HairManip: High Quality Hair Manipulation via Hair Element Disentangling

Huihuang Zhao[a,b,*], Lin Zhang[b], Paul L. Rosin[c], Yu-Kun Lai[c], Yaonan Wang[a]

[a]*National Engineering Laboratory for Robot Visual Perception and Control Technology, Hunan University, China*
[b]*School of Computer Science and Technology, Hengyang Normal University, 421002, China*
[c]*School of Computer Science & Informatics, Cardiff University, UK*

Abstract

Hair editing is challenging due to the complexity and variety of hair materials and shapes. Existing methods employ reference images or user-painted masks to edit hair and have achieved promising results. However, discrepancies in color and shape between the source and target hair can occasionally result in unrealistic results. Therefore, we propose a new hair editing method named HairManip, which decouples the hair information from the input source image into shape and color components. We then train hairstyle and hair color editing sub-networks to handle this complex information independently. To further enhance editing efficiency and accuracy, we introduce a latent code preprocessing module that effectively extracts meaningful features from hair regions, thereby improving the model's editing capabilities. The experimental results demonstrate that our method achieves significant results in editing accuracy and authenticity, thanks to the carefully designed network structure and loss functions. Code can be found at https://github.com/Zlin0530/HairManip.

Keywords: Hair editing, Preprocessing module, Editing capabilities

*Corresponding author
Email addresses: happyday.huihuang@gmail.com (Huihuang Zhao), zlin880530@gmail.com (Lin Zhang), RosinPL@cardiff.ac.uk (Paul L. Rosin), LaiY4@cardiff.ac.uk (Yu-Kun Lai), yaonan@hnu.edu.cn (Yaonan Wang)

1. Introduction

Hair is composed of a large number of soft and fine strands. It is precisely because of the flexibility and malleability of these strands that individual hair exhibits a wide range of shapes and colors. This complexity poses significant challenges to hair editing, which has attracted considerable attention and research from numerous scholars [1, 2, 3]. In practical applications, hair processing tasks also have a wide range of applications and practical value, such as in the production of animated movies, short video creation, game design, virtual reality, and more [4, 5].

In recent years, the development of computer vision [6, 7, 8] and dataset distillation [9, 10] has opened up endless possibilities for generative models [11, 12]. Generative adversarial networks (GANs) [13], as the foremost image generation method, have significantly improved the quality and resolution of generated images through its development. DCGAN [14] applied convolutional techniques to GANs, greatly enhancing the realism of generated images. StyleGAN [15, 16] made it possible to generate high-resolution face images by implementing layered control over the latent code. StyleGAN-based methods [17, 18, 19] generate high-fidelity face images by predicting edit directions in latent space. Recent emerging diffusion models [20, 21] have unique advantages in generative modeling. They generate realistic images by learning to progressively increase information decay caused by noise and subsequently eliminate noise using learned patterns.

Hair editing mainly involves transferring the hairstyle of a reference image onto a source image. In recent years, many deep learning techniques have been applied using generative adversarial networks (GANs) to study hair editing tasks [1, 2, 4, 22]. Some approaches use "user-painted" sketches [2, 23] or masks [22, 24] as inputs to the corresponding networks for hair editing. CtrlHair [1] not only allows manipulation of hair through reference photos and masks but also enables editing of hair by simply sliding a set of control bars. The emergence of the CLIP model [25] links the text and image domains and it supports a range of related work on text-based hair editing methods [3, 17, 26, 27, 28]. StyleCLIP-LM [17] infers text-guided image editing direction by a trained latent mapper, which in turn enables face attribute editing including hair editing. In order to provide users with a more user-friendly and intuitive hair editing interaction, Wei et al. proposed the HairCLIP [3] method, which unifies text and images in the same domain and designs a modulation module to achieve hair editing using a combination of text and

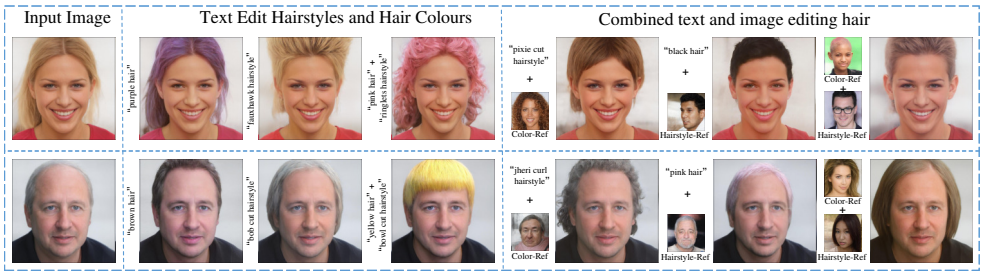

Fig.1. Our single framework supports hairstyle and hair color editing individually or jointly, and conditional inputs can come from either image or text domain.

images. StyleMC [26] provides an efficient solution to the hair editing task requiring only a few seconds to find a stable global orientation for text editing, and then using the found orientation to manipulate the image. FFCLIP [27] designs an automatic latent mapping, allowing the model to edit hair based on free-form text prompts. DeltaEdit [28] introduces text and image Delta spaces, where a Delta Mapper is trained to learn changes between image features. During the inference phase, it predicts editing directions based on changes in text features, enabling text-driven image editing without the need for text during training (text-free). However, despite these methods achieving hair editing tasks in various forms and delivering some impressive results, the diversity and flexibility in hair texture, shape, and color have led to less than ideal manipulation outcomes. For example, during the process of hair editing, some generated hairstyles may not fit well with the source image, and the color transformation may not meet the expected color requirements. As a result, the hair transfer outcomes may lack accuracy and realism.

We believe that the key to addressing the aforementioned issues lies in decoupling the complex hair information into shape and color attributes, and designing separate hair editing networks to handle these attributes discretely. Our work is inspired by StyleCLIP [17], which can divide facial attributes into different levels based on semantic information. However, we have found that it does not effectively disentangle hair elements and does not capture the finer-grained features of the hair part well. Therefore, it is challenging but desired to explore how to effectively decouple hair features and tailor a corresponding hair editing network to achieve precise and meticulous editing of complex hair information.

To achieve this goal, we propose a new hair editing framework named HairManip. Specifically, to better decouple hair information, we have de-

signed a new latent code pre-processing module that can further extract relevant features of the hair from the latent code. Then, the hair is separated into shape and color based on semantic levels. And based on the hierarchical division of facial attributes in StyleCLIP, we further divide the hair information into four categories: coarse, medium, fine, and extra fine. Based on our experience, coarse and medium-level semantic information corresponds to hairstyle attributes, while fine and extra fine-level semantic information corresponds to hair color attributes. The purpose of this decoupling of hair is to enable more precise and detailed editing of the hair. Additionally, to achieve direct and effective manipulation of hairstyle and hair color, we separately train corresponding hairstyle and hair color editing sub-networks. We also introduce new loss functions to constrain the entire model, enabling more fine-grained and better decoupling editing of the hair.

We conducted quantitative and qualitative experiments, as well as extensive ablation experiments. The experimental results show that our method outperforms existing methods in terms of editing accuracy, visual aesthetics of hair editing results, and realism. Fig.1 displays the specific results of some hairstyle manipulations.

Overall, our contributions can be divided into the following three aspects:

(1) We achieve the disentangled editing of hairstyle and hair color by decoupling the hair attributes, including shape and color. Furthermore, we have designed a latent code pre-processing module that performs pre-processing on the latent code of the source image. This preprocessing enhances its focus on the hair region's characteristics and untangles the hair information, separating it into shape and color attributes.

(2) To achieve precise and detailed control over the discrete hairstyle and hair color information, we propose an innovative hair editing network. This network consists of a pair of twin sub-networks, each responsible for handling the hairstyle and hair color information, respectively. First, we encode the hair information of the editing conditions into features. Then, we map them to the feature space of the source image. Finally, we predict the changes in the latent code based on the input conditions. This approach enables accurate hair transfer while ensuring high-quality and realistic editing results.

(3) Our goal is to achieve high-quality hair editing, and to accomplish this, we have introduced smooth L1 loss and cosine similarity loss to effectively constrain the hair editing network. This allows our model to better handle hair editing tasks and improves the quality of generated images.

2. Related Work

2.1. Generative adversarial networks

Generative Adversarial Networks (GANs) are a type of deep learning model based on game theory, proposed by Goodfellow et al [13]. It has achieved good results in fields such as image generation [29, 30, 31, 32, 33] and image style transfer [34, 35]. Over time, the images generated by GANs have become increasingly realistic, but one of its main challenges is controlling its output, e.g., changing specific attributes in facial images, such as pose, facial shape, and hairstyle, etc. StyleGAN [15, 16] has provided a feasible solution for this problem as a cutting edge technique in image synthesis. More specifically, in StyleGAN, the input noise vector Z of 512 dimensions is passed through 8 fully connected layers to map it into a latent space vector w also of 512 dimensions. The latent code w is then fed into the generator network G, which employs hierarchical controls to achieve editing of different granularity facial attributes. Therefore, many methods rely on StyleGAN to perform image editing tasks [17, 36, 37]. However, StyleGAN is characterized as an unsupervised approach, which makes it difficult to achieve controlled facial image synthesis effects. Therefore, inspired by HairCLIP [3], this paper designs a high-quality hair editing network that supports text and images as input for manipulating hair based on StyleGAN and leveraging CLIP's powerful cross-modal text-image representation capability [25].

