SketchMetaFace: A Learning-based Sketching Interface for High-fidelity 3D Character Face Modeling

Zhongjin Luo, Dong Du, Heming Zhu,
Yizhou Yu, *Fellow, IEEE,* Hongbo Fu, Xiaoguang Han✉, *Member, IEEE*

Abstract—Modeling 3D avatars benefits various application scenarios such as AR/VR, gaming, and filming. Character faces contribute significant diversity and vividity as a vital component of avatars. However, building 3D character face models usually requires a heavy workload with commercial tools, even for experienced artists. Various existing sketch-based tools fail to support amateurs in modeling diverse facial shapes and rich geometric details. In this paper, we present *SketchMetaFace* - a sketching system targeting amateur users to model high-fidelity 3D faces in minutes. We carefully design both the user interface and the underlying algorithm. First, curvature-aware strokes are adopted to better support the controllability of carving facial details. Second, considering the key problem of mapping a 2D sketch map to a 3D model, we develop a novel learning-based method termed "Implicit and Depth Guided Mesh Modeling" (IDGMM). It fuses the advantages of mesh, implicit, and depth representations to achieve high-quality results with high efficiency. In addition, to further support usability, we present a coarse-to-fine 2D sketching interface design and a data-driven stroke suggestion tool. User studies demonstrate the superiority of our system over existing modeling tools in terms of the ease to use and visual quality of results. Experimental analyses also show that IDGMM reaches a better trade-off between accuracy and efficiency.

Index Terms—Sketch-based 3D Modeling, Face Modeling, Neural Network.

1 Introduction

Creating 3D virtual avatars is a prolonged research topic in computer graphics and benefits various usage scenarios such as filming, gaming, and art designing. Typically, this process is a highly skilled task, as experienced artists need to spend several days or even months formally sculpting high-fidelity 3D faces with vivid surface details using commercialized 3D modeling tools (e.g., ZBrush, MAYA, and 3D MAX). To assist amateur users in freely instantiating their ideas as professional modelers, researchers in computer graphics and human-computer interaction have designed systems that allow users to model 3D shapes with freehand sketches based on geometric principles [1], [2], [3], [4], [5], [6]. Although traditional sketch-based 3D modeling systems, such as Teddy [1] and FiberMesh [2], enable amateur users to create 3D models, they usually require tremendous manual work to specify complex geometry.

Thanks to the recent progress in deep learning, the understanding of freehand sketches and the quality of single-view generation have reached an unprecedented level. Several intelligent sketch-based modeling systems have been developed to enable novice users to create visually plausible 3D models within a few minutes [7], [8], [9]. Closest to our work, DeepSketch2Face [10] presents the first deep learning-based sketching system for modeling 3D faces by mapping 2D sketches into a parametric space for face generation. However, considering the limited representation power of the parametric model, DeepSketch2Face can only produce 3D human faces with fixed styles and cannot be used for sculpting expressive skin wrinkles. SimpModeling [11] proposed a two-phase scheme that allows for diverse animalmorphic head modeling using 3D sketches. Nevertheless, it is challenging for users to work with as it relies on complicated and user-unfriendly 3D interactions. Additionally, the system struggles to generate fine-grained details due to the ambiguity of mono-typed strokes and the limited capability of PIFu [12], [13].

In this paper, we design and present *SketchMetaFace*, a powerful sketch-based 3D face modeling system that addresses the following challenges:

Accuracy. Recent learning-based sketching systems [10], [11], [14], [15], [16] allow novice users to create visually-plausible 3D models with a few strokes. However, they are not capable of designing shapes with fine-grained details. To assist users in conveying their ideas more accurately, we adopt curvature lines [6], [14], [17] in learning-based 3D face modeling. We will demonstrate how the curvature-aware strokes significantly boost the quality of detailed surfaces generated from sketches.

Although existing models [18], [19], [20], [21], [22] can map 2D images, including sketch images, to 3D shapes, they may fail to generate watertight 3D meshes with delicate details. A straightforward way to produce shapes with surface details is to blend high-quality multi-view depth maps generated by image translation [23]. Nonetheless, it is nontrivial to fuse the generated depth maps seamlessly into a watertight mesh. An alternative approach is to adopt the pixel-aligned implicit function (PIFu) [12], [13] to reconstruct watertight 3D shapes from single images. However, PIFu exhibits bounded performance in generating high-fidelity geometric details. Inspired by the fact that the depth map produced by image translation contains more intriguing details than PIFu-generated shapes, we propose IDGMM, i.e., **I**mplicit

- *Z. Luo, D. Du, H Zhu, and X. Han are with the School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, China. Y. Yu is with the Department of Computer Science at the University of Hong Kong. H. Fu is with the School of Creative Media, City University of Hong Kong.*
- *X. Han is the corresponding author. E-mail: hanxiaoguang@cuhk.edu.cn*

Manuscript received xxxx.

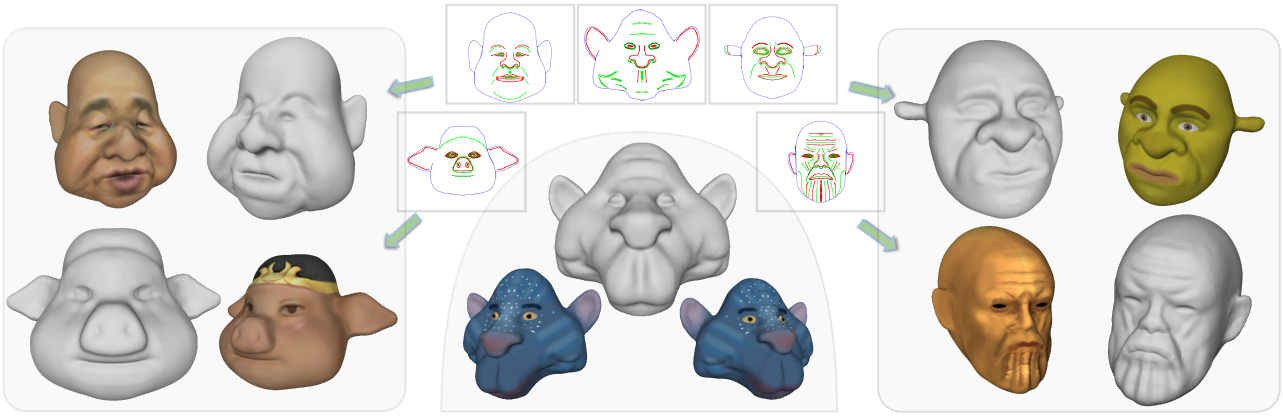


Fig. 1: We present *SketchMetaFace*, a novel sketching system designed for amateur users to create high-fidelity 3D character faces. With curvature-aware strokes (valley strokes in green and ridge strokes in red), novice users can smoothly customize detailed 3D heads. Note that our system only outputs geometry without texture and texturing is achieved using commercial modeling tools.

and **D**epth **G**uided **M**esh **M**odeling. It enjoys the merits of mesh, depth-map and implicit representations to produce high-quality 3D shapes from curvature-aware sketch images.

Usability. While curvature-aware strokes empowers users to create 3D faces with fine-grained details, it may increase their cognitive load. To address this issue, we interview potential users and thoroughly analyse their requirements. We design our system based on the analyzed requirements and formulate a coarse-to-fine interactive scheme: users first get used to the system with mono-typed sketches and then switch to fine-detail crafting with curvature-aware strokes soon as users get familiar with the system. We also carefully design a stroke suggestion component that bridges the gap between coarse and detailed sketching. Moreover, to follow the "as-2D-as-possible" principle, we keep the placement of ears as the only 3D interaction in our system.

To demonstrate the effectiveness of our system, we carefully conduct user studies, from which we conclude that our proposed system exhibits better usability than existing sketch-based 3D face modeling systems [10], [11]. Our system allows amateur users to create diverse shapes with fine-grained geometric details. By conducting comparisons against existing inference algorithms for mapping a single sketch to a 3D model, we demonstrate that results generated by our proposed IDGMM better reflect the appearances of the input sketches. Ablation studies are conducted to justify each design in our interface and algorithm. The contributions of our paper can be summarized as follows:

- We present a novel, easy-to-use sketching system that allows amateur users to model high-fidelity 3D character faces in minutes (as seen in Fig. 1).
- We carefully design a user interface: 1) the face modeling work follows a coarse-to-fine scheme and relies mainly on intuitive 2D freehand sketches; 2) we adopt curvature-aware strokes for modeling geometric details; 3) we introduce a data-driven suggestion tool to ease the cognitive load throughout the sketching process.
- We propose a novel method, i.e., Implicit and Depth Guided Mesh Modeling (IDGMM), which fuses the advantages of mesh, implicit, and depth representations for detailed geometry inference from 2D sketches.

2 Related Work

In this section, we will present relevant studies on 3D avatar modeling, geometrical sketch-based modeling, and data-driven sketch-based modeling. We are aware of the breathtaking progress in sketch-based 2D image generation of faces [24], [25]. However, we will not discuss these in detail due to the page limit.

2.1 3D Face from 2D Image

Creating visually plausible 3D avatars is a long-standing computer graphics and vision problem. Compared with 3D face reconstruction methods, which take multi-view [26], [27] or monocular video [28], [29] as input, single image reconstruction (SVR) and sketch-based modeling provide more casual means for novices to customize 3D faces. Single-image 3D face reconstruction can be roughly divided into two streams, namely, photo-realistic human face reconstruction and caricature face reconstruction.

The works on single-image photo-realistic face reconstruction can be further separated into two genres, i.e., parametric and shape-from-shading methods. However, neither can be directly adopted for modeling detailed 3D faces. Parametric-based models [30], [31], [32] fall short in representing shapes with novel and customized surface details. Shape-from-shading-based methods [33], [34] suffer from deriving geometric clues from non-photo-realistic image inputs, e.g., caricature images and sketches.

Compared with single-image realistic 3D faces generation, which has been extensively studied and achieved exceptionally high quality, the researches on 3D caricature avatars are relatively sparse. A possible reason is that caricature 3D faces are shapes with more diversified geometry, making them extremely hard to be regularized into a single parametric model losslessly. Some work [35], [36], [37] introduced deformations to increase the capability of parametric models. However, their works are still far from generating high-fidelity 3D caricature shapes of various styles. More importantly, given that most single-image caricature face modeling methods require high-quality images as input, novice users cannot further customize the shape as they wish.

Recently, researchers have also explored various schemes for interactive modeling from 2D sketch images [10], [16], [18], [19], [20], [38], [39]. In line with our work, DeepSketch2Face [10] proposed a sketch modeling system that allows users to create

caricature heads from scratch. Their method relies on a CNN-based model to parse a user-drawn sketch as the parameters for a morphable face model. However, since the 3D caricature shape is confined to the parametric caricature face model, DeepSketch2Face cannot faithfully reflect large deformations and wrinkle details presented in the sketch. To address this issue, SAniHead [16] proposed a view-surface collaborative mesh generative network, which turns dual-view freehand sketches into animalmorphic heads. Nevertheless, it fails to synthesize novel shapes deviating from training datasets due to the restricted generalization ability of their network. Our system utilizes the advantages of mesh, depth-map, and implicit representations to generate high-quality 3D shapes from curvature-aware sketch images.

2.2 Geometrical Sketch-based Modeling

Designing free-form 3D shapes via freehand sketching has drawn considerable attention in recent decades [40]. Igarashi et al. [1] pioneer by proposing the first sketch-based modeling system that allows users to create 3D shapes from scratch by sketching 2D contour lines. A large stream of subsequent researches [41], [42], [43], [44], [45], [46] has mainly focused on designing novel interpolation functions to interpolate sketched contours lines smoothly. Unlike the sketch-based modeling systems mentioned above, which take 2D sketches as input, Fibermesh [2] allows users to model free-form surfaces by sketching and manipulating 3D curves. While Fibermesh [2] and its follow-up systems [47], [48] reduce the ambiguity remarkably with explicitly defined 3D curves, they are not capable of or are not friendly for novice users to carve organic surface details (e.g., skin wrinkles).

