (e.g., scaling beyond 0.8 1.2 times or translating beyond 3 to 3 pixels on the individual parts of a sketch), and we perform smaller refinements on the input sketch in terms of both geometry and structure (see Figure 1).

In the part beautification stage, we argue that existing deep sketch representations (i.e., CNN features from rasterized pixel maps [25] or RNN features from stroke sequences [26]) are insufficient to perform beautification on users' conceptual freehand sketches, as shown in Figure 2. CNN representations are widely utilized in natural image synthesis and can easily generate plausible and novel samples by interpolating existing images [27] or the disentangled semantic attributes [28]. However, different from natural images, sketches are usually too sparse with few valid elements or points that cannot be smoothly interpolated with CNN representations (see the comparison of sketch interpolation in Figure 2). As for RNN representations, existing methods cannot guarantee a precise reconstruction of input sketches (see the comparison of sketch reconstruction in Figure 2), not

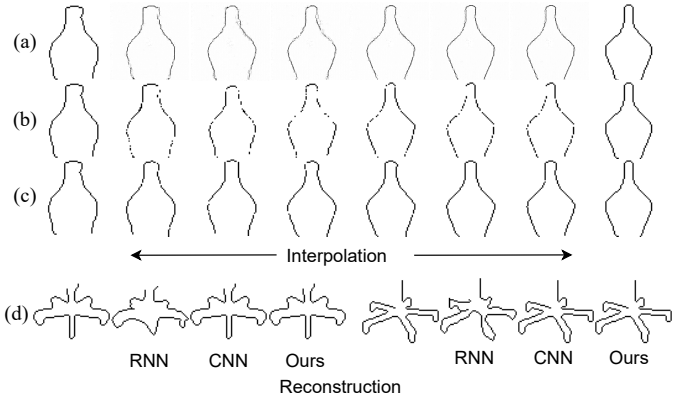

Fig. 2. Sketch interpolation based on CNN (a), CNN after binarization with the threshold of 0.5 (b), and our implicit representation (c), and reconstruction with different representations (d). Please zoom in to examine the details, e.g., blurry and diffusion artifacts surrounding the interpolated strokes based on CNN (a), as well as broken strokes and missing points with CNN after binarization (b).

to mention the more challenging interpolation task. In this paper, we propose a novel implicit sketch representation, which is not only able to represent sketches effectively but also can construct a continuous and smooth manifold to synthesize and instantiate users' conceptual drawings by interpolating the existing sketches.

In the structure beautification stage, to simulate the structural errors that exist in the user's input sketches, we apply random affine transformations to the ground-truth sketches in our training dataset and enforce our designed sketch assembly model to learn these transformations between the transformed part sketches and their ground truth counterparts. Then, simply feeding the input sketches to our sketch assembly model usually causes learning oscillation and failures in model convergence due to the sparsity of the input sketches. To address this issue, we use part-level bounding boxes to enhance the spatial feature of the separate part sketches, where the part bounding boxes (Figure 8 (e)) are used to specify the spatial location of individual part sketches and the spatial relationship between different part

sketches in a user's drawing. We propose an IOU (intersection over union) metric which uses the overlapping between the bounding boxes of transformed part sketches and their corresponding part ground truth to further evaluate the performance of the sketch assembly model.

Training our beautification framework requires a considerably large amount of part-annotated sketch data. Considering the limited training sketches, we collect a novel training dataset of part-labeled sketches by rendering the edge maps with the semantic labels under the best view [29] from the existing 3D shape repositories, i.e., PartNet [30], SDM-Net [31], and COSEG dataset [32]. We evaluate the beautification results in terms of faithfulness to the input sketch and the beautification quality. Intuitively, faithfulness can be evaluated quantitatively and via a user study, while the beautification quality can only be evaluated via a user study due to its subjectiveness. In this work, we perform a quantitative evaluation on faithfulness and a perceptual study on beautification quality to interpret the performance of sketch beautification methods more comprehensively.

Our contributions in this work are:

- We introduce a novel beautification framework for sketches of man-made objects, which consists of a part beautification module and a structure beautification module.
- We propose a novel sketch implicit representation that can represent sketches effectively. Converting the input freehand sketches to implicit representations is the primary and essential step in our part beautification stage. To the best of our knowledge, we are the first to represent sketches with implicit functions.
- We design a novel sketch assembly model to learn the spatial transformations for the sparse representations to achieve structure beautification.
- We collect a large-scale part-labeled sketch dataset consisting of 17,172 sketches spanning 9 object categories. We will release the data and code to the research community.

2 Related Work

In this section, we review the previous approaches that are closely related to our work in these categories: sketch beautification, sketch representations, and implicit representations.

2.1 Sketch Beautification

In the past decades, numerous algorithms were proposed to beautify freehand drawings. The earliest algorithmic beautification method with aesthetic constraints dates back to SketchPad [33]. Later, Pavlidis and Van Wyk [34] proposed an algorithm for beautifying figures as a post-process. Igarashi et al. [35] presented a system for rapid geometric design that beautified strokes in an interactive way. Recently, great strides have been made in the sketch simplification [17], [18], rough sketch cleanup [14] and sketch vectorization [16]. Since these methods do not attempt to change or beautify the global structure of input sketches, we do not conduct a detailed review here. We refer the interested readers to [36] for an insightful survey.

The more related works are iCanDraw [20], ShadowDraw [21], and ShipShape [22]. iCanDraw and ShadowDraw can output a sketch with a reasonable layout and more realistic details by providing reference scaffolds or shadow layouts for users. However, as discussed before, limited by their heuristic and iterative settings during drawing, they cannot beautify the existing sketches. For ShipShape, this method only performed a curve-level beautification in an interactive way by enforcing some geometric constraints to the afterwards curves based on the previous strokes, such as symmetry, reflection, and arc fitting. These constraints are difficult to apply to the existing sketches in practice without knowing the drawing order of the component strokes. Besides, during its beautification process, extra efforts from users on the structure adjustment are still needed. Instead, we study the part-level beautification since the part labels are more accessible with existing segment approaches [37], [38], [39] or user-specified annotations. We further reduce user labors with our sketch assembly model by performing structure beautification automatically.

A similar beautification problem was also explored in EZ-Sketching, where Su et al. [40] proposed a three-level optimization framework to beautify a sketch traced over an image. Their beautification process heavily relies on the image being traced, which is missing in our task. In DeepFaceDrawing, Chen et al. [41] presented a robust portrait image synthesis method, which can handle rough or incomplete sketches as input. By decomposing a face sketch into face components and projecting the face component sketches to the component-level sketch manifolds, the input freehand sketches can be implicitly refined for generating photo-realistic face images. As the layouts or structures of human portraits are almost fixed, their approach is not suitable for our beautification task. To handle the diverse structure of sketched man-made objects, we propose a novel sketch assembly module, as inspired by [24].

2.2 Sketch Representations

A sketch is generally represented as either a rasterized binary-pixel map [42] or vector sequences [26], [43], or both [44]. Since the sketch datasets TU-berlin [45] and Quickdraw [26] were introduced, the research community has seen many works in sketch understanding. For example, Yu et al. [25] proposed a multi-scale and multi-channel CNN framework with a larger kernel size for sparse raster sketches. Ha and Erc [26] introduced a sequence-to-sequence variational auto-encoder with the bidirectional RNN backbone for vectorized sketches. Later, the CNN and RNN representations [46], [47] were combined to better represent sketches in their retrieval and recognition tasks, respectively. Most of the subsequent works [44], [48] basically inherited and employed the off-the-shelf backbones for the specified tasks. However, since the CNN-based representations are designed to learn the distribution of all the pixels in the 2D image space rather than the solo valid points on strokes, they tend to generate broken segments with shadow effects (Figure 2 (a) and (b)) during the interpolation process. The RNN-based representations failed to reconstruct input sketches

