Learning to Cartoonize Using White-box Cartoon Representations

Anonymous CVPR submission

Paper ID 6791

Abstract

This paper presents an approach for image cartoonization. By observing the cartoon painting behavior and consulting artists, we propose to separately identify three white-box representations from images: the surface representation that contains smooth surface of cartoon images, the structure representation that refers to the sparse color-blocks and flatten global content in the celluloid style workflow, and the texture representation that reflects high-frequency texture, contours and details in cartoon images. A Generative Adversarial Network (GAN) framework is used to learn the extracted representations and to cartoonize images.

The learning objectives of our method are separately based on each extracted representations, making our framework controllable and adjustable. This enables our approach to meet artists' requirements in different styles and diverse use cases. Qualitative comparisons and quantitative analyses, as well as user studies, have been conducted to validate the effectiveness of this approach, and our method outperforms previous methods in all comparisons. Finally, the ablation study demonstrates the influence of each component in our framework.

1. Introduction

Cartoon is a popular artistic form that has been widely applied in diverse scenes. Modern cartoon animation workflows allow artists to use a variety of sources to create contents. Some famous products have been created by turning real-world photography into usable cartoon scene materials, where the process is called image cartoonization.

The variety of cartoon styles and use cases require task-specific assumptions or prior knowledge to develop usable algorithms. For example, some cartoon workflows pay more attention to global palette themes, but the sharpness of lines is a secondary issue. In some other workflows, sparse and clean color blocks play a dominant role in artistic expression, but the themes are relatively less emphasized.

These variants pose non-trivial challenges to black-box end-to-end models, *e.g.*, [21, 50, 5], when faced with diverse demands of artists in different use cases, and simply change the training dataset does not help. Especially, CartoonGAN [5] is designed for image cartoonization, in which a GAN framework with a novel edge loss is proposed, and achieves good results in certain cases. But using black-box model to directly fit the training data decreased its generality and stylization quality, causing bad cases in some situations.

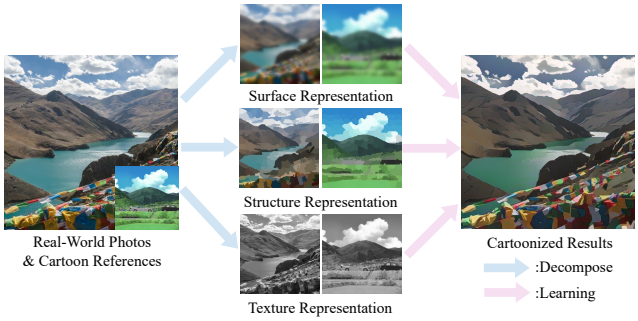


Figure 1. A simple illustration of our method. Images are decomposed into three cartoon representations, which guide the network optimization to generate cartoonized real-world photos.

To address the above-mentioned problems, we made extensive observations on human painting behaviors and cartoon images of different styles, and also consulted several cartoon artists. According to our observations, which is shown in Figure 2, we propose to decompose images into several cartoon representations, and list them as follows:

Firstly, we extract the *surface* representation to represent the smooth surface of images. Given an image $\boldsymbol{I} \in \mathbb{R}^{W\times H\times 3}$, we extract a weighted low-frequency component $\boldsymbol{I}_{sf} \in \mathbb{R}^{W\times H\times 3}$, where the color composition and surface texture are preserved with edges, textures and details ignored. This design is inspired by the cartoon painting behavior where artists usually draw composition drafts before the details are retouched, and is used to achieve a flexible and learnable feature representation for smoothed surfaces.

Secondly, the *structure* representation is proposed to effectively seize the global structural information and sparse color blocks in celluloid cartoon style. We extract a segmentation map from the input image $\boldsymbol{I} \in \mathbb{R}^{W\times H\times 3}$ and then apply an adaptive coloring algorithm on each segmented regions to generate the structure representation $\boldsymbol{I}_{st} \in \mathbb{R}^{W\times H\times 3}$. This representation is motivated to em-

Figure 2. Cartoon images with some typical and common features: 1. Global structures are composed of sparse color blocks and clear boundaries; 2. Details are outlines by sharp and clear contours; 3. Surfaces are flat and smooth.

ulate the celluloid cartoon style, which is featured by clear boundaries and sparse color blocks. The structure representation is of significance for generating the sparse visual effects, as well as for our method to be embedded in the celluloid style cartoon workflow.

Thirdly, we use the *texture* representation to contain painted details and edges. The input image $\boldsymbol{I} \in \mathbb{R}^{W \times H \times 3}$ is converted to a single-channel intensity map $\boldsymbol{I}_t \in \mathbb{R}^{W \times H \times 1}$, where the color and luminance are removed and relative pixel intensity is preserved. This feature representation is motivated by a cartoon painting method where artists firstly draw a line sketch with contours and details, and then apply color on it. It guides the network to learn the high-frequency textural details independently with the color and luminance patterns excluded.

It is worth noting that the extractions of the surface representation and the texture representation are differentiable, which enables the cartooniaztion problem to be optimized end-to-end within a Generative Neural Networks (GAN) framework. Moreover, the separated cartoon representations make our method scalable and controllable for practical use cases and easy to meet diversified artistic demands with task-specific fine-tuning.

