DrawingSpinUp: 3D Animation from Single Character Drawings

JIE ZHOU, City University of Hong Kong, China
CHUFENG XIAO, Hong Kong University of Science and Technology, China
MIU-LING LAM, City University of Hong Kong, China
HONGBO FU*, Hong Kong University of Science and Technology, China

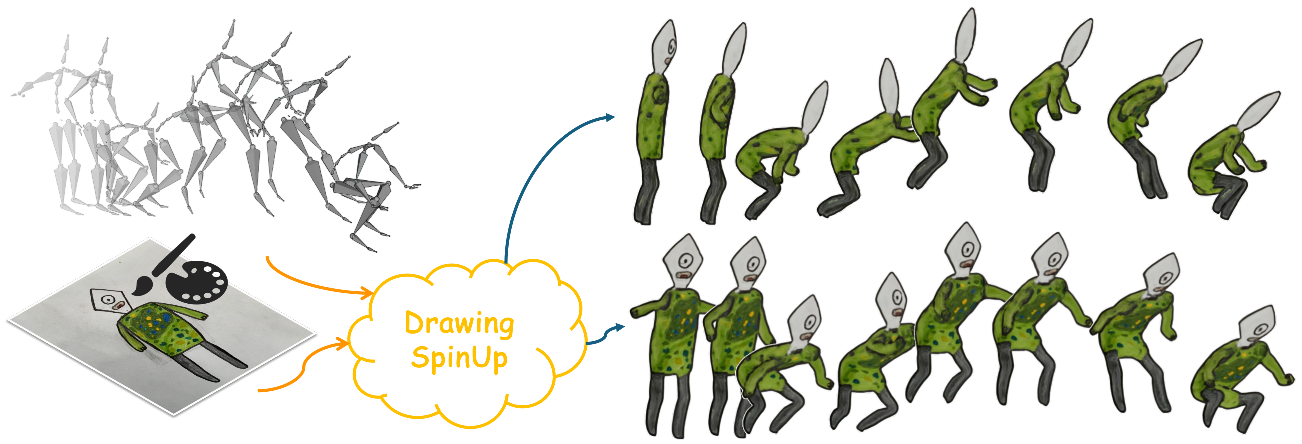

Fig. 1. Our *DrawingSpinUp* produces visually vivid 3D character animations (Right) given single input drawings (Bottom-Left) and target motions (Top-Left).

Animating various character drawings is an engaging visual content creation task. Given a single character drawing, existing animation methods are limited to flat 2D motions and thus lack 3D effects. An alternative solution is to reconstruct a 3D model from a character drawing as a proxy and then retarget 3D motion data onto it. However, the existing image-to-3D methods could not work well for amateur character drawings in terms of appearance and geometry. We observe the contour lines, commonly existing in character drawings, would introduce significant ambiguity in texture synthesis due to their view-dependence. Additionally, thin regions represented by single-line contours are difficult to reconstruct (e.g., slim limbs of a stick figure) due to their delicate structures. To address these issues, we propose a novel system, *DrawingSpinUp*, to produce plausible 3D animations and breathe life into character drawings, allowing them to freely spin up, leap, and even perform a hip-hop dance. For appearance improvement, we adopt a removal-then-restoration strategy to first remove the view-dependent contour lines and then render them back after retargeting the reconstructed character. For geometry refinement, we develop a skeleton-based thinning deformation algorithm to refine the slim structures represented by the single-line contours. The experimental evaluations and a perceptual user study show that our proposed method outperforms the existing 2D and 3D animation methods and generates high-quality 3D animations from a single character drawing. Please refer to our project page (https://drawingspinup2024.github.io) for the code and generated animations.

CCS Concepts: • **Computing methodologies** → **Animation**; *Non-photorealistic rendering*.

Additional Key Words and Phrases: Character Drawing, 3D Animation, Non-photorealistic Rendering, Style Transfer

ACM Reference Format:
Jie Zhou, Chufeng Xiao, Miu-Ling Lam, and Hongbo Fu*. 2024. *DrawingSpinUp*: 3D Animation from Single Character Drawings. In *SIGGRAPH Asia 2024 Conference Papers (SA Conference Papers '24), December 3–6, 2024, Tokyo, Japan.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3680528.3687593

1 INTRODUCTION

Character drawing plays a crucial role in art and design, significantly enhancing storytelling, gaming, and animation [Huang 2018; Lasseter and Rafael 1987; Librande 1992]. Beyond professionals, character drawing is a popular medium for amateurs, including children [Cox 2013; Smith et al. 2023], to unleash their inspiration by creating imagined characters like superheroes and fairy creatures. Given such a single drawing, could we bring the human-drawn character to life, e.g., making it run, leap, and dance on paper?

To animate a character drawing, existing methods (e.g., [Hornung et al. 2007] and [Smith et al. 2023]) typically retarget 2D motions onto the character by deforming its shape in 2D image space via as-rigid-as-possible (ARAP) [Igarashi et al. 2005]. Unfortunately,

Corresponding author: Hongbo Fu (fuplus@gmail.com).
Authors' addresses: Jie Zhou, jzhou67-c@my.cityu.edu.hk, City University of Hong Kong, Hong Kong, China; Chufeng Xiao, chufengxiao@outlook.com, Hong Kong University of Science and Technology, Hong Kong, China; Miu-Ling Lam, miu.lam@cityu.edu.hk, City University of Hong Kong, Hong Kong, China; Hongbo Fu*, fuplus@gmail.com, Hong Kong University of Science and Technology, Hong Kong, China.

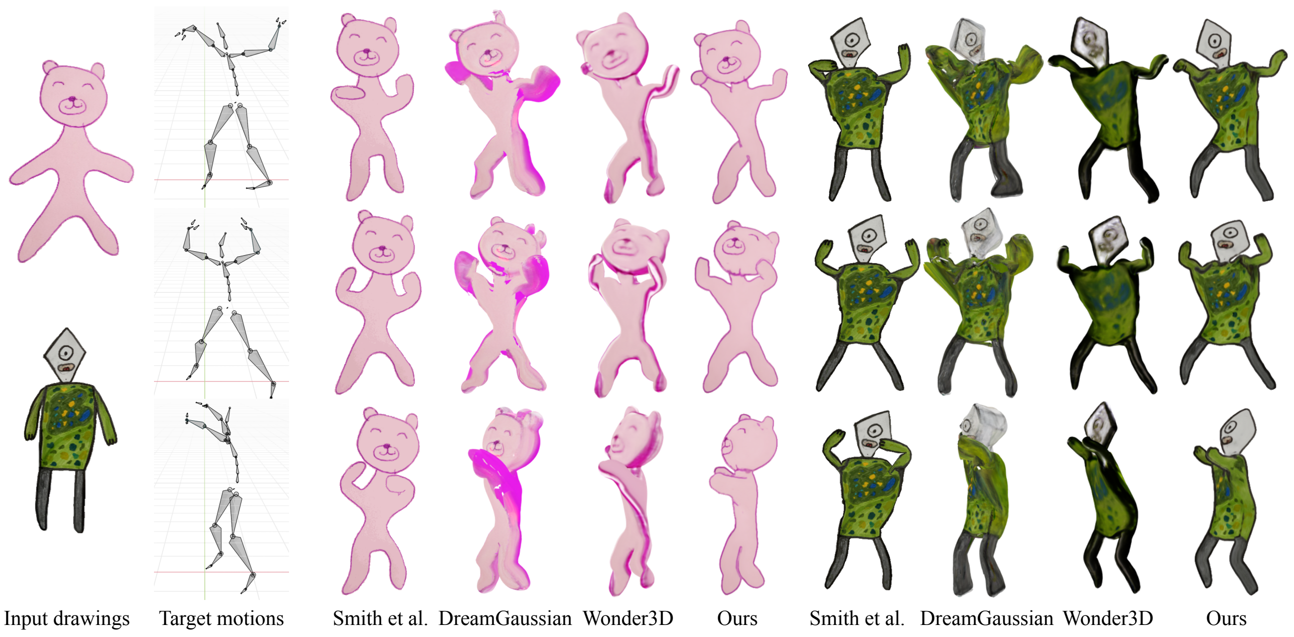

Fig. 2. Our *DrawingSpinUp* produces visually more pleasing character animation results given input drawings and target motions than the existing 2D and 3D animation techniques.

these methods cannot work well for 3D motions beyond the 2D image plane, e.g., rotating the character's body or tilting its head, as shown in Fig. 2 (results of Smith et al.). This intuitive limitation stems from the lack of a 3D sense. To afford 3D motions, a straightforward solution is to reconstruct 3D models as proxies from character drawings and then retarget 3D motions onto them. However, existing image-to-3D methods (e.g., One-2-3-45 [Liu et al. 2024b], One-2-3-45++ [Liu et al. 2024a], DreamGaussian [Tang et al. 2023], Wonder3D [Long et al. 2024], LRM [Hong et al. 2024]) always struggle to reconstruct visually pleasing results from amateur character drawings due to the domain gap between photo-realistic images and human-drawn sketches, as shown in Fig. 2 (results of DreamGaussian and Wonder3D).