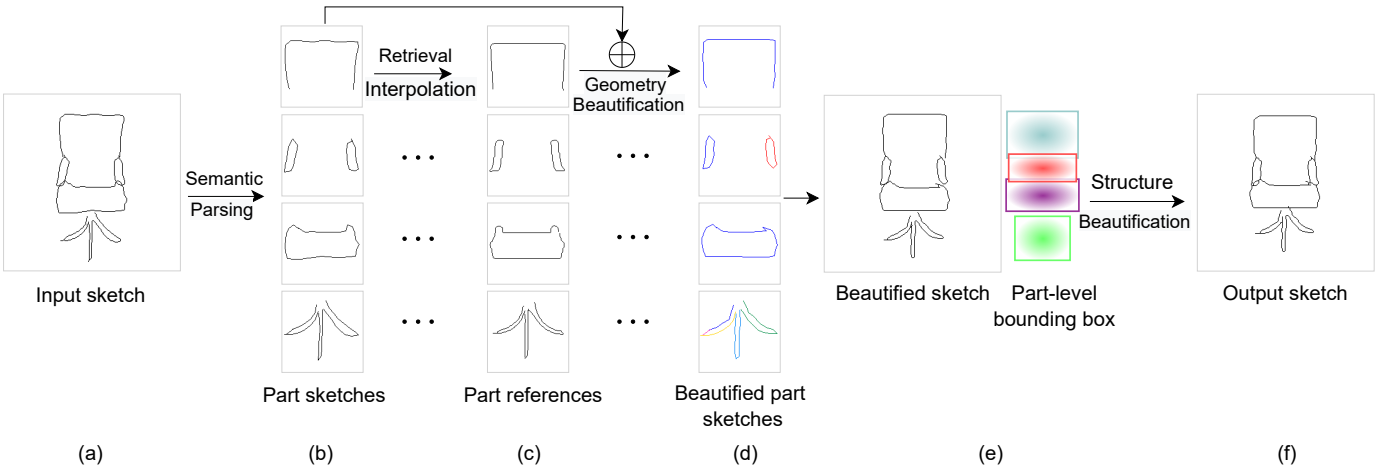

Fig. 3. System pipeline of sketch beautification. For an input sketch (a), we first parse it to individual part sketches (b), and then synthesize the corresponding part references (c) by retrieval and interpolation. After performing geometry beautification on the part sketches (b) towards part references (c), we obtain the new part sketches (d) with well-beautified geometry (see geometry differences between (b) and (d)). During the stage of structure beautification, we adjust the imperfect structure (notice the misalignment of chair arms) of the intermediate output (e) with the help of part-level bounding boxes (e) and generate the final beautified sketch (f). Different colors in beautified part sketches (d) indicate the different strokes. The colorful bounding boxes (e) denote the scales and spatial locations of different part sketches in the image space (256×256).

with vector sequences precisely (Figure 2 (d)). Alternatively, Sketchformer [43] proposed a transformer-based representation for encoding a freehand sketch input in a vector form. Despite its success in generating more plausible sketch interpolation compared to RNN-based representations, this method still fails to faithfully reconstruct complicated part sketches (e.g., chair legs) and tends to produce shifting issues during interpolation due to their stored relative position for sketch points in their vectorized format of input sketches. Therefore, we turn to implicit functions to better represent part sketches in this work.

2.3 Implicit Representations

Recently, implicit functions have attracted extensive attention in the research community [49], [50], [51], [52], [53], [54], since they can represent 3D shapes in a continuous and smooth implicit field. Existing implicit representations typically used a spatial function to represent a shape by mapping the inside and outside points distinguishably. In DeepSDF, Park et al. [52] utilized a signed distance function (SDF) to represent a watertight shape in 3D space where inside and outside points are respectively mapped to negative and positive values, and the underlying surface is implicitly represented by a zero-crossing surface. Similarly, Mescheder et al. [53] represented a target shape with a continuous occupancy function, indicating the probability of points being occupied by a shape. However, different from watertight 3D shapes, sketches are often created with no concept of *inside* or *outside* and are usually too sparse with limited valid points or pixels in the 2D image space. Hence, the aforementioned implicit representations cannot be directly applied to the part sketches. To address this issue, we design an intuitive 2D implicit function (we call it a hit function) for part sketches, where we map all sampled points of a canonical 2D space to two exclusive statuses: whether its projection to the image space of the part sketch *hits* the stroke or not (see Figure 5).

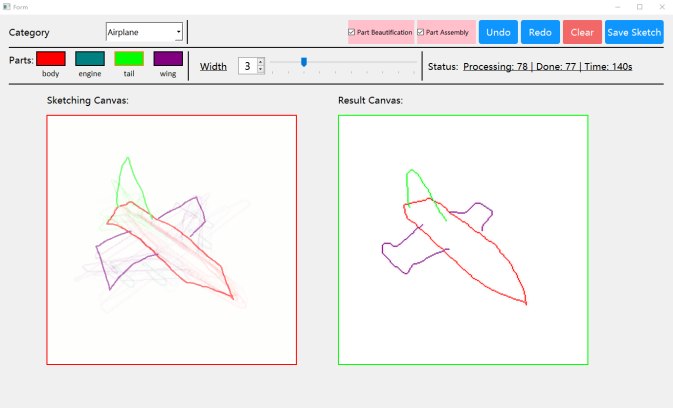

Fig. 4. Our sketching interface.

3 Methodology

In this paper, we mainly focus on freehand sketches created for man-made objects that can be segmented into parts. We treat this kind of sketches as a combination of individual part sketches, since 3D man-made objects often consist of semantically meaningful parts [55], [56], [57]. Our framework for sketch beautification is illustrated in Figure 3. We first parse an input freehand sketch depicting a man-made object like a chair to part-level sketches, and then beautify the individual part sketches in the part beautification module. In this module, we construct an implicit manifold for each part. For each part, we can search and incorporate the geometry features from its key neighbors in this manifold for geometry beautification. "Incorporating" the geometry features happens in the linear interpolation (Section 3.2.2) and part geometry beautification (Section 3.2.3). Finally, the beautified part sketches are recomposed as the final beautified sketch by our sketch assembly model, which performs the structure beautification task.

3.1 Sketching Interface

Our algorithm requires semantic segmentation of an input sketch. Sketch segmentation can be done automatically, as demonstrated in previous works [37], [38], [39]. However, for simplicity, we design a simple interface (Figure 4) for users to interactively provide part-level semantic information while sketching. The key enablers of this system are the part beautification component and the structure beautification component. Once activated on the interface (through "Part Beautification" and "Structure Beautification" buttons), the two functions will run in the background and beautify the user's input automatically after the user finishes one complete sketch. We provide more details of the mechanism of these two functions in Sections 3.2 and 3.3.

Before sketching, a user first needs to select a target object category from one of our prepared nine classes (e.g., airplane, chair, table). Our interface then shows the corresponding set of part labels. It also provides part-level shadows to provide initial geometry and structure guidance, similar to ShadowDraw [21]. Unlike ShadowDraw, we do not update the underlying shadows dynamically during the drawing process since we hope to give more freedom to users and not influence them too much. In this interface, we provide several basic drawing tools for users to amend their drawings, such as clear, undo, redo, etc. By clicking on a part label (e.g., airplane engine), the strokes drawn afterwards will be labeled with the selected label automatically. In this way, we obtain a freehand sketch with well-segmented parts. Note that our system saves each drawn sketch in both raster and vector graphics formats. For a better drawing experience, our interface supports the drawing of strokes with varying thicknesses. However, such strokes are preprocessed by extracting the skeletons [58] from the user's sketch input to have one-pixel width for further processing. Note that we utilize the "skeletonize" operation for real-time processing of freehand sketches when implementing our sketching interface, rather than directly rasterizing the recorded vectorized stroke sequences since the latter's performance highly depends on the number of input strokes.

3.2 Part Beautification

Given a rasterized freehand sketch with part annotations, we treat its component sketches separately and beautify them individually in the corresponding part-level implicit manifolds. Different from CNN representations created for the dense and high-frequency RGB pixels, our novel implicit representation works for the low-frequency and sparse sketched points in the 2D image space. Existing implicit representations are not suitable for our sketch points in the 2D space, since a 2D sketch is a kind of more discrete representation having no inside and outside spaces and occupying no continuous regions compared with the closed 3D shapes. We thus design our own sketch implicit representation where we first sample points from a 2D canonical space. We then project these points to the same-size image space of sketches and record whether these points could hit any stroke of a sketch. Finally, a discrete sketch can be represented by the sampled points (hitting sketches) of this continuous 2D canonical field implicitly (see Figure 5).

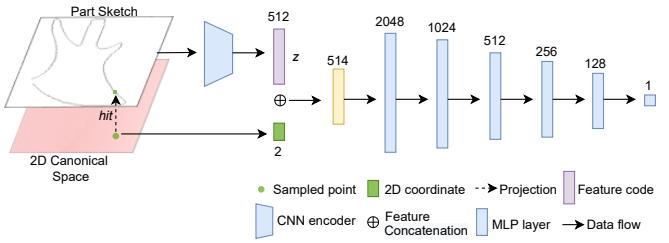

Fig. 5. Illustration of our pipeline for learning a sketch implicit representation.