2.2. Text-based hair manipulation methods

In recent years, research on manipulating images through text has received widespread attention and significant progress has been made [25, 28, 38, 27, 39, 40]. As there is currently no dedicated method for using text to edit hair, text-based hair editing methods mainly rely on text-guided image processing methods to achieve their goals [17, 28, 27, 40]. The optimization strategy of TediGAN [40] using fragment similarity loss requires a large amount of training data to learn the features; otherwise, it may lead to the target image's hairstyle being poorly adapted to the source image. Each text in StyleCLIP [17] is associated with a separately trained mapper, which ensures more precise generated results. However, this method is not flexible and highly time-consuming. To address the above-mentioned issues, FFCLIP [27] leverages the latent space of StyleGAN [15, 16] and the text embedding space of CLIP [25] to propose a Free-Form CLIP (FFCLIP) method. This method

unifies text prompts into a single model, eliminating the need to train additional models for each text prompt, greatly improving the model efficiency. However, FFCLIP sometimes fails to handle the details of images satisfactorily. Lyu et al. introduced a new text-driven framework called DeltaEdit [28]. In DeltaEdit, the Delta in both the image and text spaces has a good alignment distribution between the visual feature differences of CLIP images and the embedding differences of CLIP texts. Based on the CLIP delta space, DeltaEdit can be trained and inferred in a zero-shot manner, greatly reducing the cost of training.

2.3. Image-based hair manipulation methods

Hair, as an important component of facial image attributes, has been widely studied and investigated by scholars [1, 2, 4, 3, 18, 22, 41]. The Loho method [22] mainly performs two optimizations in the noise space of StyleGAN2 to complete the hairstyle transfer of a given reference image. However, the resulting images generated by this method may have significant artifacts. Michigan [2] and Barbershop [4] utilize binary masks to represent the three attributes of shape, structure, and appearance in the hair region. They edit and process the masks of the reference image and source image, merging their features to achieve hair transfer tasks. However, generating various masks for different attributes and performing calculations and processing for each mask is resource-intensive and time-consuming. CtrlHair [1] and HairCLIP [3] provide users with friendly hair editing interfaces and can generate high-quality hair transfer images. CtrlHair [1] proposes a generative adversarial network with a decoupled multi-dimensional Gaussian distribution, which allows users to complete hair editing tasks by referring to reference images, masks, or a set of sliders. HairCLIP [3] maps text and image information together into a unified multimodal vector space, and edits hair based on both text and image information. However, CtrlHair and HairCLIP may sometimes lose the detailed information of hair attributes, making it impossible to achieve a perfect transformation of hair details. Different from Michigan [2] and Barbershop [4] who used masks to divide the hair attributes, we divide the latent code w into four hierarchical levels. Coars and medium level features correspond to the hairstyle features, and fine and extra fine level features correspond to the hair colour features, and specially design the corresponding editing network for each level, aiming at editing the hairstyle and hair colour attributes more efficiently and flexibly, which makes the generated images more accurate and natural.

3. Method

3.1. Overview

Given the input source image I and the reference conditions of hairstyle and hair color, our HairManip method attempts to perform precise and effective editing on the hair region of the input source image. The specific editing process is summarized as follows:

$$E(w, r_s, r_c) = \Big(S_c(w_c, r_s), S_m(w_m, r_s), C_f\big(w_f, r_c\big), C_{xf}\big(w_{xf}, r_c\big)\Big), \quad (1)$$

where latent code w is the input source image parsed by StyleGAN inversion method "e4e" [42], and w_c, w_m, w_f and w_{xf} denote the features of different layers of latent code w, respectively. The variables r_s and r_c represent the editing conditions for hairstyle and hair color, respectively. These conditions are generated by the image and text encoders of CLIP [25], from the input text or image-based hairstyle and hair color reference conditions. $r_s \in \{r_s^t, r_s^i, 0\}, r_c \in \{r_c^t, r_c^i, 0\}$. Here, the superscript t and i refer to reference conditions from text and images respectively, and 0 means no constraint is provided. The entire hair editing network, denoted as E, is composed of two subnetworks: the hairstyle editing subnetwork $S \in \{S_c, S_m\}$ (where c and m refer to coarse and medium levels) and the hair color editing subnetwork $C \in \{C_f, C_{xf}\}$ (where f and xf refer to fine and extra fine levels), which respectively handle different dimensions of the latent code w. The purpose of this design is to train a more sophisticated network that achieves precise and detailed control over complex and diverse hairstyles and hair colors, making them more visually appealing.

As mentioned above, the hairstyle and hair color editing conditions $r_s \in \{r_s^t, r_s^i, 0\}$ and $r_c \in \{r_c^t, r_c^i, 0\}$ can come from either text or images, or be manipulated jointly by both. And when $r_c = 0$, only the hairstyle is edited, and when $r_s = 0$, only the hair color is edited. This provides users with a simple, user-friendly, and efficient way to interact.

The main purpose of hair editing is to edit the hairstyle and hair color. However, due to the complexity and diversity of hairstyles and hair colors, hair editing tasks are challenging. Therefore, we propose a GAN-based high-quality hair editing method (HairManip) as shown in Fig.2, which aims to tackle the challenging task of hair editing due to the complexity and diversity of hairstyles and hair colors. The method consists of three parts: an encoding

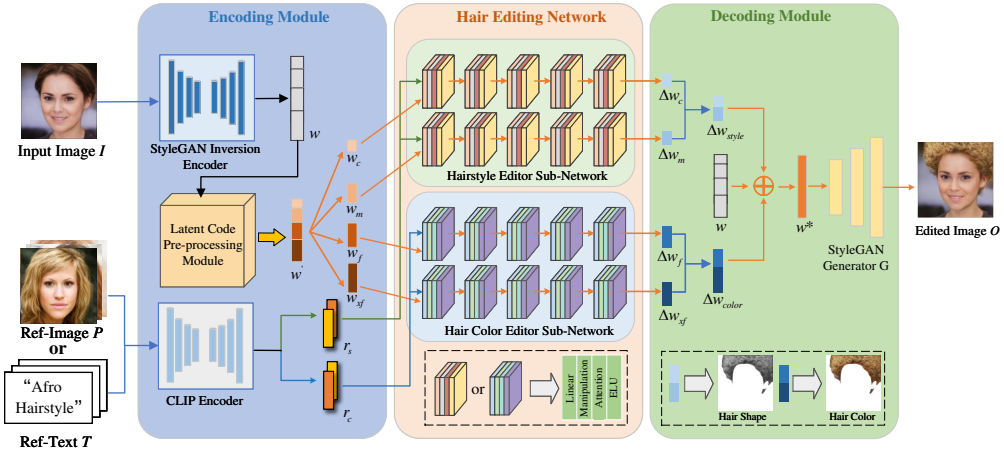

Fig.2. Overview of the HairManip framework. HairManip consists of three main components: an encoding module, a hair editing network, and a decoding module. The input source image and editing conditions are passed through the encoding module to obtain the corresponding latent code, hairstyle editing information r_s, and hair color editing information r_c. The hair editing network is used to predict the corresponding changes in hairstyle and hair color. Finally, the decoding module outputs the edited result image.

module, a hair editing network, and a decoding module. These three modules will be explained in detail below.

The training algorithm for the HairManip framework is shown in Algorithm 1.

3.2. Framework

In daily life, some people are very concerned about their hairstyle. A good hairstyle can give others a different feeling. When getting a haircut, most users express their hairstyle and hair color requirements to hairdressers through images or interactive descriptions. In order to allow users to express their hair color and style preferences clearly and accurately, we follow the principle of HairCLIP and use the excellent performance of StyleGAN in image processing and editing, as well as the powerful text/image mapping ability of CLIP, to encode the text and image into a 512-dimensional editing condition embedded in the CLIP latent space of text and image. The purpose of embedding the editing conditions into the latent space is to unify the textual and visual forms of editing conditions under one framework, and enable joint editing of hair based on both text and images.

Algorithm 1 Training Algorithm of HairManip.