We apply our method on a variety of real-world photos on diverse scenes and generate their cartoonized counterparts in different styles. Experimental results show that our method can generate images with harmonious color, pleasing artistic styles, sharp and clean boundaries, and significantly fewer artifacts as well. We also show that our method outperforms previous state-of-the-art methods through qualitative experiments, quantitative experiments, and user studies. Finally, ablation studies are also conducted to illustrate the influence of each cartoon representations. To conclude, our contributions are summarized as follows:

- We propose three cartoon representations based on our observation of cartoon painting behavior: the surface representation, the structure representation, and the texture representation. Image processing modules are then introduced to extract each representation.
- A GAN-based controllable image cartoonization framework is optimized with the guide of extracted representations. Users can adjust the style of model output by balancing the weight of each representation.
- Extensive experiments have been conducted to show that our method can generate high-quality cartoonized images. Our method outperforms existing methods in qualitative comparison, quantitative comparison, and user preference.

2. Related Work

2.1. Image Smoothing and Surface Extraction

Image smoothing [39, 15, 11, 30, 4] is an extensively studied topic. Early methods are mainly filtering based [39, 15] and optimization-based methods later became popular. Farbman *et al*. [11] utilized weighted least square to constrain the edge-preserving operator, Min *et al*. [30] solved global image smoothing by minimizing a quadratic energy function, and Bi *et al*. [4] proposed a L1 transformation for image smoothing and flattening problem. Xu and Fan *et al*. [46, 9, 10] introduced end-to-end networks for image smoothing. In this work, we adapt a differentiable guided filter [44] to extract smooth, cartoon-like surface from images, enabling our model to learn structure-level composition and smooth surface that artists have created in cartoon artworks.

2.2. Superpixel and structure Extraction

Super-pixel segmentation [12, 32, 31, 2, 25] groups spatially connected pixels in an image with similar color or gray level. Some popular super-pixel algorithms [12, 32, 31] are graph-based, treating pixels as nodes and similarity between pixels as edges in a graph. Gradient ascent based algorithms [6, 42, 2, 25] initialize the image with rough clusters and iteratively optimize the clusters with gradient ascent until convergence. In this work, we follow the felzenszwalb algorithm [12] to develop a cartoon-oriented segmentation method to achieve a learnable structure representation. This representation is significant for deep models to seize global content information and produce practically usable results for celluloid style cartoon workflows.

2.3. Non-photorealistic Rendering and Neural Style Transfer

Non-photorealistic Rendering (NPR) methods represent image content with artistic styles, such as pencil sketching [45, 29], paints [13, 21], watercolor [41, 7]. Image cartoonization is also extensively studied from filtering based method [36] to end-to-end neural network [5], covering the use cases of photos [5], videos [43], and portraits [34, 47].

Neural Style Transfer methods [13, 21, 8, 17, 27] are popular among NPR algorithms, which synthesis images with artistic style by combining the content of one image and the style of another image. Gatys *et al*. [13] jointly optimized a style loss and a content loss to generate stylize images with a style-content image pair. Johnson *et al*. [21] accelerated this process by training an end-to-end network with perception

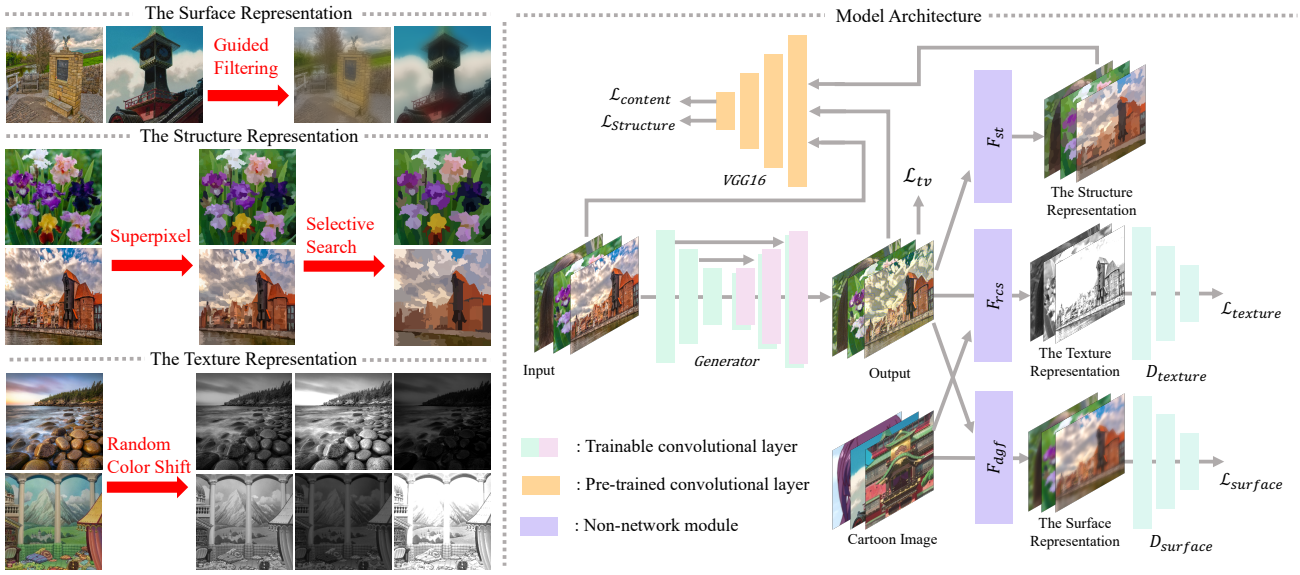


Figure 3. Our proposed image cartoonization system

loss. Several works [8, 17, 27] later proposed models to synthesis images with different styles. Our method, different from the above-mentioned methods that take a single image as a style, learns the cartoon data distribution from a set of cartoon images. This allows our model to synthesis high-quality cartoonized images on diverse use cases.

