ICG Report Team 1
Pencil Photo Sketch with Elastic Shading

呂栢頤, D09944015
Mai Tan Ha, D10922024
Wei-I Lin, R10922076

June 1, 2022

1 Motivation

When writing *hw2*, we discover the **intermediates** of **edge detection**, such as **gradient**, **Laplacian** and **hysteretic thresholding** in **Canny method**, is similar to **human-like** sketch of image. And there are some visual effect when we use **transfer function** to change the intensity histogram distribution in figure 1. It is similar to **pencil sketching/drawing**, which is everyone familiar with the art style. Our question is: 'Can we transform the real-world image into the pencil sketch style picture?'

Two primary components of image rendering are **outlines** and **shading** that reflect differences in the amount of light falling on a region with its intensity, tone, and texture. Furthermore, there are different techniques for pencil sketching art tips for beginners to create beautiful pictures.

Take use the skills in *digital image processing*, we could obtain **outlines** as **edge detection**; **shading** as part of **distorting** and **texture analysis**. If we build the great combination of these techniques, make the images as pencil sketching is possible.

Besides, as the **deep learning style transfer**, or **deep style** [1], is popular recently. It is intuitively connected to style transfer as the pencil sketching is the type among the

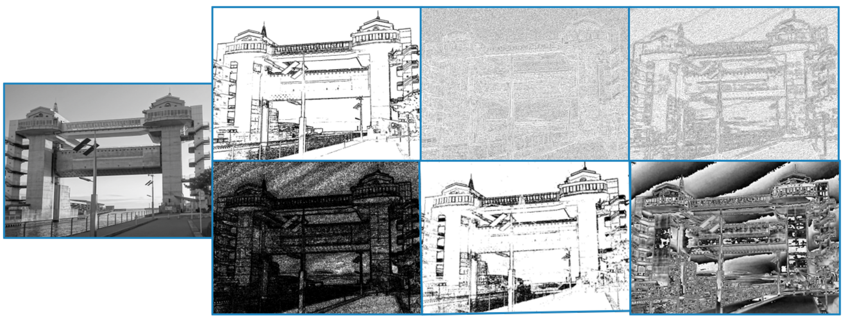

Figure 1: Intermediates and results in *hw2*

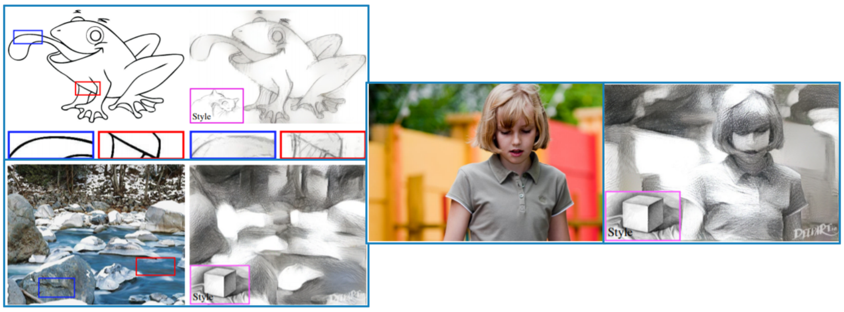

Figure 2: Examples of **style transfer**

Figure 3: Goal of this project

art style. We could collect the classic paints from The VANGOGH GALLERY, pinterest and Google image as **style image**. Then transform the image as this specific style like figure 2.

In summary, given the photo/images, we want to transform them into the **pencil sketch** style. And we're going to compare the effects from traditional **image processing** with fancy **deep style** method figure 3.

2 Problem definition

Our goal is to transform the photos/images into **pencil sketch** style. In this project, we focus on three scenarios:

Task 1: Beginner sketching: flower, avatar, and life photo of team members.

Task 2: Scene sketching: day and night of buildings, such as 101, NTU.

Task 3: Video sketching: NTU College of Social Sciences building, mother and daughter, couples in the sunshine.