Unlike photo-realistic images merely with texture details, we observe character drawings often exhibit diverse types of strokes, including interior lines and contour lines, as shown in Fig. 3 (b)-(c). Specifically, interior lines appear inside a character and represent texture patterns similar to those in photo-realistic images. Different from interior lines, contour lines represent stylized character boundaries, which are view- and motion-dependent lines and are absent in photo-realistic images. Existing image-to-3D models are always trained on photo-realistic images, thus tending to mistake artistic contour lines for internal texture details. This ambiguity can lead to appearance degradation and even affect shape reconstruction quality, as shown in Fig. 8 (c). Moreover, these methods (e.g., Wonder3D [Long et al. 2024]) generally generate several fixed-view images and then fuse them to get a 3D model. Because this pipeline inherently handles contours as view-independent textures, re-training/fine-tuning them on 3D shapes with contour rendering still fails to address contour issues. Additionally, single-line contours are commonly used to depict shapes and structures in an abstract manner, e.g., slim limbs in stick figures, as shown in Fig. 3 (d). Since such depiction is rarely seen in the training images, the models pre-trained on photo-realistic images might fail to reconstruct such delicate structures.

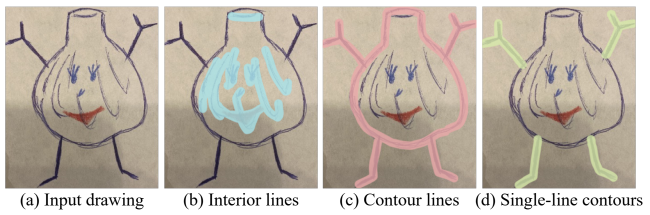

Fig. 3. An example of a character drawing with diverse types of strokes, highlighted in blue (b), red (c) and green (d).

Based on the above observations, we present *DrawingSpinUp*, the first 3D-aware animation system for generating non-planar animations from single character drawings given 3D motions. Our key idea is to recognize the contour lines and process them separately to adapt to the reconstruction prior of the pre-trained image-to-3D models. Specifically, we adopt a removal-then-restoration strategy to handle the contour lines. We first design a network to remove contour lines in a character drawing and inpaint textures in the removed regions. Next, we utilize a pre-trained image-to-3D model to reconstruct a textured geometry as a proxy for 3D motion retargeting. Finally, we propose a geometry-aware stylization network to render view- and motion-dependent contour lines for each frame of the retargeted character and enhance the internal textures to match the style of the input image. In addition, to ensure the quality of geometry reconstruction, we develop a skeleton-based thinning deformation algorithm to refine the slim structures indicated by the single-line contours. We select 120 representative amateur character drawing samples of various styles as test inputs from the Amateur Drawing Dataset collected by Smith et al. [2023] to verify the efficiency of our method. The results of extensive experiments and a perceptual user study show that our *DrawingSpinUp* can achieve plausible 3D reconstruction in geometry and appearance from single character drawings and generate vivid 3D animations, superior to existing 2D and 3D animation methods.

2 RELATED WORK

In this section, we will summarize prior works closely related to character animation based on single drawings.

2.1 2D Character Animation

Igarashi et al. [2005] introduce an as-rigid-as-possible (ARAP) deformation algorithm that enables users to intuitively deform a 2D shape by dragging keypoints, laying the foundation for animation in 2D image space. Building on this, several subsequent methods [Hornung et al. 2007; Smith et al. 2023] typically project 3D motions onto a 2D plane to facilitate character deformation using ARAP. Live Sketch [Su et al. 2018] adopts a stroke-preserving ARAP method for animating sketches while preserving the shape of user-specified strokes. AniClipart [Wu et al. 2024] further develops a differentiable ARAP algorithm to optimize and warp a clipart to a new pose, guided by text-to-video priors. However, these 2D animation methods could not directly work for 3D animations due to single-view inputs and the lack of a 3D sense, as shown in Fig. 2 (results of Smith et al.). Additionally, some data-driven methods approach this task as a conditional generation problem and utilize deep learning methods for character reanimation through motion transfer [Chan et al. 2019; Hu 2024; Xu et al. 2024]. However, these methods are still limited to generating results from a preset viewpoint. In contrast, our proposed system can effectively transfer 3D motions to a character drawing by explicitly reconstructing a 3D model from the drawing and using it for motion retargeting, allowing viewpoint freedom.

2.2 3D Character Animation

To retarget 3D motions, many researchers have proposed to reconstruct 3D models as proxies from single-view drawings. Early methods often construct 3D meshes based on silhouette and skeleton via inflation [Buchanan et al. 2013; Igarashi et al. 1999; Nealen et al. 2007; Schmidt et al. 2005; Tai et al. 2004]. Monster Mash [Dvorožňák et al. 2020] incorporates 3D inflation with a layered deformation model to casually produce a smooth 3D mesh and animation given a single-view sketch. Extended from Fibermesh [Nealen et al. 2007], CreatureShop [Zhang et al. 2022] proposes an oblique-view modeling method to create fully-textured 3D character models by transferring textures between two intrinsically symmetric body parts. Parametric shape models have also been widely used in character reconstruction. PhotoWakeUp [Weng et al. 2019] proposes a 2D warping method to deform a skinned multi-person linear (SMPL) model [Loper et al. 2015] to fit the character silhouette of a single photo to create an animatable mesh. ReenactArtFace [Qu et al. 2023] reconstructs a 3D artistic face through a 3D morphable model (3DMM) [Paysan et al. 2009] and a 2D parsing map from an input artistic image. However, the above methods either ignore the back texture or simply mirror and duplicate the given front-view texture onto the back of a character, failing to address the texture issue effectively. To address this issue, many data-driven methods [Chen et al. 2023; Luo et al. 2023; Peng et al. 2024] have been proposed to train generative models on a specific dataset to learn the missing textures. However, these methods are specifically tailored towards particular forms, such as formulated cartoon or anime characters. Unfortunately, there is a scarcity of large-scale pairs of hand-drawn character drawings and 3D assets for training since it is tedious and expensive to collect such data with subjective and artistic distortions. Recent advances in novel view synthesis [Hong et al. 2024; Liu et al. 2024b, 2023; Long et al. 2024; Tang et al. 2023] have brought a new solution, which is to exploit the powerful 3D prior capabilities of these pre-trained image-to-3D models to directly predict the missing textures. In this paper, we utilize a pre-trained diffusion model [Long et al. 2024] to generate multi-view images to compensate for the lack of back textures.