Required: input source image I; image-based editing conditions P; text-based editing conditions T; StyleGAN inversion encoder E_{SI}; CLIP encoder E_{CLIP}; hairstyle information r_s; hair color information r_c; hairstyle editor network S_c and S_m; hair colour editor network C_f, C_{xf}.

latent code $w = E_{SI}(I)$;
r_s, $r_c = E_{CLIP}(P,T)$;
w' = pre-processing module(w);
optimizer.zero_grad();
$\Delta w = \Big(S_c(w_c, r_s), S_m(w_m, r_s), C_f\big(w_f, r_c\big), C_{xf}\Big(w_{xf}, r_c\Big)\Big)$:
  repeat 5 times:
    x = linear layer(w');
    x' = manipulation modules(x, r_s, r_c);
    x'' = attention(x');
    w' = elu(x'');
  $\Delta w = \Delta w + w'$;
$w^* = \Delta w + w$;
$min_{loss} = loss_{text}(w^*, T) + loss_{image}(w^*, P) + loss_{rp}(w^*, w)$;
loss.backward();
optimizer.step();

Output: Hair editing network HairManip model.

Encoding Module. The purpose of the encoding module is mainly to encode the input source image into its corresponding latent code w through the StyleGAN Inversion method e4e [42]. After obtaining the latent code, it is fed into the latent code pre-processing module to extract relevant information related to the hair. As shown in Fig.3, the latent code pre-processing module follows a simple design, consisting of blocks composed of four linear layers and activation functions. Then, based on the semantic level division of latent code in StyleGAN, latent code is further divided into four parts (coarse, medium, fine, and extra fine), corresponding to w_c, w_m, w_f and w_{xf} in Eq. (1). The input hairstyle and hair color editing conditions are embedded in a CLIP latent space of text and image through a text and image encoder E_{CLIP}, including the image-based encoder E_{CI} and text-based encoder E_{CT}. The hairstyle editing condition $r_s \in \{r_s^t, r_s^i, 0\}$ and the hair color condition $r_c \in \{r_c^t, r_c^{\mathrm{i}}, 0\}$ can come from text, image, or no input. In mathematical

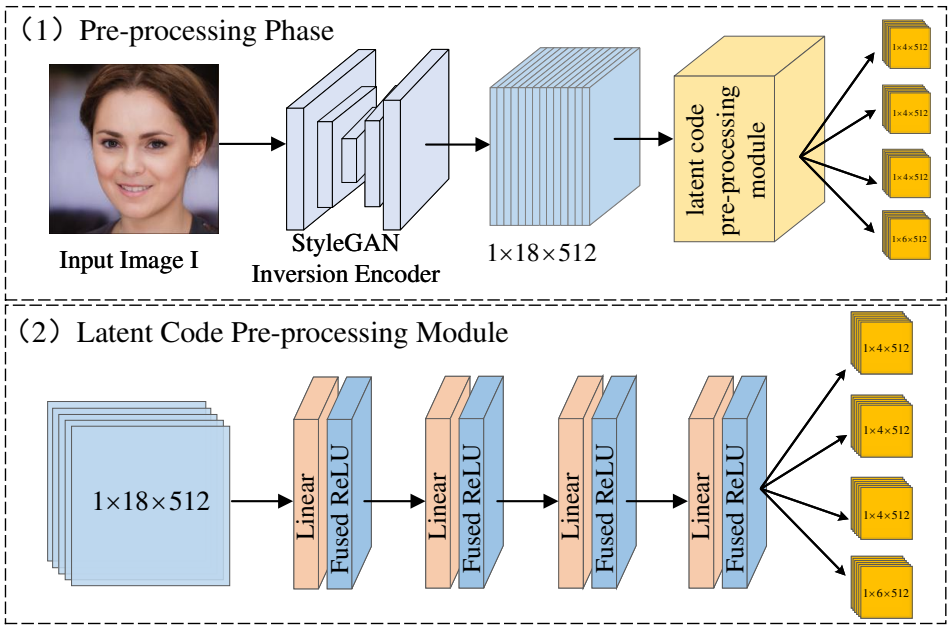

Fig.3. Details of the latent code pre-processing module. The role of the latent code pre-processing module is to further extract features from the source image. It mainly consists of four stacked linear layers and activation functions.

terms, it follows the following formula:

$$w = E_{SI}(I); r_s^i, r_c^i = E_{CI}(P); r_s^t, r_c^t = E_{CT}(T), \tag{2}$$

where E_{SI}, E_{CI} and E_{CT} represent the StyleGAN inversion encoder, CLIP image encoder, and CLIP text encoder, respectively. The variables I, T and P represent the source image, input text/image reference conditions T/P, respectively. The latent code $w \in \mathbb{R}^{512}$.

Hair Editing Network. In order to achieve more precise and detailed hair editing, our hair editing network E is mainly composed of hairstyle editing sub-network $S \in \{S_c, S_m\}$ and hair color editing sub-network $C \in \{C_f, C_{xf}\}$. The key code of hair editing algorithm is shown in Algorithm 2.

Each sub-network of the hair editing network is composed of two parts that handle latent codes at different semantic levels. Each part consists of 5 modules, including a linear layer, a manipulation module, a multi-head attention, and a non-linear activation layer (ELU). The multi-head attention mechanism aims to enable the model to capture more diverse hair feature

Algorithm 2 Key Code of Hair Editing Algorithm.

Required: pre-processed latent code w'; hair editing network E; Hair editing information r; Processed editorial information $f_\gamma(r)$, $f_\beta(r)$; latent code variation Δw, initialized to 0.

pre-processed latent code $w' = w_c + w_m + w_f + w_{xf}$;
hair editing network E $= S(S_c, S_m) + C(C_f, C_{xf})$;
for E, x in $(S_c, S_m, C_f, C_{xf}), (w_c, w_m, w_f, w_{xf})$:
    $\Delta w = E(x)$:
        repeat 5 times:
            $x =$ linear layer(x);
            $x' =$ manipulation module(x, r):
                $r_1 =$ linear layer(r);
                $r_2 =$ layer norm(r_1);
                $r_3 =$ leaky relu(r_2);
                $f_\gamma(r), f_\beta(r) =$ linear layer(r_3);
                $x' = norm(x) \times \Big(1 + f_\gamma(r)\Big) + f_\beta(r)$;
            $x' =$ attention(x');
            x=elu(x');
        $\Delta w = \Delta w + x$;

Output Δw.

information, and the role of the manipulation module is to predict information about the latent code w in the direction of the editing condition. The specific implementation process is as follows:

$$x' = \Big(1 + f_\gamma(r)\Big)(x - \mu_x)/\sigma_x + f_\beta(r), \tag{3}$$

where x is an intermediate variable obtained from the latent code w through the linear layer in the hair editing network, and x' is x computed through the manipulation module. μ_x and σ_x are the mean and standard deviation of x, respectively, $f_\gamma(r)$ and $f_\beta(r)$ are calculated through two linear layers, a normalization layer, and a leaky ReLU activation layer. The specific implementation process is shown in Fig.4.

Decoding Module. The role of the decoding module is to feed the edited latent code w^* into the pre-trained StyleGAN generator G, to generate the

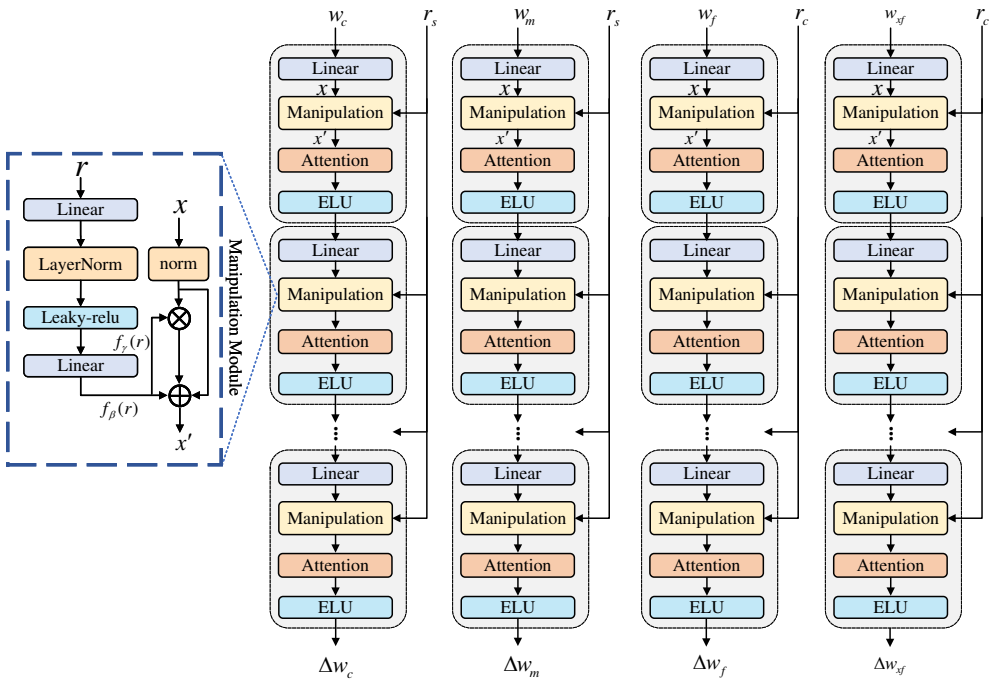

Fig.4. Architecture of Hair Editing Network. The Hair Editing Network consists of two parts: a hairstyle editing sub-network and a hair color editing sub-network, which respectively handle hair editing information and latent code fragments at different semantic levels. Each hair editing sub-network is composed of 5 simple blocks, each of which includes a linear layer (Linear), a manipulation module, a multi-head attention, and an activation layer (ELU).

edited result image, as shown below.

$$O = G(w + \Delta w), \tag{4}$$

where G represents the StyleGAN generator, $E \in \{S, C\}$ represents the hair editing network, and $\Delta w = E(w, r_s, r_c)$ represents the editing vector the measures the change in the latent code w.