The first task is also the guide of the beginner in the art course. And we know **human hand sketching** has the following features:

1. Shape of lines is shaken.
2. Long lines are drawn divided into short strokes.

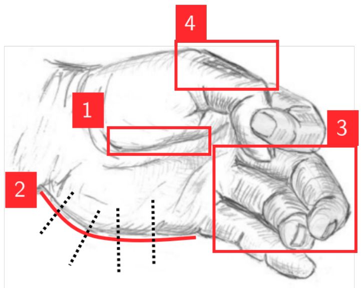

Figure 4: An example of hand-written sketch

Figure 5: An example of urban sketching

3. Variation in line brightness.
4. Main contours are drawn with multiple lines/several times.

An example of human-like sketch is shown in figure 4 [2].

The second task is advanced skills in sketching. Urban sketching figure 5 is about observing the world, witnessing, and recording. Here are some tips, pay attention to the silhouette and ensure the angle and sight with proportionally accurate. Then emphasize the light source on different objects, such as buildings, clouds, background, and things that appeal to you.

The last task is a special case. It is only the thing that computers can do. We could try the pencil-style effects of continuous photos/images than merge them into a video figure 6.

3 Algorithm

In this project, we implement and compare three models:

1. Traditional algorithm. **combine sketch and tone** [3].
2. Deep style algorithm. **deep style transfer** [1] and its improvement [4].
3. SOTA deep learning algorithm. **Im2pencil** [5].

The first one is the traditional image processing method. And the second and third methods are deep learning-based.

Figure 6: An example of video sketching

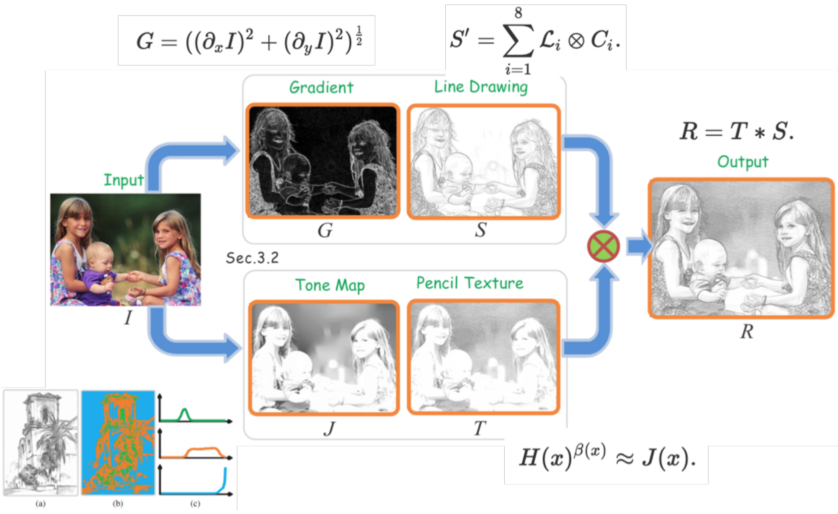

Figure 7: Overview of traditional algorithm in [3]

In the following subsections, we discuss these methods core idea and critical properties affect our results.

3.1 Traditional algorithm

Based on the observation of people's pencil sketch images in daily life, we can divide the drawing process into two steps:

1. To outline the shape of objects.
2. To render the picture, for example, penciling repeatedly and gently.

According to the description above, we can design a framework as the combination of skills in traditional image processing figure 7.

For **line drawing**, first, we need an effective tool to get the contour. The strategy here is to find the **gradient map** (G) of the original image (I).

$$G = \left((\partial_x I)^2 + (\partial_y I)^2\right)$$

The second is a line drawing by **convolution**. Convolution aggregates G with nearby pixels in which are 1 along the specific directions and the other values are 0. Considering the anti-aliasing problem, the convolution kernel is obtained by bilinear interpolation. The size of the convolution kernel is proposed as $\frac{1}{30}$ of the width or height of the original image. That is

$$G_i = L_i \circledast G,$$

where $\circledast$ means convolution and L_i is the line segment at the i^{th} direction. After obtaining the convolution result of all directions, for each pixel point, the response of the corresponding direction with the maximum convolution value is set as

$$C_i(p) = \begin{cases} G(p), & \text{if } \arg\max_i \{G_i(p)\} = i \\ 0, & \text{otherwise} \end{cases}$$

In the end, we convolve the response map set $\{C_i\}$ obtained above from the various directions again, it imitates human-like line-drawing process and resisting noise.

$$S^{'} = \sum_{i=1}^{8} L_i \circledast C_i$$

For **tone mapping**, according to the observation and analysis of a large number of hand-drawn pencil images, the distributions of their histogram are very different from photo images we took, but this paper finds three types of **transfer function** of description about pencil sketch.