3.2.1 Sketch Implicit Representation

An implicit field is typically defined by a continuous function over 2D/3D space. In this work, we take a sketch as an implicit function (hit function) defined over the continuous coordinates P in a 2D canonical space, as given in Equation 1.

$$H(p) = \begin{cases} 1; & p\text{'s projection hits a stroke;} \\ 0; & \text{otherwise.} \end{cases} \quad (1)$$

Here, we define the valid points (those projections that hit a stroke in a sketch) as 1, and the invalid points (those projections that hit the empty background) as 0. The underlying sketch is represented by the points with positive values, that is $H(\ ) = 1$, similar to the zero-isosurface points in 3D implicit representations. We then employ a neural network f to approximate the implicit function $H(p)$ as $f(p)$. Inspired by [54], where Chen and Zhang proposed an implicit decoder (IM-NET) to map a shape feature vector and a point coordinate to a value indicating the inside/outside status of a point relative to the shape, we adapt IM-NET to our sketch implicit function to indicate the status (hits a stroke or not) of a sampled 2D point (from the 2D canonical space) concatenated with a feature code z of an input sketch (see Figure 5). As for the sketch encoder, we design a CNN encoder (see the supplemental materials) to extract the feature code z from the input sketch. Conditioned on the feature code, the implicit function can be further formulated as $f(z; p)$.

Here, we adopt Multi-layer Perceptrons (MLPs) with rectified linear unit (ReLU) as the implicit function f. To represent part sketches in a uniform 2D space, we center-crop and resize all the part sketches to a $128 \quad 128$ scale. We illustrate the model for learning our sketch implicit representation in Figure 5. The part sample is first encoded to a 512D feature code and then a sampled 2D coordinate is concatenated with the feature code as the input to the implicit model. After the model converges, an input (part) sketch can be represented as the implicit representation z and interpreted back to the 2D image space by sampling points to f.

Loss function. We use a mean squared error between ground truth labels and predicted labels for each point as the loss function for training sketch implicit function f. Specifically, we formulate it as follows:

$$L(\ ) = \sum_{p2P} kf(p) \quad H(p)k^2; \quad (2)$$

where P refers to the point set sampled from the $128 \quad 128$ 2D space and $H(p)$ is the ground truth value in our dataset given the query point p.

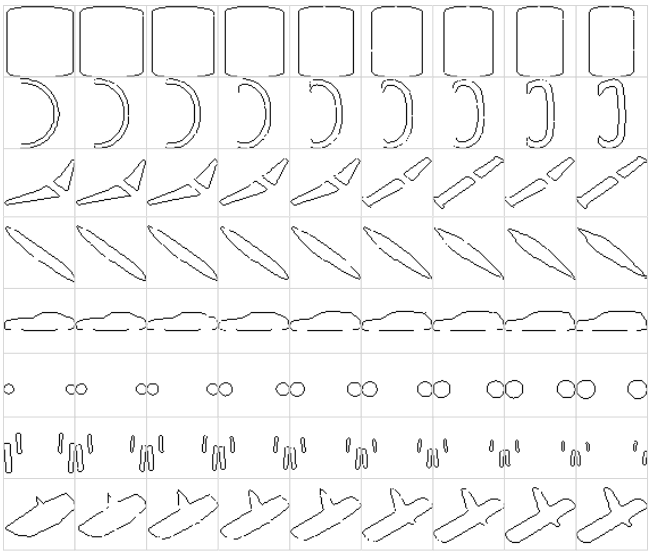

Fig. 6. Smooth interpolation of the leftmost and rightmost samples with our implicit representations.

3.2.2 Retrieval and Interpolation

Previous works [27], [41] have shown that interpolation tends to give good beautification results. Furthermore, due to the continuous nature of our defined implicit function (hit function) for sketches over 2D space and sufficient training samples, our implicit sketch representation works well in producing smoothly interpolated and good results in the learned implicit field/space by linear interpolation. With the part-level implicit sketch representations of our dataset $\{v_1; v_2; \cdots; v_N\}$, we can synthesize the novel samples continuously and smoothly by linear interpolation of implicit representations of existing sketches, as shown in Figure 6. To further bridge the gap between the user's conceptual free-hand sketches and the existing part sketches in our dataset, we adopt the same retrieval-and-interpolation strategy as [41] to instantiate the user's conceptual sketches with on-hand samples in our dataset by local linear interpolation. Specifically, we first take the implicit representation v_q of an input part sketch as the query to retrieve its top K neighbors in our dataset. In the following steps, we construct a part-level manifold with the retrieved K neighbors as the basis vectors $\{v_{t1}; v_{t2}; \cdots; v_{tK}\}$, and project the v_q to this manifold as Equation 3.

$$\min \|v_q - \sum_{i=1}^{K} w_i v_{ti}\|_2^2; \quad s.t. \sum_{i=1}^{K} w_i = 1 \tag{3}$$

where we set $K = 3$ in our implementation and calculate the unknown weight parameter w_i for each basis vector v_{ti} by solving this constrained least-squares problem. We finally obtain the projected implicit representation v_p by linearly interpolating the basis vectors locally:

$$v_p = \sum_{i=1}^{K} w_i v_{ti}. \tag{4}$$

Obtaining the projected implicit representation v_p, we further interpret it back to the 2D image space as the beautified reference for the input conceptual sketch (see Figure 3 (c)).

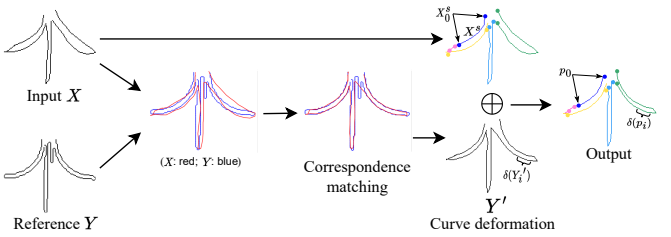

Fig. 7. Illustration of our pipeline for part-level geometry beautification. Some results of Equations 5 and 6 are annotated here.

3.2.3 Part-level Geometry Beautification

In this work, we consider the part-level beautification as a deformation process that deforms each part sketch towards the corresponding reference obtained from the retrieval-and-interpolation step rather than directly replacing the input with the reference. In this way, we hope to preserve the users' original drawing intentions as much as possible. Thereby, given each input part sketch and its reference, we design the beautification pipeline as follows: correspondence matching and curve deformation. The former step performs a coarse shape-level registration that transforms all points of the input sketch to approximate the general shape of the reference. Based on the output of correspondence matching, the latter step then conducts a fine-grained curve-level deformation that deforms the input sketch towards the above intermediate output while keeping the original endpoints of the input curves unchanged, as illustrated in Figure 7.

For correspondence matching between the input part sketch and the reference sketch, we adopt a classical non-rigid registration method [59]. Despite its robustness, this method fails to produce reasonable correspondence results efficiently when applied to our scenario. We speed up this method from more than 3 seconds to around 0.3 seconds by adding two weight decay parameters and that decrease exponentially with the increase of the iteration as shown in Equation 5. Here we refer to the sketched points of the input and the reference as X and Y, respectively.

$$\begin{aligned} E_c &= w_1 E_{match} + w_2 E_{rigid} + w_3 E_{arap}; \\ E_{match} &= \sum_{i=1}^{n} \|z_i - P_Y(z_i)\|_2^2; \\ E_{rigid} &= \sum_{i=1}^{n} \|z_i - R(x_i + t)\|_2^2; \\ E_{arap} &= \sum_{i=1}^{n} \sum_{j \in N_i} \|k(z_j - z_i) - R(x_j - x_i)\|_2^2; \end{aligned} \tag{5}$$

where $P_Y(\cdot)$, R, and t refer to the closest point in reference point set Y, the rotation matrix, and translate parameters, respectively. We initialize z_i with x_i in the input point set X, and set $w_1 = 1; w_2 = 100; w_3 = 0.9,$ $= 0.4$, and $= 0.9$. Note that we solve the overall objective function E_c in an iterative way. In our experiment, $= 15$ iterations are sufficient to produce a satisfying result (see an example of Y^0 in Figure 7).

