Zero-Shot Text-to-Parameter Translation for Game Character Auto-Creation

Rui Zhao[1], Wei Li[2], Zhipeng Hu[1], Lincheng Li[1]*, Zhengxia Zou[3]*, Zhenwei Shi[3], Changjie Fan[1]
[1]Netease Fuxi AI Lab, [2]Nankai University, [3]Beihang University
{zhaorui10, zphu, lilincheng, fanchangjie}@corp.netease.com,
liwei@dbis.nankai.edu.cn, {zhengxiazou, zhenweishi}@buaa.edu.cn

Figure 1. Game characters created by the proposed text-to-parameter translation (T2P) given different text prompts. The front view and three side views are shown for each character.

Abstract

Recent popular Role-Playing Games (RPGs) saw the great success of character auto-creation systems. The bone-driven face model controlled by continuous parameters (like the position of bones) and discrete parameters (like the hairstyles) makes it possible for users to personalize and customize in-game characters. Previous in-game character auto-creation systems are mostly image-driven, where facial parameters are optimized so that the rendered character looks similar to the reference face photo. This paper proposes a novel text-to-parameter translation method (T2P) to achieve zero-shot text-driven game character auto-creation. With our method, users can create a vivid in-game character with arbitrary text description without using any reference photo or editing hundreds of parameters manually. In our method, taking the power of large-scale pre-trained multi-modal CLIP and neural rendering, T2P searches both continuous facial parameters and discrete facial parameters in a unified framework. Due to the discontinuous parameter representation, previous methods have difficulty in effectively learning discrete facial parameters. T2P, to our best knowledge, is the first method that can handle the optimization of both discrete and continuous parameters. Experimental results show that T2P can generate high-quality and vivid game characters with given text prompts. T2P outperforms other SOTA text-to-3D generation methods on both objective evaluations and subjective evaluations.

1. Introduction

Role-Playing Games (RPGs) are praised by gamers for providing immersive experiences. Some of the recent popular RPGs, like Grand Theft Auto Online[1] and Naraka[2], have opened up character customization systems to players. In such systems, in-game characters are bone-driven and controlled by continuous parameters, like the position,

*Corresponding Authors.

[1]https://www.rockstargames.com/GTAOnline
[2]http://www.narakathegame.com

rotation, scale of each bone, and discrete parameters, like the hairstyle, beard styles, make-ups, and other facial elements. By manually adjusting these parameters, players can control the appearance of the characters in the game according to their personal preferences, rather than using predefined character templates. However, it is cumbersome and time-consuming for users to manually adjust hundreds of parameters - usually taking up to hours to create a character that matches their expectations.

To automatically create in-game characters, the method named Face-to-parameter translation (F2P) was recently proposed to automatically create game characters based on a single input face image [38]. F2P and its variants [39, 41] have been successfully used in recent RPGs like Narake and Justice, and virtual meeting platform Yaotai. Recent 3D face reconstruction methods [2, 7, 26, 33, 42–44] can also be adapted to create game characters. However, all the above-mentioned methods require reference face photos for auto-creation. Users may take time to search, download and upload suitable photos for their expected game characters. Compared with images, text prompts are more flexible and time-saving for game character auto-creation. A very recent work AvatarCLIP [10] achieved text-driven avatar auto-creation and animation. It optimizes implicit neural networks to generate characters. However, the created characters are controlled by implicit parameters, which lack explicit physical meanings, thus manually adjusting them needs extra designs. This will be inconvenient for players or game developers to further fine-tune the created game characters as they want.

To address the above problems, we propose text-to-parameter translation (T2P) to tackle the in-game character auto-creation task based on arbitrary text prompts. T2P takes the power of large-scale pre-trained CLIP to achieve zero-shot text-driven character creation and utilizes neural rendering to make the rendering of in-game characters differentiable to accelerate the parameters optimization. Previous works like F2Ps give up controlling discrete facial parameters due to the problem of discontinuous parameter gradients. To our best knowledge, the proposed T2P is the first method that can handle both continuous and discrete facial parameters optimization in a unified framework to create vivid in-game characters. F2P is also the first text-driven automatic character creation suitable for game environments.

Our method consists of a pre-training stage and a text-to-parameter translation stage. In the pre-training stage, we first train an imitator to imitate the rendering behavior of the game engine to make the parameter searching pipeline end-to-end differentiable. We also pre-train a translator to translate the CLIP image embeddings of random game characters to their facial parameters. Then at the text-to-parameter translation stage, on one hand, we fine-tune the translator on un-seen CLIP text embeddings to predict continuous parameters given text prompt rather than images, on the other hand, discrete parameters are evolutionally searched. Finally, the game engine takes in the facial parameters and creates the in-game characters which correspond to the text prompt described, as shown in Fig 1. Objective evaluations and subjective evaluations both indicate our method outperforms other SOTA zero-shot text-to-3D methods.

Our contributions are summarized as follows:

1) We propose a novel text-to-parameter translation method for zero-shot in-game character auto-creation. To the best of our knowledge, we are the first to study text-driven character creation ready for game environments.

2) The proposed T2P can optimize both continuous and discrete parameters in a unified framework, unlike earlier methods giving up controlling difficult-to-learn discrete parameters.