2.4. Generative Adversarial Networks

Generative Adversarial Network(GAN) [14] is a state-of-the-art generative model that can generate data with the same distribution of input data by solving a min-max problem between a generator network and a discriminator network. It is powerful in image synthesis by forcing the generated images to be indistinguishable from real images. GAN has been widely used in conditional image generation tasks, such as image inpainting [33], style transfer [35], image cartoonization [5], image colorization [48]. In our method, we adopt adversarial training architecture and use two discriminators to enforce the generator network to synthesis images with the same distribution as the target domain.

2.5. Image-to-Image Translation

Image-to-Image Translation [20, 18, 24, 50] tackles the problem of translating images from a source domain to another target domain. Its applications include image quality enhancement [19], stylizing photos into paints [21, 35], cartoon images [5] and sketches [26], as well as grayscale photo colorization [49] and sketch colorizaiton [48]. Recently, bi-directional models are also introduced for inter-domain translation. Zhu *et al*. [50] performs transformation of unpaired images(i.e. summer to winter, photo to paints).

In this paper, we adopt an unpaired image-to-image translation framework for image cartoonization. Unlike previous black-box models that guide network training with loss terms, we decompose images into several representations, which enforces network to learn different features with separate objectives, making the learning process controllable and tunable.

3. Proposed Approach

We show our proposed image cartoonizaiton framework in Figure 3. Images are decomposed into the surface representation, the structure representation, and the texture representations, and three independent modules are introduced to extract corresponding representations. A GAN framework with a generator G and two discriminators D_s and D_t is proposed, where D_s aims to distinguish between surface representation extracted from model outputs and cartoons, and D_t is used to distinguish between texture representation extracted from outputs and cartoons. Pre-trained VGG network [37] is used to extract high-level features and to impose spatial constrain on global contents between extracted structure representations and outputs, and also between input photos and outputs. Weight for each component can be adjusted in the loss function, which allows users to control the output style and adapt the model to diverse use cases.

3.1. Learning From the Surface Representation

The surface representation imitates cartoon painting style where artists roughly draw drafts with coarse brushes and have smooth surfaces similar to cartoon images. To smooth images and meanwhile keep the global semantic structure, a differentiable guided filter is adopted for edge-preserving filtering. Denoted as $\mathcal{F}_{dgf}$, it takes an image $\boldsymbol{I}$ as input and itself as guide map, and returns extracted surface representation $\mathcal{F}_{dgf}(\boldsymbol{I}, \boldsymbol{I})$ with textures and details removed.

A discriminator D_s is introduced to judge whether model outputs and reference cartoon images have similar surfaces, and guide the generator G to learn the information stored in the extracted surface representation. Let $\boldsymbol{I_p}$ denote the input photo and $\boldsymbol{I_c}$ denote the reference cartoon images, we formulate the surface loss as:

$$\begin{aligned}\mathcal{L}_{surface}(G, D_s) = &logD_s(\mathcal{F}_{dgf}(\boldsymbol{I}_c, \boldsymbol{I}_c)) \\ &+ log(1 - D_s(\mathcal{F}_{dgf}(G(\boldsymbol{I}_p), G(\boldsymbol{I}_p))))\end{aligned} \tag{1}$$

3.2. Learning From the Structure representation

The Structure representation emulates flatten global content, sparse color blocks and clear boundaries in celluloid style cartoon work-flow. We at first use felzenszwalb algorithm to segment images into separate regions. As superpixel algorithms only consider the similarity of pixels and ignore semantic information, we further introduce selective search [40] to merge segmented regions and extract a sparse segmentation map $\boldsymbol{T}$. Denoted as $\mathcal{F}_{seg}$, the segmentation algorithm is described in Algorithm 1. For the similarity metric, we consider both pixel value and spatial coordinate, and defined it as the euclidean distance in (x, y, l, a, b) 5 dimension space, where x and y represent coordinates and l, a, b represent pixel value in LAB color space.

Algorithm 1 Segmentation for the Structure Representation

Input: Input image $\boldsymbol{I}$, number of regions $\boldsymbol{N}$
Output: Extracted segmentation map $\boldsymbol{T} = \mathcal{F}_{seg}(\boldsymbol{I})$

```
Get superpixel segmentation: T_sp = F_felzenszwalb(I);
Initialize similarity set S;
for each neighboring superpixel pair (t_i, t_j) in T_sp do
    Calculate similarity s(t_i, t_j);
    S = S ∪ s(t_i, t_j);
end for
Initialize segmentation map T = T_sp;
while number of superpixels > N do
    Get highest similarity s(t_i, t_j) = max(S);
    Merge corresponding superpixels t_k = s_i ∪ s_j;
    Removing similarities regarding t_i and t_j from T;
    Calculating similarity set S_k of t_k;
    S = S ∪ S_k, T = T ∪ t_k;
end while
return Extracted segmentation map T
```

Standard superpixel algorithms color each segmented region with average pixel value, which lowers global contrast, darkens images and causes hazing effect on the final results (shown in Figure 5(a)). We thus propose an adaptive coloring algorithm, and formulate it in Equation 2, where we find $\gamma_1 = 20$, $\gamma 2 = 40$ and $\mu = 1.2$ generate good results. The results of our coloring algorithm are shown in Figure 4, this effectively enhances the contrast of images, and reduces hazing effect in the final results (shown in Figure 5(b)).

$$\boldsymbol{S}_{i,j} = (\theta_1 * \bar{\boldsymbol{S}} + \theta_2 * \tilde{\boldsymbol{S}})^{\mu} \tag{2}$$

(a) (b)

Figure 4. Adaptive coloring algorithm. (a) shows average colored segmentation map and (c) shows adaptively colored segmentation map, which are brighter and have higher contrast.