3.3. Loss Functions

Given an input image, HairManip aims to manipulate the hair based on the text and image provided by the user. To achieve this, we define relevant text editing and image editing losses. In addition, to make the hair editing results more visually appealing, we define a non-hair region preservation

loss to ensure that information in non-hair regions (such as background and identity) is well-preserved while editing the hair regions. In summary, we use three types of losses to train our hair editing network, all the weight values of the losses in the experiments are continuously updated iteratively based on the model's performance and training objectives. Our complete objective function is as follows:

$$\mathcal{L} = \lambda_t \mathcal{L}_t + \lambda_i \mathcal{L}_i + \lambda_{rp} \mathcal{L}_{rp}, \tag{5}$$

where $\mathcal{L}_{st}$ represents the text editing loss, $\mathcal{L}_i$ represents the image editing loss, and L_{rp} represents the region preservation loss. According to test analysis, it was found that setting λ_t, λ_i and λ_{rp} to 2, 1, and 1 respectively yielded the best model performance, consistent with the results obtained in HairCLIP. In the following sections, we will provide detailed descriptions of these loss functions.

Text Edit Loss. The text editing loss consists of two parts: hairstyle text editing loss and hair color text editing loss, as follows:

$$\mathcal{L}_t = \lambda_{st} \mathcal{L}_{st} + \lambda_{ct} \mathcal{L}_{ct}, \tag{6}$$

where $\mathcal{L}_{st}$ and $\mathcal{L}_{ct}$ represent the hairstyle text editing loss and the hair color text editing loss, respectively. λ_{st} and λ_{ct} are the weights assigned to their respective losses, both with a default value of 1. Note that if there is no need to use text editing hairstyles or text editing hair colors, their corresponding loss weights λ_{st} and λ_{ct} will be set to 0. We use CLIP to measure the cosine distance between the resulting hair-edited image and the text prompts (hairstyle text prompt and hair color text prompt) in the latent space:

$$\mathcal{L}_{st} = 1 - \cos\Big(E_{CI}\Big(G\Big(w + E\Big(w, r_s^t, r_c\Big)\Big)\Big), r_s^t\Big), \tag{7}$$

$$\mathcal{L}_{ct} = 1 - \cos\Big(E_{CI}\Big(G\Big(w + E\Big(w, r_s, r_c^t, \Big)\Big)\Big), r_c^t\Big), \tag{8}$$

where E represents the hair editing network, $G\Big(w+E\Big(w, r_s^t, r_c\Big)\Big)$ and $G\Big(w+E\Big(w, r_s, r_c^t\Big)\Big)$ respectively represent the resulting images generated by editing the hairstyle through image and text, E_{CI} is the image encoder of CLIP, and $\cos(\cdot)$ represents cosine similarity.

Image Edit Loss. As with the text editing loss, the image editing loss consists of two parts: hairstyle image editing loss and hair color image editing

loss, represented as:

$$\mathcal{L}_i = \lambda_{si}\mathcal{L}_{si} + \lambda_{ci}\mathcal{L}_{ci}, \tag{9}$$

where λ_{si} and λ_{ci} are the weights of hairstyle image editing loss and hair color image editing loss respectively, with values of 5 and 0.02 for $\mathcal{L}_{si}$ and $\mathcal{L}_{ci}$. If there is no need to use image editing hairstyles or image editing hair colors, their corresponding loss weights $\mathcal{L}_{si}$ and $\mathcal{L}_{ci}$ will be set to 0. In addition, to better measure the similarity between two hairstyles, we take the masks of the hair regions in both images, and then encode them into the latent space of CLIP using the image encoder of CLIP, to measure the degree of match between them:

$$\mathcal{L}_{si} = 1 - \cos\Big(E_{CI}\Big(\mathbf{x}_E * P_h\big(\mathbf{x}_E\big)\Big), E_{CI}\big(\mathbf{x} * P_h(\mathbf{x})\big)\Big), \tag{10}$$

where $\mathbf{x}_E = G\left(w + E\left(w, r_s^i, r_c\right)\right)$ represents the result image generated by the image-based hairstyle editing information $r_s^i = E_{CI}\left(\mathbf{x} * P_h(\mathbf{x})\right)$ and hair color editing information $r_c \in \{r_c^t, r_c^i, 0\}$. $\mathbf{x}$ represents the reference hairstyle image. P is a pre-trained facial parsing network [43], and $P_h\big(\mathbf{x}_E\big)$ denotes the mask of the hair region generated from the result image $\mathbf{x}_E$. However, for the image hair color loss, we calculate the mean color difference between the hair region of the color reference image and the result image:

$$\mathcal{L}_{ci} = \|avg\left(\mathbf{x}_E * P_h\big(\mathbf{x}_E\big)\right) - avg\big(\mathbf{x} * P_h(\mathbf{x})\big)\|_1, \tag{11}$$

where $\mathbf{x}_E = G\Big(w + E\Big(w, r_s, r_c^i\Big)\Big)$ represents the resulting image generated by editing the hair color information in image form based on the hair color editing information $r_c^i = E_{CI}\left(\mathbf{x} * P_h(\mathbf{x})\right)$ and hairstyle editing information $r_s \in \{r_s^t, r_s^{\mathrm{i}}, 0\}$. $\mathbf{x}$ represents the reference image for hair color.

Region Preservation Loss. When editing hair, to ensure that non-hair areas such as identity information and background information remain unchanged, we define a region preservation loss as follows:

$$\begin{aligned}\mathcal{L}_{rp} =& \lambda_{id}\mathcal{L}_{id} + \lambda_{bg}\mathcal{L}_{bg} + \lambda_{norm}\mathcal{L}_{\mathrm{norm}} + \lambda_{sl}\mathcal{L}_{sl} \\ &+ \lambda_{sim}\mathcal{L}_{sim} + \lambda_{s\_mc}\mathcal{L}_{s\_mc},\end{aligned} \tag{12}$$

where $\mathcal{L}_{id}$, $\mathcal{L}_{bg}$, $\mathcal{L}_{\mathrm{norm}}$, $\mathcal{L}_{sl}$, $\mathcal{L}_{sim}$ and $\mathcal{L}_{s\_mc}$ represent identity loss, background loss, L2 norm, smooth L1 loss, cosine similarity loss and the loss to preserve hair color during hairstyle editing when editing hair. As different losses impose varying levels of constraint and importance on the model, appropriate

loss weights can indeed enhance the model's performance. Based on empirical evidence and validation, it has been found that setting the corresponding loss weights λ_{id}, λ_{bg}, λ_{norm}, λ_{sl}, λ_{sim} and λ_{s_mc} to 0.3, 1, 0.8, 0.8, 0.4, and 0.02 respectively yields the best model performance. The following will provide a detailed introduction to these losses:

$$\mathcal{L}_{id} = 1\text{- }\cos\Big(R\Big(G\big(w{+}E\big(w, r_s, r_c\big)\big)\Big), R(G(w))\Big), \tag{13}$$

where $G\Big(w + E\Big(w, r_s, r_c\Big)\Big)$ represents the resulting image generated by the source image and the editing condition passed through the hair editing network E. R is the ArcFace [44] network used to extract facial information, while $G(w)$ is the image of the source generated by the StyleGAN generator G. In addition, to preserve the background information as much as possible, we followed the principle of HairCLIP to design the corresponding background loss:

$$\mathcal{L}_{bg} = \|(\mathbf{x}_E - \mathbf{x}_w) * (P_{nh}(\mathbf{x}_{_E}) \cap P_{nh}(\mathbf{x}_{_w}))\|_2, \tag{14}$$

where $\mathbf{x}_w = G\big(w\big)$, P represents the facial parsing network [43], and $P_{nh}(\mathbf{x}_E)$ represents the mask of non-hair areas for $\mathbf{x}_E$.