3) The proposed text-driven auto-creation paradigm is flexible and friendly for users, and the predicted physically meaningful facial parameters enable players or game developers to further finetune the game character as they want.

2. Related Work

2.1. Parametric Character Auto-Creation

Character auto-creation has been an emerging research topic because of its significance in role-playing games, augmented reality, and metaverses. Some methods on this topic are recently proposed. Tied Output Synthesis (TOS) learns to predict a set of binary facial parameters to control the graphical engine to generate a character that looks like the human in input photo [49]. Face-to-Parameter translation (F2P) is proposed to optimize a set of continuous facial parameters to minimize the distance between the generated game character's face and the input photo [38]. In F2P's following works [39, 41], the framework is improved to achieve fast and robust character creation. The PockerFace-Gan is proposed to decouple the expression features and identity features in order to generate expression-less game characters [40]. Borovikov et al. applies domain engineering and predict the facial parameters in a global-local way, considering the face as a hierarchical ensemble of general facial structure and local facial regions [3]. These methods all need reference photos to create characters, while we aim at creating characters based on text input.

2.2. 3D Face Reconstruction

3D face reconstruction also aims to generate a 3D face given single or multi-view 2D facial images. 3D morphable model (3DMM) [1] and its variants [2, 6, 9, 12, 19] are representative methods in the literature. They first parameterize a 3D face mesh data and then optimize it to match the facial identity, expression, and texture of given reference im-

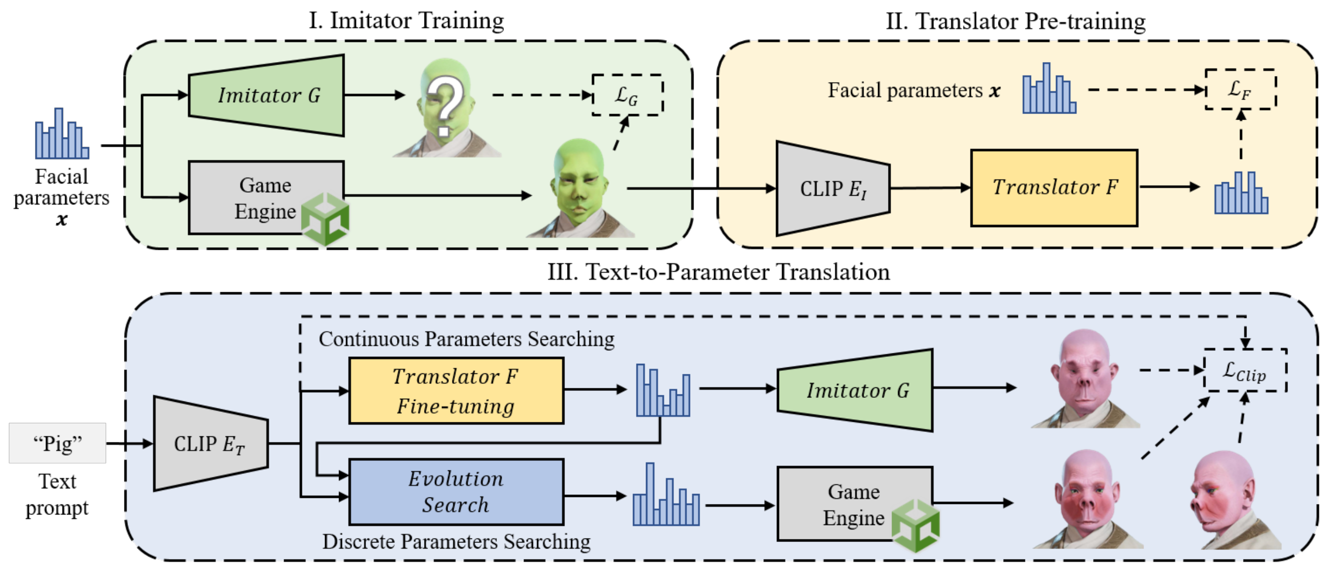

Figure 2. An overview of the proposed T2P. E_I and E_T denote the CLIP image encoder and text encoder, respectively. An imitator is trained to mimic the game engine and achieve differentiable rendering. A translator is pre-trained to translate the CLIP image embeddings to continuous facial parameters. When creating game characters given text prompts, T2P searches continuous facial parameters by fine-tuning the translator and searches discrete facial parameters by the evolution search. Finally, the facial parameters are fed into the game engine to render the in-game characters.

ages. Taking advantage of deep Convolutional Neural Networks (CNNs), high-level image representations are used to improve the predicting of the morphable model coefficients [7, 13, 44]. The recently proposed MeInGame firstly reconstructs the face as a 3DMM model and then transfers the face to game mesh keeping their topology [20]. It also predicts texture map and lighting coefficients from input images to improve the outlook of the game mesh.