To emboss interpolated surfaces with sharper details, various methods introduce sketches with different semantics [49], [50] or curvature cues [6], [17] to formulate more determined constraints. However, additional inputs may significantly increase novice users' cognitive load. Inspired by BendSketch [6], our system allows users to draw with curvature-aware strokes, which serve as a less ambiguous means for users to specify the bulge and sink on faces accurately. To reduce the cognitive load of using curvature-aware strokes, we introduce a carefully designed sketch suggestion module to support amateurs in getting familiar with our system intuitively.

2.3 Data-driven Sketch-based Modeling

The recent decade has witnessed the emergence of data-driven methods for sketch-based 3D shape generation thanks to large-scale 3D datasets. The data-driven sketch-based modeling systems can be roughly divided into two streams regarding the shape generation approaches, i.e., retrieval-based and learning-based.

Retrieval-based methods [51], [52], [53], [54] consume a freehand sketch for the query and search for the most similar shape from the data warehouse as the reconstruction output. Fan et al. [55] propose a suggestive interface with shadow guidance to guide object sketching. However, shadow guidance may introduce severe visual cluttering for sketches with different semantics. Xie et al. [56] proposed to retrieve candidate object parts from a database with part sketches for further assembly. Recently, deep neural networks have been applied for retrieval-based sketch modeling systems [57], which have shown their superiority compared to their traditional learning-based counterparts in handling noisy sketch input created by novice users. However, limited by the capacity of the data warehouse, retrieval-based sketch modeling may produce shapes that drift away from input sketches.

In recent years, learning-based solutions have been popular for sketch-based 3D shape generation and editing [10], [11], [15], [18], [19], [20], [38], [39], [58], [59], [60], [61], [62], [63], [64]. For example, Nishida et al. [64] proposed inferring urban building parameters from freehand sketches with convolutional neural networks, while Huang et al. [62] presented an interactive modeling system that infers parameters for procedural modeling from sketches. DeepSketch2Face [10] proposed a deep regression model that converts a sketch into the parameters of a morphable 3D caricature face model. However, the above parametric regression-based methods work only for 3D shapes within a specific category that can be easily parameterized. Du et al. [63] adopted implicit learning to produce artificial object parts from sketches and proposed a deep regression model to predict the position of the parts, while Sketch2CAD [15] enables users to achieve controllable part-based CAD object modeling by sketching in context. SimpModeling [11] utilized a coarse-to-fine modeling scheme, allowing users to create desired animalmorphic heads with 3D curves and on-surface sketching. We argue that 2D sketching would be more intuitive than 3D sketching since most novice users are more familiar with 2D interfaces and interactions. Furthermore, SimpModeling falls short in generating fine-grained geometric details due to the ambiguity of mono-typed strokes and the bounded capability of its shape-generation network. In this paper, our system allows users to create 3D high-fidelity facial models with 2D curvature-aware sketches intuitively.

3 User Interface

This section first summarizes the requirements of designing sketch-based modeling for novice users to customize high-fidelity faces of highly diversified styles. On top of the design goals, we will introduce the crucial designs of our system and justify how they reflect the design goals. Please refer to the accompanying video for sketch-based modeling in action.

3.1 Design Requirements and Analysis

In the design process of our sketch-based 3D face modeling system, we interviewed 11 participants with different levels of modeling experience to analyze the demands for a user-friendly sketch-based modeling interface. Three of these participants were modelers with more than five years of experience in 3D modeling, while the rest were novice users with no or little knowledge of 3D modeling. Based on the responses, we summarize the following design goals and the corresponding design choices for our system:
Coarse to Fine (R1). After briefly introducing the background knowledge about sketch-based 3D shape modeling, we first discuss whether users prefer working in a top-down or bottom-up manner. All experts and most novice users preferred to model the shape in a top-down manner. Therefore, our proposed sketch-based modeling system allows users to model 3D faces in a coarse-to-fine manner [11]. In the coarse stage, users can design the contour and the attachment of the faces (e.g., ears). After users finish designing a coarse head shape, they will move on to the fine-grained shape modeling stage, where they can carve geometrical details such as wrinkles, mouths, eyes, etc. Note that we treat ears as attachments and adjust their position through 3D interactive operations in the coarse stage since it is difficult to determine the 3D location of attachments just via frontal-view sketching.

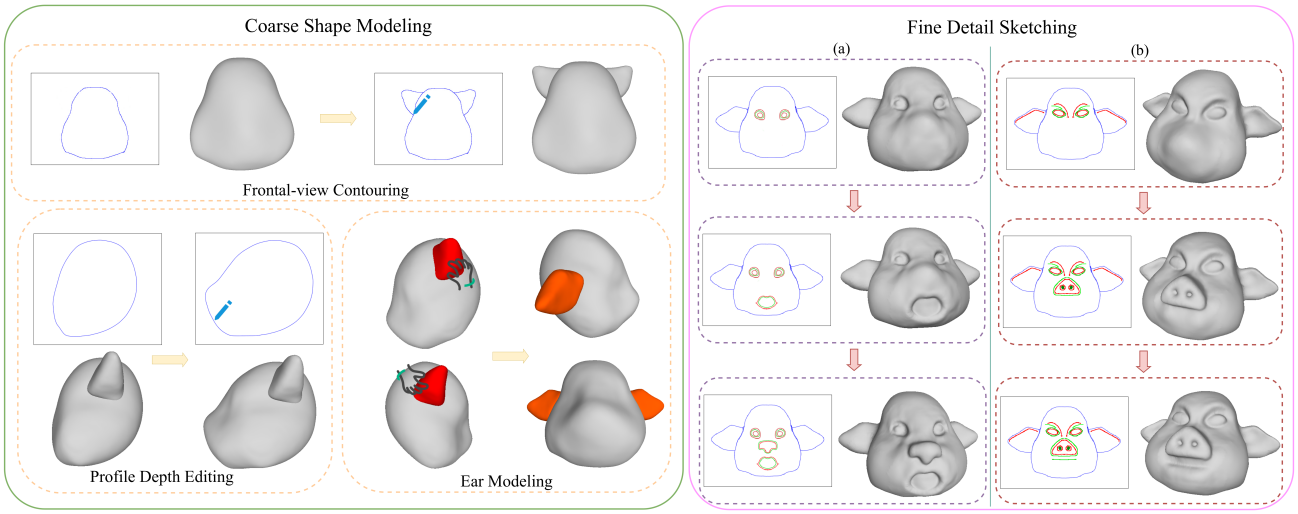


Fig. 2: An illustration of the interactions supported by our system. In the *Coarse Shape Modeling* stage, users may define coarse 3D faces with frontal-view contouring, profile depth editing, and ear modeling. In the *Fine Detail Sketching* stage, users can further carve fine-grained surface details with the curvature-aware strokes.

As 2D as Possible (R2). When discussing whether 3D interactions should be dominant in the system, most novice users mentioned that they prefer to express their ideas through 2D drawings. Interestingly, even professional modelers agree that 2D interactions should be the dominant interaction for the system, as they believe novices may get bored with manipulating the cameras and the 3D shapes. To this end, our system follows the "as-2D-as-possible" principle. Users can finish most of the design only with a 2D sketch pad, and 3D interactions (e.g., tuning the layout of ears) are introduced only when necessary.

Agile and Precise (R3). While some amateurs mentioned that they want to carve a 3D face carefully according to a reference character face, others only intend to customize a visually-plausible 3D face with a few strokes. Hence, our system allows users to customize 3D faces with different degrees of interaction complexity, as shown in the demo video. Novice users can quickly orchestrate a visually plausible 3D face with the dedicated sketch stroke suggestion module. The sketch stroke suggestions also serve as a decent initialization for detailed face modeling. For users who are interested in carving customized surface details, we provide curvature-aware strokes that allow the specification of surface details to be more precise.

3.2 Coarse Shape Modeling

To support the design requirements mentioned in Section 3.1, in our system, the modeling of high-fidelity 3D faces is decomposed into coarse shape modeling and fine detail sketching (**R1**). Users may start designing a coarse 3D face by drawing face contour lines on the 2D sketching pad view, as illustrated in Fig. 2. Novice users could switch to the symmetric sketching mode. Under this mode, mirror-symmetrical strokes will be generated as the user draws on the sketch pad. In this stage, our system can produce a 3D model in a real-time manner by responding to each drawing operation.

Profile Depth Editing. The essence of our system lies in eliminating 3D user interactions (**R2**). However, the generated 3D faces with single-view contour strokes lack depth variances along the z-axis due to the missing constraints on the depth channel. To this end, we deliberately design a profile depth editing interaction scheme that allows users to specify the face contours in the lateral view. Once users switch to the depth editing mode, a new canvas will appear with an initial side-view rendered 3D face contour. As seen in Fig. 2, novice users may design shapes with sharp-variant depth by revising the profile sketch without directly manipulating the 3D shapes.

Ear Modeling. The attachments of 3D faces, i.e., the ears, play an essential role in shaping a virtual character's characteristics and styles. Unlike nose, eyes, and mouth, ears (and other face attachments) are of greater diversity in 3D layout, making it challenging to use only frontal-view sketching to express. To this end, our system uses separate meshes to represent the face and the ears for better expressiveness. Users may customize the ears by drawing their contour lines on the 2D sketch pad view, like specifying the coarse head shape. Specifically, the ears (also for other attachments like horns) are sketched on individual canvas layers, which facilitate users to manipulate their 2D attachment layouts and help the backend models learn diversified attachment shapes. As illustrated in Fig. 2, users can modify the 3D layout of the ears in the 3D view for more precise control of the generated shape. Users can also copy attachments as in RigMesh [3]. It is worth mentioning that layout manipulation and attachment copying are the only 3D operations in the whole modeling procedure (**R2**).

3.3 Fine Detail Sketching

After the user customizes the coarse face shape, they may further characterize the detailed facial geometry, e.g., eyes, noses, mouth, and wrinkles. Although previous works, e.g., DeepSketch2Face [10] and SimpModeling [11], allow users to edit surface details through 2D and 3D sketching, they fall short in generating diversified and controllable surface details due to the ambiguous mono-typed sketch strokes.

Curvature-aware Strokes. We adopt curvature-aware strokes [6] to alleviate the sketch's ambiguity, enabling users to carve surface details precisely (R3). Specifically, two types of strokes (i.e., ridge and valley) are defined. Before each stroke drawing, the user needs to pick a type first. Different stroke types are visualized with different colors (i.e., red for ridge and green for valley). Our

system also supports tunable depth for each stroke, which defines the curvature amplitude, i.e., greater depth (darker color) means a higher ridge or deeper valley.

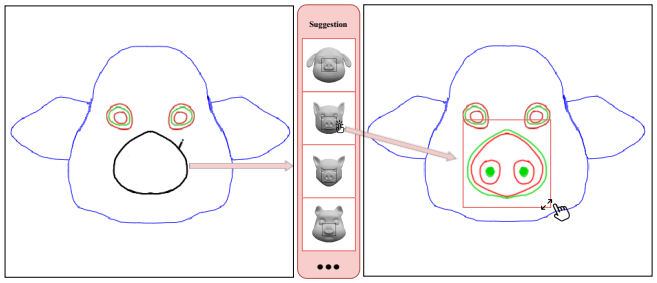


Fig. 3: An illustration of our stroke suggestion component. Soon after users specify the style, target region, and facial components to be modeled, the stroke suggestion component retrieves the relevant curvature-aware strokes. Users may also manipulate the layout for the retrieved strokes through dragging and scaling.