(a) W/O adaptive coloring (b) With adaptive coloring

Figure 5. Removal of hazing effect. (a) shows results trained with average colored structure representations, which suffer from hazing effects. (b) shows results trained with adaptively colored structure representations, which are brighter and free from hazing effects.

$$(\theta_1, \theta_2) = \begin{cases} (0, 1) & \sigma(\boldsymbol{S}) < \gamma_1, \\ (0.5, 0.5) & \gamma_1 < \sigma(\boldsymbol{S}) < \gamma_2, \\ (1, 0) & \gamma_2 < \sigma(\boldsymbol{S}). \end{cases}$$

We use high-level features extracted by pre-trained VGG16 network [37] to enforce spatial constrain between our results and extracted structure representation. Let $\mathcal{F}_{st}$ denote the structure representation extraction, the structure loss $\mathcal{L}_{structure}$ is formulated as:

$$\mathcal{L}_{structure} = \|VGG_n(G(\boldsymbol{I}_p)) - VGG_n(\mathcal{F}_{st}(G(\boldsymbol{I}_p)))\| \tag{3}$$

3.3. Learning From the Textural Representation

The high-frequency features stored in cartoon images are desired as learning objectives, but luminance and color information make it easy to distinguish between cartoon images and real-world photos. We thus propose a random color shift algorithm $\mathcal{F}_{rcs}$ to extract single-channel texture representation from color images, which retains high-frequency texture information and helps avoid the influence of color and luminance.

$$\mathcal{F}_{rcs}(\boldsymbol{I}_{rgb}) = (1-\alpha)(\beta_1 * \boldsymbol{I}_r + \beta 2 * \boldsymbol{I}_g + \beta_3 * \boldsymbol{I}_b) + \alpha * \boldsymbol{Y} \tag{4}$$

In Equation 4, $\boldsymbol{I}_{rgb}$ represents 3-channel RGB color images, $\boldsymbol{I}_r, \boldsymbol{I}_g$ and $\boldsymbol{I}_b$ represent three color channels, and $\boldsymbol{Y}$ represents standard grayscale image converted from RGB color image. We set $\alpha = 0.8$, β_1, β_2 and $\beta_3 \sim U(-1, 1)$. As is shown in Figure 3, the random color shift can generate random intensity maps with luminance and color information removed. A discriminator D_t is introduced to distinguish texture representations extracted from model outputs and cartoons, and guide the generator to learn the clear contours and fine textures stored in the texture representations.

$$\begin{aligned}\mathcal{L}_{texture}(G, D_t) = &logD_t(\mathcal{F}_{rcs}(\boldsymbol{I}_c)) \\ &+ log(1 - D_t(\mathcal{F}_{rcs}(G(\boldsymbol{I}_p))))\end{aligned} \tag{5}$$

3.4. Full model

Our full model is a GAN based framework with one generator and two discriminators. It is jointly optimized with features learned from three cartoon representations and could be formulated in Equation 6. By adjusting and balancing λ_1, λ_2, λ_3 and λ_4, it could be easily adapted to various applications with different artistic style.

$$\begin{aligned}\mathcal{L}_{total} &= \lambda_1 * \mathcal{L}_{surface} + \lambda_2 * \mathcal{L}_{texture} \\ &+ \lambda_3 * \mathcal{L}_{structure} + \lambda_4 * \mathcal{L}_{content} + \lambda_5 * \mathcal{L}_{tv}\end{aligned} \tag{6}$$

The total-variation loss $\mathcal{L}_{tv}$ [3] is used to impose spatial smoothness on generated images. It also reduces high-frequency noises such as salt-and-pepper noise. In Equation 7, *H*, W, C represent spatial dimensions of generated images.

$$\mathcal{L}_{tv} = \frac{1}{H * W * C} \| \bigtriangledown_x (G(\boldsymbol{I}_p)) + \bigtriangledown_y (G(\boldsymbol{I}_p)) \| \tag{7}$$

The content loss $\mathcal{L}_{content}$ is used to ensure that the cartoonized results and input photos are semantically invariant, and the sparsity of $\mathcal{L}_1$ norm allows for local features to be cartoonized. Similar to the structure loss, it is calculated on pre-trained VGG16 feature space:

$$\mathcal{L}_{content} = \| VGG_n(G(\boldsymbol{I}_p)) - VGG_n(\boldsymbol{I}_p) \| \tag{8}$$

3.5. Post processing and Style Interpolation

To remove undesired high-frequency noises caused by adversarial training and maintain good perceptual quality, we use a differentiable guided filter $\mathcal{F}_{dgf}$ for post processing during training. Denote the network input as $\boldsymbol{I}_{in}$ and network output as $\boldsymbol{I}_{out}$, we formulated the post-processing in Equation 9, where $\boldsymbol{I}_{in}$ is used as guide map:

$$\boldsymbol{I}_{out} = \mathcal{F}_{dgf}(\boldsymbol{I}_{in}, G(\boldsymbol{I}_{in})) \tag{9}$$