To keep the user's original intention as much as possible, after calculating the intermediate output Y^0 of the first step, we further perform the curve deformation that deforms the recorded strokes $\{X^s\}$ (s is the stroke index) of the user's input X towards Y^0 following Equation 6. As we mentioned

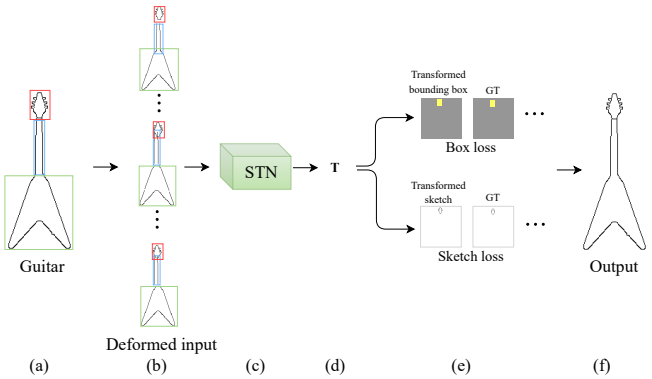

Fig. 8. Illustration of our pipeline for learning structure beautification of the part-deformed sketch. (a) Ground truth. (b) Synthesized part-deformed sketches. (c) STN backbone of sketch assembly model. (d) Learned transformation matrix. (e) Losses for part-level bounding boxes and sketches. (f) Reassembled output.

before, our interface can also record the strokes during the drawing process.

$$
\begin{aligned}
E_d &= E_{position} + E_{shape};\\
E_{position} &= \sum_{s2S} \sum_{n} kp_0 \quad X_0^s k;\\
E_{shape} &= \sum_{s2S} \sum_{i=1} k\ (p_i) \quad (Y_i^0)k:
\end{aligned}
\tag{6}
$$

Here, $(k) = k_{i\ 1} + k_{i+1} \quad 2k_i$, representing the local feature at each point. X_0^s represents the two endpoints of a stroke X^s of the input sketch X while p_0 refers to the corresponding endpoints of the optimized curve. Y_i^0 represents the closest point of Y^0 to the point p_i of the optimized curve under the Euclidean measurement (see Figure 7). We optimize the energy function in the stroke level. The term $E_{position}$ enforces the optimized curve to have the same start point and the end point as the input stroke, and the term E_{shape} constrains the optimized curve to have the same shape (curvature) as the corresponding reference stroke, as illustrated in Figure 7. In this way, the final beautified sketch keeps the same start points and the end points as those in the input sketch. We can think of our system as redrawing the input strokes based on their original endpoints in a more professional way.

3.3 Structure Beautification

Through the part beautification module, the geometry of input individual part sketches can be well refined. However, there is still a structure inconsistency existing in the input sketch. To address this issue, we further introduce a structure beautification module to adjust the imperfect structure of the input sketch with the deformed parts. The basis of this module is the sketch assembly model designed to learn spatial transformations for beautifying the deformed sketches. Figure 8 illustrates the workflow of training our sketch assembly model, where we first simulate the structure inconsistency by randomly applying affine transformations to the individual part sketches in our dataset. In our implementation, we combine the affine transformation operations of random scaling (ranging from 0:8 to 1:2 folds) and random translations (varying in x and y directions with 3 to 3 pixels offsets) in the image space

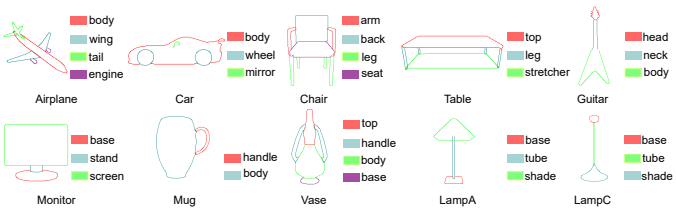

Fig. 9. Different categories and part annotations in our dataset. Note that the lamp class has two different labeling strategies (see the part annotations of LampA and LampC).

Categories	#Samples	#Parts	Source
Chair	5,962	4	PartNet
Table	4,440	3	PartNet
Airplane	2,467	4	SDM-Net
Car	1,813	3	SDM-Net
Guitar	741	3	SDM-Net
Monitor	559	3	SDM-Net
Vase	298	4	COSEG
Mug	211	2	COSEG
LampA	510	3	COSEG
LampC	171	3	COSEG

TABLE 1
Data distribution of our synthesized sketch dataset.

(see Figure 8 (b)). As there is no obvious rotation observed in part sketches and our part beautification module is able to reduce such rotation effects on part sketches, we do not apply the random rotations to our training data here. We then design a sketch assembly network (Figure 8 (c)) with the backbone of the spatial transform network (STN, in short) [60] to learn the inverse mapping that transforms the distorted parts back to their ground-truth locations (see Figure 8 (a)). To prevent training oscillation caused by the sparse input (sketch), we further introduce the part-level bounding boxes M (Figure 8 (e)) to enhance the spatial features of the part sketches S.

Loss function. We use the following loss function for training the assembly network:

$$
\begin{aligned}
L = \sum_{p2P} & \ {}_1 k S_p^a \quad S_p^{GT} k + {}_2 k M_p^a \quad M_p^{GT} k \\
& + \ {}_3 k T_p \quad T_p^{Identity} k_2^2;
\end{aligned}
\tag{7}
$$

where we regard the three loss terms as sketch loss, bounding box loss, and regularization loss respectively. S_p^a and M_p^a are the part sketch and the part bounding box with the random affine transformation, respectively, while S_p^{GT} and M_p^{GT} are the corresponding ground truth in our dataset (see Figure 8 (e)). To stabilize the learning process, we further constrain a regularization loss on the learned transformation matrix T_p to enforce it to have a slow and mild update rather than a large oscillation during the learning process. $T_p^{Identity}$ refers to the Identity mapping matrix. We set ${}_1 = 100$, ${}_2 = 1$, and ${}_3 = 1$ in the experiment.

4 Experiment

We conducted extensive experiments on freehand sketches drawn by 8 volunteers with our sketch interface. Two of them were professional interior designers with years of drawing experience, and the rest were ordinary students

aged 26 to 29 with no professional drawing skills. Given an object category, we asked the invited volunteers to sketch the man-made objects in their minds as casually as possible. They can draw any shape of part sketches around the common regions of the part shadows, like the airplane presented in the sketching canvas of Figure 4. Finally, we collected more than 200 freehand sketches and 15-45 sketched objects that span diverse part geometry and global structures for each category. Figures 1 and 10 show some representative drawings and the corresponding beautification results. Please find more details of our collected freehand sketches and additional beautification results in the supplemental materials. We also construct a large dataset of part-labeled synthesized sketches to train the part-level sketch implicit representations and the sketch assembly model in our sketch beautification framework. We show some rendered part-annotated sketches under the best view from the existing 3D shape repositories, including PartNet [30], SDM-Net [31], and COSEG dataset [32] in Figure 9. Our sketch dataset contains 17,172 sketches with clear part annotations distributed over 9 man-made object categories that are commonly observed in daily life [45]. Figure 9 gives a representative sketch under each category. We provide more details of the data distribution of our synthesized sketch dataset in Table 1.

4.1 Implementation Details

We implemented our sketch implicit model and sketch assembly model with the PyTorch framework [61] and used the Xavier initialization [62]. To train sketch implicit representations for our 128 128 part sketch, we sampled both the valid sketched points and the invalid points from the 2D image space (total 16,384 points). We demonstrate the parameter structures of the sketch implicit model and sketch assembly model in the supplemental materials. The two models were trained on an NVIDIA RTX 2080Ti GPU and optimized by the Adam optimizer (1 = 0.9 and 2 = 0.999) with the learning rate of 5e 5 and 1e 4, respectively. Here we trained the two models to full convergence until the learning rate decayed to relatively small. Note that training the sketch implicit model takes around 48 hours on a single GPU with the batch size of 1 for one category on average. The batch size for training sketch assembly is 64. The iteration epochs for the two models were set as 800 and 600, respectively. Although it takes a long time to train these two models during the training stage, our whole sketch beautification pipeline only spends around 1 second to beautify an entire sketch.

4.2 Baselines

To verify the effectiveness of our proposed sketch beautification pipeline, we compare our approach with existing methods both qualitatively and quantitatively. We use four baselines in our experiment and detail them as follows.