Stroke Suggestions. While curvature-aware strokes significantly improve the controllability of our system, they inevitably bring additional cognitive load for novice users. To address this, we carefully design a data-driven stroke suggestion tool. Consider a scenario when a user wishes to draw a pig nose on the face, as illustrated in Fig. 3. Our system allows the user to pick the "nose" type and select a "pig" style first, and then draw a contour to specify the rough shape and the location where they wish to place the nose. After that, a set of strokes with the specified category, as well as the corresponding shapes, is retrieved from the database and presented as "Suggestion". The user can picks one which can be placed automatically or after manually adjusting the location and size. Users were provided 20 suggestions each time, and the retrieved sketches are editable. With such a suggestion tool, amateurs can quickly compile a neat 3D face model with the high-quality sketch strokes in the database and kick off instantiating their ideas on a decent basis. The suggestion tool is implemented by a retrieval neural network based on the auto-encoder structure, please refer to the supplemental materials for details.

Instant Shape Preview. An instant preview of the 3D shape could serve as guidance for further sketching. However, due to the geometry complexity, the model inference in the stage of fine-detail sketching takes around 0.5s, making it unable to support real-time response. Our video shows that we adopt image space rendering and generate the frontal-view normal map as a real-time shape preview. Please refer to the supplemental materials for the implementation details of the instant preview module.

4 METHODOLOGY

In this section, we present the details of the backend models that support the interactive sketching interface.

Overview. Following our coarse-to-fine interface design, we discuss the algorithms used for the two stages accordingly. In the coarse stage, as illustrated in Fig. 4, we propose a part-separated implicit learning method that maps the coarse input sketch S_r to separated part meshes (i.e., face and attachments). After the user tunes the part layout, these separated meshes are merged into a single mesh M_c. We then render the outer contour [65] of M_c into the sketch image S_c, on which users can add fine strokes in the detail sketching stage.

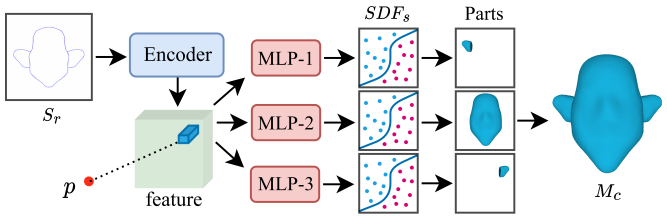


Fig. 4: An illustration of o our part-separated coarse modeling of a 3D face with an outline sketch input S_r. It shows the generation of three parts of a face region and two ears using PIFu, and then assembles and merges them to obtain a coarse model M_c.

In the detail sketching stage, users may further craft fine-grained surface details through sketching on the rendered coarse sketch image S_c. To generate detailed geometry M_f from the fine sketch S_f, as shown in Fig. 5, we propose IDGMM, which learns a progressive deformation from M_c to M_f, under the guidance of both the learned implicit field (SDF) and the learned depth map from S_f.

4.1 Preliminary

Before introducing the proposed model, we will briefly review some relevant concepts and building blocks.

Pix2Pix. Given a source image I_s, Pix2Pix [23] learns a mapping from I_s to a target image I_t, i.e., $f : I_s \rightarrow I_t$ in an adversarial manner. Commonly, a U-Net is adopted to model this translation, and the conditional GAN loss and the reconstruction loss (L_1 or L_2 loss) are used for training. In our model, the Pix2Pix module is adopted for translations among sketch images, depth maps, and normal maps.

Implicit Learning. Recently, various deep representations have been used for 3D shape reconstruction, e.g., voxels, point clouds, meshes, and implicit fields. Among them, implicit field-based methods achieve state-of-the-art performance [66], [67], [68]. There are two commonly used formulations to model implicit surfaces: occupancy and signed distance function (SDF). Occupancy is a continuous function g_o that maps a query point $p \in R^3$ to a binary status $o \in \{0, 1\}$, indicating inside/outside of the surface. SDF is a function g_s that maps p to its signed distance s to the underlying surface. A multi-layer perception (MLP) is usually adopted for approximating g_o or g_s.

PIFu. Among the works relevant to single image 3D shape reconstruction, pixel-aligned implicit function (PIFu) outperforms its counterparts in generating results better matching input images. Specifically, PIFu models a function h to map $p \in R^3$ with a projected pixel-aligned feature f_p to an occupancy o or SDF value d, i.e., $h : \{p, f_p\} \rightarrow o/d$. Firstly, an hourglass architecture [69] is applied on I to obtain a feature map I_f. Then, p is projectd onto I_f to obtain f_p. MLP is used to model h. Our system also requires input-aligned results, so we adopt PIFu as the base module for shape inference from sketch images. Our method uses SDF-based PIFu since it is more suitable for providing deformation guidance.

PIFu with Normal Input. As a follow-up work of PIFu, PIFuHD [13] proposed a coarse-to-fine pixel-aligned implicit shape learning pipeline to generate more geometry details. More specifically, it utilizes PIFu as the coarse-level learning and adopts generated normal maps for fine-level learning. Inspired by PIFuHD, we infer normal maps from the input sketch images with Pix2Pix to assist in learning fine-grained surface details. Similar to the design

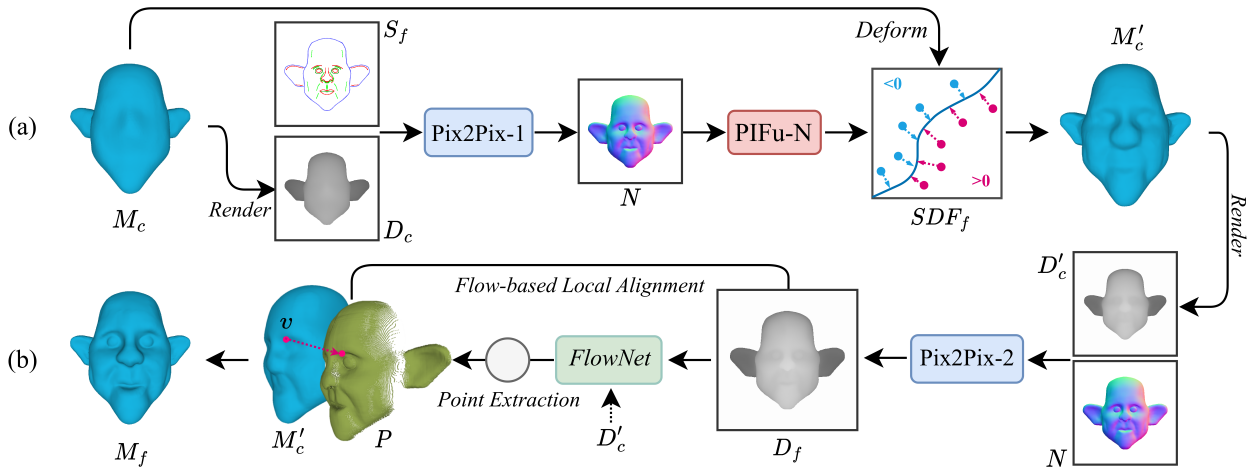


Fig. 5: The architecture of our IDGMM. (a) Taking a coarse mesh M_c as input, it is first rendered into a depth map D_c. D_c together with the input fine sketch S_f are fed into Pix2Pix-1 to generate a normal map N. N is applied to generate an implicit field using PIFu-N. Under the guidance of the SDF field, M_c is deformed to obtain an updated mesh M'_c. (b) We then render M'_c into a depth map D'_c, which is enhanced to D_f with a Pix2Pix-2 module. After a flow-based local depth alignment, we obtain a high-quality point cloud P from the warped depth map. P is locally aligned with M'_c and used to guide mesh refinement from M'_c to the resulting mesh M_f. Note that the process of sketching is iterative, and the mesh obtained at step (n-1) is used as the input M_c for the n-th step.

proposed in PIFuHD, we maintain a tiny MLP that extracts local image features from the inferred normal maps to generate high-frequency details. In the following sections, we will use PIFu-N to denote our PIFu with normal input.

4.2 Coarse Modeling

In the coarse stage, users only need to draw a rough outline for a desired face, i.e., the face contour and attachment contours (e.g., ears). A straightforward way to generate a coarse model from the outline sketch S_r is to use PIFu, which maps S_r to an implicit field. Subsequently, Marching Cubes [70] can be adopted to extract a mesh from the implicit field. However, as the attachments and the face are represented with a single mesh, users cannot directly manipulate the layout for the attachments, thus significantly weakening users' control over modeling results.

4.2.1 Part-separated PIFu

To boost the controllability of our system, we present a novel part-separated PIFu. Let's first consider a common scenario where a face contains a left ear and a right ear. As shown in Fig. 4, three different PIFu modules are used to model the three parts separately. They use different MLPs but share a common encoder that maps S_r to feature maps. In our implementation, the number of parts is fixed. The MLPs designed for ear parts can also be used to generate ear-like attachments, such as horns.

The 3D location of each ear is kept without any normalization during training, which makes the network learn the layout of ears automatically. After obtaining the implicit field of each part, we extract separated meshes from them (for better efficiency, 64^3 resolution is used for marching cube). After users manipulate 3D ear placements, those meshes are merged into a single one with a *corefine-and-compute-union* operation provided by CGAL [1]. After this step, we apply a remeshing method [71] to get M_c.

1. CGAL: the Computational Geometry Algorithms Library. https://www.cgal.org/.

Although our curvature-aware strokes contain a "depth" attribute for depth controlling, it can only model local depth. Thus we provide a profile sketching tool for global depth editing (as seen in Fig. 2). Specifically, the profile contour is treated as the handle to define a Laplacian deformation [72]. Since M_c in the coarse stage is in a low resolution, the Laplacian deformation can be performed in real-time.

4.2.2 Training

The part-separated PIFu is trained in a fully-supervised manner. For each character face mesh M in the dataset, we render its contours as a sketch image input. To prepare the ground truth data for training our part-separated PIFu used in the coarse stage, we smooth faces meshes M, and then segment them into distinct parts (i.e., faces and attachments). The ground-truth SDF values for each part are calculated in the world coordinates. During training, we use the L_1 metric to measure the difference between the predicted SDF values and the ground truth.

4.3 IDGMM: Implicit and Depth Guided Mesh Modeling

In the fine stage, M_c is first rendered into a new contour map S_c. Then users will draw curvature-aware strokes over S_c, and we denote the updated sketch image as S_f. This section discusses the method to map S_f to a model denoted as M_f. It resembles the shape of S_c but contains local geometric details reflected by S_f, as illustrated in Fig. 5.

Recently, many deep-learning-based methods [12], [13], [66], [67], [68] have been proposed to map a sketch image to a 3D model. A straightforward solution is to apply PIFu-based methods [12], [13] and extract the surface mesh with Marching Cubes (MC) [70]. However, MC is time-consuming (5 seconds 256^3 iso-value grid) when extracting high-resolution surfaces and fails to meet the requirements for interactive editing. To this end, we apply the field-guided deformation formula to speed up the extraction of detailed surfaces from implicit fields.

Specifically, our method takes M_c and S_f as input and learns the displacement for each vertex on M_c with the help of both the implicit and depth-map representations. Before conducting deformation, we subdivide the regions [71] where detail strokes are drawn to better afford geometric details. Note that the sketching process is iterative, and the input M_c at the n-th step is the resulting mesh at step (n-1). For simplicity, we still use M_c to represent the input coarse mesh of each step.

4.3.1 Implicit-guided Mesh Updating

Inspired by the work [73], SimpModeling [11] proposed a strategy for mesh deformation under the guidance of implicit fields, but it is inefficient: 1) SimpModeling utilizes an occupancy field and needs to determine the updated vertices by a dense sampling way; 2) to stabilize the deformation, the Laplacian deformation technique [72] is adopted.