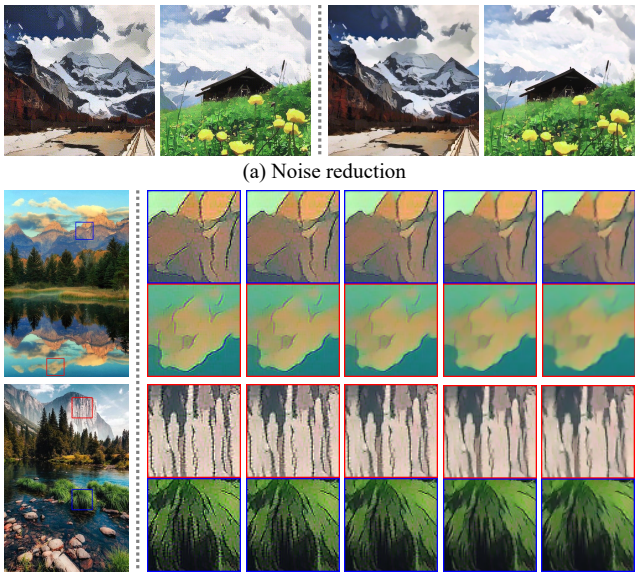
We further find that, by interpolating the $\boldsymbol{I}_{out}$ and $G(\boldsymbol{I}_{in})$, the sharpness of edges and details could be effectively tuned without having to fine-tune the network parameters. The interpolation is formulated in Equation 10, and the results of post-processing are shown in Figure 6.

$$\boldsymbol{I}_{interp} = \delta * \boldsymbol{I}_{out} + (1 - \delta) * G(\boldsymbol{I}_{in}) \tag{10}$$

4. Experimental Results

4.1. Experimental Setup

Implementation. We implement our GAN based framework with TensorFlow [1]. The generator and discriminator architectures are described in the supplementary material. Patch discriminator [20] is adopted to simplify calculation and enhance discriminative capacity. We use Adam [23] optimization algorithm to optimize both networks. Learning rate and batch size are set to $2 * 10^{-4}$ and 16 during training.



(a) Noise reduction

(b) Image interpolation

Figure 6. Illustration of post-processing. (a) Our post processing can effectively turn noisy images(left) to clean images(right). (b) The sharpness of details could be adjusted. δ = 0.0, 0.25, 0.5, 0.75, 1.0 from left to right. Zoom in for details.

For stability, we at first pre-train the generator with the content loss for 50000 iterations, and then jointly optimize the GAN based framework. Training is stopped when reaching 100000 iterations or on convergency. All results shown in this paper, unless specially mentioned, are generated with $\lambda_1 = 1, \lambda_2 = 10, \lambda_3 = 2 * 10^3, \lambda_4 = 2 * 10^3, \lambda_5 = 10^4$.

Dataset. Both human face and landscape data are collected for generalization on diverse real-world scenes. For real-world photos, we collect 10000 images from the FFHQ dataset [22] for the human face and 5000 images from the dataset in [50] for landscape. For cartoon images, we collect 10000 images from animations for the human face and 10000 images for landscape. Producers of collected animations include Kyoto animation, P.A.Works, Shinkai Makoto, Hosoda Mamoru, and Miyazaki Hayao. For validation set, we collect 3011 animation images and 1978 real-world photos. Images used in user study are collected from the Internet and Microsoft COCO [28] dataset. During training, all images are resized to 256*256 resolution, and face images are feed only once in every 5 iterations.

Previous Methods. We compare our method with three algorithms that represent Neural Style Transfer [21], Image-to-Image Translation [50] and Image Cartoonization [5] respectively. For fair comparisons, We train CycleGAN and Fast Neural Style with the same training data and feeding schedule as our method, and set all the hyper-parameters by default or follow the author's recommendation. For CartoonGAN, we directly use the four official pre-trained models.

Evaluation metrics. In qualitative experiments, we present results with details of four different methods and

(a) Landscape (b) City views (c) People (d) Animals (e) Plants

Figure 7. Results of our method in different scenes. Zoom in for details

(a) Enhance Texture Representation (b) Enhance Surface Representation (c) Enhance Structure Representation

Figure 8. Output quality could be controlled by adjusting weight of each representation. Zoom in for details.

Methods	[21]	[5]	[50]	Ours
LR, CPU(ms)	639.31	1947.97	1332.66	**64.66**
LR, GPU(ms)	16.53	13.76	9.22	**3.58**
HR, GPU(ms)	48.96	148.02	106.82	**17.23**
Parameter(M)	1.68	11.38	11.13	**1.48**

Table 1. Performance and model size comparison, LR represents 256*256 resolution, while HR represents 720*1280 resolution

original images, as well as qualitative analysis. In quantitative experiments, we use Frechet Inception Distance (FID) [16] to evaluate the performance by calculating the distance between source image distribution and target image distribution. In the user study, candidates are asked to rate results of different methods between 1 to 5 in cartoon quality and overall quality, higher scores mean better quality.

Time Performance and Model Size. Speeds of four methods are compared on Intel i7-8750h CPU and Nvidia Tesla V-100 GPU. Shown in Table 1, our model is the fastest among four methods on all devices and all resolutions, and has the smallest model size. It is worth mentioning that our model can process a 720*1280 image on GPU within only 17.23ms, which enables it for real-time High-Resolution video processing task at the speed of more than 50 FPS.