One of the baselines is the Laplacian smoothing algorithm [63]. We implement this method on the stroke level to process the input freehand sketches. The next one is a naive instance-level and retrieval-based approach. We take the user's freely sketched input as the query and retrieve its top-1 candidate from the dataset as the beautified result. In our implementation, we use the HOG [64] descriptor, which is efficient and able to capture the spatial features of the sparse sketches. Then, we further designed a part-level retrieval baseline. Different from instance retrieval using the entire input sketch as the query, we first parse the user's freehand sketch to individual parts and utilize the separated part sketches (not the entire sketch) to retrieve the corresponding top-1 part results from the part-annotated dataset with the HOG descriptor, and finally replace the original part sketches with the retrieved part candidates as the final beautified output. Another baseline is a learning-based sketch simplification approach Mastering Sketching [14]. Although this generative method is designed to remove the superfluous details (i.e., the repetitive scribbles) and synthesize (or strengthen) the important lines for the rough sketches, we utilize it in our sketch beautification task to generate the beautified results for the input freehand sketches conditioned on the strokes that are identified as important by its learned model. In our implementation, we directly utilize their original trained inference model to process the user's freehand drawings. Note that since the outputs of Mastering sketching are generated with bold effects, we post-process them with a thinning operation [58] to remove the redundant pixels and conserve the clear skeletons as final beautification results. For the closely related work ShipShape [22], since it was fully integrated into a quite old version of Adobe Illustrator and its plugin is no longer accessible today, we can only discuss the difference between our approach and ShipShape (Section 2.1) instead of conducting experimental comparisons.

4.3 Performance Evaluation

We evaluate our method and compare it with the existing methods: Laplacian smoothing, instance retrieval, part retrieval, and Mastering Sketching on freely drawn man-made object sketches. Figure 10 shows beautified results of different methods on the freehand sketches. The visual comparisons show that our method produces the most satisfying results across various categories of sketches, regardless of the local geometry or the global structure.

We observe that the outputs of Laplacian smoothing and Mastering Sketching basically keep the same shape as the input sketches. Compared with the input, the former results lose some sharp features (see car body, chair seat, and guitar body in Figure 10 (c), (d), and (e) respectively) and the latter are generated with bold or shadow effects. Although Mastering Sketching succeeds in healing some small seams between parts (e.g., the head and neck parts of the guitar sketch in Figure 10 (e)) by generating more pixels (along the two strokes of the guitar neck towards the guitar head based on the original sketches), it fails to beautify the larger seams, e.g., defects between the neck and body parts of the guitar sketch in Figure 10 (e).

When visually inspected, the results produced by instance retrieval have the most dissimilar appearance compared with the input sketches. Due to the freedom and abstraction of hand drawings as well as the limited scale of our synthesized sketch dataset for retrieval, instance retrieval cannot find a pleasing beautified counterpart from the existing sketch dataset for the user's input sketch. Although part retrieval tries further to approximate the input

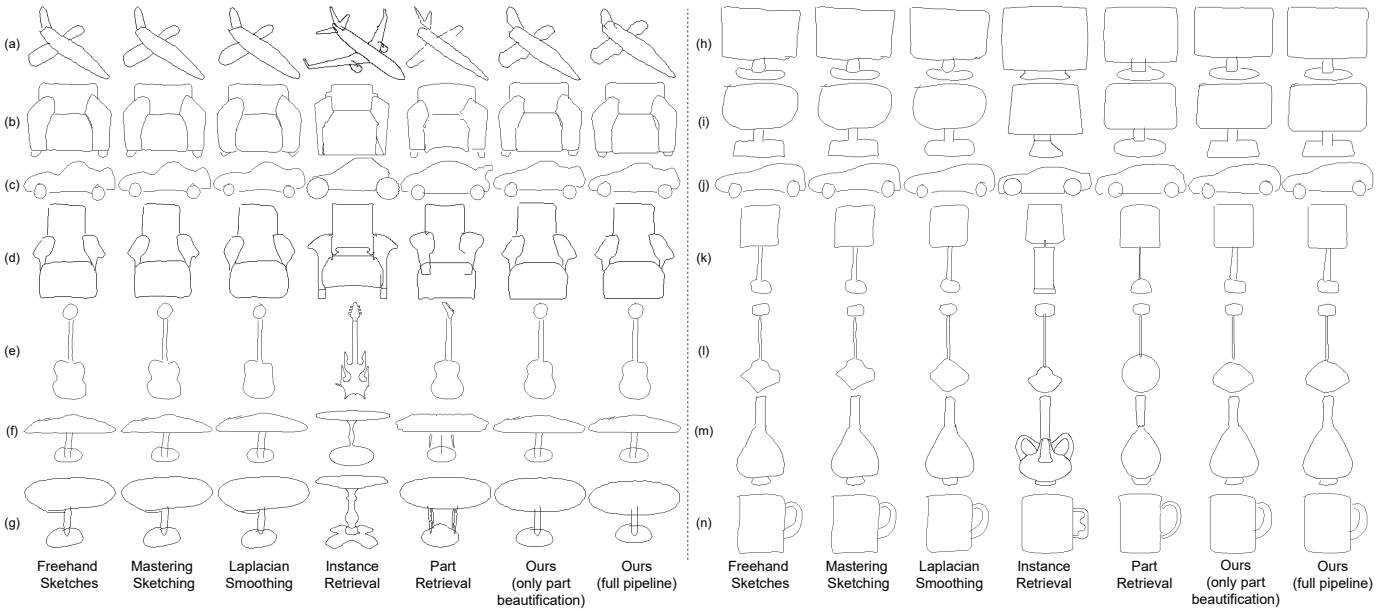

Fig. 10. Comparison of different approaches for sketch beautification. Please zoom in for better visualization.

sketches from a more fine-grained level, it only alleviates the dissimilar phenomenon. Still, these retrieval-based methods have no way to maintain the user's original drawing intentions.

To better demonstrate our sketch beautification method, we further present the intermediate outputs of the part beautification module in Figure 10. Unlike previous learning-based or retrieval-based methods, our part beautification module produces orderly lines, curves, and other primitives in part sketches by redrawing the input part strokes, leverages the knowledge of a large number of synthesized part-level sketches, and also retains shapes that resemble the original inputs without introducing extra strokes or missing any strokes. However, as shown in Figure 10, some structure defects like seams (b), penetration (j), and misalignment (g) between part sketches still exist in the intermediate outputs of our part beautification module. This further emphasizes the importance of structural adjustment. When combined with the other component, our structure beautification module, we can obtain the most beautified outputs with pleasing part geometries and convincing global structures. However, since there is no existing standard criteria to evaluate our proposed sketch beautification approach, we give our own insights and present the evaluation strategy as follows: In fact, beautifying a sketch means adding changes to it while respecting the original sketch. There is no perfect solution here, since two main aspects (i.e., improving beautification quality while respecting the original sketch) need to be balanced during the beautification. Regarding faithfulness to the input sketch and beautification quality, we further conduct a quantitative evaluation on the beautification faithfulness and a perceptual study on the beautification quality in Subsections 4.3.1 and 4.3.2.

Methods	mCD ↓	mEMD($\times 10^2$) ↓
Instance Retrieval	15.55	5.92
Part Retrieval	8.22	5.20
Laplacian Smoothing	3.44	4.52
Mastering Sketching	**1.40**	**2.04**
Ours (only part beautification)	4.59	4.62
Ours (full pipeline)	6.84	5.04

TABLE 2
Quantitative evaluation on the faithfulness of the beautified results (produced by different sketch beautification methods) to the input sketches.

4.3.1 Quantitative Evaluation on Faithfulness

To quantitatively evaluate the performance of different beautification approaches in preserving the user's original drawing intentions (i.e., faithfulness), we report the statistic values of two metrics for the aforementioned methods in Table 2. To measure the difference between a beautified result and the user's original freehand sketch, we adopt the Chamfer Distance-L2 (CD) and Earth Mover's Distance (EMD) as the evaluation metrics (lower is better) for the faithfulness evaluation in the sketch beautification task. The former metric is employed to measure the point-wise distance between the sketched objects in two sketches and the latter one is utilized to compute the distribution-level distance of two point distributions over the entire image space (256 256). Given a pair of the user's freely sketched input and the beautified output produced by one of the compared methods, we calculate the Chamfer Distance of the only valid sketch pixels (i.e., excluding the background pixels) in the two sketches. When computing the Earth Mover's Distance, we preserve both the sketch pixels and the background pixels and treat the pair of the input and output sketches as two distributions. Finally, we average these two metrics over pairs of sketches before and after beautification in our collected freehand sketch dataset and present the mean values of CD and EMD in Table 2.