To preserve the visual attributes of the input source image, we minimize the L2 norm of the manipulation steps in the latent space:

$$\mathcal{L}_{norm} = \left\|E\big(w, r_s, r_c\big)\right\|_2. \tag{15}$$

Furthermore, we also adopted the smooth L1 loss and cosine similarity loss to make the generated result image visually more realistic and consistent with human perception. It follows the following mathematical formulas:

$$\mathcal{L}_{sl} = smooth_{L1}(w, w + \Delta w), \tag{16}$$

$$\mathcal{L}_{sim} = 1 - \cos(w + E(w, r_s, r_c), w). \tag{17}$$

To improve the flexibility of the model in editing hair, we introduced a loss $\mathcal{L}_{s\_mc}$ that keeps hair color unchanged when editing hairstyles. The principle of this loss is the same as the hair color image loss $\mathcal{L}_{ci}$.

$$\mathcal{L}_{s\_mc} = \|\,avg\left(\mathbf{x}_E * P_h\left(\mathbf{x}_E\right)\right) - avg\left(\mathbf{x}_w * P_h\big(\mathbf{x}_w\big)\right)\|_1, \tag{18}$$

where $\mathbf{x}_E = G\Big(w + E\Big(w, r_s, 0\Big)\Big)$, $r_s \in \{r_s^t, r_s^{\mathrm{i}}\}$ and $\mathbf{x}_w$ is the reconstructed real image of the source image by the StyleGAN generator G.

4. Experiments

4.1. Implementation Details

In this section, we used the high-quality CelebA-HQ dataset as the training set for the HairManip hair editing network to train and evaluate the model. The CelebA-HQ dataset is an upgraded version of the CelebA face dataset, consisting of over 30,000 high-resolution (1024×1024) face images and related attribute labels. For the text editing conditions, we followed the previous work of HairCLIP [3] and used 44 hairstyle text descriptions and 12 hair color text descriptions as our text editing conditions.

During the training phase, we followed the data partition of CelebA-HQ dataset as done by e4e and divided it into training and test sets. We trained and tested the HairManip hair editing network on a computer with an RTX3090 GPU. For the training parameters, we set the learning rate to 0.0005, batch size to 1, and used the Adam [45] optimizer with β_1 and β_2 set to 0.9 and 0.999, respectively. The best experimental results were obtained after 50k iterations of training. During the training process, the hair editing network trains the hairstyle and hair color editing networks based on different forms of random inputs for editing conditions, including text and images. Additionally, in order to better evaluate the HairManip model, we tested the model's running efficiency and the quality metrics PSNR and SSIM of the generated results. We also invited 50 volunteers to evaluate and score the accuracy and authenticity of the results generated by different methods.

4.2. Quantitative and Qualitative Comparison

Since our method uses text and image-based editing conditions for precise hair editing, we will compare and analyze it with methods that generate images driven by text and hair editing methods, respectively, in order to verify the effectiveness of our network framework and carefully designed loss functions.

Comparison with text-driven image generation methods. To validate the effectiveness of our method in text-based hair editing, we set up ten text editing conditions (including text-based hair color editing, text-based hairstyle editing, and text-based hairstyle and hair color editing) and compared them with existing text-driven image methods such as StyleCLIP [17], TediGAN[40], HairCLIP [3] and DeltaEdit [28]. Fig.5 shows the corresponding comparison results. (Due to FFCLIP [27] not releasing a pre-trained model, we do not compare our method with FFCLIP here). The images

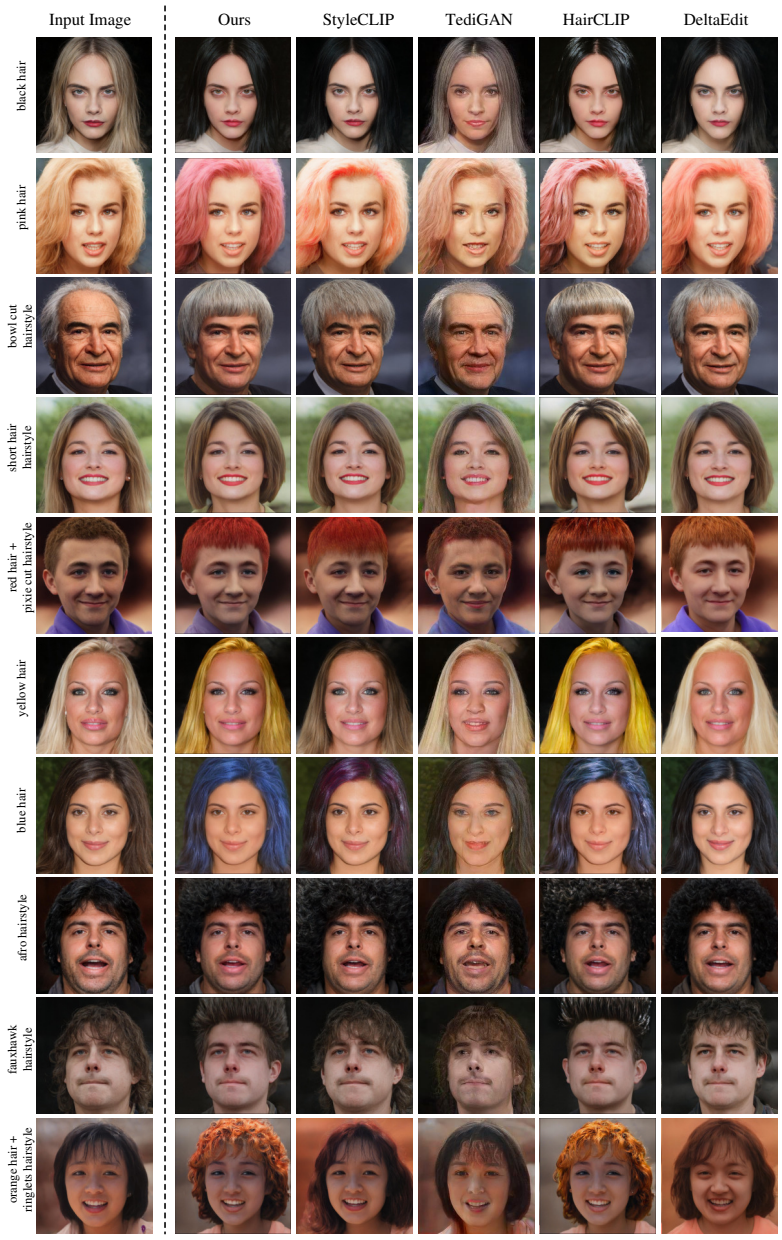

Fig.5. HairManip compares with the latest methods in visualizing hair editing through text. The text descriptions are displayed on the far left, including text editing for hair color, text editing for hairstyle, and text editing for both hairstyle and color.

generated by StyleCLIP and TediGAN were obtained based on their official parameter optimization strategies. The manipulation strength of StyleCLIP was set to 4.1 by default, while the loss-weight-clip of TediGAN was set to 0.1 and the number of iterations was set to 200.

As we can see from Fig.5, StyleCLIP trained a separate mapper for each text editing condition, which enhances its ability to manipulate hairstyles. As a result, StyleCLIP can generate reasonably accurate results for certain hair editing tasks. However, it is important to note that excessive manipulation can affect the realism of the image (as seen in the example of the fauxhawk hairstyle), which is consistent with the findings presented in the HairCLIP paper. TediGAN's generated images are not ideal for all given editing conditions. This is because TediGAN's optimization strategy using the fragment similarity loss requires sufficient training data to learn features, otherwise, the matching results may be inaccurate. The results generated by HairCLIP show noticeable white spots in the images. DeltaEdit suffers from significant identity loss in the edited images. It is worth noting that StyleCLIP, TediGAN, and DeltaEdit have not delivered satisfactory results in hair color transformation tasks. This is because these three methods are designed to handle multiple tasks and were not specifically designed for hair editing. They also lack specialized network structures to deal with the complexity of hair information. Furthermore, the intricate shapes and appearances of hair, leading to semantic diversity, make it challenging to apply the aforementioned facial editing methods to hair editing with semantic variations. For example, the DeltaEdit method trains a Delta Mapper to learn the mapping from changes in image features to changes in style space and uses text feature changes to achieve various facial attribute edits. However, it does not have dedicated modules to handle these features, which can result in the loss or difficulty in learning some feature information when mapping image features to the style space. By sharing the editing conditions in the latent space of textual and image inputs, our method is able to train multiple hair editing conditions effectively. Furthermore, through the latent code pre-processing module, we extract useful learned features, which contribute to more precise generation of the resulting images. At the same time, we separate hairstyle and hair color information and feed them separately into carefully designed hairstyle and hair color editing sub-networks, aiming to achieve more detailed and precise manipulation of the hair. In addition, we added a new smooth L1 loss function and cosine similarity loss function optimization strategy to make the generated hair editing results more realistic and visually appealing.

Table 1
Comparison of the realism and efficiency of our method with current methods in generating result images. Higher PSNR and SSIM scores are better.