2.3. Zero-Shot Text-Driven Generation

Zero-shot content generation is recently made possible by the powerful multimodel representation and generalization capabilities of CLIP [32]. Combining the CLIP with variational autoencoder or diffusion model, DALL-E [31], DALL-E 2 [31] and Imagen [30] achieved high-quality zero-shot text-to-image synthesis, and sparked widespread discussion. Text-driven image translation and manipulation, and human image generation are also explored [8, 15, 16, 18, 25, 48, 50–52]. Taking advantage of CLIP, zero-shot text-driven 3D object generation and manipulation methods made rapid advances [5, 14, 17, 23, 37, 46]. The most recently proposed Dreamfusion uses Imagen to supervise the Neural Radiance Fields network (NeRF) [24] to generate 3D object [27]. The most related work to ours named AvatarCLIP was recently proposed to achieve zero-shot text-driven 3D avatar generation and animation [10]. Given a text prompt, AvatarCLIP first generates a coarse shape by code-book-based retrieval, guided by CLIP. Then the coarse shape is used to initialize a NeuS network [47] to generate the implicit representation. Finally, the implicit 3D avatar is optimized to sculpt fine geometry and generate texture. This method treats the 3D human generation as a NeuS optimization process. However, the implicit representation makes it difficult to implement in games and unfriendly to user interaction. As a comparison, our created bone-driven game characters are controlled by explicit parameters with physical meanings. This enables players and game developers to further edit the created characters according to their needs.

3. Method

Fig. 2 shows an overview of the proposed T2P. We first train an imitator to simulate the game engine and pre-train a translator to translate the CLIP image embeddings to continuous facial parameters. Then, to achieve text-to-parameter translation, given the text prompts, we fine-tune the translator to predict continuous parameters and combine the evolution search to optimize discrete parameters.

3.1. Imitator

We train a neural imitator to mimic the behavior of the game engine in order to differentiate the rendering of in-game characters. It takes in continuous facial parameters $\boldsymbol{x}$ and renders the front view of the game character $\boldsymbol{y}$. Different from the F2P [38] taking a similar generator network architecture of DC-GAN [29], we add a positional encoder at the input-end of the renderer to improve the facial param-

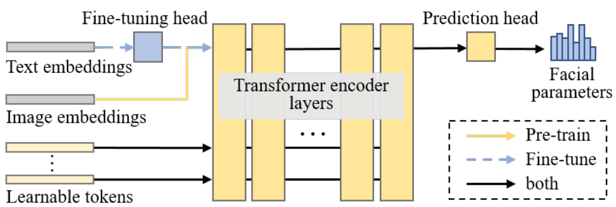

Figure 3. The architecture of our translator. The translator contains a set of transformer encoder layers, several learnable tokens, a fine-tuning head, and a prediction head. The translator is firstly pre-trained on CLIP image embeddings and then fine-tuned on CLIP text embeddings to predict continuous facial parameters. When fine-tuning the translator, only the parameters of the fine-tuning head are updated.

eters parsing on complex textures and geometry. We treat the imitator training as a regression problem to minimize the pixel-wise distance between the images rendered by the game engine and the imitator. To avoid the blurry rendered pixels, we use L1 loss as the loss function to train the imitator:

$$\begin{aligned}\mathcal{L}_G(\boldsymbol{x}) &= E_{\boldsymbol{x}\sim u(\boldsymbol{x})}\{||\boldsymbol{y}-\hat{\boldsymbol{y}}||_1\} \\ &= E_{\boldsymbol{x}\sim u(\boldsymbol{x})}\{||G(\boldsymbol{x})-Engine(\boldsymbol{x})||_1\},\end{aligned} \tag{1}$$

where $G(\boldsymbol{x})$ and $Engine(\boldsymbol{x})$ represent the image rendered by the imitator and game engine, respectively.

To prepare the training data, we randomly sample 170K continuous facial parameters $\boldsymbol{x}$ from a multidimensional uniform distribution $u(\boldsymbol{x})$. We feed these parameters into the game engine to render out the facial images. Then these facial parameters and image pairs are split into 80% and 20% for training and validation.

3.2. Continuous Parameters Searching

We aim to train a translator to predict continuous facial parameters based on CLIP text embeddings. To reduce the learning difficulty, we first pre-train the translator on CLIP image embeddings and then fine-tune it on text CLIP embeddings. The main reason is that text-parameter pairs are expensive to collect, while image-parameter pairs can be infinitely generated with the game engine.

We take the randomly sampled facial parameters and rendered image pairs mentioned in section 3.1 as training data. The rendered images are fed into the CLIP image encoder to collect image embeddings. Then we build a translator F based on a transformer encoder, and train it to map the image embeddings $\boldsymbol{e}_I$ into facial parameters $\boldsymbol{x}$, as shown in Fig. 3. The object function is defined as the L1 reconstruction loss between the true facial parameters and the predicted ones:

$$\mathcal{L}_F(\boldsymbol{e}_I,\boldsymbol{x}) = E_{\boldsymbol{e}_I\sim u(\boldsymbol{e}_I)}\{||F(\boldsymbol{e}_I)-\hat{\boldsymbol{x}}||_1\}. \tag{2}$$