- Shadow as Gaussian distribution. $p_3(v) = \frac{1}{\sqrt{2\pi\sigma_d}} \exp\left(-\frac{(v-\mu_d)^2}{2\sigma_d^2}\right)$
- Mild-tone as uniform distribution. $p_2(v) = \begin{cases} \frac{1}{u_b - u_a}, & \text{if } u_a \leq v \leq u_b \\ 0, & \text{otherwise} \end{cases}$
- Highlight as exponential distribution. $p_1(v) = \begin{cases} \frac{1}{\sigma_b} \exp\left(-\frac{1-v}{\sigma_b}\right), & \text{if } v \leq 1 \\ 0, & \text{otherwise} \end{cases}$

where μ_d, σ_d are mean and standard deviation, u_a, u_b are range of intensity, and σ_b is mean. Then we combine them together by

$$p(v) = \frac{1}{Z} \sum_{i=1}^{3} w_i p_i(v)$$

where Z is normalisation term and w_i is weight of each transfer function. Empirically, we set the hyper-parameters above as table 1. And we apply $p(v)$ on each pixels to get

w_1	w_2	w_3	σ_b	u_a	u_b	μ_d	σ_d
52	37	11	9	105	225	90	11

Table 1: Hyper-parameters in traditional algorithm

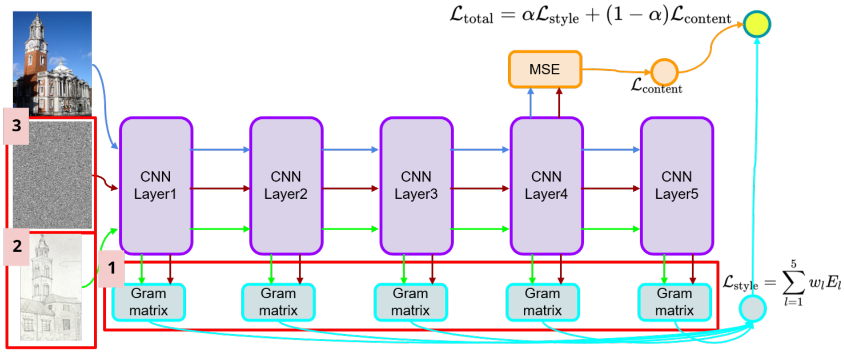

Figure 8: Overview of deep style algorithm in [1]

$J(x)$.

Second is pencil texture rendering, this paper collects 20 pencil textures $H(x)$ and learns the **level of intensity** β^*.

$$\beta^* = \arg\min_{\beta} \|\beta \log H(x) - \log J(x)\|_2^2 + \lambda \|\nabla \beta\|_2^2$$

where $\lambda = 0.2$ is regularisation term.

Finally, we can get the result image by element-wise multiply between **line drawing** S & **tone mapping** $T = H^{\beta^*}$.

$$R = T * S$$

where $*$ is element-wise product.

3.2 Deep style

In this project, we are going to implement [1], which is the earliest and famous deep learning-based style transfer. The core idea of this paper uses a pre-trained **convolutional neural network (CNN)**, VGG-19 to extract and merge the **content representation** of one image with the **style representation** of another from different levels.

Compare with traditional image processing method, **content** is defined as the semantic information of an image, i.e. the **line drawing**. While **style** is defined as the **textural** information of an image, i.e. the **tone mapping**. But we train them simultaneously in this framework.