	Ours	StyleCLIP	TediGAN	HairCLIP	DeltaEdit
PSNR	**28.6**	23.2	24.1	27.8	26.96
SSIM	**0.94**	0.87	0.79	0.92	0.89
Time Consuming	**4.1s**	99s	240s	**3.3s**	7.7s

This will be more evident in the ablation experiment.

In addition, we evaluated the quality of the generated images by using a randomly selected set of 11,000 images from the CelebA-HQ dataset as reference images. We employed the PSNR and SSIM metrics to assess the preservation of background and identity information in the generated images. We also compared the efficiency of our method with five other approaches, and the detailed experimental results are presented in Table 1. From Table 1, it can be observed that our method outperforms the other methods.

Comparison with hair editing methods. In order to further validate the effectiveness of our method, we compared the HairManip method with the state-of-the-art hair editing methods, Barbershop [4], HairCLIP [3] and CtrlHair [1]. The specific comparison results are shown in Fig.6. From the figure, it can be seen that the Barbershop method heavily relies on whether the pose of the source image and the hairstyle reference image are consistent. When the pose difference between the source image and the hairstyle reference image is small, the hairstyle can be successfully transferred, as shown in the second row of the results. However, when the pose difference is large, the hairstyle generated by Barbershop cannot well transfer the hairstyle of the reference image. HairCLIP is able to transfer hairstyles successfully even when the poses of source images and reference images are different. This is because HairCLIP replaces the similarity metric space with the CLIP latent space and embeds the hair region of reference images in CLIP as a conditional input. With this method, transfer between non-aligned hairstyles can be achieved. However, HairCLIP sometimes makes mistakes in hairstyle and hair color transfer, as shown in the first and third rows of the comparison results. This is because HairCLIP only roughly divides the hair information of the source image latent code into different semantic levels and trains the hair mapper network accordingly, which cannot effectively capture the features of hairstyle and hair color. In contrast to these methods, we fur-

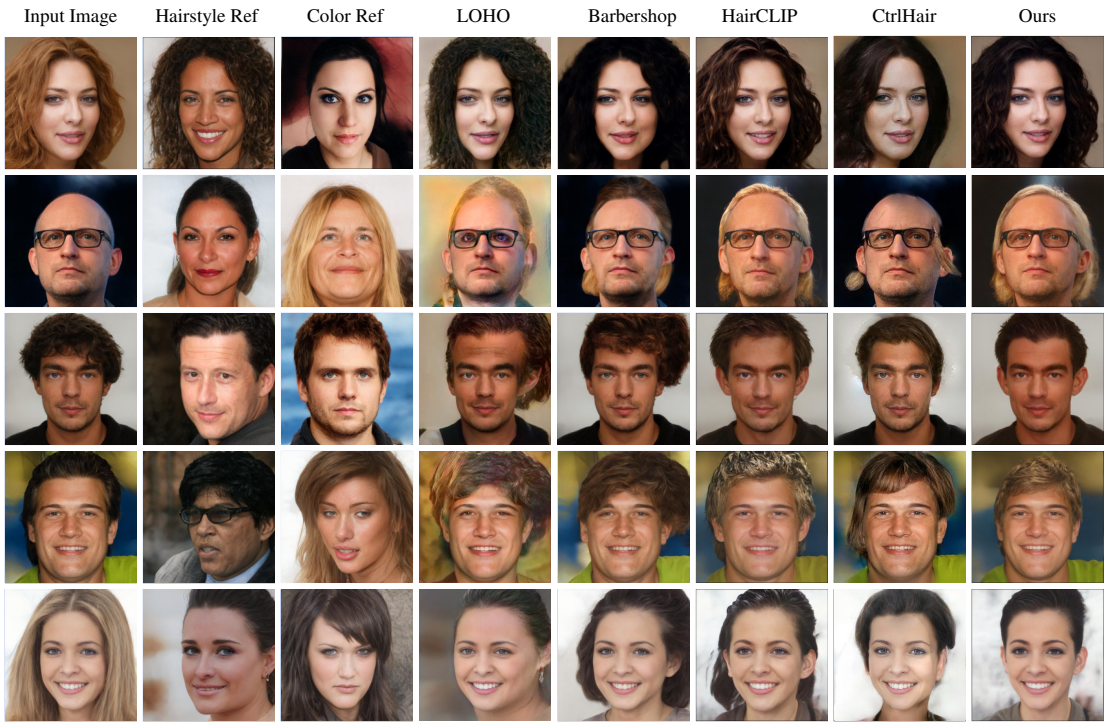

Fig.6. Comparison with current state-of-the-art hair editing methods. The first three columns indicate the source image, hairstyle reference image, and hair color reference image, respectively.

ther extract effective information from the source image's latent code on the basis of HairCLIP and design more sophisticated sub-networks for hairstyle and hair color editing to handle complex hairstyle and color information. Experimental results show that our method can perform hair editing tasks more outstandingly, and the edited results are more realistic and visually appealing.

User evaluation. To further evaluate our model, we invited 50 participants of different genders with ages ranging from 18 to 60 years to rate and evaluate various models from the two categories. For the text-driven methods, we provided 25 sets of results, with each set consisting of a source image and a randomly generated result image from one of the 12 text editing conditions. Participants were asked to rate the accuracy (Acc) and visual realism (Real) of each editing task based on the provided editing conditions. Ratings were given on a scale of 1 to 5, where 1 represented the best and 5 represented

Table 2
User evaluation experiments are conducted with text-driven generated image methods and hair editing methods. The Acc and Real metrics represent the editing accuracy and visual realism of the generated images, and the lower the Acc and Real scores, the higher the ranking and the better the performance.

	Metrics	
Methods	Acc.	Real.
Text-Driven Methods		
Ours	**1.53**	**1.37**
StyleCLIP	3.57	3.93
TediGAN	4.75	4.88
HairCLIP	2.76	2.33
DeltaEdit	2.39	2.49
Hair Editing Methods		
Ours	**1.32**	**1.49**
LOHO	3.82	3.79
Barbershop	4.86	4.72
HairCLIP	2.41	2.82
CtrlHair	2.56	2.18

the worst. Similarly, for the hair editing methods, we also provided 25 sets of results using randomly selected reference images from the CelebA-HQ dataset. Each set consisted of a source image, a reference image, and the resulting edited image. Participants were asked to rate the editing accuracy and realism of each set of results. Ratings were given on a scale of 1 to 5, where 1 represented the best and 5 represented the worst. The specific evaluation results are shown in Table 2. From the Table 2, it can be observed that our method achieved the best accuracy and realism scores among the two categories of methods, further validating the superiority of our approach.

4.3. Ablation Analysis

In this section, we conducted qualitative and quantitative ablation experiments as well as hair interpolation experiments on the proposed HairManip method, aiming to validate the effectiveness of our network structure and and carefully designed loss functions.

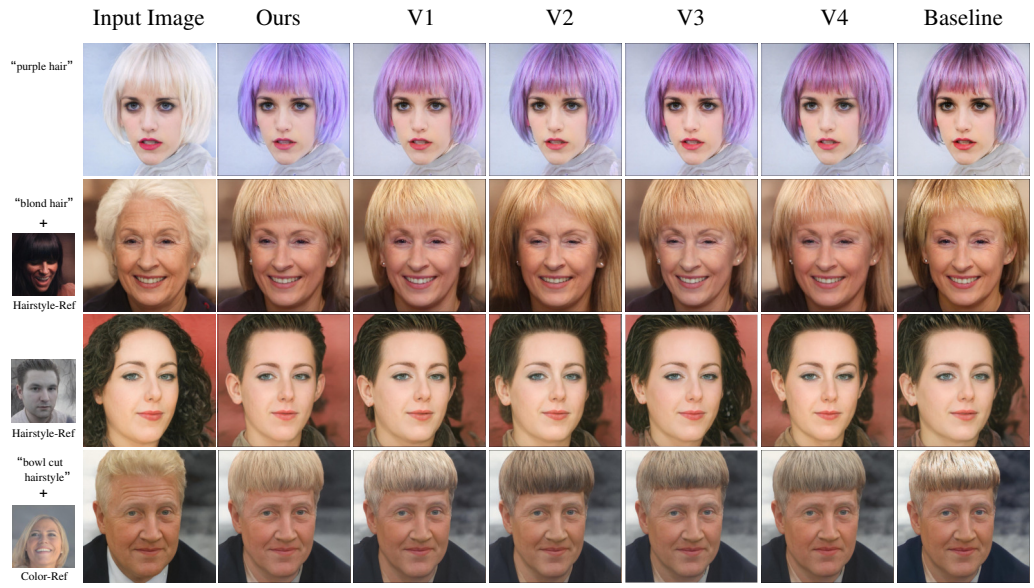

Fig.7. Analysis of ablation experiments and the leftmost component represents the hair editing condition.