When T2P creates game characters given text prompts, there is no image embeddings available. Though the CLIP is trained to pull the text and image pairs close to each other in the embedding space, there are still gaps between the two modalities. We, therefore, fine-tune the translator to fit the input text embeddings. Inspired by the recent prompt tuning study [53], we fix the parameters of the transformer and fine-tune a tiny tuner head. The translator is trained to map the text embeddings $\boldsymbol{e}_T$ to facial parameters $\boldsymbol{x}$. Then the facial parameters are fed into the imitator to render the image of the game character. The fine-tuning object function is to minimize the cosine distance between the given text embeddings $\boldsymbol{e}_T$ and the image embeddings of the rendered image:

$$\begin{aligned}\mathcal{L}_{CLIP}(\boldsymbol{e}_T,\boldsymbol{x}) &= 1-cos(\boldsymbol{e}_T, E_I(G(\boldsymbol{x}))) \\ &= 1-cos(\boldsymbol{e}_T, E_I(G(F(\boldsymbol{e}_T)))),\end{aligned} \tag{3}$$

where E_I is the CLIP image encoder. The parameters of the fine-tuned head w are iteratively updated as follows,

$$w \leftarrow w - \eta_t \frac{\partial \mathcal{L}_{CLIP}}{\partial w}, \tag{4}$$

where η_t is the learning rate at tth iteration. We follow the snapshot ensembles [11] and set the learning rate using the cosine annealing schedule with warm restarts (SGDR) [22] to encourage the translator to converge to and escape from local minima:

$$\eta_t = \eta_{min} + \frac{1}{2}(\eta_{max}-\eta_{min})(1+cos(\frac{N_t}{N}\pi)), \tag{5}$$

where η_{min}, η_{max}, and η_t denote the minimum, maximum, and current learning rate, respectively. N denotes the number of iterations between two warm restarts, and N_t denotes the number of iterations since the last restart. Each time the N_t equals N, the current iteration is called a snapshot point, and we save the predicted facial parameters at this point. These facial parameters are then used to initialize the first population of the evolution search.

3.3. Discrete Parameters Searching

In the bone-driven face model, besides continuous facial parameters controlling its bones, discrete facial elements (like the hairstyle, beard styles, and make-up) are also important. However, these elements are difficult for the imitator to learn, because they are discrete and highly changeable. Unlike previous methods that ignore discrete parameters during optimization, we propose to evolutionally search them by directly interacting with the game engine. Evolutionary algorithms have been widely used in reinforcement learning and neural architecture search [21, 36], where the objective function can be optimized without using any gradient information.

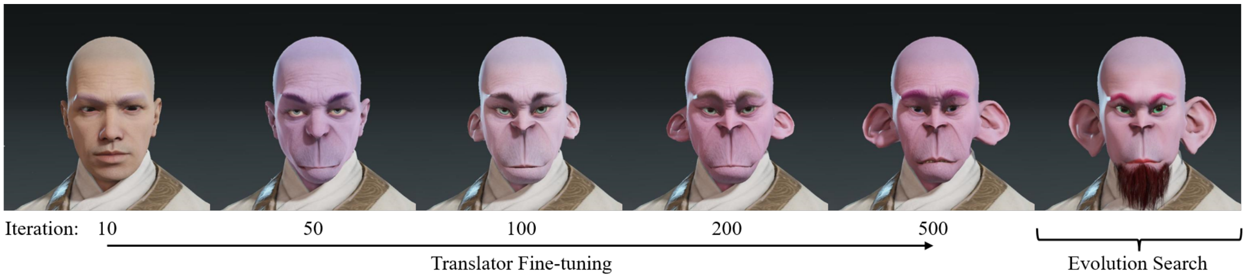

Figure 4. Game characters created by the proposed T2P given the text prompt "monkey". The first five game characters are created by the translator at different fine-tuning iterations. The last one is created by the evolution search, adding a discrete facial element, a beard.

Here we perform a text-driven evolution search to find the optimum discrete facial parameters. The initial generation contains random initialized discrete parameters as well as the continuous facial parameters predicted by the translator. To impose supervision on 3D views, we render out two images for each game character, one for front view $\boldsymbol{y}_{front}$ and one for side view $\boldsymbol{y}_{side}$. The facial parameters are scored by the CLIP model as follows,

$$\begin{aligned} S_{CLIP} = \alpha cos(E_T(T), E_I(\boldsymbol{y}_{front})) \\ + (1-\alpha) cos(E_T(T'), E_I(\boldsymbol{y}_{side})), \end{aligned} \tag{6}$$

where α is the weight coefficient, T is the given text prompt, T' is the automatically adjusted text prompt for the side view, E_T is the CLIP text encoder and E_I is the CLIP image encoder. Then k random pairs of facial parameters are selected as parents to produce the next generation through crossover and mutation. For the crossover step, child $\boldsymbol{x}^c$ is generated by randomly choosing a value from parents $\boldsymbol{x}^f$ and $\boldsymbol{x}^m$ at each position i,

$$P(x_i^c = x_i^f) + P(x_i^c = x_i^m) = 1. \tag{7}$$

For the mutation step, each child parameter $\boldsymbol{x}^c$ is added random noise at multiple randomly selected position i,

$$x_i^{c\prime} = x_i^c + noise. \tag{8}$$

The newly generated children's parameters together with the better ones of the parents' parameters are selected as the next generation and get involved in the looping selection, crossover, and mutation. The evolution process terminates until the CLIP score is converged.