Consistent with what is observed from Figure 10, as

shown in Table 2, our method outperforms its retrieval-based competitors while falling behind Mastering Sketching and Laplacian smoothing quantitatively. It is reasonable that the results of Mastering Sketching have the closest distance to the input sketches since this method inherits (or accepts) all the input strokes with limited pixel-level beautification. For Laplacian smoothing, since this method only slightly changes the position of the sketched points by smoothing the local-level strokes, it is easier for this method to achieve a remarkable performance (the second ranking) in faithfulness evaluation (see Table 2). While the fine-grained part retrieval outperforms the instance retrieval significantly, it is still inferior to our method. The beautified outputs of our part beautification module and our full pipeline achieve competitive performance to Laplacian smoothing in preserving the user's original drawing intentions (the third and the fourth rankings in Table 2). It is expected that the results of our part beautification module have the closer distance to the user's input sketches compared with our full pipeline (as reported in Table 2) since the part beautification module only beautifies the curve shape of the part sketches without adjusting the scales or locations of the part sketches. Just as we discussed before, there are still a lot of structure errors (e.g., seams, penetration, and misalignment) in the intermediate beautified outputs, as shown in Figure 10 (Ours (only part beautification)). To correct such structural errors during the structure beautification stage, a small range of structural refinement (including pixel-wise translations and scalings) is applied to the beautified part sketches. Therefore, it is inevitable that our full pipeline combining the part beautification and structure beautification modules slightly enlarges the distance to the input freehand sketches. For the faithfulness aspect of the sketch beautification task, since our framework is proposed to beautify both the local geometry and global structure of input sketches, our method cannot keep the input sketches almost unchanged as in Mastering Sketching. But our approach does not heavily change the input sketches as instance and part retrieval, and can also be regarded as a larger "enhancement" operation (making the original input sketches aesthetically more beautiful in both local curve and global shape) to some extent. For the evaluation on the beautification quality of the different sketch beautification methods, we demonstrate it with a user study in the next section.

4.3.2 Perceptual Study on Beautification Quality

It is known that the concept of beautification is highly relevant to human preference and perception. Therefore, to evaluate the beautification quality of the beautified sketches produced by different methods in Section 4.2 (namely, Laplacian smoothing (LS), Mastering Sketching (MS), instance retrieval (IR), part retrieval (PR), our part beautification module only (Ours (PB)) and our full approach), we further conducted a user study to evaluate the performance of these approaches in the sketch beautification task.

Specifically, we first randomly picked a set of 15 input sketches from the drawn freehand sketches from all the categories in our dataset. We then applied the aforementioned six sketch beautification methods to each input sketch to generate the beautified results. Figure 10 displays some representative examples of inputs and outputs used in our user study. The evaluation was done through an online questionnaire. There were in total 23 participants (15 males and 8 females) in this study. We showed each participant sets of an input sketch and six beautified sketches generated by the compared approaches in random order set by set. Each participant was asked to select the most beautiful result in each set regarding the visual aesthetics from both aspects of the local geometry and the global structure. Finally, we received 23 (participants) 15 (input sketches) = 345 subjective evaluations.

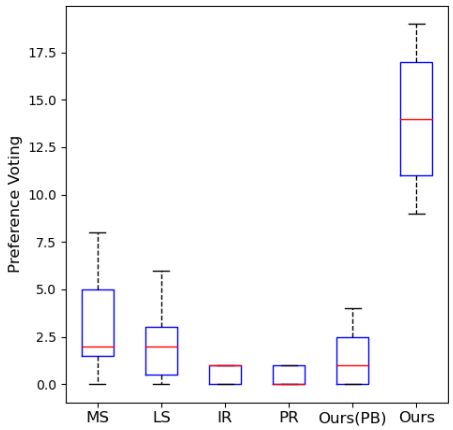

Fig. 11. Box plots of the average preference voting over the prepared questions (beautification quality) for each method. MS, LS, IR, and PR stand for Mastering Sketching, Laplacian smoothing, Instance retrieval, and Part retrieval, respectively. Ours (PB) and Ours refer to the part beautification module and the full pipeline of our sketch beautification approach, respectively.

Figure 11 shows the statistics of the voting results. We conducted one-way ANOVA tests on the preference voting results, and found significant effects for aesthetic preference to our method ($F = 65.88$, $p < 0.001$). Then further paired t-tests show that our method (mean: 13.93) got significantly more votes than all the other methods, Mastering Sketching (mean: 3.13, [$t = 2.14$, $p < 0.001$]), Laplacian smoothing (mean: 2.47, [$t = 2.14$, $p < 0.001$]), instance retrieval (mean: 1.00, [$t = 2.14$, $p < 0.001$]), part retrieval (mean: 0.47, [$t = 2.14$, $p < 0.001$]), and our part beautification module (mean: 2.00, [$t = 2.14$, $p < 0.001$]).

To summarize, our approach achieved the best performance that significantly surpasses its competitors under human aesthetic perceptions. Although there are slightly higher deviations in the beautified results generated by our method compared with the original input sketches, our beautified sketches are still able to be faithfully voted as the best beautification results for the given sketches by the users. In addition, our approach successfully beautifies the local-level part geometry and corrects the global-level structural errors of input sketches. In this way, our proposed method helps to improve the beautification quality of input freehand sketches significantly. We show more beautification results of our method (including the intermediate outputs after part beautification and final results after structure beautification) in the supplemental materials.

4.3.3 Ablation Study

Since we have demonstrated the roles of the part beautification module and the structure beautification module of

Input	Loss	mIOU(%)↑	mCD↓	mEMD($\times 10^2$)↓
sk	skloss	0.00	$+\infty$	$+\infty$
sk	(sk+regu)loss	83.38	5.85	6.75
sk+bb	bbloss	91.87	2.79	5.42
sk+bb	(sk+bb)loss	91.83	2.79	5.42
sk+bb	(sk+bb+regu)loss	**91.89**	**2.78**	**5.40**

TABLE 3
Ablation study of our designed mechanism for the sketch assembly model. The sk and bb in the first column are the inputs of the sketch and the bounding box, respectively. The skloss, bbloss, and reguloss in the second column refer to the supervision losses of the sketch, the bounding box, and the regularization term, respectively.

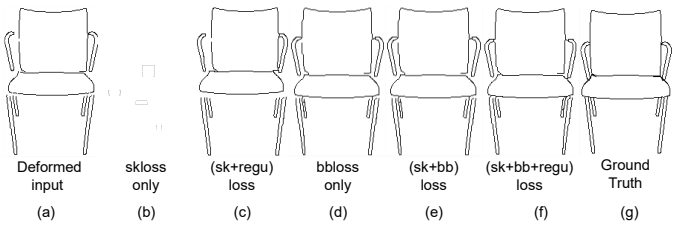

Fig. 12. Visual comparison of different supervision losses during the training process. Please zoom in for better visualization (in particular for "skloss only").

our sketch beautification pipeline qualitatively and quantitatively in the above two subsections, and detailed the key components of the part beautification module in Section 3.2, we do not conduct the redundant ablation studies for the part beautification module here.

Hence, in this subsection, we mainly focus on validating the effectiveness of the key components in the structure beautification module. Since the sketch is a kind of sparse representation, it is a nontrivial task for our sketch assembly model to represent and learn its spatial transformations precisely and effectively. To validate our designed mechanism of the sketch assembly model, we perform ablation studies of its key components in turn, namely, the sketch loss, the bounding box loss, and their combinations with the regularization loss (see Table 3).

Specifically, we use 1,000 randomly sampled synthesized sketches of the Chair category as the ground truth since this category has the most complex and challenging structures in our dataset. We then apply random affine transformations to the parts of the ground truth samples five times and take these 5,000 deformed sketches with the warped parts as the benchmark input. Finally, we evaluate the performance of the trained sketch assembly models under different training strategies by measuring the disparity between their assembly outputs and the ground truth. Note that although the 1,000 ground truth sketches are within the same set used for retrieval in our beautification pipeline in Section 3.2.2, the benchmark (5,000 deformed sketches) for testing here is already significantly different from the original set after five times of random affine transformations. To better reflect the performance of the methods in the sketch assembly task, we compute a region-based and part-level IOU metric on bounding boxes of the transformed part outputs and the corresponding part ground truth, as formulated in the following equation:

$$IOU = \frac{1}{N_p} \sum_{p2P} \frac{M_p^a \cap M_p^{GT}}{M_p^a + M_p^{GT}}; \tag{8}$$

where N_p is the number of parts P in a sketch, M_p^a and M_p^{GT} are the part-level bounding boxes of the assembled sketches and ground-truth sketches, respectively. We also calculate the Chamfer Distance-L2 (CD) and Earth Mover's Distance (EMD) between the assembled and ground-truth sketches to further verify the performance of the ablated sketch assembly models. Table 3 reports the mean values of the above metrics. We also show the performance of different components in the sketch assembly task qualitatively in Figure 12.