Table 3
The specific meaning of each symbol in the analysis of the ablation experiment.

	Hair editing network	Pre-processing module	Smooth L1 loss	Cosine similarity loss
Ours	✔	✔	✔	✔
V1	✔	✘	✘	✘
V2	✘	✔	✘	✘
V3	✘	✘	✔	✘
V4	✘	✘	✘	✔
Baseline	✘	✘	✘	✘

Superiority of each component. Specifically, to evaluate the effectiveness of different components on the generated images, we randomly selected editing methods for either text or images (including editing hairstyle or hair color separately for text or image; editing hairstyle for image while editing hair color for text; editing hairstyle for text while editing hair color for image) to edit the source image. By conducting ablation experiments on each component one by one while keeping other components unchanged, we eval-

Table 4
Quantitative ablation experiments for individual components.

Metrics	Ours	V1	V2	V3	V4	Baseline
PSNR	**31.00**	29.2	30.63	29.04	28.95	28.71
SSIM	**0.92**	0.91	0.90	0.89	0.86	0.89

uated the impact of each component on the generated results. The specific experimental results are shown in Fig.7. The meanings of each symbol are provided in Table 3, and the Baseline refers to the HairCLIP method. From Fig.7, it can be observed that V1 contributes to improving the effectiveness of hairstyle transformation, while V2, V3, and V4 contribute to improving the effectiveness of hair color transfer. To visually assess the effectiveness of the components within the proposed HairManip method, we conducted tests using 15 randomly selected text editing conditions to evaluate the PSNR and SSIM metrics of each component and the Baseline. The specific comparison results are shown in Table 4, which represents the average values across fifteen text editing conditions. The quantitative comparison results in Table 4 once again visually demonstrate the importance of each component to the model and the superiority of our method compared to the Baseline, aligning with the conclusions drawn in Fig.7. In conclusion, the results of quantitative and qualitative ablation experiments demonstrate the effectiveness and necessity of our carefully designed network structure and loss functions. The HairManip model can perform hair editing tasks more excellently, and the generated result images are more natural and visually appealing.

Effectiveness of network structure. We compared the HairManip method with two other models, which were variants derived from HairManip, to validate the effectiveness of the proposed hair editing network. Fig.8 displays the specific comparison results, where the editing condition in the first row is "brown hair", and the editing condition in the second row is "bob cut hairstyle". (a) represents a variant model where the hair color editing sub-network is removed from HairManip. (b) represents a variant model where the hairstyle editing sub-network is removed from HairManip. As can be seen in Fig.8, after removing the hair color editing sub-network, (a) does not transfer hair color effectively, and after removing the hairstyle editing sub-network, (b) does not transfer the hairstyle effectively. In contrast, our method excels at both hairstyle and hair color editing tasks. The experimental results further demonstrate the necessity and effectiveness of our unique

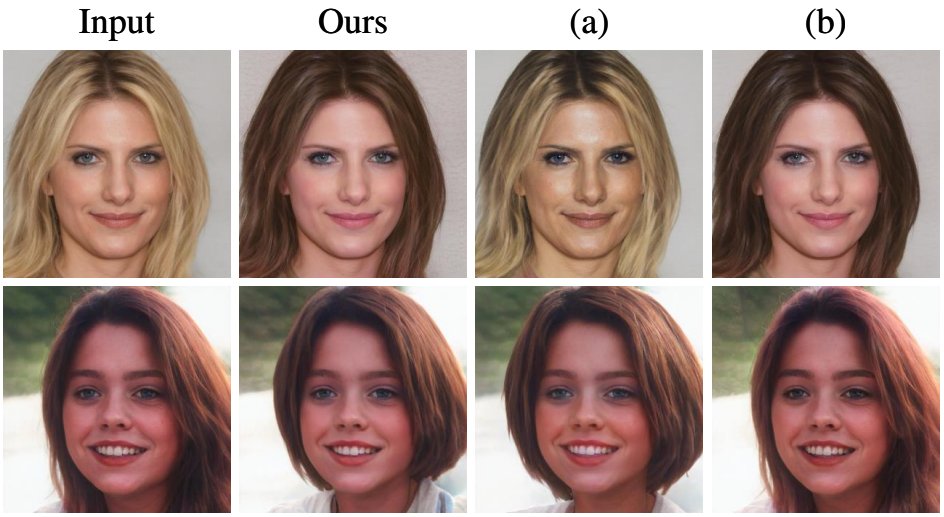

Fig.8. Comparison with two variants of the method. The first row represents editing the hair color of the input image using the text editing condition "brown hair", and the second row represents editing the hairstyle of the input image using the hairstyle editing condition "bob cut hairstyle".

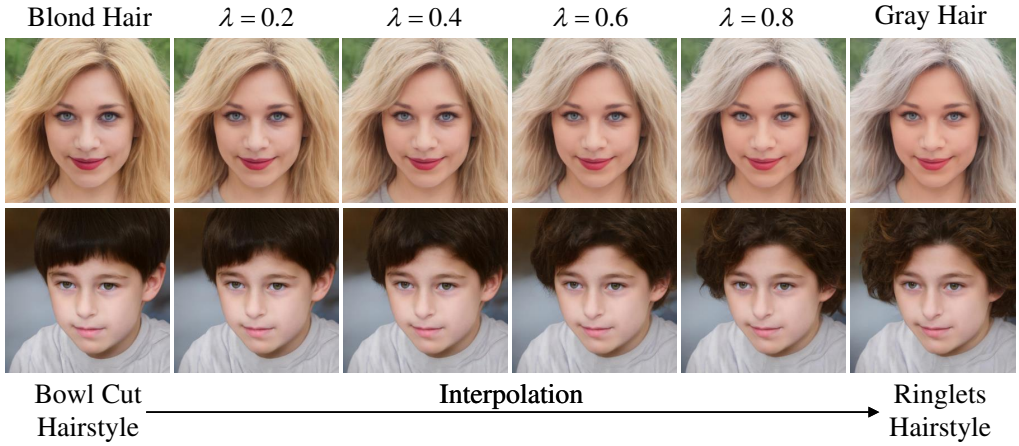

Fig.9. Results of the hair interpolation experiment. We gradually increase the weight λ from 0 to 1, and the interpolated image is transferred continuously from blond hair to gray hair, and from bowl cut hairstyle to ringlets hairstyle.

hairstyle and hair color editing networks.

Hair Interpolation. To verify that the HairManip method can perform

precise and detailed editing of hair attributes, we conducted hair interpolation experiments between two edited latent codes w_A and w_B. Specifically, we generated a new latent code w_O by linearly blending latent codes w_A and w_B, meaning $w_O = \lambda w_A + (1 - \lambda)w_B$. As shown in Fig.9, the interpolated image is gradually converted from blond hair (bowl cut hairstyle) to gray hair (ringlets hairstyle) by controlling λ to gradually increase from 0 to 1 at 0.2 intervals. From Fig.9, it can be observed that by controlling the designated weight λ, we can achieve fine-grained control over hair attributes.

5. Conclusions

In this work, we proposed a new hair editing method called HairManip, which can achieve precise and detailed manipulation of hair through user-inputted text or image editing conditions. To make the generated images more realistic and natural, we naturally divided hair into hairstyle and hair color attributes, and fed them into our separately designed hair editing network for training. Meanwhile, we added corresponding preprocessing modules and loss functions to further improve the efficiency and accuracy of the model. Extensive experimental results show that this method performs outstandingly in hair editing tasks. Compared to similar methods, it can more accurately capture the characteristics and details of hair, and the generated result images are more realistic.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China [grant numbers 61772179], Hunan Provincial Natural Science Foundation of China [grant numbers 2020JJ4152,2022JJ50016], and Postgraduate Scientific Research Innovation Project of Hunan Province [grant numbers CX20221285,CX20231265].

References

[1] X. Guo, M. Kan, T. Chen, S. Shan, GAN with multivariate disentangling for controllable hair editing, in: Computer Vision–ECCV 2022:

17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XV, Springer, 2022, pp. 655–670.

[2] T. Zhentao, C. Menglei, C. Dongdong, L. Jing, C. Qi, Y. Lu, S. Tulyakov, Y. Nenghai, MichiGAN: Multi-input-conditioned hair image generation for portrait editing, ACM Transactions on Graphics 39 (2020) 95.

[3] T. Wei, D. Chen, W. Zhou, J. Liao, Z. Tan, L. Yuan, W. Zhang, N. Yu, HairCLIP: Design your hair by text and reference image, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 18072–18081.

[4] P. Zhu, R. Abdal, J. Femiani, P. Wonka, Barbershop: GAN-based image compositing using segmentation masks, ACM Transactions on Graphics (TOG) 40 (2021) 1–13.

[5] Y. Bao, Y. Qi, A survey of image-based techniques for hair modeling, IEEE Access 6 (2018) 18670–18684.