The procedure for neural style transfer is to optimise the **blank image** we are generating (generated image) w.r.t. the **content loss** and **style loss** in CNN figure 8. Content loss is defined as a mean square error (MSE) of feature maps between content image P^l and generated image F^l.

$$\mathcal{L}_{\text{content}}(P^l, F^l) = \frac{1}{2}\|F^l - P^l\|_2^2$$

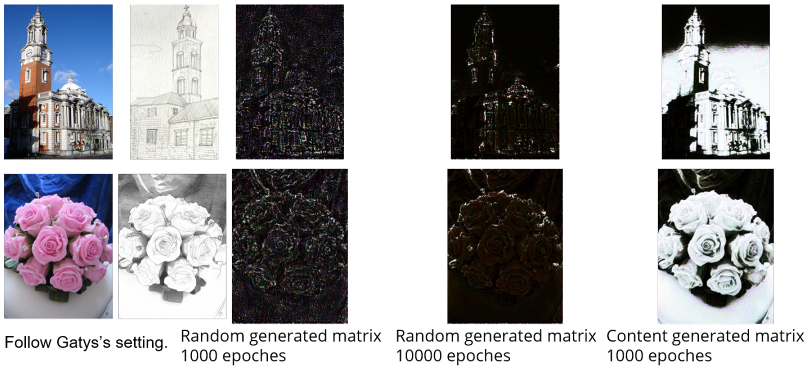

Figure 9: Gram loss is difficult to train

where $l = 4$ in our experiments.

On the other hand, style loss is defined as the difference between Gramian matrix of the feature map of generated image $G^l = {F^l}^\top F^l$ and style image $A^l = {S^l}^\top S^l$. Then the contribution of layer l to the style loss is

$$E_l = \frac{1}{(2N^l M^l)^2} \|F^l - G^l\|_2^2$$

where N^l, M^l are the size of style and generated feature maps. And the style loss is

$$\mathcal{L}_{\text{style}}(S^l, F^l) = \sum_{l=1}^{5} w_l E_l$$

Finally, we initialize the generated matrix with white noise. And perform gradient descent on it to minimize features discrepancy with total loss $\mathcal{L}_{\text{total}}$.

$$\mathcal{L}_{\text{total}}(P, S, F) = \alpha \mathcal{L}_{\text{style}}(S, F) + (1 - \alpha) \mathcal{L}_{\text{content}}(P, F)$$

where hyper-parameter α controls the level of style.

In our experiments, we find out there are **three** critical factors:

1. Style loss. The difference between feature maps of the generated image and the style image.
2. Style image. We could get a different style of pencil sketching by the style image.
3. Initial generated matrix. Start from the different generated matrix, such as random, content, and style.

The first factor is essential to the training procedure. Even if we carefully adjust and grid-search the hyper-parameters, such as α, w_l, some images are too difficult to train with limited time figure 9.

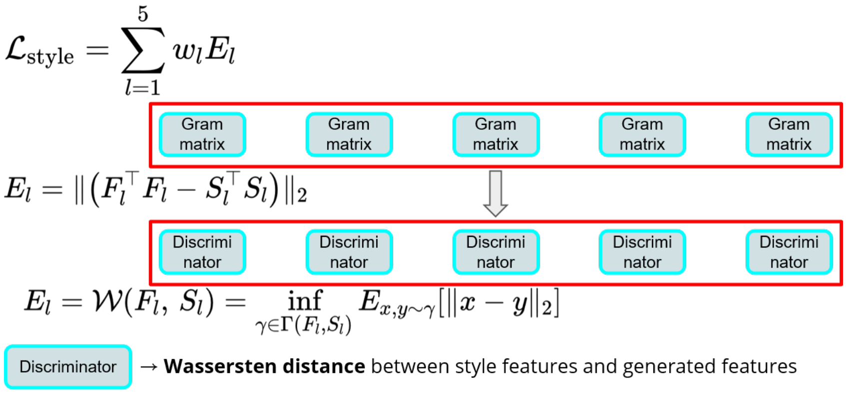

Figure 10: From Gramian matrix loss to Wassersten distance

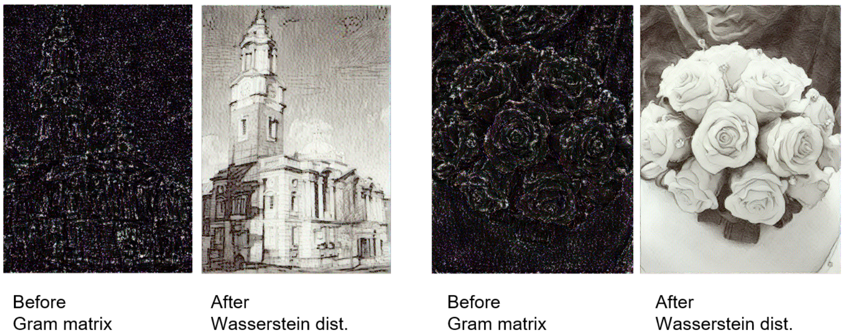

Figure 11: WGAN could get better results in less epochs

Inspired by [4], we could treat feature map of style as **probability distribution**. And use **Wassersten distance** as the metric of distribution distance between generated image and style image.