In contrast, we update M_c directly with the guidance of the continuous SDF field to keep robustness during deformation, which dramatically reduces the computational cost of the Laplacian deformation (i.e., updating each vertex $\mathbf{v} \in M_c$ via $\mathbf{v}' = \mathbf{v} + g_s(\mathbf{v})\mathbf{n}$, where $\mathbf{n}$ indicates the normal of $\mathbf{v}$ and $g_s(\mathbf{v})$ is the SDF value of $\mathbf{v}$). The above updating mechanism could be performed iteratively for multiple times, but its enhancement was slight. Hence, we only perform one iteration to reduce the computational burden and leave the remaining detail enhancement work to the depth-guided deformation stage. We denote the new mesh after updating as M_c'.

A direct way to learn the SDF function from S_f is by applying PIFu-N on S_f. However, It may lead to a misalignment between the generated SDF field and the coarse mesh M_c, thus challenging the deformation. Therefore, as illustrated in Fig. 5, we render M_c into a depth map D_c, and feed D_c and S_f together into a Pix2Pix module to infer a normal map N for conducting PIFu-N.

4.3.2 Depth-guided Mesh Refinement

Although normal-assisted PIFu can model details better than other existing methods, generating details as reflected in the normal map is still practically challenging. Our experiments found that the learned depth maps contain richer geometric details than the learned implicit fields. Thus we propose a depth-guided deformation method to enhance M_c' further. Specifically, as illustrated in Fig. 5, we first render M_c' into a depth map D_c' and feed it together with N into a new Pix2Pix module for generating a depth map D_f with sharper details than D_c'. Here, we use N instead of S_f since N has already captured the geometric information from S_f and can ease the learning procedure.

Without Depth Alignment. To transfer geometric details from D_f to M_c', a straightforward way is to first convert D_f to a point cloud P and then fit M_c' to P. Specifically, for each vertex $\mathbf{v}$ of M_c', we retrieve K closest points in P and employ the inverse distance weighting algorithm [74] to directly update the position of $\mathbf{v}$.

Flow-based Local Depth Alignment. Although the design of the method discussed above well guarantees global alignment between P and M_c', there is no guarantee for local alignment. Implicit-guided mesh updating is hard to ensure the alignment of local geometry (e.g., nose) between the M_c' and S_f (thus, both N and D_f may also suffer from misalignment). Directly fitting M_c' to D_f tends to cause artifacts due to the local misalignment between them, as shown in Fig. 6. Multiple iterations and extensive smoothings are required to obtain stable results, which is inefficient and may result in blurry geometric details. To address this issue, we propose a flow-based alignment method. More specifically, we train a FlowNet [75] to take D_f and D_c' as input and output a warping field. The warping field is applied to align D_f to M_c' and generate an aligned/warped depth D_f'. Then a high-quality point cloud P can be extracted from D_f'. Thus, P is also locally aligned with M_c'. The good alignment between P and M_c' facilitates the registration of local geometric details from P to M_c'. As a result, the final mesh M_f is close to M_c' but with more local details, instead of being completely aligned with S_f. The alignment of the sketch, depth maps, and normal map used in Fig. 5 is shown in Fig. 7. Although a minor global misalignment exists between M_f and S_f, the resulting mesh is still plausible and convincing, as illustrated in Fig. 9. Thanks to the local alignment, we found that one iteration of the depth-guided mesh refinement is enough to reconstruct vivid details stably (the improvement of multiple iterations is slight), reducing the computational cost.

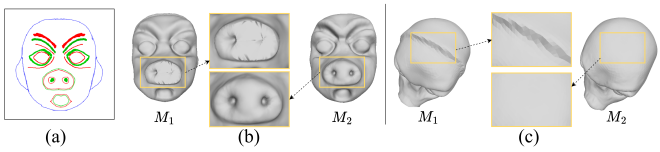


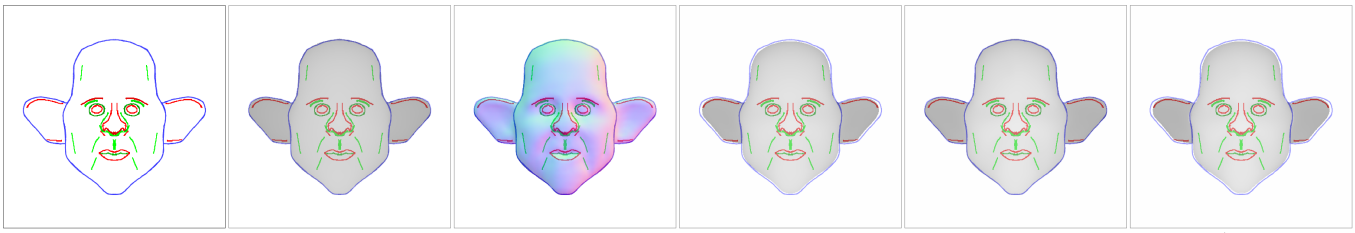
Fig. 6: An illustration of results without and with Flow-based Local Depth Alignment. (a) the input sketch. (b) the front view of the results. (c) the top view of the results. Our flow-based alignment (M_2) resolves the artifacts caused by directly fitting M_c' to D_f without depth alignment (M_1).

4.3.3 Training

IDGMM is backed by four learning-based models: Pix2Pix-1 that maps $S_f \oplus D_c$ ($\oplus$ indicates concatenation) to N, Pix2Pix-2 that maps $D_c' \oplus N$ to D_f, PIFu-N and FlowNet. All the networks are trained separately and in a fully-supervised manner: 1) To train Pix2Pix-1, for each ground-truth mesh M (which contains rich details), we render its ridge and valley lines as input fine strokes, using the tool provided by Suggestive Contours [65]. The stroke types are encoded by the channel of red or green colors, and the depth is encoded with the shades of the color. Specifically, the ridge is encoded in (c, 0, 0) and the valley in (0, c, 0), $c = 255 - |k|$, where k is the curvature of a line segment. Thus the smaller value of c, the visually greater color depth (i.e., visually darker), representing the higher ridge or deeper valley. In our experiments, the trained model can generalize well to strokes of varying widths, though the strokes in the training set are in a constant width. 2) We smooth M to be M_s and use it as M_c to render depth maps as D_c for training Pix2Pix-1 (N is rendered from M). 3) We put M into a 128^3 SDF field (noted as g_M^{128}) and extract the mesh M_l. Then we render M_l into a depth map to approximate D_c' for training Pix2Pix-2. 4) We subdivide M to get M' with dense points and deform M' under the guidance of g_M^{128} to generate a new mesh M_g. We render M' and M_g to depth maps to approximate D_f and D_c'. As M_g and M' are topologically consistent, it is easy to obtain a dense flow as supervision to train FlowNet.

5 Results and Evaluation

In this section, we will evaluate our system from two aspects, namely, **system usability** (Section 5.1) and **algorithm effectiveness** (Section 5.2).



(a) S_f (b) $D_c/M_c + S_f$ (c) $N + S_f$ (d) $D'_c/M'_c + S_f$ (e) $D_f + S_f$ (f) $D'_f/P + S_f$

Fig. 7: An illustration of the alignment of the sketch, depth-maps, and normal-map used in Fig. 5. The overlapping images of S_f and D_c, N, D'_c, D_f, D'_f are shown above. Note that D_c is rendered by M_c, while D'_c is rendered by M'_c. P is extracted from the warped depth (denoted as D'_f here) generated by FlowNet. The resulting mesh M_f of IDGMM is close to M'_c but with more local details, instead of being completely aligned with S_f.

5.1 Evaluation on System Usability

Apparatus. Our user interface was implemented with QT and deployed on a desktop workstation with one Intel i5 @2.7GHz CPU and 8GB of memory. Users can interact with the system with a computer mouse or a pen tablet. The neural-network-based backend model was implemented with Pytorch 1.8.1 and deployed on a server with one Intel i7 @4.2GHz CPU, 16 GB of memory, and one NVIDIA GTX 3090 GPU graphics card. To support the training of our proposed algorithms for modeling high-fidelity 3D heads, we merged the existing datasets of 3DAnimalHead [11] and 3DCaricShop [76], resulting in 4,528 high-quality models in total. Then we split these data into 9:1 for training and testing in our experiments. Please refer to our supplemental materials for the implementation details of the neural networks.

Participants. Our key objective is to create a 3D modeling system that is easy to use for amateur users without 3D modeling experience. To verify this, we invited 16 subjects (P1-P16, aged 18 to 32) to participate in this evaluation session, none of whom had experience in 3D modeling. Six of them (P2, P3, P6, P7, P8, P12) had professional 2D drawing experience, and the remaining had limited drawing experience. Before the modeling session, each participant was given 10 minutes to watch an video showing the basic operations of our system. After the tutorial, each user had 20 minutes to get familiar with our system. All the participants were asked to perform comparison and usability studies.

5.1.1 Comparison Study

We first conducted a comparison study on different modeling systems to demonstrate the superiority of our system. After thoroughly reviewing existing sketch-based character modeling systems, we chose DeepSketch2Face [10] and SimpModeling [11] for comparison since these systems can be easily accessed. For DeepSketch2Face, its released system was used. We asked the authors of SimpModeling to provide their system to us. ZBrush is a powerful commercial software for assisting professional artists in creating arbitrary 3D models. We also added ZBrush to our informal comparison on face modeling. For a fair comparison, all 16 subjects were also given 10 minutes to learn through a tutorial and 20 minutes to get familiar with each of the other systems before the formal user study. In the formal session, each user was given a shading image of a 3D model as a reference. She/he was requested to create 3D models referring to the given image using the four compared systems (i.e., DeepSketch2Face, SimpModeling, *SketchMetaFace*, and ZBrush) in random order. Note that all the tools provided by SimpModeling and ZBrush are 3D interactive operations, while most operations of DeepSketch2Face and *SketchMetaFace* focus on the 2D canvas.

Fig. 8 shows the reference images, the created models with the four systems, and the corresponding modeling time. Compared to DeepSketch2Face and SimpModeling, our system supported users to create more appealing shapes and craft more vivid surface details. The geometric shape and surface details created by our system are closer to the reference models. Compared to ZBrush, our system took less time for users to create visually reasonable 3D models. To complete each model, each user took around 2-5 minutes to use DeepSketch2Face, around 7-15 minutes with SimpModeling, around 5-9 minutes with our system, and around 10-18 minutes with ZBrush. Most participants complained that DeepSketch2Face was hard to use as it could only output human faces (mainly because of the limited parametric space of the human face). They mentioned that SimpModeling could create coarse shapes and some minor details, but it was challenging to learn and use. We observed that most subjects got stuck in the coarse shape modeling process with SimpModeling and ZBrush. Some even gave up adjusting coarse shapes and directly turned to sculpting surface details. “The 3D operations are difficult to use, and I need to speed time adjusting the shape. I am disappointed with SimpModleing and ZBrush”, as commented by P8. “3D interactions are extremely unfriendly to me. I need to switch perspectives frequently. These frequent switching operations make me irritable” (P11). Most subjects enjoyed the modeling process defined by *SketchMetaFace*. Some participants reported that *SketchMetaFace* was user-friendly and allowed for creating vivid avatar heads easily. They also pointed out that our system saved much time and labor in generating 3D heads. “*SketchMetaFace* is much better than SimModeling. The coarse shape modeling provided by *SketchMetaFace* is easier and can save me a lot of time. The curvature-aware strokes allow me to craft details freely in an intuitive way” (P6). “It is very cool to create 3D models by drawing sketches. I am looking forward to using *SketchMetaFace* in the future.” P1 suggested that the 3D sculpting tools (e.g., smooth and crease) provided by ZBrush could be added to the fine stage, supporting users in further fine-tuning geometric details.