3.4. Implementation Details

Network architecture. Our imitator consists of a positional encoder with four fully-connected layers and a generator with six transposed convolution layers. The generator is similar to DCGAN's generator [29], except that its Tanh activation of the output layer is removed to encourage a better convergence. The translator consists of eight Transformer encoder layers [45], each of them having eight multi-attention heads, and sixteen input tokens. The first token is the CLIP embeddings and the other tokens are learnable. We concatenate a prediction head with one single fully-connected layer after the Transformer. The fine-tuning head of the translator is a three layers perceptron with a bottleneck architecture.

Training details. The imitator and translator are both trained using SGD optimizer [4]. We set the momentum to 0.9 and set the weight decay to 5e-4. For imitator pretraining, the learning rate is set to 1e-3 and is reduced to 0.98x per 30 epochs, and the training is stopped after 500 epochs. For translator pre-training, the learning rate is set to 1e-4 and is reduced to 0.1x at the 600th epoch and the training is stopped at the 1000th epoch. We randomly sample 170K facial parameters and corresponding rendered images of in-game characters pairs to train the imitator and translator. For translator fine-tuning, the minimum and maximum learning rates are set to $\eta_{min} = 0$ and $\eta_{max} = 1$, respectively, and the number of iterations between two warm starts N is set to 10 for the SGDR learning rate scheduler. Fine-tuning is stopped when the CLIP scores are no longer improved by more than 100 iterations.

Evolution search. The facial parameters predicted by the translator at the last 5 snapshot points are selected as initial values. Each set of facial parameters contains 269 continuous parameters and 62 discrete parameters, and the initialized values of these discrete parameters are set to zeros, which means these facial elements do not appear at the beginning. These 5 sets of facial parameters together with 5 more random ones are the first population for the evolution search. We found that updating continuous parameters together with discrete parameters in the evolution search achieves better results. The number of selected pairs of parents is set to 10. The weight coefficient α is set to 0.8. The crossover rate is set to 0.4 and the mutation rate is set to 0.05.

Prompt engineering. To enhance the text prompts, we follow the CLIP [28] and adapt prompt ensembling to the given text prompts. We preset 12 template sentences, such as "{} head rendered in a game engine", and then fill the

Figure 5. In-game fictional characters created by the proposed T2P given different text prompts. The results in the first row are created by the translator. The results in the second row are created by the evolution search.

"{}" with the input text prompt. We calculate the CLIP text embeddings of the filled sentences and take their mean value as the input text embeddings for the translator and evolution search. For evolution search, we further add "side view of" to the template sentences when calculating the CLIP score of the rendered images of the side view.

4. Experimental Results and Analysis

4.1. Game Character Auto-Creation

Fig. 4 shows the game characters created by T2P given the text prompt "monkey". The first five images show the in-game characters created by the translator at different fine-tuning iterations. The in-game character gradually grows from a normal human face to look like a monkey. The evolution search further searches discrete facial elements and also slightly improves continuous parameters. The last image of Fig. 4 shows the evolution search adds a beard to the character to make it more vivid. In this process, the proposed T2P is enabled to search both continuous and discrete facial parameters to optimize the in-game character to be consistent with the given text prompt and vivid. Fig. 5 shows more results of fictional character creation. Results in the first row are controlled by continuous parameters, and results in the second row are added discrete facial elements.

T2P can create characters with animal heads, as shown in Fig. 4, fictional characters, as shown in Fig. 5, and celebrities, as shown in Fig. 6, and characters conditioned on complactied text prompts, as shown in Fig. 7. These results show the powerful zero-shot game character auto-creation ability of the proposed T2P. By inputting only a text prompt, T2P can generate a vivid character, which is more flexible and time-saving for players or game developers compared to manual customization.

4.2. Comparison with Other Methods

We compare the proposed method with AvatarCLIP [10] and DreamFusion [27]. The comparison includes objective evaluations and subjective evaluations. Since DreamFusion is not open source yet, we use the community implementation version of it, named Stable-Dreamfusion[1]. This version uses the open-source stable diffusion model [34] to drive the 3D object generation. We only compare the heads generated by these methods. This may introduce unfairness, thus we will never claim superiority besides the head part.

We feed 24 different text prompts into these two methods and our proposed T2P to generate characters respectively. Three examples are shown in Fig. 8. For objective evaluations, we compare the Inception Score [35], CLIP Ranking-1, and their speed (run on NVIDIA A30), as shown in Table 1. For each method, CLIP Ranking-1 calculates the ratio of its created characters ranked by CLIP as top-1 among the characters created by all three methods. The evaluation scores show the proposed T2P outperforms the other two methods and runs at a much faster speed.

For subjective evaluations, we invite 20 volunteers to evaluate the generation results in terms of realistic degree and consistency with the given text. They are asked to focus on the heads and faces of the characters and score them from 1 to 5, where 1 is the worst and 5 is the best. The evaluation results are shown in Table 1. Evaluation results show our method consistently outperforms the other two methods. We also notice that AvatarCLIP performs good at celebrities generation, Dreamfusion is good at fictional characters generation, while our method performs better at both types, just as shown in Fig. 8.