In our experiments, we found that only utilizing the sketch loss cannot supervise the learning process of spatial transformations of sparse sketches. Due to limited valid pixels in sketches, the network tends to shift its focus from sketched pixels to the background pixels during the training process, even under the L1 loss of the input and output sketches. With the number of training epochs increasing, the model degrades rapidly and ignores the sketched pixels until the part sketches are rescaled to none. Hence, the mean IOU value of 'skloss only' model in Table 3 is 0. The other two metrics also show the failure of training on sketch input with the sketch loss only. This can be further witnessed in Figure 12 (b). Note that we screen-captured this subfigure of "skloss only" at the very beginning of training since the part sketches would disappear soon after several further epochs. Along with the degradation of "skloss only" model, we observed a significant increment in the parameters of the learned transformation matrices. We further designed and added the regularization loss to penalize this huge oscillation of the learned transformation matrices. However, although the input part sketches would no longer shrink to nothing by applying the additional regularization loss to the learned transformation matrices, the network still failed to learn meaningful spatial transformations (just random translation or scaling under the constraint of the regularization loss), as shown in Figure 12 (c). Only by introducing the bounding box input and the corresponding bounding box loss, the networks were able to learn the spatial transformation of sparse sketches stably and effectively (see the inputs containing 'bb' and the losses with 'bbloss' in Table 3 and Figure 12 (d-f)). With the combination of the sketch loss, the bounding box loss, and the regularization loss, the network achieved the best performance, as shown in Table 3 quantitatively and Figure 12 (f) qualitatively. These results further confirm the necessity and importance of our design choices for the structure beautification module.

5 Conclusion and Future Work

We have introduced an intuitive and effective beautification approach for freehand sketches depicting man-made objects by conducting part-level geometry beautification and global structure refinement sequentially. As one of the key components in our approach, the sketch implicit model can be easily plugged into contemporary deep neural networks for a variety of tasks relevant to sketches including sketch recognition, classification, retrieval, reconstruction, and generation thanks to its promising representation capacity in precisely reconstructing the input sketch (Figure 2), and

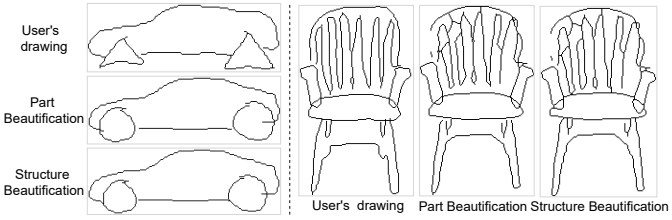

Fig. 13. Failure cases of our sketch beautification method.

in generating novel, smooth, and continuous samples by interpolating among two/multiple existing sketches (Section 3.2.2). The other component, i.e., the sketch assembly model, provides a robust and effective solution for compositing sparse 2D components by having the part-level bounding boxes. The whole beautification pipeline could further inspire and boost the downstream applications that operate over freehand sketches as input. However, beautifying a freehand sketch under an arbitrary view is still challenging for our method. Furthermore, the voting results in our perceptual study (Figure 11) indicate that structure beautification has the strongest effect on user preference when perceiving beautification quality. Therefore, further exploration of how and which factors (e.g., more obvious spatial changes) cause such a preference difference when users perceive geometry and structure changes is a valuable and promising direction for beautification tasks. We leave these for future work to explore.

Our method has some limitations. First, in our method, all the training samples of the sketches of man-made objects are rendered under the canonical view (best view) for each category. This limits the adaptation ability of our method to freehand sketches with large view variations or multi-view sketches. Although this is partially solved by adding a shadow guidance to constrain the user's input, we are interested in addressing this issue in a more elegant way [9]. Second, a key merit of our method is the representative power of the sketch implicit representations that can interpolate the sketched points smoothly and continuously. But this implicit representation requires a longer training time than the CNN representations on the same input data, since the implicit model needs to sample all the coordinates and remember the ground truth value of each point in the 2D image space. One possible solution for speeding up the training process is to change the sampling strategy, e.g., by keeping sampling the points close to the strokes instead of sampling all the points from the whole 2D space. Third, while our method is able to instantiate conceptual freehand sketches, our approach could fail if input sketches are drawn too poorly or too complexly, as shown in Figure 13. If a user draws a very unnatural sketch (see the triangle-like wheels of a car on the left-top, our method might not follow the user's drawing intention and even totally changes the geometry of part sketches drawn by the user. This violates the beautification constraints we set in this paper (as discussed in Section 1). If a user sketches a very complex part (see the chair back with too many sticks on the right), our approach also cannot beautify such a part. This is due to the failure of the correspondence matching step in Section 3.2.3. It is known that correspondence matching is still an open problem in the research community. Therefore, the improvement on registration and correspondence matching could also further boost the performance of our method. Lastly, users should ideally be allowed to adjust the degree of beautification (more significant beautification would lead to larger deviation from the input sketch). In our approach, the beautification function is designed in a closed and automatic way for efficiency reasons. In the future, we would like to extend our sketch beautification approach to allow for more user control.

ACKNOWLEDGEMENTS

We thank the anonymous reviewers for their constructive comments. This work was partially supported by grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (No. CityU 11212119, 11206319, and 11205420), the Chow Sang Sang Group Research Fund/Donation, and the Centre for Applied Computing and Interactive Media (ACIM) of the School of Creative Media, CityU.

REFERENCES

[1] J. Delanoy, M. Aubry, P. Isola, A. A. Efros, and A. Bousseau, "3d sketching using multi-view deep volumetric prediction," *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, vol. 1, no. 1, pp. 1–22, 2018.

[2] Z. Lun, M. Gadelha, E. Kalogerakis, S. Maji, and R. Wang, "3d shape reconstruction from sketches via multi-view convolutional networks," in *2017 International Conference on 3D Vision (3DV)*. IEEE, 2017, pp. 67–77.

[3] B. Wang, J. Sun, and B. Plimmer, "Exploring sketch beautification techniques," in *Proceedings of the 6th ACM SIGCHI New Zealand chapter's international conference on Computer-human interaction: making CHI natural*, 2005, pp. 15–16.

[4] C. L. Zitnick, "Handwriting beautification using token means," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 4, pp. 1–8, 2013.

[5] D. Smirnov, M. Bessmeltsev, and J. Solomon, "Learning manifold patch-based representations of man-made shapes," in *International Conference on Learning Representations*, 2020.

[6] W. Su, X. Yang, and H. Fu, "Sketch2normal: deep networks for normal map generation," in *SIGGRAPH Asia 2017 Posters*, 2017, pp. 1–2.

[7] J. Chen and Y. Fang, "Deep cross-modality adaptation via semantics preserving adversarial learning for sketch-based 3d shape retrieval," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 605–620.

[8] G. Dai, J. Xie, F. Zhu, and Y. Fang, "Deep correlated metric learning for sketch-based 3d shape retrieval," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[9] D. Yu, L. Li, Y. Zheng, M. Lau, Y.-Z. Song, C.-L. Tai, and H. Fu, "Sketchdesc: Learning local sketch descriptors for multi-view correspondence," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 5, pp. 1738–1750, 2020.

[10] C. Li, H. Pan, A. Bousseau, and N. J. Mitra, "Sketch2cad: Sequential cad modeling by sketching in context," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 6, pp. 1–14, 2020.

[11] Y. Gryaditskaya, M. Sypesteyn, J. W. Hoftijzer, S. C. Pont, F. Durand, and A. Bousseau, "Opensketch: a richly-annotated dataset of product design sketches." *ACM Trans. Graph.*, vol. 38, no. 6, pp. 232–1, 2019.

[12] C. Li, H. Pan, A. Bousseau, and N. J. Mitra, "Free2cad: parsing freehand drawings into cad commands," *ACM Transactions on Graphics (TOG)*, vol. 41, no. 4, pp. 1–16, 2022.

[13] Q. Su, X. Bai, H. Fu, C.-L. Tai, and J. Wang, "Live sketch: Video-driven dynamic deformation of static drawings," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 2018, pp. 1–12.