5.1.2 Usability Study

In this study, each participant was asked to freely create at least one model without restrictions on result diversity, result quality, or time duration. Fig. 9 shows a gallery of models created by these participants, which reflect the expressiveness of our system. It can be seen from this figure that our system supports amateurs in geometrical modeling to create character faces with diversified

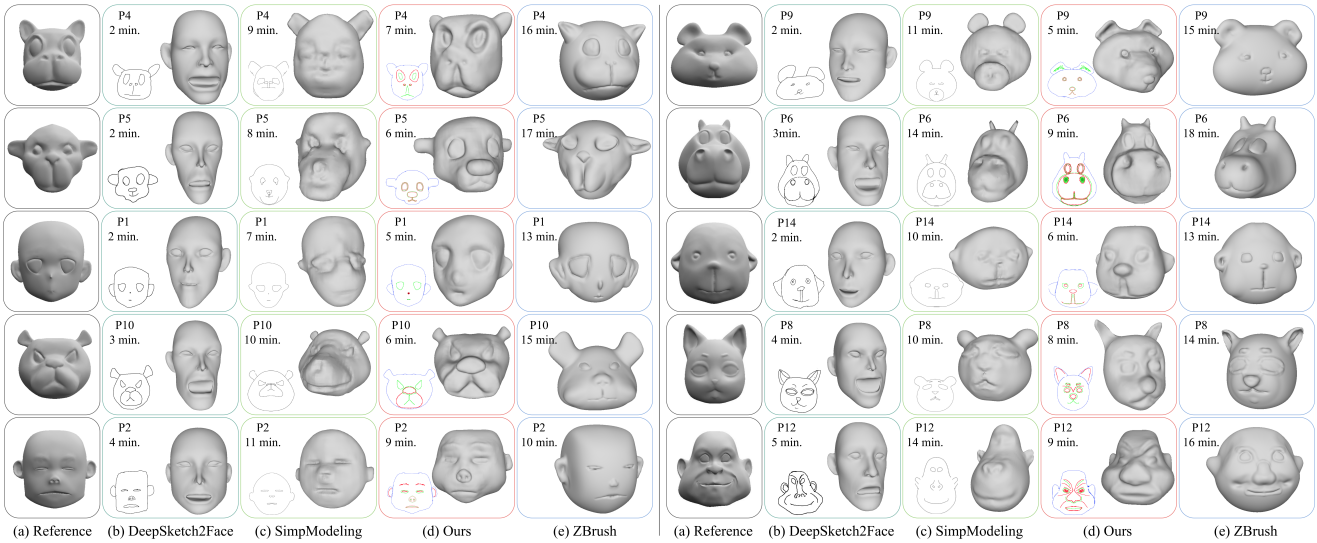


Fig. 8: Comparison between our system against the state of the arts. The results in each row were created by the same user given a reference in (a). For each system, we show the sketch, resulting model, drawing time, and the corresponding participant.

shapes and rich geometric details. All of the participants felt that our system was powerful in creating diversified avatar heads, and they were deeply impressed by the simplicity, intuitiveness, and controllability of our system. It is worth mentioning that two of the participants said they enjoyed the process very much and expressed their desire to learn 3D modeling.

Most of the participants liked the intuitive stroke suggestion tool, which was quite helpful for them in figuring out the meaning of curvature-aware strokes. We observed that the participants with great drawing skills (i.e., P2, P3, P6, P7, P8, and P12) quickly became used to working with the curvature-aware strokes thanks to the suggestion tool. Once grasping curvature-aware strokes, they preferred to paint each part of the model from scratch and customize desired details by themselves instead of searching for a specific structure using the stroke suggestion module. P6 commented "The stroke suggestion tool is a very nice and useful function for assisting me in understanding the usage of curvature-aware strokes." We received similar positive comments from P7 and P12: "With the help of the stroke suggestion function, I can easily understand how to depict geometric structures using curvature-aware strokes" (P7); "The curvature-aware strokes are useful and powerful for carving models' details, like wrinkles" (P12). Other participants tended to use the stroke suggestion function throughout the whole modeling process due to their limited drawing skills. "The suggestion module is easy and intuitive to use. I do not need to spend much time thinking about how to paint a correct sketch. It avoids frequent modifying operations" (P1). "The suggestion module is convenient and friendly for me. It reduces a lot of manual operations and allows me to create diversified results in a very easy way" (P5). "I can make funny and realistic results by simply searching and integrating different parts in minutes (two eyes, a nose, and a mouth)" (P10).

The participants also provided some constructive comments. For example, P4 said, "It would be better to allow me to search for a suitable head contour in the coarse modeling stage, just like searching for a nose or a mouth in the fine stage." One potential solution is collecting a coarse shape database and applying the retrieval mechanism in the coarse-shape modeling stage. "Although the profile depth editing tool allows me to adjust models in the side view, the system still fails to create an elephant's nose. I do not know how to create an elephant's nose using the tools provided by *SketchMetaFace*." said P2. Enlarging our datasets and adopting multi-view drawing in the coarse stage would be a possible solution for this problem.

5.1.3 Questionnaire Analysis

At the end of the comparison study, each participant was required to complete a System Usability Scale (SUS) questionnaire and a NASA Task Load Index (NASA-TLX) questionnaire to evaluate the usability and workload of our system. We found that the overall SUS score of our system was 79, out of a scale of 100 (DeepSketch2Face: 64, SimpModeling: 38, ZBrush: 41), indicating the good usability of our system [77]. In Fig. 10(a), we show the mean scores for all the individual SUS questions. For the questions with the odd numbers, the higher the SUS scores, the better; for the rest of the questions, the lower the SUS scores, the better. The scores of Q1 and Q9 suggest that the participants appreciated our system and were satisfied with the models created by our system. From Q2-4, Q7-8, and Q10, we can conclude that our system supported amateur users creating desired 3D head models easily and intuitively, indicating the good user efficiency and usability of our system. The scores of Q5-6 show that the participants also recognized our system's well-designed modeling pipeline and tools. Although the high scores of Q3 and Q7 indicate that DeepSketch2Face is easy to use, the participants were disappointed with its results, leading to low scores for Q1 and Q9. The high scores of Q2, Q4, Q6, Q8, and Q10 and the low scores of Q3, Q7, and Q9 all suggest that SimpModleing and ZBrush are unfriendly for these amateur uses. Grasping these two systems is extremely hard for them.

Fig. 10(b) illustrates the average score for each question in the NASA-FLX questionnaire. The results of our systems are also positive. Compared to SimpModeling and ZBrush, our system's mental demand, physical demand, temporal demand, effort, and frustration are at an extremely low level. It implies that our system does not require users to pay a lot of concentration and effort

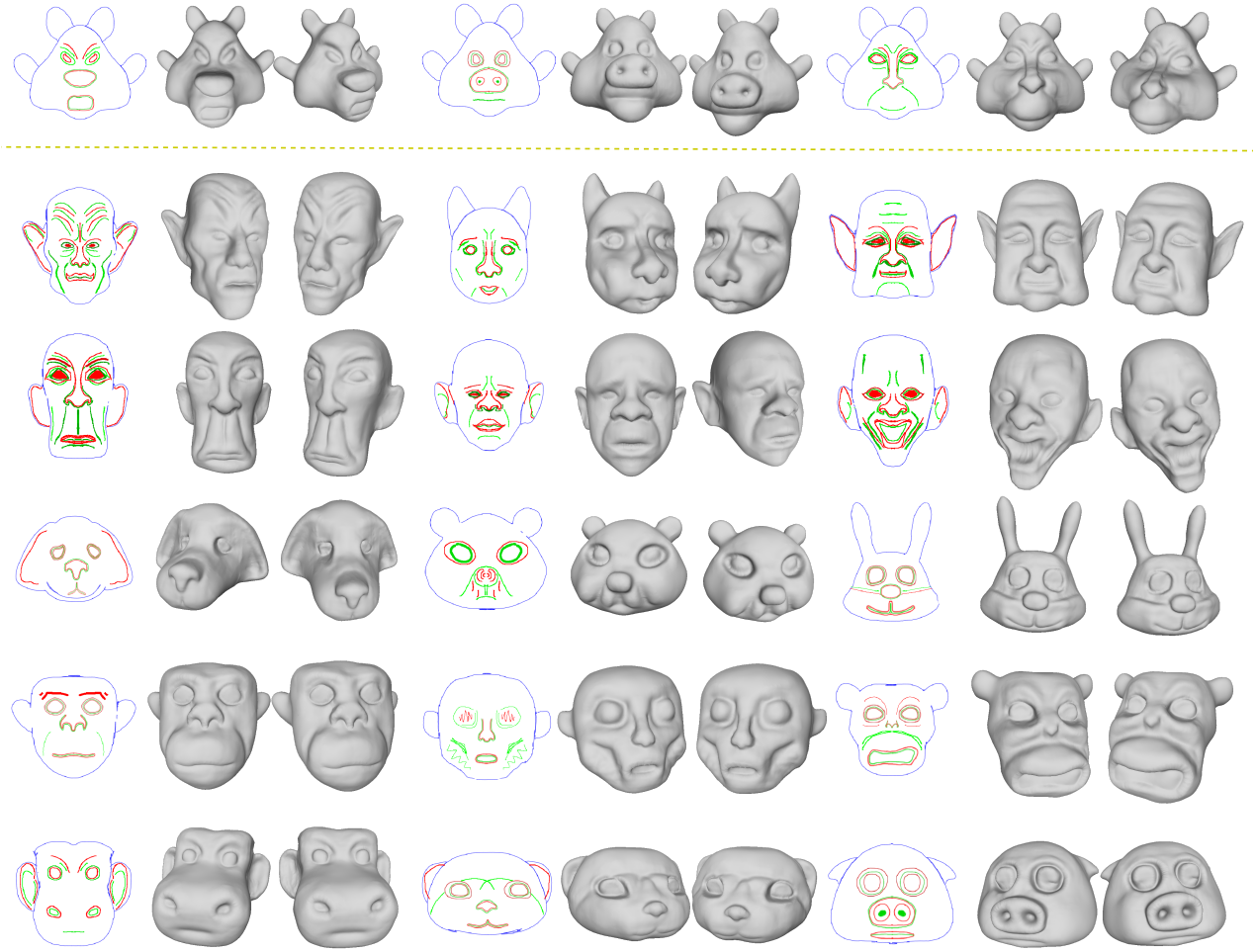


Fig. 9: The gallery of our results. All models are created by amateur users who are trained to use our system with a tutorial. Thanks to the easy-to-use two-stage modeling design and the stroke suggestion component, the users can complete each model design in 5-9 minutes. The three results in the first row were created using the same coarse mesh but applying different surface details.