4.3. Ablation Studies

We conduct ablation studies to analyze the importance of the proposed translator and evolution search. We run our framework with three settings, including 1) only evolution search 2) only translator and 3) both translator and evolution search. The details of these settings are as follows.

1) **Evolution Search.** The translator is removed from the framework and the evolution search is used to directly search both continuous and discrete facial parameters given text prompts.

2) **Translator.** The evolution search is abandoned, and the translator is fine-tuned to translate the given text prompts into continuous facial parameters and gives up controlling discrete parameters.

3) **Full Implementation.** Given text prompts, the translator is fine-tuned to predict continuous facial parameters. Then, the evolution search further searches discrete parameters and also improves the continuous ones.

Fig. 9 shows the CLIP scores increasing curves with the

[1]https://github.com/ashawkey/stable-dreamfusion

Figure 6. In-game celebrities created by the proposed T2P. This figure shows the front view and the side view for each character.

Method	Objective Evaluations			Subjective Evaluations	
	Inception Score ↑	CLIP Ranking-1 ↑	Time ↓	Reality ↑	Consistency with Text ↑
DreamFusion [27]	1.60 ± 0.12	16.67%	254.50min	1.85 ± 1.02	2.23 ± 1.39
AvatarCLIP [10]	1.37 ± 0.31	16.67%	177.79min	1.97 ± 0.53	2.14 ± 0.66
T2P (ours)	$\mathbf{1.65} \pm 0.21$	$\mathbf{66.66\%}$	**359.47**s	$\mathbf{3.87 \pm 0.47}$	$\mathbf{3.34 \pm 0.53}$

Table 1. Comparison results of DreamFusion, AvatarCLIP, and the proposed T2P in terms of objective and subjective evaluations.

T2P running in 300 seconds. The means and standard deviations are calculated based on 100 times repeat running driven by one text prompt. As shown in the figure, the full implementation of our method always outperforms the other two. The translator is optimized rapidly to find optimal continuous parameters but can not further improve the CLIP scores because of lacking discrete facial elements. Compared with the translator, the evolution search is quite slow but can reach a higher CLIP score. The full implementation of T2P takes advantage of both translator and evolution search and achieves fast and better optimization.

We further test different settings of proposed T2P on 100 different text prompts to evaluate their performance. Table 2 shows the results. The first row is the result of directly using the pre-trained translator to predict continuous facial parameters, and the second row is the result of fine-tuning translator to predict parameters. The fine-tuned one can achieve a higher CLIP score, which indicates the necessity of fine-tuning. The CLIP scores of only using the evolution search and the full version of T2P are shown in the third and fourth rows, respectively. The full version of T2P achieves the highest CLIP score because it can search both continuous and discrete facial parameters to create better in-game characters.

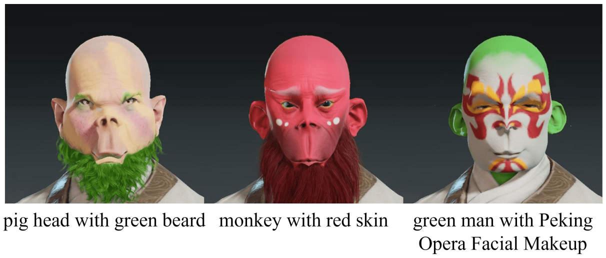

Figure 7. In-game characters created by the proposed T2P given complicated prompts.

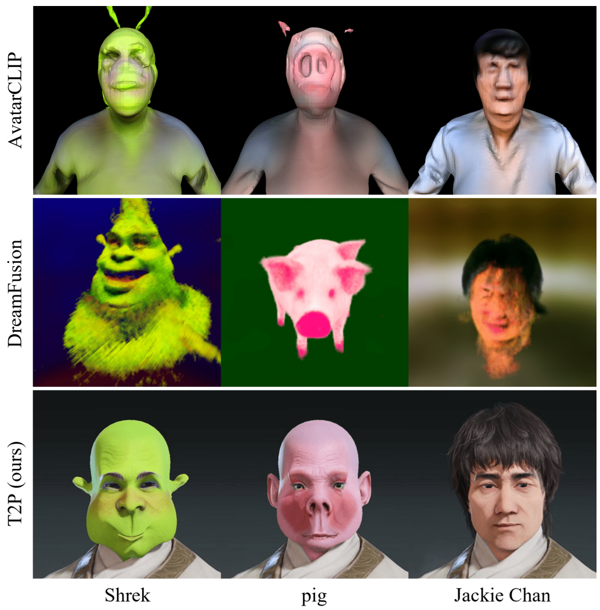

Figure 8. Comparison of AvatarCLIP, DreamFusion, and the proposed T2P. Each column shows the 3D characters created by these methods given the same text prompt.

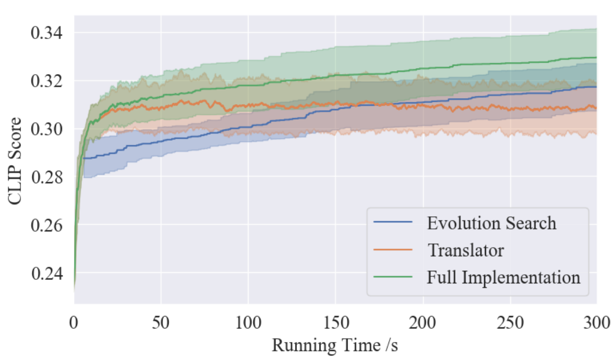

Figure 9. Curves of CLIP scores increasing within 300s under three different module settings.