[14] E. Simo-Serra, S. Iizuka, and H. Ishikawa, "Mastering sketching: adversarial augmentation for structured prediction," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 1, pp. 1–13, 2018.

[15] H. Ye, K. C. Kwan, and H. Fu, "3d curve creation on and around physical objects with mobile ar," *IEEE Transactions on Visualization & Computer Graphics*, no. 01, pp. 1–1, 2021.

[16] G. Orbay and L. B. Kara, "Beautification of design sketches using trainable stroke clustering and curve fitting," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 5, pp. 694–708, 2011.

[17] D. P. Van Mossel, C. Liu, N. Vining, M. Bessmeltsev, and A. Sheffer, "Strokestrip: joint parameterization and fitting of stroke clusters," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–18, 2021.

[18] C. Liu, E. Rosales, and A. Sheffer, "Strokeaggregator: Consolidating raw sketches into artist-intended curve drawings," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–15, 2018.

[19] S.-H. Bae, R. Balakrishnan, and K. Singh, "Ilovesketch: as-natural-as-possible sketching system for creating 3d curve models," in *Proceedings of the 21st annual ACM symposium on User interface software and technology*, 2008, pp. 151–160.

[20] D. Dixon, M. Prasad, and T. Hammond, "icandraw: Using sketch recognition and corrective feedback to assist a user in drawing human faces," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2010, pp. 897–906.

[21] Y. J. Lee, C. L. Zitnick, and M. F. Cohen, "Shadowdraw: real-time user guidance for freehand drawing," *ACM Transactions on Graphics (TOG)*, vol. 30, no. 4, pp. 1–10, 2011.

[22] J. Fišer, P. Asente, S. Schiller, and D. Sỳkora, "Advanced drawing beautification with shipshape," *Computers & Graphics*, vol. 56, pp. 46–58, 2016.

[23] E. Simo-Serra, S. Iizuka, K. Sasaki, and H. Ishikawa, "Learning to simplify: fully convolutional networks for rough sketch cleanup," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, pp. 1–11, 2016.

[24] N. Schor, O. Katzir, H. Zhang, and D. Cohen-Or, "Componet: Learning to generate the unseen by part synthesis and composition," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 8759–8768.

[25] Q. Yu, Y. Yang, Y.-Z. Song, T. Xiang, and T. Hospedales, "Sketch-a-net that beats humans," in *Proceedings of the British Machine Vision Conference (BMVC)*. BMVA Press, 2015, pp. 1–12.

[26] D. Ha and D. Eck, "A neural representation of sketch drawings," in *International Conference on Learning Representations*, 2018.

[27] W. Wu, K. Cao, C. Li, C. Qian, and C. C. Loy, "Transgaga: Geometry-aware unsupervised image-to-image translation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 8012–8021.

[28] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 8110–8119.

[29] H. Dutagaci, C. P. Cheung, and A. Godil, "A benchmark for best view selection of 3d objects," in *Proceedings of the ACM workshop on 3D object retrieval*, 2010, pp. 45–50.

[30] K. Mo, S. Zhu, A. X. Chang, L. Yi, S. Tripathi, L. J. Guibas, and H. Su, "Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 909–918.

[31] L. Gao, J. Yang, T. Wu, Y.-J. Yuan, H. Fu, Y.-K. Lai, and H. Zhang, "Sdm-net: Deep generative network for structured deformable mesh," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 6, pp. 1–15, 2019.

[32] O. Sidi, O. van Kaick, Y. Kleiman, H. Zhang, and D. Cohen-Or, "Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering," in *Proceedings of the 2011 SIGGRAPH Asia Conference*, 2011, pp. 1–10.

[33] I. E. Sutherland, "Sketchpad a man-machine graphical communication system," *Simulation*, vol. 2, no. 5, pp. R–3, 1964.

[34] T. Pavlidis and C. J. Van Wyk, "An automatic beautifier for drawings and illustrations," *ACM SIGGRAPH Computer Graphics*, vol. 19, no. 3, pp. 225–234, 1985.

[35] T. Igarashi, S. Matsuoka, S. Kawachiya, and H. Tanaka, "Interactive beautification: A technique for rapid geometric design," in *ACM SIGGRAPH 2007 courses*, 2007, pp. 18–es.

[36] C. Yan, D. Vanderhaeghe, and Y. Gingold, "A benchmark for rough sketch cleanup," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 6, pp. 1–14, 2020.

[37] Z. Huang, H. Fu, and R. W. Lau, "Data-driven segmentation and labeling of freehand sketches," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 6, pp. 1–10, 2014.

[38] L. Li, H. Fu, and C.-L. Tai, "Fast sketch segmentation and labeling with deep learning," *IEEE computer graphics and applications*, vol. 39, no. 2, pp. 38–51, 2018.

[39] L. Yang, J. Zhuang, H. Fu, X. Wei, K. Zhou, and Y. Zheng, "Sketchgnn: Semantic sketch segmentation with graph neural networks," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 3, pp. 1–13, 2021.

[40] Q. Su, W. H. A. Li, J. Wang, and H. Fu, "Ez-sketching: three-level optimization for error-tolerant image tracing." *ACM Trans. Graph.*, vol. 33, no. 4, pp. 54–1, 2014.

[41] S.-Y. Chen, W. Su, L. Gao, S. Xia, and H. Fu, "Deepfacedrawing: Deep generation of face images from sketches," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 72–1, 2020.

[42] R. G. Schneider and T. Tuytelaars, "Sketch classification and classification-driven analysis using fisher vectors," *ACM Transactions on graphics (TOG)*, vol. 33, no. 6, pp. 1–9, 2014.

[43] L. S. F. Ribeiro, T. Bui, J. Collomosse, and M. Ponti, "Sketchformer: Transformer-based representation for sketched structure," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 14 153–14 162.

[44] A. K. Bhunia, P. N. Chowdhury, Y. Yang, T. M. Hospedales, T. Xiang, and Y.-Z. Song, "Vectorization and rasterization: Self-supervised learning for sketch and handwriting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 5672–5681.

[45] M. Eitz, J. Hays, and M. Alexa, "How do humans sketch objects?" *ACM Transactions on graphics (TOG)*, vol. 31, no. 4, pp. 1–10, 2012.

[46] P. Xu, Y. Huang, T. Yuan, K. Pang, Y.-Z. Song, T. Xiang, T. M. Hospedales, Z. Ma, and J. Guo, "Sketchmate: Deep hashing for million-scale human sketch retrieval," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8090–8098.

[47] L. Li, C. Zou, Y. Zheng, Q. Su, H. Fu, and C.-L. Tai, "Sketch-r2cnn: An rnn-rasterization-cnn architecture for vector sketch recognition," *IEEE transactions on visualization and computer graphics*, vol. 27, no. 9, pp. 3745–3754, 2020.

[48] A. K. Bhunia, S. Koley, A. F. U. R. Khilji, A. Sain, P. N. Chowdhury, T. Xiang, and Y.-Z. Song, "Sketching without worrying: Noise-tolerant sketch-based image retrieval," *arXiv preprint arXiv:2203.14817*, 2022.

[49] Y. Deng, J. Yang, and X. Tong, "Deformed implicit field: Modeling 3d shapes with learned dense correspondence," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 286–10 296.

[50] Z. Chen, A. Tagliasacchi, and H. Zhang, "Bsp-net: Generating compact meshes via binary space partitioning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 45–54.

[51] S. Saito, Z. Huang, R. Natsume, S. Morishima, A. Kanazawa, and H. Li, "Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2304–2314.

[52] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "Deepsdf: Learning continuous signed distance functions for shape representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 165–174.

[53] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3d reconstruction in function space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4460–4470.

[54] Z. Chen and H. Zhang, "Learning implicit fields for generative shape modeling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5939–5948.

[55] J. Li, K. Xu, S. Chaudhuri, E. Yumer, H. Zhang, and L. Guibas, "Grass: Generative recursive autoencoders for shape structures," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 1–14, 2017.

[56] H. Wang, N. Schor, R. Hu, H. Huang, D. Cohen-Or, and H. Huang, "Global-to-local generative model for 3d shapes," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 6, pp. 1–10, 2018.

[57] R. Wu, Y. Zhuang, K. Xu, H. Zhang, and B. Chen, "Pq-net: A generative part seq2seq network for 3d shapes," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2020, pp. 826–835.

[58] T. Y. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Communications of the ACM*, vol. 27, no. 3, pp. 236–239, 1984.