2.3 Contour Rendering

As discussed in Section 1, non-photorealistic contour lines of character drawings may degrade reconstruction quality due to its nature of view- and motion-dependence. A few works have recognized this issue and managed to mitigate it. PAniC-3D [Chen et al. 2023] proposes a line-infilling method to translate anime character images to render-like images more conducive to 3D reconstruction. It extracts lines with DoG operator, which deals with all lines equally, thus a facial landmark detector has to be used to keep certain lines around key facial features preserved. Qu et al. [2023] propose to synthesize contour lines for artistic faces by leveraging the input parsing map and a contour loss. Due to the use of uniform-thickness contour extraction strategy, their method might fail to handle the style of contour lines with inconsistent thickness and various textures. For stylizing contours, existing stylization methods [Bénard et al. 2013; Fišer et al. 2016] that learn example-based styles are a potential solution, but they do not take the underlying 3D geometry into account and thus could not generate stylized contours with multi-view consistency. Liu et al. [2021] incorporate a 3D shape with a line drawing generated from Neural Contours [Liu et al. 2020] for stroke stylization. Instead of merely stylizing lines in [Liu et al. 2021], our task also needs to harmonize the generated contours with the interior textures of the given character drawing. Inspired by the prior works, we first train a network to remove these view-dependent contour lines to prevent them from confusing the 3D reconstruction process and then design a geometry-aware stylization network to restore the contours, producing animations consistent with the input drawing style.

3 METHOD

We aim to build a system that can generate 3D animations by applying 3D target motions to a single character drawing. We follow the work of Smith et al. [2023] to preprocess a human-drawn character drawing, including detection, segmentation, and pose estimation. Given a single character drawing, the foreground segmentation

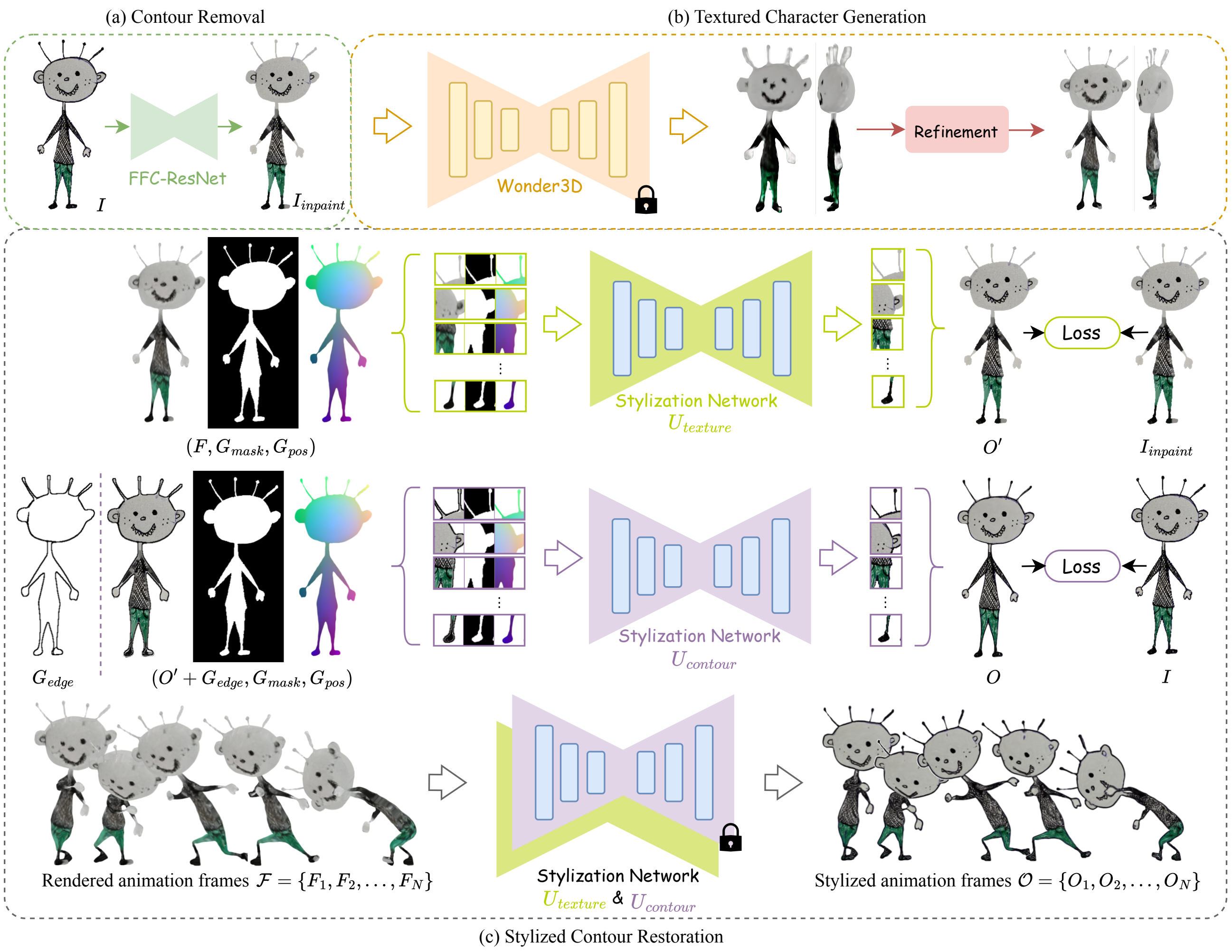
(a) Contour Removal
(b) Textured Character Generation
FFC-ResNet
I
$I_{inpaint}$
Wonder3D
Refinement
(F, G_{mask}, G_{pos})
Stylization Network
$U_{texture}$
Loss
O'
$I_{inpaint}$
G_{edge}
$(O' + G_{edge}, G_{mask}, G_{pos})$
Stylization Network
$U_{contour}$
Loss
O
I
Rendered animation frames $\mathcal{F} = \{F_1, F_2, \dots, F_N\}$
Stylization Network
$U_{texture}$ & $U_{contour}$
Stylized animation frames $\mathcal{O} = \{O_1, O_2, \dots, O_N\}$
(c) Stylized Contour Restoration

Fig. 4. The pipeline of our *DrawingSpinUp*. (a) We first remove and inpaint the contour regions of the input drawing via an FFC-ResNet. (b) We use a pre-trained Wonder3D to generate a coarse 3D geometry and then refine its shape and texture. (c) We propose a two-stage geometry-aware stylization network to restore the stylized contours across animation frames.

mask, the predicted joint keypoints, and a target 3D motion, our system *DrawingSpinUp* generates a vivid 3D animation while preserving the consistent artistic style with the input drawing. Fig. 4 shows the pipeline of *DrawingSpinUp*. We first remove and inpaint the contour regions of the input drawing via an image-to-image translation network (Section 3.1). Next, we use a pre-trained diffusion model and a neural surface reconstructor to generate a coarse 3D geometry (Section 3.2.1). Then we develop a shape refinement strategy to deal with the noisy surface and elongated structures depicted by thin strokes (Section 3.2.3). We automatically rig the character based on the predicted joint keypoints and then retarget the given 3D motion onto it to generate an initial animation (Section 3.3). Finally, we propose a geometry-aware stylization network to restore the stylized contours across animation frames (Section 3.4). We will present the details in the following subsections.

3.1 Contour Removal

The style and thickness of contours vary significantly across different drawings and can even be non-uniform within a single drawing. Therefore, using a distance transform with fixed parameters to extract contours is inadequate for all cases, leading to unclear results or excessive removal. To address this, we structure the contour prediction task as an image-to-image translation problem. Given an input drawing I and its segmented foreground mask M, we predict the corresponding contour mask M_c. We use an FFC-ResNet [Suvorov et al. 2022] as the generator of our contour removal network. We choose FFC-ResNet because contour lines are typically found at the boundaries of objects, and Fast Fourier Convolution (FFC) [Chi et al. 2020] has a large receptive field that covers the entire image, allowing for more accurate predictions of contour regions compared to vanilla convolution.

We use the 3DBiCar dataset [Luo et al. 2023] as the training dataset. We render front-view images and contour lines of different thicknesses from textured 3D biped cartoon characters in 3DBiCar with Blender [Blender Development Team. 2022]. Next, we stylize the contour lines with random colors and add them to the images to imitate the styles of amateur drawings.