$$E_l = \mathcal{W}(F^l, S^l) = \inf_{\gamma \in \Gamma(F^l, S^l)} \mathbb{E}_{x,y\sim\gamma}\left[\|x-y\|_2\right]$$

where $\mathcal{W}$ is Wassersten distance. Thus we can follow the idea from **WGAN** [6]. Build the **discriminators** to approximate the Wassersten distance. Finally, take style image as the real image, generated matrix as result of generator, iteratively train discriminator and generator to make their distribution closer. The results show the improvement of replacing Gramian matrix loss with WGAN figure 11.

The second factor is choosing the style image. As we know the core idea of style transfer is **overlapping** the style image on the content image. There is a dramatic effect of different style images, moreover, it could **destroy** our generated matrix in case figure 12.

The third factor is the initialization of generated matrix. We could see the different results starting from random noise or content generated matrix figure 13. The significant feature is **histogram of intensity distribution**. Regardless of random noise or content generated matrix, it cannot easily reach the pencil sketching style.

To solve the above issues, we provide a new idea of combining the traditional algorithm and the deep style. We use the result of the traditional algorithm as a style image as

Figure 12: Style is difficult to select

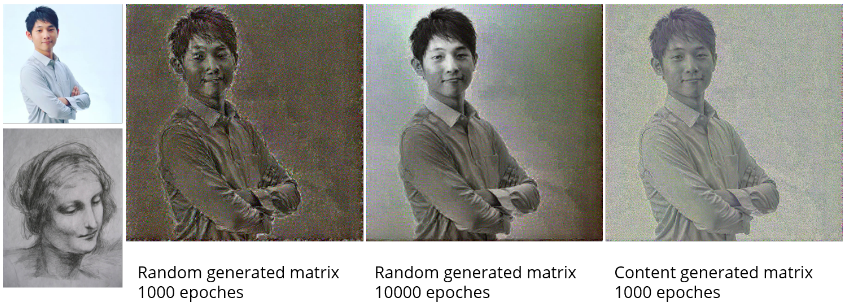

Figure 13: Difference initial generated matrix

well as the initial generated matrix. It could get **stable edge information** but **rich tone information** pencil sketching. And we reveal our improvement in experiments.

3.3 Im2pencil

Im2pencil is a SOTA pencil-style sketching deep learning method. It combines the traditional edge detection and deep learning method to create the different **outlines** (line drawing) and **shading** (tone mapping) pencil style image figure 14.

The core idea of **Im2pencil** is design one CNN to capture predefined clear and rough outlines, the other CNN to render predefined hatching, blending, cross-hatching, and stippling shading. The goal of outline CNN and shading CNN is transforming edge map and tone map into pencil style sketching figure 15. At first, they collect many pencil drawings on the web. Then manually annotate these images with outline style labels as well as shading style labels as the training dataset figure 16.

Next step, they use the Extended Difference-of Gaussians (XDoG) filter to obtain outline drawing. And Guided Filter (GF) to acquire a tone map figure 17. Then train in outline branch and shading branch.