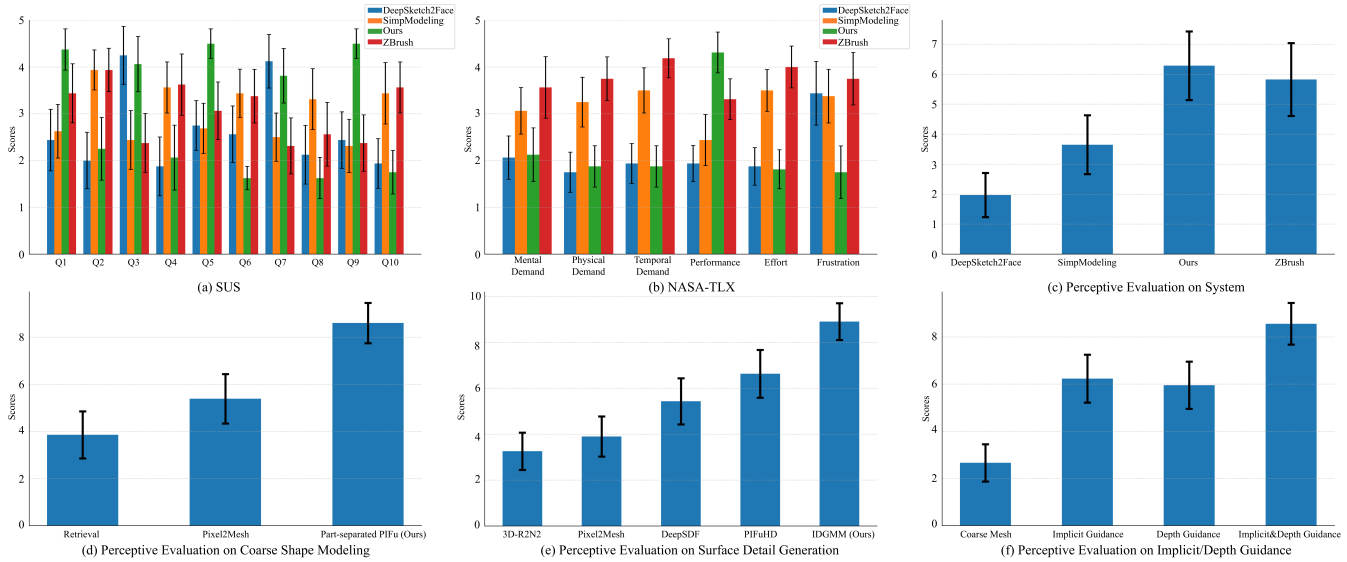


Fig. 10: (a) Mean scores of SUS in a 5-point scale. (b) Mean scores of NASA-TLX in a 5-point scale. (c) Perceptive evaluation on results of the compared systems. (d) Perceptive evaluation on coarse shape modeling. (e) Perceptive evaluation on surface detail generation. (f) Perceptive evaluation on implicit/depth guidance. Each error bar represents the standard deviation of the corresponding mean.

when using it. The higher performance score of our system reflects that the participants were also more satisfied with their modeling results with our system. The lower performance score and the higher frustration score of SimpModeling and ZBrush than those of our system suggest that it was hard for the participants to create desired results using 3D operations. The lower performance score of DeepSketch2Face demonstrates that the participants were unsatisfied with the results generated by its algorithm, which also leads to a high frustration level.

We conducted a subjective user study to evaluate the faithfulness (i.e., the degree of fitness to reference images/models) of synthesized results. We randomly chose a set of results from the comparison study, containing 15 reference models and the corresponding results created by the participants using the four above systems. We invited 50 subjects to participate in this subjective evaluation through an online questionnaire. Most subjects had no 3D modeling experience, and none had participated in the previous studies. We showed the participants five images for each case (15 cases in total), including the input sketch and the four modeling results by the compared systems, placed side by side in random order. Each participant was asked to score each result based on the faithfulness to the reference model (1 denoting the lowest fitness and 10 for the highest fitness). Fig. 10(c) shows the mean score of each system for this study. This figure shows that the 3D models created by amateurs with our system in the comparison study received relatively higher marks than the counterpart systems, implying that our system could assist novice users in creating desired 3D heads. Statistical analysis also showed that the scores significantly differed across the compared systems. Specifically, we ran Shapiro-Wilk normality tests on the collected data and found non-normality distributions ($p < 0.001$). We thus conducted Kruskal-Wallis tests on the faithfulness scores and found significant effects. Paired tests between our system and each of the compared ones confirmed that our system (mean: 6.28) could effectively support amateurs in creating significantly more faithful results to the reference models than the other systems, i.e., DeepSketch2Face (mean: 1.96, $p < 0.001$), SimpModeling (mean: 3.64, $p < 0.001$) and ZBrush (mean: 5.82, $p = 0.008$). More details can be found in our supplementary material.

5.2 Evaluation on Algorithm Effectiveness

Comparison on Part-separated Mesh Inference. There are some alternative methods [21], [57], [78] for inferring part-separated meshes from an input sketch. To verify the generalization ability of part-separated PIFu, we choose two representative alternative methods for comparison. One is a retrieval-based method [57], denoted as Retrieval and the other one is a deformation-based method [21], denoted as Pixel2Mesh. The qualitative comparisons are presented in Fig. 11, where we can see that our results align much better with the input sketches.

Comparisons on Sketch2Mesh. The core problem of our system is to learn the mapping from S_f to a detailed mesh. To evaluate the superiority of IDGMM, we selected four existing representative methods for comparison: 3D-R2N2 [79], Pixel2Mesh [21], DeepSDF [67] and PIFuHD [13] (the method used by SimpModeling). All these methods took S_f and D_c as input for fairness. Fig. 12 and Tab. 1 show the results of this comparison. Both qualitative and quantitative results demonstrate the superiority of our method. Although PIFuHD performs not badly on quantitative measurements, the qualitative results show that our proposed

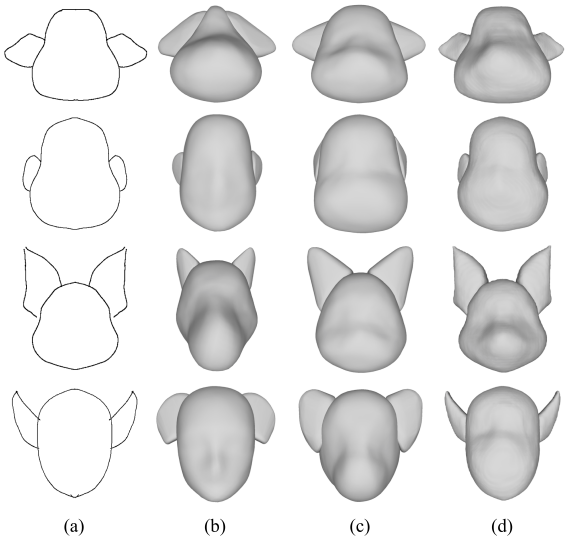


Fig. 11: Qualitative comparisons on part-separated mesh inference from an input sketch (a). (b) The results of retrieval. (c) The results of Pixel2Mesh. (d) The results of our part-separated PIFu.

algorithm (IDGMM) performs much better than PIFuHD on geometric details synthesis. Meanwhile, PIFuHD requires a time-consuming mesh extraction process from an implicit field (around 5.0s for one result generation). SimpModeling slightly reduces PIFuHD's time consumption by sampling points along the normal directions and applying local Laplacian deformation (1.0s for one result generation). Our IDGMM combines the advantages of mesh, continuous SDF, and depth map representations, making it very powerful not only in generating detailed 3D geometry but also in inference efficiency (around 0.5s for one result generation).

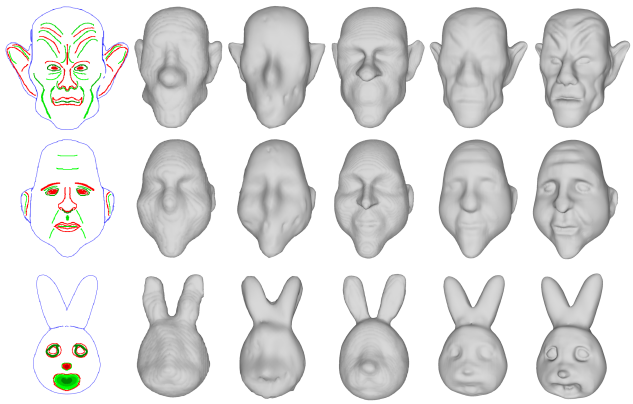


Fig. 12: Qualitative comparisons of our IDGMM with four existing methods for Sketch2Mesh inference.

Ablation Study on Implicit/Depth Guidance. There are two key components in our proposed IDGMM: implicit-guided mesh updating and depth-guided mesh refinement. To verify the indispensability of these two modules, we compared IDGMM with two alternative settings: 1) without implicit guidance - we use D_c and N as input to generate D_f and corresponding warped P, which is then used to guide the deformation from M_c. 2) without depth guidance, i.e., M_c' shown in Fig. 5. Qualitative

TABLE 1: Quantitative comparison with our proposed IDGMM with four existing methods for Sketch2Mesh inference. We adopt IoU, Chamfer-L_2, and normal consistency to evaluate the results.

	IoU ↑	Chamfer-L_2 ($\times 10^2$) ↓	Normal-Consis. ↑
3D-R2N2	0.858	0.149	0.929
Pixel2Mesh	0.882	0.123	0.937
DeepSDF	0.894	0.117	0.949
PIFuHD	0.911	0.103	0.955
Ours	**0.915**	**0.099**	**0.956**

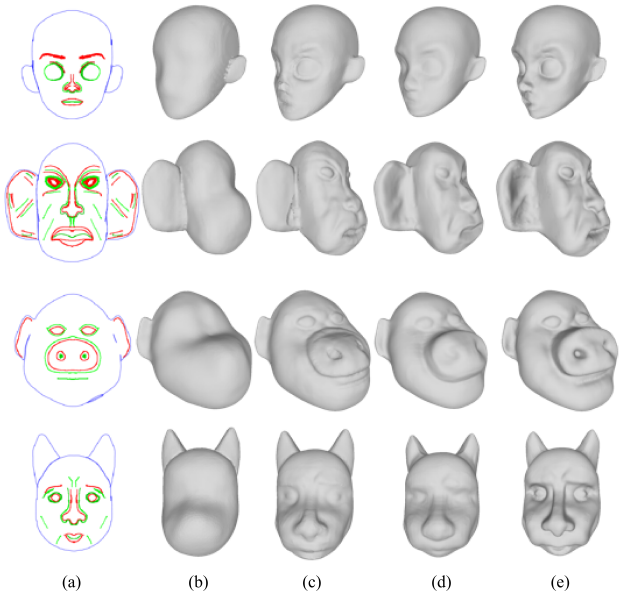


Fig. 13: Ablation study on implicit/depth guidance. From left to right: (a) input sketch; (b) coarse mesh (i.e., M_c in Fig. 5); (c) resulting mesh with only depth guidance (without implicit guidance); (d) resulting mesh with only implicit guidance (without depth guidance, i.e., M_c' in Fig. 5); (e) resulting mesh with both guidance (i.e., M_f in Fig. 5).

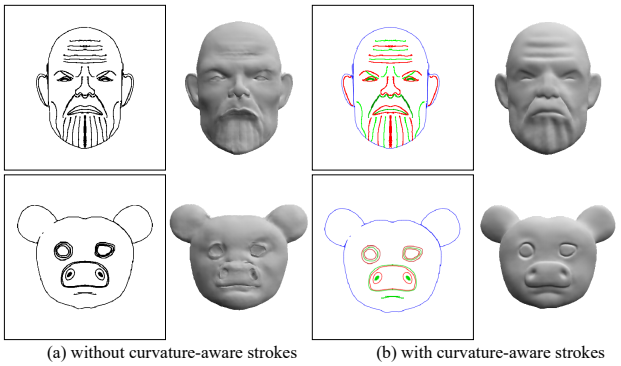


Fig. 14: Ablation study on without/with curvature-aware strokes. Using curvature-aware strokes significantly helps enhance the quality of the generated geometric details.

results are shown in Fig. 13. The resulting meshes with both implicit and depth guidance outperform the other two options on surface detail generation, implying the necessity of the implicit-guided and depth-guided modules.

Ablation Study on Curvature-aware Strokes. The common option to represent sketches is using strokes without any curvature-aware attributes (e.g., DeepSketch2Face and SimpModeling), which is hard to depict complex surface details, as seen in the left part of Fig. 14. The right part of Fig. 14 shows the great capability of curvature-aware strokes in representing rich geometric details.