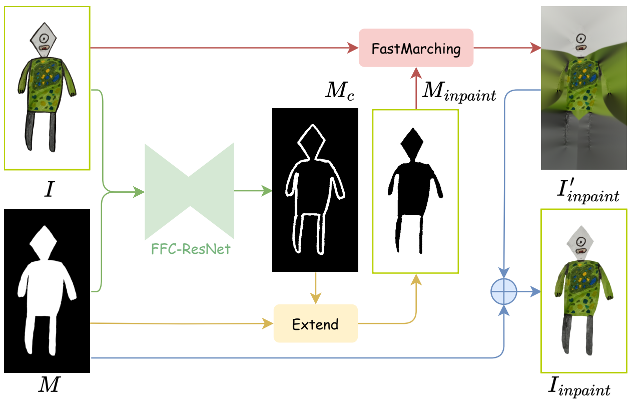

Fig. 5. The process of contour removal.

After obtaining the predicted contour mask M_c, we remove the original contour by inpainting the masked region with the interior texture of the input drawing, as illustrated in Fig. 5. To eliminate the impact of the background color (always white) on the inpainting region, we extend M_c by adding the background region $(1 - M)$ into it to get the inpainting region mask $M_{inpaint}$:

$$M_{inpaint} = M_c \cup (1 - M). \tag{1}$$

Next, we inpaint the pixels within $M_{inpaint}$ heuristically based on a fast marching method [Telea 2004]. That is, each pixel in $M_{inpaint}$ is replaced by a normalized weighted sum of the neighboring pixels in $(1 - M_{inpaint})$. In a word, we compute the inpainted drawing $I_{inpaint}$ without contour by

$$\begin{aligned} I'_{inpaint} &= FastMarching(I, M_{inpaint}); \\ I_{inpaint} &= I'_{inpaint} \cdot M + I \cdot (1 - M). \end{aligned} \tag{2}$$

Please refer to supplemental materials for more details and a comparison of different methods for contour removal.

3.2 3D Character Generation

3.2.1 Coarse Reconstruction. To reconstruct a 3D character from a single contour-free drawing, we use a pre-trained diffusion model, Wonder3D [Long et al. 2024], to produce multi-view normal maps and color images. We choose Wonder3D as our backbone since it uses an orthographic camera setting, which helps to keep strong generalizations on amateur character drawings. We apply an off-the-shelf segmentation network, IS-Net [Qin et al. 2022], to segment these normal maps to get the corresponding foreground masks. Then we utilize a neural surface reconstructor, Instant-NSR [Guo 2022; Zhao et al. 2022], to reconstruct a textured geometry from these 2D representations. However, the obtained shape and texture are both far from satisfactory. As observed from Fig. 8 (d), the thin structures of the reconstructed shape might become much thicker than those in the input drawings. In some cases, there could even be adhesion on the surface, as shown in Fig. 6, leading to geometry artifacts. Besides, the predicted texture appears blurry and loses details. These phenomena may result from the roughness of mask prediction and the misalignment of multi-view information.

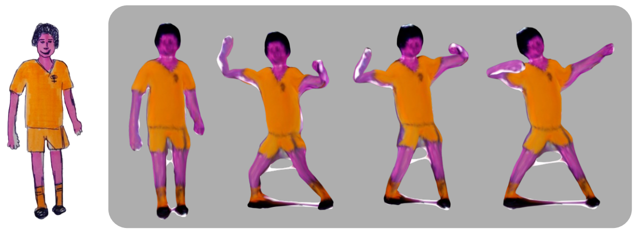

Fig. 6. An example of surface adhesion.

3.2.2 Shape Cutting. To refine the reconstructed shape, we employ front-view cutting on the queried SDF using the front-view mask M. Thus, the trimmed geometry can be defined by the 0 level set:

$$\mathcal{G} = \left\{ (x, y, z) \in \mathbb{R}^3 \mid f(x, y, z) \leq 0, M(X, Y) = 1 \right\}, \tag{3}$$

where $f(\cdot)$ is the SDF and (X, Y) is the projected 2D coordinates on the front-view plane of any 3D sampling point (x, y, z). Then we pass the level set to the Marching Cubes algorithm [Lorensen and Cline 1998] to extract the trimmed geometry, as shown in Fig. 8 (e). However, the front-view cutting operation can only change the silhouette of the front but not the thickness of the side.

3.2.3 Skeleton-based Thinning Deformation. To reduce the thickness of the side, we design a skeleton-based shape deformation algorithm to thin these regions without changing the front-view boundary. We solve it as a bi-harmonic problem [Botsch and Kobbelt 2004]. Given a trimmed geometry $\mathcal{G}$, we denote its vertices as $\mathbf{v}$ and its faces as $\mathbf{f}$. Thus, we can easily thin $\mathcal{G}$ by deforming it following

$$\begin{aligned} \mathbf{v}' &= \mathbf{v} + \mathbf{d}; \\ \mathbf{d} &= \mathcal{B}(\mathbf{v}, \mathbf{f}, \mathbf{h}, \mathbf{d_h}). \end{aligned} \tag{4}$$

where $\mathbf{d}$ represents a deformation field obtained from $\mathcal{B}(\cdot)$ that computes bi-harmonic maps using a uniform Laplacian operator [Jacobson et al. 2018]. We estimate the whole deformation field d based on the known displacements $\mathbf{d_h}$, where $\mathbf{h}$ denotes handle indices.

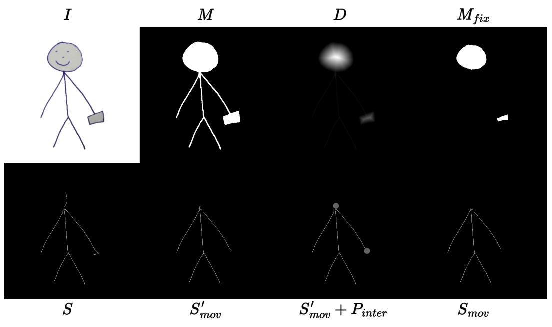

Fig. 7. An example of handle vertex location.

We next explain how we locate the handle vertices. We divide handle vertices into two categories (fixed and move-needed) to determine whether a local structure requires thinning. To distinguish

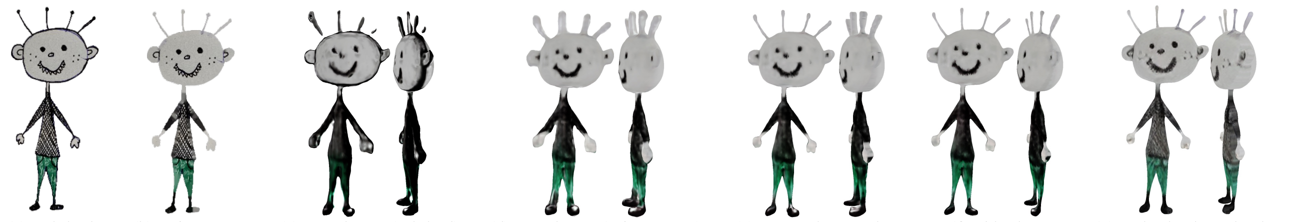

Fig. 8. Results of different processing stages.

them, as illustrated in Fig. 7, we extract a distance map D and a skeleton S from M via a medial-axis extraction algorithm [Lee et al. 1994]. Based on them, we can filter two masks M_{fix} and S_{mov} respectively, indicating the regions of fixing and move-needed vertices, following

$$\begin{aligned} M_{fix} &= M \cdot (D \geq \theta_1), \\ S'_{mov} &= S \cdot (D \leq \theta_2), \\ S_{mov} &= IR(S'_{mov}), \end{aligned} \tag{5}$$

where θ_1 and θ_2 are distance thresholds and should meet $\theta_1 > \theta_2$. We use $\theta_1 = 11$ and $\theta_2 = 6$ in our experiments. Considering that the thinning deformation may affect adjacent areas (e.g., thinning the top of the neck may damage the face that is fixed), we remove the pixels within an intersection area P_{inter} from S'_{mov} via $IR(\cdot)$. Then we can easily distinguish fixed vertices P_{fix} and move-needed vertices P_{mov} based on M_{fix} and S_{mov}. We compute the desired displacement for each vertex in P_{mov} by querying the distance value from D. Finally, we update P_{mov} through bilateral deformation following Eq. 4. For sharp edges generated by cutting and thinning, we use Laplacian smoothing [Vollmer et al. 1999] to smooth the surface. Fig. 8 (f) shows an example of thinning the character's hair and limbs based on the proposed method. Please refer to supplemental materials for more details.