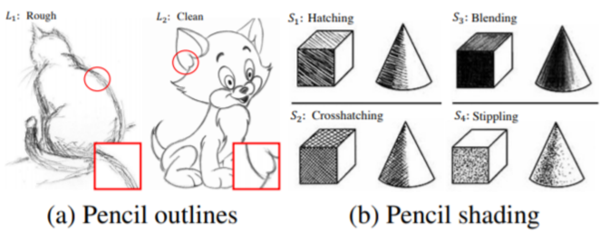

Figure 14: Twe control factors in **Im2pencil**: outlines and shading

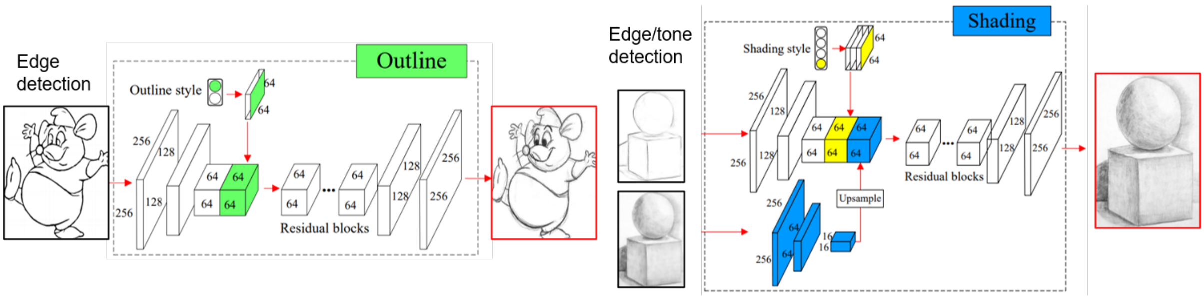

Figure 15: Overview of **Im2pencil** algorithm in [3]

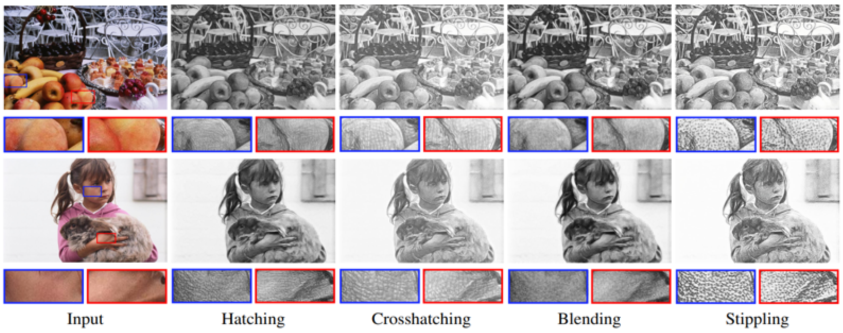

Figure 16: Annotations of pre-define styles are required

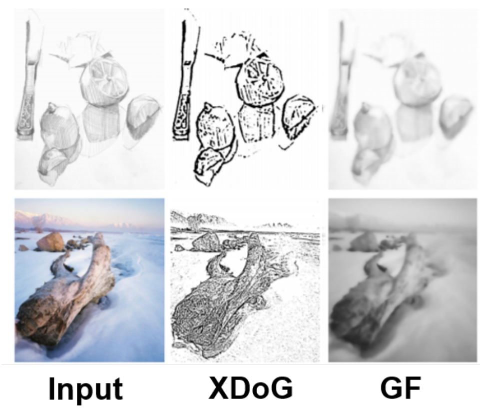

Figure 17: Intermediates of XDoG (edge) & GF (tone)

Figure 18: Results of traditional algorithm on beginner sketching & scene sketching

In this project, we use their pre-trained model to predict/ reference our dataset. We could extract the outline drawing and tone map, select desired styles to conduct style transfer procedures.

4 Experimental results

First, we will discuss the results of the traditional algorithm. Second, conduct several experiments with deep style and show how we improve it with the train WGAN framework. Third, results of combining traditional algorithm and deep style. Finally, we compare the above methods with **Im2Pencil**, which is an emerging method in recent.

4.1 Traditional algorithm

The traditional algorithm gives a stable and good performance on different scenarios, including the night scene figure 18. We could adjust the **thickness** of line drawing as well as **intensity & texture** of tone mapping to get desired results.

Although the traditional algorithm seems robust, it causes the problem when we apply it to video the scenario. We could discover that there is **lens**/mask-like side effects in front of our eyes. i.e. only the edge changes with time but not for background texture. It is caused by $R = T * S$, the tone mapping comes from the same pencil style image. When the scenes are similar but our sight moves, the traditional algorithm gives artificial results figure 19.