4.4. Facial Paramter Interpolation

Since the generated characters are controlled by parameters with explicit physical meanings, users can further adjust the outlook of the characters as they want. One can also interpolate different facial parameters to create a new character, as shown in Fig. 10. The first row shows the interpolation between the monkey and Thanos, in which the new facial parameters are calculated as follows,

Translator	Evolution Search	CLIP Score
fixed	×	27.29 ± 3.10
fine-tuned	×	34.85 ± 3.15
×	✓	35.31 ± 2.26
fine-tuned	✓	35.72 ± 2.70

Table 2. Results of ablation studies. Four versions of the proposed method are compared.

Figure 10. Examples of the facial parameter interpolation of game characters.

$$\boldsymbol{x}_{new} = \beta \boldsymbol{x}_{monkey} + (1-\beta)\boldsymbol{x}_{Thanos}, \tag{9}$$

where β is the interpolation coefficient decreasing from 1 to 0. The results in the second row of Fig. 10 show the interpolation between the monkey and Shrek. Besides, more than two characters can also be interpolated. We believe the benefits of the facial parameters controlling bone-driven game characters can give players a higher degree of freedom in character customization.

5. Conclusion

We propose a novel method called "text-to-parameter translation" to create bone-driven in-game characters given text prompts. Our method achieves high-quality zero-shot creation of in-game characters and can search both continuous and discrete facial parameters in a unified framework. The proposed text-driven framework is flexible and time-saving for users, and the created bone-driven characters with physically meaningful facial parameters are convenient for users to further edit as they want. Experimental results show our method achieves high-quality and vivid zero-shot text-driven game character auto-creation and outperforms other SOTA text-to-3D generation methods in terms of objective evaluations, speed, and subjective evaluations.

References

[1] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 187–194, 1999. 2

[2] James Booth, Epameinondas Antonakos, Stylianos Ploumpis, George Trigeorgis, Yannis Panagakis, and Stefanos Zafeiriou. 3d face morphable models" in-the-wild". In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 48–57, 2017. 2

[3] Igor Borovikov, Karine Levonyan, Jon Rein, Pawel Wrotek, and Nitish Victor. Applied monocular reconstruction of parametric faces with domain engineering. *arXiv preprint arXiv:2208.02935*, 2022. 2

[4] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010. 5

[5] Zehranaz Canfes, M Furkan Atasoy, Alara Dirik, and Pinar Yanardag. Text and image guided 3d avatar generation and manipulation. *arXiv preprint arXiv:2202.06079*, 2022. 3

[6] Chen Cao, Yanlin Weng, Shun Zhou, Yiying Tong, and Kun Zhou. Facewarehouse: A 3d facial expression database for visual computing. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):413–425, 2013. 2

[7] Pengfei Dou, Shishir K Shah, and Ioannis A Kakadiaris. End-to-end 3d face reconstruction with deep neural networks. In *proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5908–5917, 2017. 2, 3

[8] Rinon Gal, Or Patashnik, Haggai Maron, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators. *ACM Transactions on Graphics (TOG)*, 41(4):1–13, 2022. 3

[9] Thomas Gerig, Andreas Morel-Forster, Clemens Blumer, Bernhard Egger, Marcel Luthi, Sandro Schönborn, and Thomas Vetter. Morphable face models-an open framework. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pages 75–82. IEEE, 2018. 2

[10] Fangzhou Hong, Mingyuan Zhang, Liang Pan, Zhongang Cai, Lei Yang, and Ziwei Liu. Avatarclip: Zero-shot text-driven generation and animation of 3d avatars. *arXiv preprint arXiv:2205.08535*, 2022. 2, 3, 6, 7

[11] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E Hopcroft, and Kilian Q Weinberger. Snapshot ensembles: Train 1, get m for free. *arXiv preprint arXiv:1704.00109*, 2017. 4

[12] Patrik Huber, Guosheng Hu, Rafael Tena, Pouria Mortazavian, P Koppen, William J Christmas, Matthias Ratsch, and Josef Kittler. A multiresolution 3d morphable face model and fitting framework. In *Proceedings of the 11th international joint conference on computer vision, imaging and computer graphics theory and applications*, 2016. 2

[13] Aaron S Jackson, Adrian Bulat, Vasileios Argyriou, and Georgios Tzimiropoulos. Large pose 3d face reconstruction from a single image via direct volumetric cnn regression. In *Proceedings of the IEEE international conference on computer vision*, pages 1031–1039, 2017. 3

[14] Ajay Jain, Ben Mildenhall, Jonathan T Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 867–876, 2022. 3

[15] Yuming Jiang, Ziqi Huang, Xingang Pan, Chen Change Loy, and Ziwei Liu. Talk-to-edit: Fine-grained facial editing via dialog. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13799–13808, 2021. 3