3.2.4 Color Back-projection. To improve the texture quality, we recolor each vertex by back-projecting multi-view color images onto 3D space, inspired by Peng et al. [2024]. In this paper, we focus exclusively on characters in a forward stance, a common posture in amateur drawings resembling an A/T-pose. This allows us to cover most textures by querying front and back view color images. For areas not visible from the front and back views, such as the inner side of the arms close to the body or the inner surface between the legs, we employ weighted colors from neighboring vertices for inpainting. Ultimately, we color each vertex to create a textured geometry, as illustrated in Fig. 8 (g).

3.3 Rigging and Retargeting

Given the contour-free 3D characters, we employ Mixamo's online rigging tool [Inc. 2022] to automatically rig them to be animation-ready 3D assets. Mixamo uses a keypoint-based auto-rigging algorithm based on eight 2D joint keypoints on the front view. We directly reuse the joint keypoints offered by Smith et al. [2023]. Given any humanoid 3D motion data, we can retarget it onto the rigged characters with Rokoko [2023] for animation rendering. As for skinning weight, we use Blender [Blender Development Team. 2022] to automatically calculate the distance between each vertex and the closest bone and assign weights accordingly.

3.4 Stylized Contour Restoration

Given the animated contour-free character, we can render a sequence of color frames $\mathcal{F} = \{F_1, F_2, ..., F_N\}$ (N is the sequence length). As illustrated in Fig. 4 (c), we now restore the original drawing style (including texture details and contour lines) for each frame of $\mathcal{F}$, taking the stylized keyframe I as a condition, to obtain the stylized frames $O = \{O_1, O_2, ..., O_N\}$. Thus, we propose a two-stage geometry-aware stylization network to address this image-to-image translation task.

3.4.1 Network Architecture. As illustrated in Fig. 4 (c), our stylization network is composed of two cascaded modified U-Nets of Futschik et al. [2019]. The first one $U_{texture}$ is responsible for restoring internal texture details, while the second one $U_{contour}$ focuses on restoring external contour lines. The network architectures of these two U-Nets are illustrated in Fig. 9. Considering that many energetic motions may require the character to tilt or even head down while the vanilla convolutional layers are sensitive to rotation, as claimed by [Finnveden et al. 2021; Hao et al. 2022; Mo and Zhao 2024], we replace all convolutional layers (except for the final layer) of $U_{texture}$ with rotation-invariant coordinate (RIC) convolutional layers [Mo and Zhao 2024] to enhance the stability of texture details.

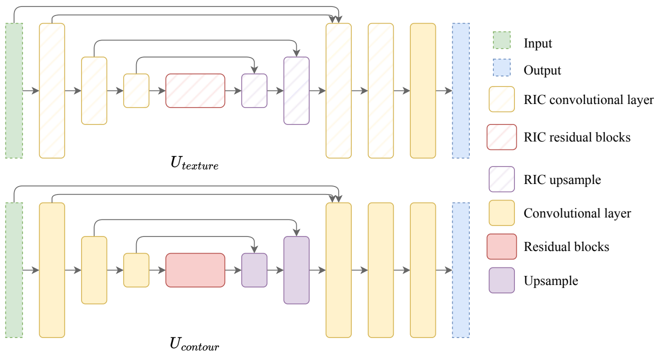

Fig. 9. The network architectures of two U-Nets in our stylization network.

3.4.2 Geometry-aware Inputs. Inspire by Jamriška et al. [2019], to maintain multi-view consistency, our stylization network takes as inputs four types of guidance channels, i.e., original color frame F, foreground mask G_{mask}, positional hint G_{pos}, and edge map G_{edge}. Specifically, G_{pos} is obtained from the normalized (x, y) coordinates of the character in a rest posture, providing the view-independent information, encouraging the source patches from I to be transferred to similar relative positions in a color frame F. G_{edge} is extracted from the Z-depth of each frame with the Canny edge detector [Canny 1986] and is used in the contour restoration stage to compensate for the view-dependent information. More specifically, the first U-Net $U_{texture}$ takes (F, G_{mask}, G_{pos}) as input and generates the middle stylized frame O'. Then we overlap G_{edge} on O' and the second U-Net $U_{contour}$ takes $(O' + G_{edge}, G_{mask}, G_{pos})$ as input and generate the final stylized frame O.

3.4.3 Patch-based Training. Learning common knowledge via a generalized stylization network for contour restoration is difficult. This is because the artistic styles of character drawings have large variances in color, thickness, and stroke style. Thus, we adopt a patch-based training strategy based on limited training samples, following Texler et al. [2020]. We use the same training strategy for $U_{texture}$ and $U_{contour}$. Specifically, we randomly sample small $k \times k (where k = 32)$ patches from all guidance channels and the ground truth. These patch pairs are then used to train the networks to generate corresponding patches with texture details or stylized contours. We adopt a combination of L1 loss, adversarial loss, and VGG loss for supervised learning. The loss between O' and $I_{inpaint}$ is used to optimize $U_{texture}$, while the loss between O and I is used to optimize $U_{contour}$. When inference, thanks to the settings of fully convolutional layers, we can feed each frame with full size to the network to finally restore internal texture details and stylized contours for a generated animation.

4 EXPERIMENTS

4.1 Runtime

Here we give the runtime of each stage on a single RTX 4090 GPU, including (1) contour removal (0.1s), (2) 3D character generation (2-3min), (3) online rigging (1-2min), (4) stylization network training (5-10min), (5) frame rendering in Blender with Eevee (0.1s/frame), and (6) stylization network inference (0.2s/frame).

The total training time is 10-15min for each character, and the inference time is 0.3s/frame. Note that we need to run Steps (1)-(4) only once to model the domain of a new character. Once the stylized network is trained, we can repeat Steps (5)-(6) to generate diverse animations given different 3D motions for the same character.

4.2 Comparison to State-of-the-Art Methods

DrawingSpinUp *VS. Smith et al. [2023].* Fig. 10 shows a comparison between Smith et al.'s method and ours, given the same 3D motion. We can observe the results generated by Smith et al. only exhibit planar motion projected from the 3D motion of the limbs, failing to capture the tilting of the body or the head of the character. Instead, our method can produce plausible 3D-aware animation that is faithful to the given 3D motion.

DrawingSpinUp *VS. Other 3D-based Animation Methods.* To our knowledge, our work is the first to support 3D animation of amateur character drawings. Thus, we mainly compare our method with those based on different image-to-3D reconstruction backbones, including DreamGaussian [Tang et al. 2023] and Wonder3D [Long et al. 2024]. Fig. 11 shows the animated results based on their reconstructed characters from the input drawings. DreamGaussian tends to produce pure color but not details for novel-view textures, leading to serious artifacts at the back of characters. Wonder3D often

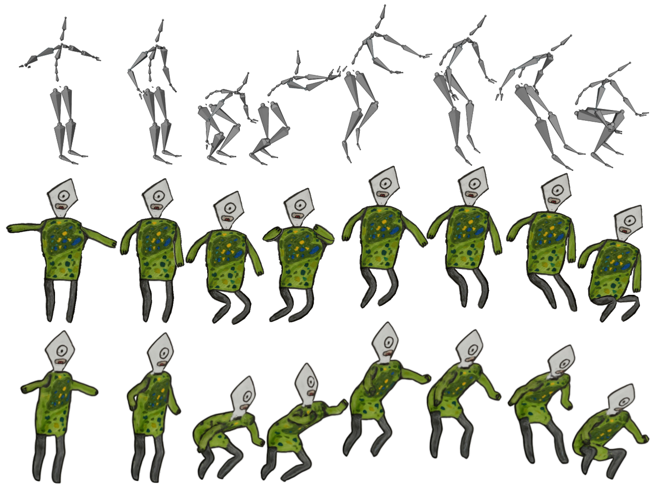
Fig. 10. *DrawingSpinUp* VS. Smith et al. [2023]. From top to bottom: target motions (front-left view), Smith et al.'s result, and ours (front-left view).

generates messy textures due to the ambiguity caused by contours. Thanks to our contour removal network and stylization network, our method can maintain consistent styles with the input drawings and produce natural animations.