Perceptive Evaluation Study. To further evaluate the effectiveness and superiority of our proposed algorithm (part-separated PIFu and IDGMM), we conducted another perceptive evaluation study. We selected 10 samples from the experiments of Comparison on Part-separated Mesh Inference (like Fig. 11), Comparisons on Sketch2Mesh (like Fig. 12), and Ablation Study on Implicit/Depth Guidance (like Fig. 13) respectively, resulting in three questionnaires. Each case in the questionnaires showed the input sketch and the results generated by different algorithms, placed side by side in random order. The 50 subjects mentioned above were also asked to evaluate each synthesized model's faithfulness (i.e., the degree of fitness to input sketches) on a ten-point Likert scale (1 = lowest fitness to 10 = highest fitness). Fig. 10(d) shows that the results generated by part-separated PIFu fit the input sketches better than Retrieval and Pixel2Mesh. Fig. 10(e) suggests that IDGMM could synthesize richer, more vivid, and more realistic geometric details than the other methods. Fig. 10(f) indicates the necessity and superiority of combining implicit and depth guidance for detailed geometry generation. For statistical analysis, we first performed Shapiro-Wilk normality tests, respectively, for the three collected data and found that all of them followed non-normality distributions ($p < 0.001$). Therefore, we conducted a Kruskal-Wallis test on the faithfulness scores for each perceptive evaluation, and the results also showed significance across different comparisons. For the evaluation of coarse shape modeling, paired tests showed that our method (mean: 8.60) performs significantly better on diverse shape generation than both Retrieval (mean: 3.85, $p < 0.001$) and Pixel2Mesh (mean: 5.38, $p < 0.001$). For the evaluation of surface detail generation, the results indicated that IDGMM (mean: 8.90) led to significantly more faithful results than the other methods, i.e., 3D-R2N2 (mean: 3.25, $p < 0.001$), Pixel2Mesh (mean: 3.89, $p < 0.001$), DeepSDF (mean: 5.43, $p < 0.001$), and PIFuHD (mean: 6.63, $p < 0.001$). For the evaluation of implicit/depth guidance, the tests suggested that depth&implicit guidance (mean: 8.55) significantly performs better on geometric detail synthesis than the alternative options, i.e., only implicit guidance (mean: 6.23, $p < 0.001$) and only depth guidance (mean: 5.95, $p < 0.001$). It is worth mentioning that the difference between depth and implicit guidance was not distinct ($p = 0.169$). This is consistent with our expectation, since both only using depth refinement and only using implicit refinement can synthesize minor details. But they fail to depict high-quality geometric details, further confirming the significant positive effect of incorporating implicit and depth refinement. All these statistical results confirmed that all our proposed algorithms significantly outperform the corresponding alternative options. More details about evaluation are provided in our supplementary material.

6 Conclusion

In this paper, we presented an easy-to-use sketching system for amateur users to create and high-fidelity 3D face models. Both the user interface and the algorithm are carefully designed. Firstly, curvature-aware strokes are utilized to assist users in easily carving geometric details. Secondly, a coarse-to-fine interface is designed. In the coarse stage, users only need to model face contours and

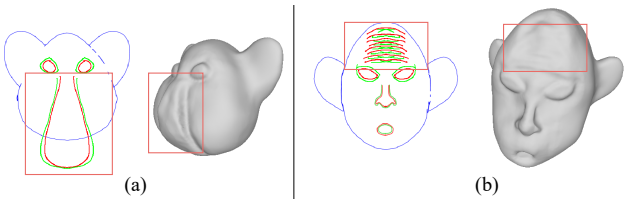


Fig. 15: Limitations of our system. Our system also suffers from limitations when a) modeling facial components or details with complex depth changes; b) strokes are placed too densely.

the 3D layout of ears. Then, in the fine stage, all interactions are operated on a 2D canvas for detail drawing. Thirdly, to support the accuracy and usability of the user interface, a novel method, named Implicit and Depth guided Mesh Modeling (IDGMM), is proposed. It combines the advantages of the implicit (SDF), mesh, and depth representations, and reaches a good balance between output quality and inference efficiency. Both evaluations of the system and algorithm demonstrate that our system is of better usability than existing systems and the proposed IDGMM also outperforms existing methods.

Although our system is able to create 3D models with diversified shapes and rich details, it also has some limitations (Fig. 15): a) As we only focus on frontal-view sketching for detail carving, some organs with complex depth changing are hard to model, such as the nose of an elephant; b) When the strokes are densely placed, it cannot produce reasonable geometric details as a large number of vertices are required in this scenario, which our current system does not support. In the future, we will enlarge our dataset to support users in modeling shapes with other categories, such as cartoon character bodies and human garments. We will also try to take multi-view sketches as input to further support the creation of complex models, such as elephants. Meanwhile, we will explore the possibilities to carve high-resolution models efficiently and support richer detail crafting effectively.

Acknowledgements. The work was supported in part by NSFC-62172348, the Basic Research Project No. HZQB-KCZYZ-2021067 of Hetao Shenzhen-HK S&T Cooperation Zone, the National Key R&D Program of China with grant No. 2018YFB1800800, the Shenzhen Outstanding Talents Training Fund 202002, the Guangdong Research Projects No. 2017ZT07X152 and No. 2019CX01X104, the Guangdong Provincial Key Laboratory of Future Networks of Intelligence (Grant No. 2022B1212010001), the Shenzhen Key Laboratory of Big Data and Artificial Intelligence (Grant No. ZDSYS201707251409055), and the Key Area R&D Program of Guangdong Province with grant No. 2018B030338001. It was also partially supported by Outstanding Yound Fund of Guangdong Province with No. 2023B1515020055, Shenzhen General Project with No. JCYJ20220530143604010, Hong Kong Research Grants Council under General Research Funds (HKU17206218), grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (No. CityU 11212119) and the Centre for Applied Computing and Interactive Media (ACIM) of School of Creative Media, CityU.

REFERENCES

[1] T. Igarashi, S. MATSUOKA, and H. TANAKA, "Teddy: A sketching interface for 3d freeform design," in *Computer graphics proceedings, annual conference series*. Association for Computing Machinery SIGGRAPH, 1999, pp. 409–416.

[2] A. Nealen, T. Igarashi, O. Sorkine, and M. Alexa, "Fibermesh: designing freeform surfaces with 3d curves," in *ACM SIGGRAPH 2007 papers*, 2007, pp. 41–es.

[3] P. Borosan, M. Jin, D. DeCarlo, Y. Gingold, and A. Nealen, "RigMesh: Automatic rigging for part-based shape modeling and deformation," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 6, pp. 198:1–198:9, Nov. 2012. [Online]. Available: http://doi.acm.org/10.1145/2366145.2366217

[4] D. Sỳkora, L. Kavan, M. Čadík, O. Jamriška, A. Jacobson, B. Whited, M. Simmons, and O. Sorkine-Hornung, "Ink-and-ray: Bas-relief meshes for adding global illumination effects to hand-drawn characters," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 2, pp. 1–15, 2014.

[5] H. Pan, Y. Liu, A. Sheffer, N. Vining, C.-J. Li, and W. Wang, "Flow aligned surfacing of curve networks," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 4, pp. 1–10, 2015.

[6] C. Li, H. Pan, Y. Liu, X. Tong, A. Sheffer, and W. Wang, "Bendsketch: modeling freeform surfaces through 2d sketching," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 1–14, 2017.

[7] Y. Zhong, Y. Gryaditskaya, H. Zhang, and Y.-Z. Song, "Deep sketch-based modeling: Tips and tricks," in *2020 International Conference on 3D Vision (3DV)*. IEEE, 2020, pp. 543–552.

[8] ——, "A study of deep single sketch-based modeling: View/style invariance, sparsity and latent space disentanglement," *Computers & Graphics*, vol. 106, pp. 237–247, 2022.

[9] P. Xu, T. M. Hospedales, Q. Yin, Y.-Z. Song, T. Xiang, and L. Wang, "Deep learning for free-hand sketch: A survey," *IEEE transactions on pattern analysis and machine intelligence*, vol. 45, no. 1, pp. 285–312, 2022.

[10] X. Han, C. Gao, and Y. Yu, "Deepsketch2face: a deep learning based sketching system for 3d face and caricature modeling," *ACM Transactions on graphics (TOG)*, vol. 36, no. 4, pp. 1–12, 2017.

[11] Z. Luo, J. Zhou, H. Zhu, D. Du, X. Han, and H. Fu, "Simpmodeling: Sketching implicit field to guide mesh modeling for 3d animalmorphic head design," in *The 34th Annual ACM Symposium on User Interface Software and Technology*, 2021, pp. 854–863.

[12] 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 International Conference on Computer Vision*, 2019, pp. 2304–2314.

[13] S. Saito, T. Simon, J. Saragih, and H. Joo, "Pifuhd: Multi-level pixel-aligned implicit function for high-resolution 3d human digitization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 84–93.

[14] C. Li, H. Pan, Y. Liu, X. Tong, A. Sheffer, and W. Wang, "Robust flow-guided neural prediction for sketch-based freeform surface modeling," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 6, pp. 1–12, 2018.

[15] 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.

[16] D. Du, X. Han, H. Fu, F. Wu, Y. Yu, S. Cui, and L. Liu, "Sanihead: Sketching animal-like 3d character heads using a view-surface collaborative mesh generative network," *IEEE Transactions on Visualization and Computer Graphics*, 2020.

[17] E. Iarussi, D. Bommes, and A. Bousseau, "Bendfields: Regularized curvature fields from rough concept sketches," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 3, pp. 1–16, 2015.

[18] 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.

[19] 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.

[20] J. Wang, J. Lin, Q. Yu, R. Liu, Y. Chen, and S. X. Yu, "3d shape reconstruction from free-hand sketches," in *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part VIII*. Springer, 2023, pp. 184–202.

[21] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, "Pixel2mesh: Generating 3d mesh models from single rgb images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 52–67.

[22] B. Guillard, E. Remelli, P. Yvernay, and P. Fua, "Sketch2mesh: Reconstructing and editing 3d shapes from sketches," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 13 023–13 032.

[23] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

[24] S.-Y. Chen, F.-L. Liu, Y.-K. Lai, P. L. Rosin, C. Li, H. Fu, and L. Gao, "Deepfaceediting: Deep face generation and editing with disentangled geometry and appearance control," *ACM Trans. Graph.*, vol. 40, no. 4, jul 2021. [Online]. Available: https://doi.org/10.1145/3450626.3459760

[25] 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.

[26] Y. Xiao, H. Zhu, H. Yang, Z. Diao, X. Lu, and X. Cao, "Detailed facial geometry recovery from multi-view images by learning an implicit function," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[27] Z. Bai, Z. Cui, J. A. Rahim, X. Liu, and P. Tan, "Deep facial non-rigid multi-view stereo," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[28] M. B R, A. Tewari, H.-P. Seidel, M. Elgharib, and C. Theobalt, "Learning complete 3d morphable face models from images and videos," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[29] P. Garrido, M. Zollhöfer, D. Casas, L. Valgaerts, K. Varanasi, P. Perez, and C. Theobalt, "Reconstruction of personalized 3d face rigs from monocular video," *ACM Trans. Graph. (Presented at SIGGRAPH 2016)*, vol. 35, no. 3, pp. 28:1–28:15, 2016.

[30] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou, "Facewarehouse: A 3d facial expression database for visual computing," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 3, pp. 413–425, 2013.

[31] Y. Deng, J. Yang, S. Xu, D. Chen, Y. Jia, and X. Tong, "Accurate 3d face reconstruction with weakly-supervised learning: From single image to image set," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[32] L. Tran and X. Liu, "Nonlinear 3d face morphable model," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7346–7355.

[33] E. Richardson, M. Sela, R. Or-El, and R. Kimmel, "Learning detailed face reconstruction from a single image," in *proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1259–1268.

[34] A. Tuan Tran, T. Hassner, I. Masi, E. Paz, Y. Nirkin, and G. Medioni, "Extreme 3d face reconstruction: Seeing through occlusions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3935–3944.