[16] Yuming Jiang, Shuai Yang, Haonan Qju, Wayne Wu, Chen Change Loy, and Ziwei Liu. Text2human: Text-driven controllable human image generation. *ACM Transactions on Graphics (TOG)*, 41(4):1–11, 2022. 3

[17] Nasir Khalid, Tianhao Xie, Eugene Belilovsky, and Tiberiu Popa. Clip-mesh: Generating textured meshes from text using pretrained image-text models. *ACM Transactions on Graphics (TOG), Proc. SIGGRAPH Asia*, 2022. 3

[18] Gwanghyun Kim and Jong Chul Ye. Diffusionclip: Text-guided image manipulation using diffusion models. 2021. 3

[19] Tianye Li, Timo Bolkart, Michael J Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4d scans. *ACM Trans. Graph.*, 36(6):194–1, 2017. 2

[20] Jiangke Lin, Yi Yuan, and Zhengxia Zou. Meingame: Create a game character face from a single portrait. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 311–319, 2021. 3

[21] Yuqiao Liu, Yanan Sun, Bing Xue, Mengjie Zhang, Gary G Yen, and Kay Chen Tan. A survey on evolutionary neural architecture search. *IEEE transactions on neural networks and learning systems*, 2021. 4

[22] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 4

[23] Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. Text2mesh: Text-driven neural stylization for meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13492–13502, 2022. 3

[24] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 3

[25] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2085–2094, 2021. 3

[26] Weilong Peng, Zhiyong Feng, Chao Xu, and Yong Su. Parametric t-spline face morphable model for detailed fitting in shape subspace. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6139–6147, 2017. 2

[27] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022. 3, 6, 7

[28] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 5

[29] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 3, 5

[30] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022. 3

[31] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021. 3

[32] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *International conference on machine learning*, pages 1060–1069. PMLR, 2016. 3

[33] Elad Richardson, Matan Sela, Roy Or-El, and Ron Kimmel. Learning detailed face reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1259–1268, 2017. 2

[34] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022. 6

[35] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016. 6

[36] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017. 4

[37] Aditya Sanghi, Hang Chu, Joseph G Lambourne, Ye Wang, Chin-Yi Cheng, Marco Fumero, and Kamal Rahimi Malekshan. Clip-forge: Towards zero-shot text-to-shape generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18603–18613, 2022. 3

[38] Tianyang Shi, Yi Yuan, Changjie Fan, Zhengxia Zou, Zhenwei Shi, and Yong Liu. Face-to-parameter translation for game character auto-creation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 161–170, 2019. 2, 3

[39] Tianyang Shi, Zhengxia Zou, Zhenwei Shi, and Yi Yuan. Neural rendering for game character auto-creation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 2

[40] Tianyang Shi, Zhengxia Zou, Xinhui Song, Zheng Song, Changjian Gu, Changjie Fan, and Yi Yuan. Neutral face game character auto-creation via pokerface-gan. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 3201–3209, 2020. 2

[41] Tianyang Shi, Zhengxia Zuo, Yi Yuan, and Changjie Fan. Fast and robust face-to-parameter translation for game character auto-creation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1733–1740, 2020. 2

[42] Ayush Tewari, Michael Zollhofer, Hyeongwoo Kim, Pablo Garrido, Florian Bernard, Patrick Perez, and Christian Theobalt. Mofa: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 1274–1283, 2017. 2

[43] Anh Tuan Tran, Tal Hassner, Iacopo Masi, Eran Paz, Yuval Nirkin, and Gérard Medioni. Extreme 3d face reconstruction: Seeing through occlusions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3935–3944, 2018. 2

[44] Anh Tuan Tran, Tal Hassner, Iacopo Masi, and Gérard Medioni. Regressing robust and discriminative 3d morphable models with a very deep neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5163–5172, 2017. 2, 3

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 5

[46] Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Clip-nerf: Text-and-image driven manipulation of neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3835–3844, 2022. 3

[47] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 3

[48] Tianyi Wei, Dongdong Chen, Wenbo Zhou, Jing Liao, Zhentao Tan, Lu Yuan, Weiming Zhang, and Nenghai Yu. Hairclip: Design your hair by text and reference image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18072–18081, 2022. 3

[49] Lior Wolf, Yaniv Taigman, and Adam Polyak. Unsupervised creation of parameterized avatars. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1530–1538, 2017. 2

[50] Weihao Xia, Yujiu Yang, Jing-Hao Xue, and Baoyuan Wu. Tedigan: Text-guided diverse face image generation and manipulation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2256–2265, 2021. 3

[51] Zipeng Xu, Tianwei Lin, Hao Tang, Fu Li, Dongliang He, Nicu Sebe, Radu Timofte, Luc Van Gool, and Errui Ding. Predict, prevent, and evaluate: Disentangled text-driven image manipulation empowered by pre-trained vision-language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18229–18238, 2022. 3

[52] Yingchen Yu, Fangneng Zhan, Rongliang Wu, Jiahui Zhang, Shijian Lu, Miaomiao Cui, Xuansong Xie, Xian-Sheng Hua, and Chunyan Miao. Towards counterfactual image manipulation via clip. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 3637–3645, 2022. 3

[53] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022. 4