4.3 Perceptual User Study

We conducted a user study to evaluate the perceptual quality of our *DrawingSpinUp* over the other compared methods via the following two metrics:

- Motion Consistency (MC): the alignment between the target motion and the generated motion;
- Style Preservation (SP): the preservation of the original style.

We conducted the user study via an online questionnaire with 15 groups of results by the compared methods (presented in a random order) and invited 53 human viewers to rate them in terms of the two metrics using a 5-point scale (1: "poor quality"; 2: "fair quality"; 3: "average quality"; 4: "good quality"; 5: "excellent quality"). The participants included a diverse demographic aged 23 to 55, with balanced gender representation. We recruited them via social media, ensuring a mix of expertise levels from beginners to professionals in animation and illustration. The average ratings for MC are 3.85 (Smith), 4.30 (DreamGaussian), 4.30 (Wonder3D), and 4.55 (Ours), respectively. The average ratings for SP are 4.18 (Smith), 3.65 (DreamGaussian), 3.50 (Wonder3D), and 4.53 (Ours), respectively.

We also conducted one-way ANOVA tests on the rating results and found a significant difference among these four methods for motion consistency (F=28.84, p<0.001) and style consistency (F=71.83, p<0.001). The further paired T-tests (with p<0.001) show that our method got a significantly higher rating in motion consistency than the other methods, i.e., Smith et al. (t=9.90), DreamGaussian (t=9.96) and Wonder3D (t=10.18). In terms of style consistency, our method also outperforms the other 3D-based methods, i.e., Smith et al. (t=4.31), DreamGaussian (t=20.74), and Wonder3D (t=23.954). The results clearly show that *DrawingSpinUp* significantly outperforms the other three methods in terms of motion consistency and style preservation.

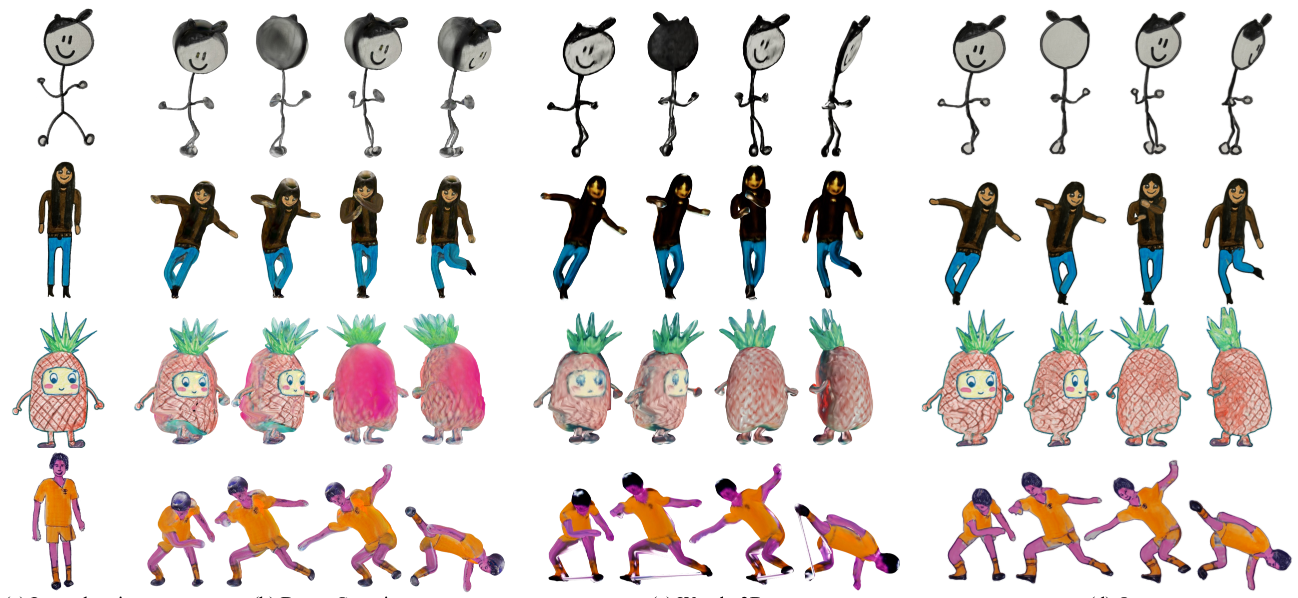

Fig. 11. *DrawingSpinUp* VS. other 3D-based animation methods.

4.4 Ablation Study

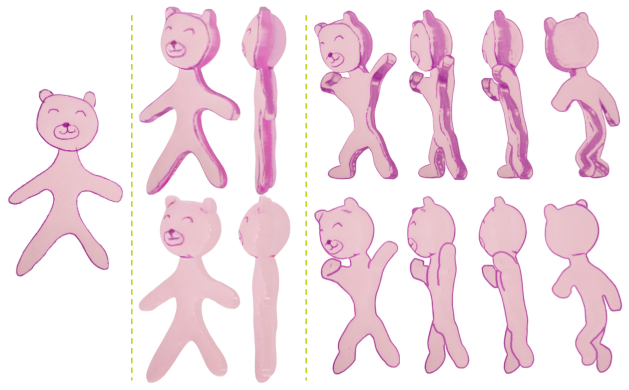

Fig. 12. Comparison between w/o (Top) and w/ (Bottom) contour removal.

Contour Removal. Fig. 12 shows the impact of contour removal. It can be seen that without this step, the contours would be considered as part of the internal texture, which cannot be solved by the following stylization step. In contrast, after contour removal, our method renders view-dependent contours to form a more plausible animation.

Cutting and Thinning. We also present a comparison of three scenarios: no-cut-no-thin (Top), only-cut (Middle), and cut-and-thin (Bottom). As shown in Fig. 13, with no shape refinement or only cutting leads to inflated results for fine structures such as hair and limbs. In contrast, our cutting-and-thinning method can handle well the elongated structures depicted by slim strokes.

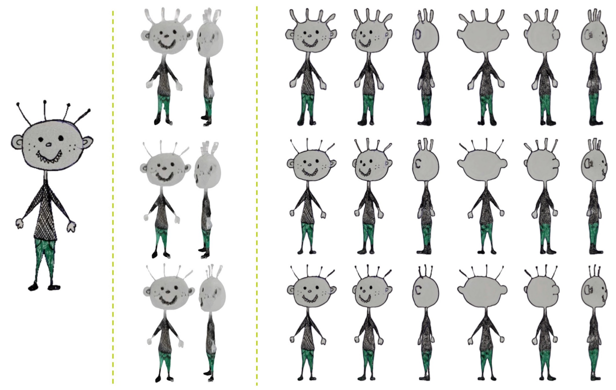

Fig. 13. Comparison of three scenarios: no-cut-no-thin (Top), only-cut (Middle), and cut-and-thin (Bottom).

Rotation Invariance. We compare the results without (Top) and with (Bottom) RIC convolution in Fig. 14. Vanilla convolution cannot work well with the head-down animation frames since it lacks the ability to align the feature maps of a transformed image with those of its original. In contrast, RIC convolution alleviates this issue, enhancing the stability of results.