Generality to diverse use cases. We apply our model on diverse real-world scenes, including natural landscape, city views, people, animals, and plants, and show the results in Figure 7. More examples of different styles and diverse use cases are shown in the supplementary material.

4.2. Validation of Cartoon Representations.

To validate our proposed cartoon representations reasonable and effective, a classification experiment and a quantitative experiment based on FID are conducted, and the results

No.	Surface	Structure	Texture	Original
Acc	0.8201	0.6342	0.8384	0.9481
FID	113.57	112.86	112.71	162.89

Table 2. Classification accuracy and FID evaluation of our proposed cartoon representation.

are shown in Table 2. We train a binary classifier on our training dataset to distinguish between real-world photos and cartoon images. The classifier is designed by adding a fully-connected layer to the discriminator in our framework. The trained classifier is then evaluated on the validation set to validate the influence of each cartoon representation.

We find the extracted representations successfully fool the trained classifier, as it achieves lower accuracy in all three extracted cartoon representations compared to the original images. The calculated FID metrics also support our proposal that cartoon representations help close the gap between real-world photos and cartoon images, as all three extracted cartoon representations have smaller FID compared to the original images.

4.3. Illustration of Controllability

As is shown in Figure 8, the style of cartoonized results could be adjusted by turning the weight of each representation in the loss function. Increase the weight of texture representation adds more details in the images. By setting $\lambda_2 = 20$, rich details such as plants on the wall and the contours of the house are preserved. This is because it regulates dataset distributions and enhances high-frequency details stored in texture representation. Smoother textures and fewer details are generated with a higher weight of surface representation. We setting $\lambda_1 = 2$ and $\lambda_2 = 10$, the details of the grassland and the mountain are smoothed. The reason

(a) Original Photo (b) Fast Neural Style (c) CycleGAN (d) CartoonGAN (e) Ours

Figure 9. Qualitative comparison of our method and previous methods. The Hosoda style, Shinkai style, Hayao style and Paprika style of CartoonGAN [5] are shown in row 1-4 respectively. Zoom in for details

Methods	Photo	Fast Neural Style [21]	CycleGAN [50]	CartoonGAN [5], average	Ours
FID to Cartoon	162.89	146.34	141.50	130.38	**101.31**
FID to Photo	N/A	103.48	122.12	75.28	**28.79**
Methods	Shinkai style of [5]	Hosoda style of [5]	Hayao style of [5]	Paprika style of [5]	Ours
FID to Cartoon	135.94	130.76	127.35	127.05	**101.31**
FID to Photo	37.96	58.13	86.48	118.56	**28.79**

Table 3. Performance evaluation based on FID

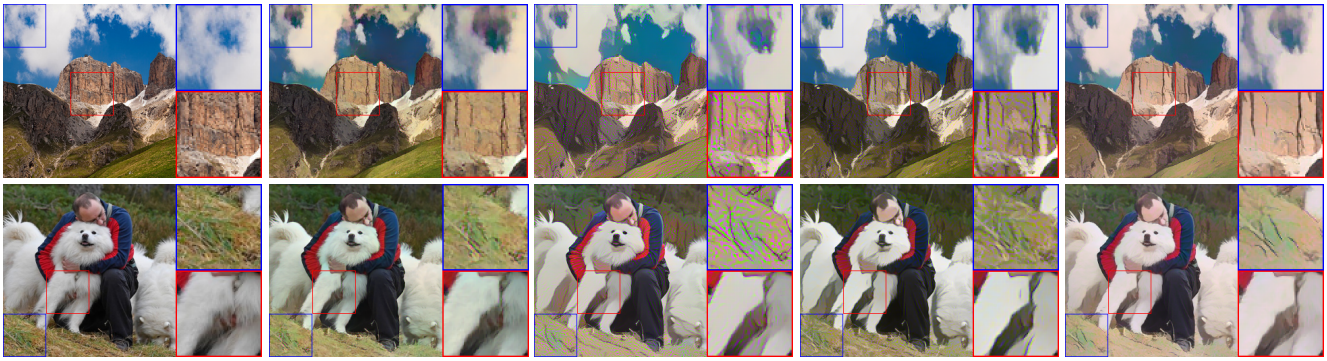
is that guided filtering smooths training samples and reduces densely textured patterns. To get more abstract and sparse features, we can increase the weight of structure representation. By setting $\lambda_3 = 4 * 10^3$ and $\lambda_4 = 6 * 10^3$, the details of the highway are abstracted into sparse color blocks. This is because the selective search algorithm flattens the training data and abstract them into structure representations. To conclude, unlike black-box models, our white-box method is controllable and can be easily adjusted.

4.4. Qualitative Comparison

Comparisons between our method and previous methods are shown in Figure 9. The white-box framework helps keep global color harmonious. While all previous methods cause unrealistic color distortion like the green sky and brown clouds, our method effectively prevents improper color modification, such as clouds, sky, and mountains. Cartoon representations also help generate smooth features and sharp boundaries. CycleGAN causes distortions in sky and clouds, CartoonGAN generates distorted building details and messy mountain textures, whereas our method generates details with clear contours and smooth textures such as castles, mountains, and clouds. Lastly, our proposed representations effectively reduce high-frequency artifacts. Fast Neural Style suffers from pepper-and-salt noises and CycleGAN causes checkerboard artifacts, but our method generates noise-free images in all shown cases. To conclude, our method outperforms previous methods in generating images with harmonious color, cartoon-like features, clear boundaries, and fewer noises. More examples are shown in the supplementary materials.