[6] S. Liu, J. Ye, R. Yu, X. Wang, Slimmable dataset condensation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 3759–3768.

[7] K. Han, Y. Wang, J. Guo, Y. Tang, E. Wu, Vision GNN: An image is worth graph of nodes, Advances in Neural Information Processing Systems 35 (2022) 8291–8303.

[8] Y. Jing, C. Yuan, L. Ju, Y. Yang, X. Wang, D. Tao, Deep graph reprogramming, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 24345–24354.

[9] T. Wang, J.-Y. Zhu, A. Torralba, A. A. Efros, Dataset distillation, arXiv preprint arXiv:1811.10959 (2018).

[10] S. Liu, K. Wang, X. Yang, J. Ye, X. Wang, Dataset distillation via factorization, Advances in Neural Information Processing Systems 35 (2022) 1100–1113.

[11] X. Yang, D. Zhou, S. Liu, J. Ye, X. Wang, Deep model reassembly, Advances in neural information processing systems 35 (2022) 25739–25753.

[12] X. Yang, J. Ye, X. Wang, Factorizing knowledge in neural networks, in: European Conference on Computer Vision, Springer, 2022, pp. 73–91.

[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, Advances in neural information processing systems 27 (2014).

[14] A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks, arXiv preprint arXiv:1511.06434 (2015).

[15] T. Karras, S. Laine, T. Aila, A style-based generator architecture for generative adversarial networks, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 4401–4410.

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

[17] O. Patashnik, Z. Wu, E. Shechtman, D. Cohen-Or, D. Lischinski, StyleCLIP: Text-driven manipulation of StyleGAN imagery, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 2085–2094.

[18] S. Khwanmuang, P. Phongthawee, P. Sangkloy, S. Suwajanakorn, StyleGAN salon: Multi-view latent optimization for pose-invariant hairstyle transfer, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 8609–8618.

[19] H. Liu, Y. Song, Q. Chen, Delving stylegan inversion for image editing: A foundation latent space viewpoint, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 10072–10082.

[20] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, B. Ommer, High-resolution image synthesis with latent diffusion models, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2022, pp. 10684–10695.

[21] X. Yang, D. Zhou, J. Feng, X. Wang, Diffusion probabilistic model made slim, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 22552–22562.

[22] R. Saha, B. Duke, F. Shkurti, G. W. Taylor, P. Aarabi, LOHO: Latent optimization of hairstyles via orthogonalization, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 1984–1993.

[23] C. Xiao, D. Yu, X. Han, Y. Zheng, H. Fu, SketchHairSalon: deep sketch-based hair image synthesis, ACM Transactions on Graphics (TOG) 40 (2021) 1–16.

[24] C.-H. Lee, Z. Liu, L. Wu, P. Luo, MaskGAN: Towards diverse and interactive facial image manipulation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 5549–5558.

[25] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al., Learning transferable visual models from natural language supervision, in: International conference on machine learning, PMLR, 2021, pp. 8748–8763.

[26] U. Kocasari, A. Dirik, M. Tiftikci, P. Yanardag, StyleMC: multi-channel based fast text-guided image generation and manipulation, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2022, pp. 895–904.

[27] Y. Zhu, H. Liu, Y. Song, Z. Yuan, X. Han, C. Yuan, Q. Chen, J. Wang, One model to edit them all: Free-form text-driven image manipulation with semantic modulations, Advances in Neural Information Processing Systems 35 (2022) 25146–25159.

[28] Y. Lyu, T. Lin, F. Li, D. He, J. Dong, T. Tan, DeltaEdit: Exploring text-free training for text-driven image manipulation, 2023.

[29] J. Wang, E. Zhang, S. Cui, J. Wang, Q. Zhang, J. Fan, J. Peng, GGD-GAN: Gradient-guided dual-branch adversarial networks for relic sketch generation, Pattern Recognition 141 (2023) 109586.

[30] P. Esser, R. Rombach, B. Ommer, Taming transformers for high-resolution image synthesis, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 12873–12883.

[31] X. Hou, L. Shen, Z. Ming, G. Qiu, Deep generative image priors for semantic face manipulation, Pattern Recognition 139 (2023) 109477.

[32] A. Khatun, S. Denman, S. Sridharan, C. Fookes, Pose-driven attention-guided image generation for person re-identification, Pattern Recognition 137 (2023) 109246.

[33] S. Gu, J. Bao, H. Yang, D. Chen, F. Wen, L. Yuan, Mask-guided portrait editing with conditional GANs, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 3436–3445.

[34] R. Abdal, P. Zhu, N. J. Mitra, P. Wonka, StyleFlow: Attribute-conditioned exploration of StyleGAN-generated images using conditional continuous normalizing flows, ACM Transactions on Graphics (ToG) 40 (2021) 1–21.

[35] S. Yang, L. Jiang, Z. Liu, C. C. Loy, Pastiche master: exemplar-based high-resolution portrait style transfer, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 7693–7702.

[36] E. Härkönen, A. Hertzmann, J. Lehtinen, S. Paris, GANSpace: Discovering interpretable GAN controls, Advances in Neural Information Processing Systems 33 (2020) 9841–9850.

[37] Y. Alaluf, O. Patashnik, D. Cohen-Or, Only a matter of style: Age transformation using a style-based regression model, ACM Transactions on Graphics (TOG) 40 (2021) 1–12.

[38] C. Xiao, Q. Yang, X. Xu, J. Zhang, F. Zhou, C. Zhang, Where you edit is what you get: Text-guided image editing with region-based attention, Pattern Recognition 139 (2023) 109458.

[39] M. Tao, H. Tang, F. Wu, X.-Y. Jing, B.-K. Bao, C. Xu, DF-GAN: A simple and effective baseline for text-to-image synthesis, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 16515–16525.

[40] W. Xia, Y. Yang, J.-H. Xue, B. Wu, TediGAN: Text-guided diverse face image generation and manipulation, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 2256–2265.

[41] T. Kim, C. Chung, Y. Kim, S. Park, K. Kim, J. Choo, Style your hair: Latent optimization for pose-invariant hairstyle transfer via local-style-aware hair alignment, in: Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XVII, Springer, 2022, pp. 188–203.

[42] O. Tov, Y. Alaluf, Y. Nitzan, O. Patashnik, D. Cohen-Or, Designing an encoder for StyleGAN image manipulation, ACM Transactions on Graphics (TOG) 40 (2021) 1–14.

[43] Z. Liu, https://github.com/switchablenorms/CelebAMask-HQ/tree/master/face_parsing/, 2021 (accessed 16 December 2022).

[44] J. Deng, J. Guo, N. Xue, S. Zafeiriou, ArcFace: Additive angular margin loss for deep face recognition, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 4690–4699.

[45] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, In, ICLR 5 (2015).

Huihuang Zhao received his PhD degree in 2010 from XiDian University. He was a Sponsored Researcher in the School of Computer Science and Informatics, Cardiff University. Now he is a Visiting Professor in National Engineering Laboratory for Robot Visual Perception and Control Technology, Hunan University. His main research interests include machine learning and image processing.

Lin Zhang received the B.S. degree of School of Computer Science and Technology from Hengyang Normal University, China, in 2022. He is currently pursuing his M.S. degree in the School of Computer Science, Hengyang Normal University, Hengyang. His current research interests include computer vision and deep learning.

Paul L. Rosin is currently a professor with the School of Computer Science and Informatics, Cardiff University, U.K. Previous posts include lecturer with the Department of Information Systems and Computing, Brunel University London, U.K., research scientist with the Institute for Remote Sensing Applications, Joint Research Centre, Ispra, Italy, and lecturer with the Curtin University of Technology, Perth, Australia. His research interests include low level image processing, performance evaluation, shape analysis, facial analysis, cellular automata, non-photorealistic rendering, and cultural heritage. For more information, please visit https://users.cs.cf.ac.uk/Paul.Rosin/.

Yukun Lai received his bachelor's degree and PhD degree in computer science from Tsinghua University in 2003 and 2008, respectively. He is currently a Professor in the School of Computer Science & Informatics, Cardiff University. His research interests include computer graphics, geometry processing, image processing and computer vision. He is on the editorial boards of *IEEE Transactions on Visualization and Computer Graphics* and *The Visual Computer*. For more information, please visit https://users.cs.cf.ac.uk/Yukun.Lai/.

Yaonan Wang received the Ph.D. degree in electrical engineering from Hunan University, Changsha, China, in 1994. He was a PostDoctoral Research Fellow with the Normal University of Defence Technology, Changsha, from 1994 to 1995. From 1998 to 2000, he was a Senior Humboldt Fellow in Germany. From 2001 to 2004, he was a Visiting Professor with the University of Bremen, Bremen, Germany. Since 1995, he has been a Professor with the College of Electrical and Information Engineering, Hunan University. His current research interests include robotics and image processing.