5 CONCLUSION

This paper has presented the first system *DrawingSpinUp* to generate vivid 3D animations by applying 3D motions to a single character drawing, while maintaining the contour style consistent with the input drawing. To reconstruct a 3D character as a proxy from the single drawing, our system borrows the reconstruction prior from a pre-trained image-to-3D diffusion model and makes it compatible

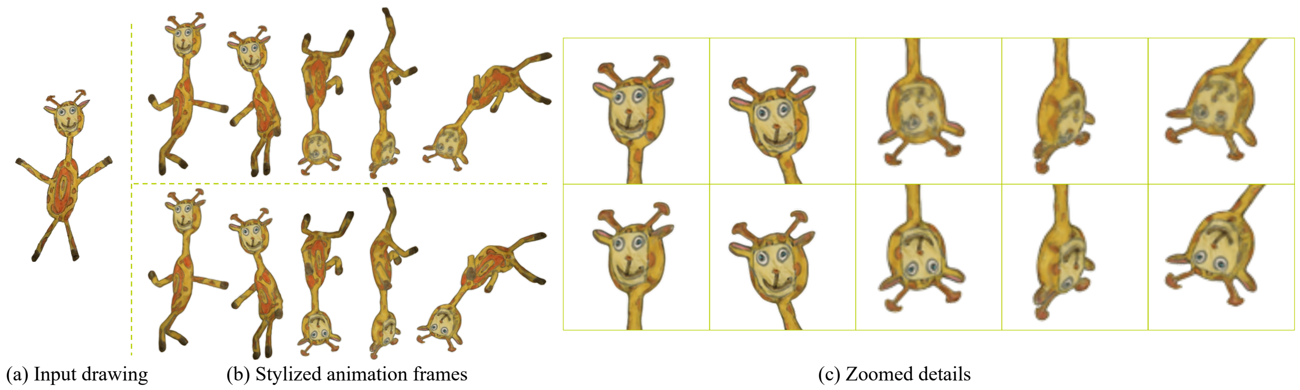

Fig. 14. Comparison between w/o (Top) and w/ (Bottom) RIC convolution.

with human-drawn drawings in terms of appearance and geometry. For appearance improvement, we adopt a removal-then-restoration strategy to first remove the view-dependent contour lines and then render them back after retargeting the reconstructed character. For shape refinement, we develop a cutting-and-thinning method to refine the slim structures represented by the single-line contours.

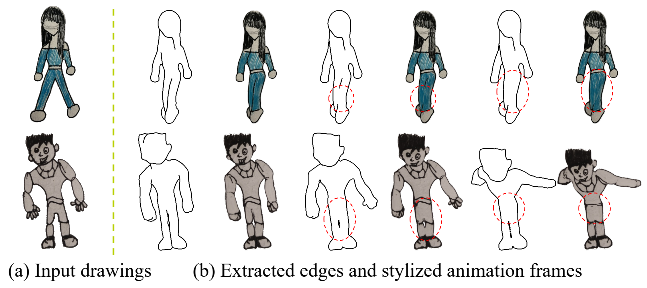

Fig. 15. Two failure cases due to inappropriate edge extraction.

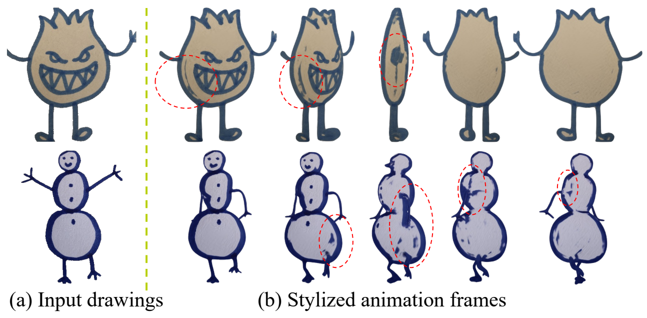

Fig. 16. Two failure cases due to too thick contour lines.

While our system can produce visually plausible 3D animations, it still has several limitations. First, we assume the characters in input drawings are approximately in a frontal A/T pose, without any self-occlusion. Some self-occlusion cases, such as crossed arms, would result in the reconstructed geometry exhibiting surface adhesion or merging of body parts. Second, our contour rendering might produce artifacts when we extract edges with inappropriate thresholds, as shown in the red dotted circles in Fig. 15. Additionally, when the contour lines of the input drawing are too thick, artifacts may appear in the generated results, as indicated by the red dotted circles in Fig. 16. Third, the rigging algorithm mainly depends on the quality of 3D reconstruction (which performs generally well thanks to our shape refinement for thin structures, e.g., the stick figure in Row 1 of Fig. 11) and keypoint detection (which can be manually corrected before rigging). However, it might produce artifacts when the drawings are highly abstract and far away from bipedal characters. We are interested in extending our method to work for general human drawings, e.g., quadruped animals.

In the future, we plan to extend our system for real-time performance, involving developing fast 3D reconstruction and auto-rigging methods, and investigate a generalized stylization network for acceleration. Although our method is not real-time, it benefits the community of character animation and non-photorealistic rendering. We envision its potential for drawing-to-animation storytelling by integrating text prompts and prototyping animation designs in cartoon films, games, VR/AR scenes, etc.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their constructive comments and suggestions, which greatly enhanced the quality of this work. Additionally, we appreciate the participants who took the time to engage in the user study. This research was supported by a start-up fund by the Hong Kong University of Science and Technology (Project No. R9913) and the Chow Sang Sang Group Research Fund (Project No. 9229119).

REFERENCES

Pierre Bénard, Forrester Cole, Michael Kass, Igor Mordatch, James Hegarty, Martin Sebastian Senn, Kurt W Fleischer, Davide Pesare, and Katherine Breeden. 2013. Stylizing animation by example. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 119–1.

Blender Development Team. 2022. Blender (Version 3.1.0) [Computer software]. https://www.blender.org

Mario Botsch and Leif Kobbelt. 2004. An intuitive framework for real-time freeform modeling. *ACM Transactions on Graphics (TOG)* 23, 3 (2004), 630–634.

Philip Buchanan, Ramakrishnan Mukundan, and Michael Doggett. 2013. Automatic single-view character model reconstruction. In *Proceedings of the international symposium on sketch-based interfaces and modeling*. 5–14.

John Canny. 1986. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence* 6 (1986), 679–698.

Caroline Chan, Shiry Ginosar, Tinghui Zhou, and Alexei A Efros. 2019. Everybody dance now. In *Proceedings of the IEEE/CVF international conference on computer* *vision*. 5933–5942.

Shuhong Chen, Kevin Zhang, Yichun Shi, Heng Wang, Yiheng Zhu, Guoxian Song, Sizhe An, Janus Kristjansson, Xiao Yang, and Matthias Zwicker. 2023. PAniC-3D: stylized single-view 3D reconstruction from portraits of anime characters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 21068–21077.

Lu Chi, Borui Jiang, and Yadong Mu. 2020. Fast fourier convolution. *Advances in Neural Information Processing Systems* 33 (2020), 4479–4488.

Maureen V Cox. 2013. *Children's drawings of the human figure*. Psychology Press.

Marek Dvorožňák, Daniel Sýkora, Cassidy Curtis, Brian Curless, Olga Sorkine-Hornung, and David Salesin. 2020. Monster mash: a single-view approach to casual 3D modeling and animation. *ACM Transactions on Graphics (ToG)* 39, 6 (2020), 1–12.

Lukas Finnveden, Ylva Jansson, and Tony Lindeberg. 2021. Understanding when spatial transformer networks do not support invariance, and what to do about it. In *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 3427–3434.

Jakub Fišer, Ondřej Jamriška, Michal Lukáč, Eli Shechtman, Paul Asente, Jingwan Lu, and Daniel Sýkora. 2016. Stylit: illumination-guided example-based stylization of 3d renderings. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 1–11.

David Futschik, Menglei Chai, Chen Cao, Chongyang Ma, Aleksei Stoliar, Sergey Korolev, Sergey Tulyakov, Michal Kucera, and Daniel Sýkora. 2019. Real-Time Patch-Based Stylization of Portraits Using Generative Adversarial Network. *Expressive* 2019 (2019), 33–42.

Yuan-Chen Guo. 2022. Instant Neural Surface Reconstruction. https://github.com/bennyguo/instant-nsr-pl.

You Hao, Ping Hu, Shirui Li, Jayaram K Udupa, Yubing Tong, and Hua Li. 2022. Gradient-Aligned convolution neural network. *Pattern Recognition* 122 (2022), 108354.

Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. 2024. LRM: Large Reconstruction Model for Single Image to 3D. In *The Twelfth International Conference on Learning Representations*.

Alexander Hornung, Ellen Dekkers, and Leif Kobbelt. 2007. Character animation from 2d pictures and 3d motion data. *ACM Transactions on Graphics (ToG)* 26, 1 (2007), 1–es.