4.5. Quantitative Evaluation

Frechet Inception Distance (FID) [16] is a wildly-used metric to quantitatively evaluate the synthesizing performance of generative models. Pre-trained Inception-V3 model [38] is used to extract high-level features of images and calculate the distance between two image distributions. We use FID to evaluate the performance of previous methods and our method. As CartoonGAN models have not been



(a) Original Photo (b) W/O Texture Representation (c) W/O Cell Representation (d) W/O Surface Representation (e) Full Model

Figure 10. Ablation study by removing each component

Methods	[21]	[5]	[50]	Ours
Cartoon quality, mean	2.347	2.940	2.977	**4.017**
Cartoon quality, std	1.021	1.047	1.437	**0.962**
Overall quality, mean	2.38	2.937	2.743	**3.877**
Overall quality, std	0.993	1.046	1.321	**0.982**

Table 4. Result of User study, higher score means better quality. Row 1 and 2 represent the mean and standard error of Cartoon quality score, row 3 and 4 represent the mean and standard error of Overall quality score.

trained on human face data, for fair comparisons, we only calculate FID on scenery dataset.

As is shown in Table 3, our method generates images with smallest FID to cartoon image distribution. This proves that results generated by our method are most similar to the cartoon images. Output of our method also has smallest FID to real-world photo distribution, indicating that our method loyally preserves image content information.

4.6. User Study

The quality of Image cartoonization is highly subjective and greatly influenced by individual preference. We conducted user studies to show how users evaluate our method and previous methods. The user study involves 30 images, each processed by our proposed method and three previous methods. 10 candidates are asked to rate every image between 1-5 in 2 dimensions, following the criterion below:
Cartoon quality: users are asked to evaluate how similar are the shown images and cartoon images.
Overall quality: users are asked to evaluate whether there are color shifts, texture distortions, high-frequency noises, or other artifacts they dislike on the images.

We collect 1200 scores in total, and show the average score and standard error of each algorithm Table 4. Our method outperforms previous methods in both cartoon quality and overall quality, as we get higher scores in both criterions. This is because our proposed representations effectively extracted cartoon features, enabling the network to synthesis images with good quality. The synthesis quality of our method is also the most stable, as our method has the smallest standard error in both criterions. The reason is that our method is controllable and can be stabilized by balancing different components. Images used in user study are shown in the supplementary materials.

4.7. Analysis of Each Components

We show the results of ablation studies in Figure 10. Ablating the texture representation causes messy details. Shown in Figure 10(a), irregular textures on the grassland and the dog's leg remains. This is due to the lack of high-frequency stored in the surface representation, which deteriorates the model's cartoonization ability. Ablating the structure representation causes high-frequency noises in Figure 10(b). Severe pepper-and-salt appear on the grassland and the mountain. This is because the structure representation flattened images and removed high-frequency information. Ablating the surface representation causes both noise and messy details. Unclear edges of the cloud and noises on the grassland appear in Figure 10(c). The reason is that guided filtering suppresses high-frequency information and preserves smooth surfaces. As a comparison, the results of our full model are shown in Figure 10(d), which have smooth features, clear boundaries, and much less noise. In conclusion, all three representations help improve the cartoonizaiton ability of our method.

5. Conclusion

In this paper, we propose a white-box controllable image cartoonization framework based on GAN, which can generate high-quality cartoonized images from real-world photos. Images are decomposed into three cartoon representations: the surface representation, the structure representation, and the texture representation. Corresponding image processing modules are used to extract three representations for network training, and output styles could be controlled by adjusting the weight of each representation in the loss function. Extensive quantitative and qualitative experiments, as well as user studies, have been conducted to validate the performance of our method. Ablation studies are also conducted to demonstrate the influence of each representation.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th Symposium on Operating Systems Design and Implementation*, pages 265–283, 2016. 5

[2] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012. 2

[3] Hussein A Aly and Eric Dubois. Image up-sampling using total-variation regularization with a new observation model. *IEEE Transactions on Image Processing*, 14(10):1647–1659, 2005. 5

[4] Sai Bi, Xiaoguang Han, and Yizhou Yu. An l 1 image transform for edge-preserving smoothing and scene-level intrinsic decomposition. *ACM Transactions on Graphics (TOG)*, 34(4):78, 2015. 2

[5] Yang Chen, Yu-Kun Lai, and Yong-Jin Liu. Cartoongan: Generative adversarial networks for photo cartoonization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9465–9474, 2018. 1, 2, 3, 5, 6, 7, 8

[6] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (5):603–619, 2002. 2

[7] Cassidy J Curtis, Sean E Anderson, Joshua E Seims, Kurt W Fleischer, and David H Salesin. Computer-generated watercolor. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 421–430. ACM Press/Addison-Wesley Publishing Co., 1997. 2

[8] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. *arXiv preprint arXiv:1610.07629*, 2016. 2, 3

[9] Qingnan Fan, Jiaolong Yang, Gang Hua, Baoquan Chen, and David Wipf. A generic deep architecture for single image reflection removal and image smoothing. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3238–3247, 2017. 2

[10] Qingnan Fan, Jiaolong Yang, David Wipf, Baoquan Chen, and Xin Tong. Image smoothing via unsupervised learning. In *SIGGRAPH Asia 2018 Technical Papers*, page 259. ACM, 2018. 2