[35] J. Liu, Y. Chen, C. Miao, J. Xie, C. X. Ling, X. Gao, and W. Gao, "Semi-supervised learning in reconstructed manifold space for 3d caricature generation," in *Computer Graphics Forum*, vol. 28, no. 8. Wiley Online Library, 2009, pp. 2104–2116.

[36] J. Zhang, H. Cai, Y. Guo, and Z. Peng, "Landmark detection and 3d face reconstruction for caricature using a nonlinear parametric model," *arXiv preprint arXiv:2004.09190*, 2020.

[37] Q. Wu, J. Zhang, Y.-K. Lai, J. Zheng, and J. Cai, "Alive caricature from 2d to 3d," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7336–7345.

[38] S.-H. Zhang, Y.-C. Guo, and Q.-W. Gu, "Sketch2model: View-aware 3d modeling from single free-hand sketches," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6012–6021.

[39] P. N. Chowdhury, T. Wang, D. Ceylan, Y.-Z. Song, and Y. Gryaditskaya, "Garment ideation: Iterative view-aware sketch-based garment modeling," in *10th International Conference on 3D Vision (3DV 2022)*.

[40] C. Ding and L. Liu, "A survey of sketch based modeling systems," *Frontiers of Computer Science*, vol. 10, no. 6, pp. 985–999, 2016.

[41] O. A. Karpenko and J. F. Hughes, "Smoothsketch: 3d free-form shapes from complex sketches," in *ACM SIGGRAPH 2006 Papers*, 2006, pp. 589–598.

[42] R. Schmidt, B. Wyvill, M. C. Sousa, and J. A. Jorge, "Shapeshop: Sketch-based solid modeling with blobtrees," in *ACM SIGGRAPH 2007 courses*, 2007, pp. 43–es.

[43] A. Bernhardt, A. Pihuit, M.-P. Cani, and L. Barthe, "Matisse: Painting 2d regions for modeling free-form shapes," in *SBM'08-Eurographics Workshop on Sketch-Based Interfaces and Modeling*. Eurographics Association, 2008, pp. 57–64.

[44] P. Joshi and N. A. Carr, "Repoussé: Automatic inflation of 2d artwork." in *SBM*, 2008, pp. 49–55.

[45] Y. Gingold, T. Igarashi, and D. Zorin, "Structured annotations for 2d-to-3d modeling," in *ACM SIGGRAPH Asia 2009 papers*, 2009, pp. 1–9.

[46] L. Olsen, F. Samavati, and J. Jorge, "Naturasketch: Modeling from images and natural sketches," *IEEE Computer Graphics and Applications*, vol. 31, no. 6, pp. 24–34, 2011.

[47] 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.

[48] R. Schmidt, A. Khan, K. Singh, and G. Kurtenbach, "Analytic drawing of 3d scaffolds," in *ACM SIGGRAPH Asia 2009 papers*, 2009, pp. 1–10.

[49] C. Shao, A. Bousseau, A. Sheffer, and K. Singh, "Crossshade: shading concept sketches using cross-section curves," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 4, pp. 1–11, 2012.

[50] B. Xu, W. Chang, A. Sheffer, A. Bousseau, J. McCrae, and K. Singh, "True2form: 3d curve networks from 2d sketches via selective regularization," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, pp. 1–13, 2014.

[51] M. Eitz, R. Richter, T. Boubekeur, K. Hildebrand, and M. Alexa, "Sketch-based shape retrieval." *ACM Trans. Graph.*, vol. 31, no. 4, pp. 31–1, 2012.

[52] B. Li, Y. Lu, F. Duan, S. Dong, Y. Fan, L. Qian, H. Laga, H. Li, Y. Li, P. Liu, M. Ovsjanikov, H. Tabia, Y. Ye, H. Yin, and Z. Xue, "3D Sketch-Based 3D Shape Retrieval," in *Eurographics Workshop on 3D Object Retrieval*, A. Ferreira, A. Giachetti, and D. Giorgi, Eds. The Eurographics Association, 2016.

[53] A. Qi, Y. Gryaditskaya, J. Song, Y. Yang, Y. Qi, T. M. Hospedales, T. Xiang, and Y.-Z. Song, "Toward fine-grained sketch-based 3d shape retrieval," *IEEE transactions on image processing*, vol. 30, pp. 8595–8606, 2021.

[54] L. Luo, Y. Gryaditskaya, T. Xiang, and Y.-Z. Song, "Structure-aware 3d vr sketch to 3d shape retrieval," *arXiv preprint arXiv:2209.09043*, 2022.

[55] L. Fan, R. Wang, L. Xu, J. Deng, and L. Liu, "Modeling by drawing with shadow guidance," in *Computer Graphics Forum*, vol. 32, no. 7. Wiley Online Library, 2013, pp. 157–166.

[56] X. Xie, K. Xu, N. J. Mitra, D. Cohen-Or, W. Gong, Q. Su, and B. Chen, "Sketch-to-design: Context-based part assembly," in *Computer Graphics Forum*, vol. 32, no. 8. Wiley Online Library, 2013, pp. 233–245.

[57] F. Wang, L. Kang, and Y. Li, "Sketch-based 3d shape retrieval using convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1875–1883.

[58] Y. Zhong, Y. Qi, Y. Gryaditskaya, H. Zhang, and Y.-Z. Song, "Towards practical sketch-based 3d shape generation: The role of professional sketches," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 9, pp. 3518–3528, 2020.

[59] Z. Cheng, M. Chai, J. Ren, H.-Y. Lee, K. Olszewski, Z. Huang, S. Maji, and S. Tulyakov, "Cross-modal 3d shape generation and manipulation," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part III*. Springer, 2022, pp. 303–321.

[60] D. Kong, Q. Wang, and Y. Qi, "A diffusion-refinement model for sketch-to-point modeling," in *Proceedings of the Asian Conference on Computer Vision*, 2022, pp. 1522–1538.

[61] W. Su, D. Du, X. Yang, S. Zhou, and H. Fu, "Interactive sketch-based normal map generation with deep neural networks," *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, vol. 1, no. 1, pp. 1–17, 2018.

[62] H. Huang, E. Kalogerakis, E. Yumer, and R. Mech, "Shape synthesis from sketches via procedural models and convolutional networks," *IEEE transactions on visualization and computer graphics*, vol. 23, no. 8, pp. 2003–2013, 2016.

[63] D. Du, H. Zhu, Y. Nie, X. Han, S. Cui, Y. Yu, and L. Liu, "Learning part generation and assembly for sketching man-made objects," in *Computer Graphics Forum*. Wiley Online Library, 2020.

[64] G. Nishida, I. Garcia-Dorado, D. G. Aliaga, B. Benes, and A. Bousseau, "Interactive sketching of urban procedural models," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, pp. 1–11, 2016.

[65] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, and A. Santella, "Suggestive contours for conveying shape," in *ACM SIGGRAPH 2003 Papers*, 2003, pp. 848–855.

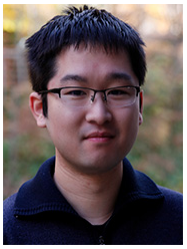
[66] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3d reconstruction in function space," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4460–4470.

[67] 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 Conference on Computer Vision and Pattern Recognition*, 2019, pp. 165–174.

[68] Z. Chen and H. Zhang, "Learning implicit fields for generative shape modeling," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5939–5948.
[69] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *European conference on computer vision*. Springer, 2016, pp. 483–499.
[70] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," *ACM siggraph computer graphics*, vol. 21, no. 4, pp. 163–169, 1987.
[71] M. Botsch and L. Kobbelt, "A remeshing approach to multiresolution modeling," in *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, 2004, pp. 185–192.
[72] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel, "Laplacian surface editing," in *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, 2004, pp. 175–184.
[73] A. Sharf, T. Lewiner, A. Shamir, L. Kobbelt, and D. Cohen-Or, "Competing fronts for coarse–to–fine surface reconstruction," in *Computer Graphics Forum*, vol. 25, no. 3. Wiley Online Library, 2006, pp. 389–398.
[74] P. M. Bartier and C. P. Keller, "Multivariate interpolation to incorporate thematic surface data using inverse distance weighting (idw)," *Computers & Geosciences*, vol. 22, no. 7, pp. 795–799, 1996.
[75] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2462–2470.
[76] Y. Qiu, X. Xu, L. Qiu, Y. Pan, Y. Wu, W. Chen, and X. Han, "3dcaricshop: A dataset and a baseline method for single-view 3d caricature face reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 236–10 245.
[77] A. Bangor, P. Kortum, and J. Miller, "Determining what individual sus scores mean: Adding an adjective rating scale," *Journal of usability studies*, vol. 4, no. 3, pp. 114–123, 2009.
[78] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, "A papier-mâché approach to learning 3d surface generation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 216–224.
[79] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, "3d-r2n2: A unified approach for single and multi-view 3d object reconstruction," in *European conference on computer vision*. Springer, 2016, pp. 628–644.

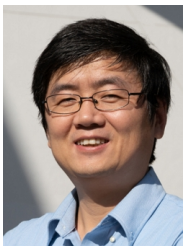
Zhongjin Luo received his B.Eng. degree in Software Engineering from Tongji University in 2019. He is currently a Ph.D. student in SSE at the Chinese University of Hong Kong, Shenzhen. His research interests are sketch-based modeling and human digitization.

Dong Du is a research associate at GAP-Lab, the Chinese University of Hong Kong, Shenzhen. He received his Ph.D. degree from the University of Science and Technology of China in 2021 and his BSc degree from Nanjing University of Science and Technology in 2014. He also attended the School of Creative Media, City University of Hong Kong as a research associate in 2017. His research mainly focuses on sketch-based modeling and 3D/4D reconstruction.



Heming Zhu is a Ph.D. student at VCAI, Max Planck Institute for Informatics, Saarland Informatics Campus. He received his master's and bachelor's degrees in computer science and technology at Zhejiang University. He also attended GAP Lab, Chinese University of Hong Kong, Shenzhen, as a research associate until 2022. His research mainly focuses on human modeling and synthesis and clothing reconstruction.

Yizhou Yu is a full professor in the Department of Computer Science at the University of Hong Kong. He was first a tenure-track and then a tenured professor at University of Illinois, Urbana-Champaign (UIUC) for 12 years. He has also collaborated with eBay Research, Google Brain and Microsoft Research in the past. He received his PhD degree in computer science from the computer vision group at University of California, Berkeley. He also holds a MS degree in applied mathematics and a BE degree in computer science and engineering from Zhejiang University. His current research interests include deep learning methods for machine intelligence, computational visual media, geometric computing, intelligent video surveillance, and biomedical data analysis.



Hongbo Fu is a full professor in the School of Creative Media, City University of Hong Kong. Before joining CityU, he had postdoctoral research trainings at the Imager Lab, University of British Columbia, Canada and the Department of Computer Graphics, Max-Planck-Institut Informatik, Germany. He received the PhD degree in computer science from the Hong Kong University of Science and Technology in 2007 and the BS degree in information sciences from Peking University, China, in 2002. His primary research interests fall in the fields of computer graphics and human computer interaction. He has served as an associate editor of The Visual Computer, Computers&Graphics, and Computer Graphics Forum.

Xiaoguang Han is now an Assistant Professor and President Young Scholar of the Chinese University of Hong Kong (Shenzhen) and the Future Intelligence Network Research Institute. He received his PhD degree from the University of Hong Kong in 2017. His research interests include computer vision and computer graphics. He has published nearly 50 papers in well-known international journals and conferences, including top conferences and journals SIGGRAPH (Asia), CVPR, ICCV, ECCV, NeurIPS, ACM TOG, IEEE TPAMI, etc. He is currently a guest editor of Frontiers of Virtual Reality, and also an associate editor of the journal of Computer and Graphics.