Li Hu. 2024. Animate anyone: Consistent and controllable image-to-video synthesis for character animation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8153–8163.

Helen Q Huang. 2018. *Character sketch: a drawing course for costume designers*. Routledge.

Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. 1999. Teddy: a sketching interface for 3D freeform design. In *ACM SIGGRAPH Conference Proceedings*. 409–416.

Takeo Igarashi, Tomer Moscovich, and John F Hughes. 2005. As-rigid-as-possible shape manipulation. *ACM transactions on Graphics (TOG)* 24, 3 (2005), 1134–1141.

Adobe Inc. 2022. Mixamo. https://www.mixamo.com. Accessed: May 5, 2024.

Alec Jacobson, Daniele Panozzo, et al. 2018. libigl: A simple C++ geometry processing library. https://libigl.github.io/.

Ondřej Jamriška, Šárka Sochorová, Ondřej Texler, Michal Lukáč, Jakub Fišer, Jingwan Lu, Eli Shechtman, and Daniel Sýkora. 2019. Stylizing video by example. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 1–11.

John Lasseter and San Rafael. 1987. Principles of traditional animation applied to 3D computer animation. *Computer Graphics* 21, 4 (1987).

Ta-Chih Lee, Rangasami L Kashyap, and Chong-Nam Chu. 1994. Building skeleton models via 3-D medial surface axis thinning algorithms. *CVGIP: graphical models and image processing* 56, 6 (1994), 462–478.

Stephen Edward Librande. 1992. *Example-based character drawing*. Ph. D. Dissertation. Massachusetts Institute of Technology.

Difan Liu, Matthew Fisher, Aaron Hertzmann, and Evangelos Kalogerakis. 2021. Neural Strokes: Stylized Line Drawing of 3D Shapes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

Difan Liu, Mohamed Nabail, Aaron Hertzmann, and Evangelos Kalogerakis. 2020. Neural Contours: Learning to Draw Lines from 3D Shapes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Minghua Liu, Ruoxi Shi, Linghao Chen, Zhuoyang Zhang, Chao Xu, Xinyue Wei, Hansheng Chen, Chong Zeng, Jiayuan Gu, and Hao Su. 2024a. One-2-3-45++: Fast single image to 3d objects with consistent multi-view generation and 3d diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10072–10083.

Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Mukund Varma T, Zexiang Xu, and Hao Su. 2024b. One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. *Advances in Neural Information Processing Systems* 36 (2024).

Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. 2023. Zero-1-to-3: Zero-shot one image to 3d object. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 9298–9309.

Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, et al. 2024. Wonder3d: Single image to 3d using cross-domain diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9970–9980.

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. 2015. SMPL: a skinned multi-person linear model. *ACM Transactions on Graphics (TOG)* 34, 6 (2015), 1–16.

William E Lorensen and Harvey E Cline. 1998. Marching cubes: A high resolution 3D surface construction algorithm. In *Seminal graphics: pioneering efforts that shaped the field*. 347–353.

Zhongjin Luo, Shengcai Cai, Jinguo Dong, Ruibo Ming, Liangdong Qiu, Xiaohang Zhan, and Xiaoguang Han. 2023. RaBit: Parametric Modeling of 3D Biped Cartoon Characters with a Topological-consistent Dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12825–12835.

Hanlin Mo and Guoying Zhao. 2024. Ric-cnn: Rotation-invariant coordinate convolutional neural network. *Pattern Recognition* 146 (2024), 109994.

Andrew Nealen, Takeo Igarashi, Olga Sorkine, and Marc Alexa. 2007. FiberMesh. *ACM Transactions on Graphics (TOG)* 26, 3 (2007), 41.

Pascal Paysan, Reinhard Knothe, Brian Amberg, Sami Romdhani, and Thomas Vetter. 2009. A 3D face model for pose and illumination invariant face recognition. In *2009 sixth IEEE international conference on advanced video and signal based surveillance*. Ieee, 296–301.

Hao-Yang Peng, Jia-Peng Zhang, Meng-Hao Guo, Yan-Pei Cao, and Shi-Min Hu. 2024. Charactergen: Efficient 3d character generation from single images with multi-view pose canonicalization. *ACM Transactions on Graphics (TOG)* 43, 4 (2024), 1–13.

Xuebin Qin, Hang Dai, Xiaobin Hu, Deng-Ping Fan, Ling Shao, and Luc Van Gool. 2022. Highly Accurate Dichotomous Image Segmentation. In *ECCV*.

Linzi Qu, Jiaxiang Shang, Xiaoguang Han, and Hongbo Fu. 2023. ReenactArtFace: Artistic Face Image Reenactment. *IEEE Transactions on Visualization and Computer Graphics* (2023).

Rokoko. 2023. Rokoko Studio Live Blender. https://github.com/Rokoko/rokoko-studio-live-blender. Accessed: May 5, 2024.

Ryan Schmidt, Brian Wyvill, Mario Costa Sousa, and Joaquim A Jorge. 2005. Shapeshop: Sketch-based solid modeling with blobtrees. In *Eurographics Workshop on Sketch-Based Interfaces and Modeling*. 53–62.

Harrison Jesse Smith, Qingyuan Zheng, Yifei Li, Somya Jain, and Jessica K Hodgins. 2023. A method for animating children's drawings of the human figure. *ACM Transactions on Graphics (ToG)* 42, 3 (2023), 1–15.

Qingkun Su, Xue Bai, Hongbo Fu, Chiew-Lan Tai, and Jue Wang. 2018. Live sketch: Video-driven dynamic deformation of static drawings. In *Proceedings of the 2018 chi conference on human factors in computing systems*. 1–12.

Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor Lempitsky. 2022. Resolution-robust large mask inpainting with fourier convolutions. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 2149–2159.

Chiew-Lan Tai, Hongxin Zhang, and Jacky Chun-Kin Fong. 2004. Prototype modeling from sketched silhouettes based on convolution surfaces. In *Computer graphics forum*, Vol. 23. Wiley Online Library, 71–83.

Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. 2023. DreamGaussian: Generative Gaussian Splatting for Efficient 3D Content Creation. In *The Twelfth International Conference on Learning Representations*.

Alexandru Telea. 2004. An image inpainting technique based on the fast marching method. *Journal of graphics tools* 9, 1 (2004), 23–34.

Ondřej Texler, David Futschik, Michal Kučera, Ondřej Jamriška, Šárka Sochorová, Menclei Chai, Sergey Tulyakov, and Daniel Sýkora. 2020. Interactive video stylization using few-shot patch-based training. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 73–1.

Jörg Vollmer, Robert Mencl, and Heinrich Mueller. 1999. Improved laplacian smoothing of noisy surface meshes. In *Computer graphics forum*, Vol. 18. Wiley Online Library, 131–138.

Chung-Yi Weng, Brian Curless, and Ira Kemelmacher-Shlizerman. 2019. Photo wake-up: 3d character animation from a single photo. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 5908–5917.

Ronghuan Wu, Wanchao Su, Kede Ma, and Jing Liao. 2024. AniClipart: Clipart Animation with Text-to-Video Priors. *arXiv preprint arXiv:2404.12347* (2024).

Zhongcong Xu, Jianfeng Zhang, Jun Hao Liew, Hanshu Yan, Jia-Wei Liu, Chenxu Zhang, Jiashi Feng, and Mike Zheng Shou. 2024. Magicanimate: Temporally consistent human image animation using diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1481–1490.

Congyi Zhang, Lei Yang, Nenglun Chen, Nicholas Vining, Alla Sheffer, Francis CM Lau, Guoping Wang, and Wenping Wang. 2022. CreatureShop: Interactive 3D Character Modeling and Texturing From a Single Color Drawing. *IEEE Transactions on Visualization and Computer Graphics* (2022).

Fuqiang Zhao, Yuheng Jiang, Kaixin Yao, Jiakai Zhang, Liao Wang, Haizhao Dai, Yuhui Zhong, Yingliang Zhang, Minye Wu, Lan Xu, et al. 2022. Human performance modeling and rendering via neural animated mesh. *ACM Transactions on Graphics (TOG)* 41, 6 (2022), 1–17.