[11] Zeev Farbman, Raanan Fattal, Dani Lischinski, and Richard Szeliski. Edge-preserving decompositions for multi-scale tone and detail manipulation. In *ACM Transactions on Graphics (TOG)*, volume 27, page 67. ACM, 2008. 2

[12] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004. 2

[13] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015. 2

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. 3

[15] Kaiming He, Jian Sun, and Xiaoou Tang. Guided image filtering. In *European Conference on Computer Vision*, pages 1–14. Springer, 2010. 2

[16] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017. 6, 7

[17] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017. 2, 3

[18] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 172–189, 2018. 3

[19] Andrey Ignatov, Nikolay Kobyshev, Radu Timofte, Kenneth Vanhoey, and Luc Van Gool. Wespe: weakly supervised photo enhancer for digital cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 691–700, 2018. 3

[20] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1125–1134, 2017. 3, 5

[21] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016. 1, 2, 3, 5, 6, 7, 8

[22] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019. 5

[23] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[24] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Diverse image-to-image translation via disentangled representations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 35–51, 2018. 3

[25] Alex Levinshtein, Adrian Stere, Kiriakos N Kutulakos, David J Fleet, Sven J Dickinson, and Kaleem Siddiqi. Turbopixels: Fast superpixels using geometric flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2290–2297, 2009. 2

[26] Yijun Li, Chen Fang, Aaron Hertzmann, Eli Shechtman, and Ming-Hsuan Yang. Im2pencil: Controllable pencil illustration from photographs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1525–1534, 2019. 3

[27] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal style transfer via feature transforms. In *Advances in Neural Information Processing Systems*, pages 386–396, 2017. 2, 3

[28] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014. 5

[29] Cewu Lu, Li Xu, and Jiaya Jia. Combining sketch and tone for pencil drawing production. In *Proceedings of the Symposium on Non-Photorealistic Animation and Rendering*, pages 65–73. Eurographics Association, 2012. 2

[30] Dongbo Min, Sunghwan Choi, Jiangbo Lu, Bumsub Ham, Kwanghoon Sohn, and Minh N Do. Fast global image smoothing based on weighted least squares. *IEEE Transactions on Image Processing*, 23(12):5638–5653, 2014. 2

[31] Alastair P Moore, Simon JD Prince, Jonathan Warrell, Umar Mohammed, and Graham Jones. Superpixel lattices. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. Citeseer, 2008. 2

[32] Greg Mori. Guiding model search using segmentation. In *Proceedings of IEEE International Conference on Computer Vision*, volume 2, pages 1417–1423. IEEE, 2005. 2

[33] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2536–2544, 2016. 3

[34] Paul L Rosin and Yu-Kun Lai. Non-photorealistic rendering of portraits. In *Proceedings of the workshop on Computational Aesthetics*, pages 159–170. Eurographics Association, 2015. 2

[35] Artsiom Sanakoyeu, Dmytro Kotovenko, Sabine Lang, and Bjorn Ommer. A style-aware content loss for real-time hd style transfer. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 698–714, 2018. 3

[36] Zoya Shahcheraghi and John See. On the effects of pre-and post-processing in video cartoonization with bilateral filters. In *Proceedings of IEEE International Conference on Signal and Image Processing Applications*, pages 37–42. IEEE, 2013. 2

[37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3, 4

[38] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016. 7

[39] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *International Journal of Computer Vision*, volume 98, page 2, 1998. 2

[40] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013. 4

[41] Tom Van Laerhoven, Jori Liesenborgs, and Frank Van Reeth. Real-time watercolor painting on a distributed paper model. In *Proceedings Computer Graphics International, 2004.*, pages 640–643. IEEE, 2004. 2

[42] Andrea Vedaldi and Stefano Soatto. Quick shift and kernel methods for mode seeking. In *European Conference on Computer Vision*, pages 705–718. Springer, 2008. 2

[43] Jue Wang, Yingqing Xu, Heung-Yeung Shum, and Michael F Cohen. Video tooning. In *ACM Transactions on Graphics (ToG)*, volume 23, pages 574–583. ACM, 2004. 2

[44] Huikai Wu, Shuai Zheng, Junge Zhang, and Kaiqi Huang. Fast end-to-end trainable guided filter. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1838–1847, 2018. 2

[45] Li Xu, Cewu Lu, Yi Xu, and Jiaya Jia. Image smoothing via l 0 gradient minimization. In *ACM Transactions on Graphics (TOG)*, volume 30, page 174. ACM, 2011. 2

[46] Li Xu, Jimmy Ren, Qiong Yan, Renjie Liao, and Jiaya Jia. Deep edge-aware filters. In *International Conference on Machine Learning*, pages 1669–1678, 2015. 2

[47] Ming Yang, Shu Lin, Ping Luo, Liang Lin, and Hongyang Chao. Semantics-driven portrait cartoon stylization. In *Proceedings of IEEE International Conference on Image Processing*, pages 1805–1808. IEEE, 2010. 2

[48] Lvmin Zhang, Chengze Li, Tien-Tsin Wong, Yi Ji, and Chunping Liu. Two-stage sketch colorization. In *SIGGRAPH Asia 2018 Technical Papers*, page 261. ACM, 2018. 3

[49] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European Conference on Computer Vision*, pages 649–666. Springer, 2016. 3

[50] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of IEEE International Conference on Computer Vision*, 2017. 1, 3, 5, 6, 7, 8