4.2 Deep style

The deep style with WGAN gives the rich style of pencil sketching. We could replace the **style images** to get different results. There are some tips to choose style images:

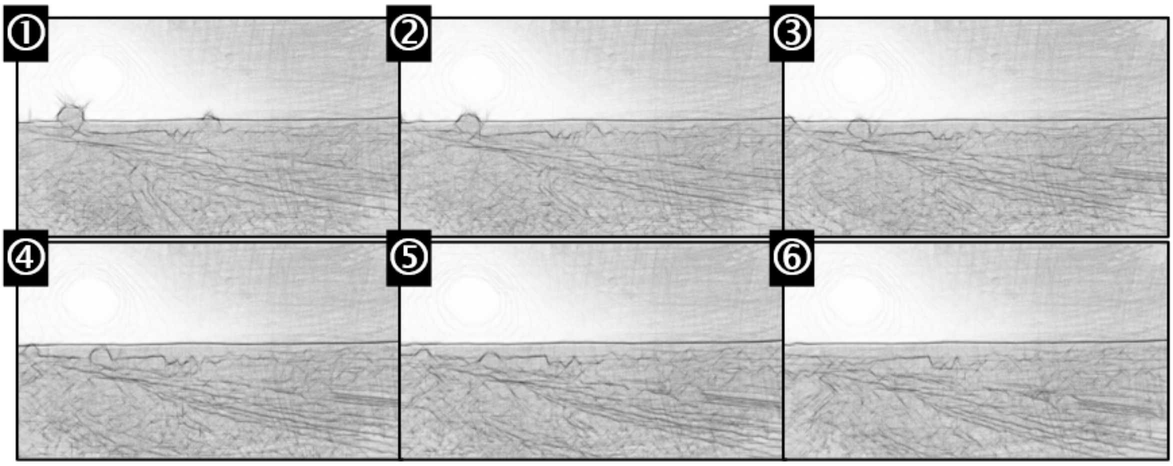

Figure 19: Results of traditional algorithm on video sketching

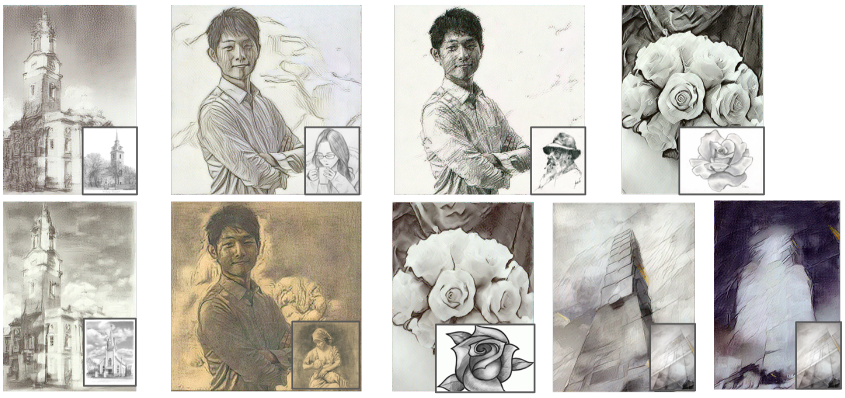

Figure 20: Results of deep style on beginner sketching & scene sketching

- Select with similar outlines of the content image.
- Select with similar tones of the content image.
- Select with simple and consistent outlines and tones. e.g. **Van Gogh**'s drawing is great in our experiments.

In figure 20, we get good on the beginner sketching scenario as these images have clear contours. But in scene sketching, it is difficult of selecting similar style images. And it often blurs on light objects in a night scenes with relatively strong contrast dark background.

4.3 Combine traditional algorithm and deep style

Reuse the results of the traditional algorithm is better than before. We could get more flexible context, tone than the traditional algorithm, but also stable than pure deep style.

Moreover, we show that the take it as initial generated matrix, the results is more similar to human-like pencil sketching figure 21. We consider that the distribution of intensity

Figure 21: Histogram is closer to style image is better

Figure 22: Compare traditional algorithm, deep style & combining on beginner sketching

is closer to the tone map in the traditional algorithm.

We list traditional algorithm, deep style with WGAN and their combination from the top row to down row. In a beginner sketching scenario figure 22, we could sketch *Five (and More) Senses* of human faces. The deep style with WGAN gives the special style of pencil sketching. And our method remedies the **loss edge** information in the traditional algorithm, e.g. *Face of Poy* & *Eyes of 新垣結衣*. Besides, it gives more variation on texture and intensity to strong the pencil style.

In scene sketching scenario figure **??**, the traditional algorithm gives stable style effects.

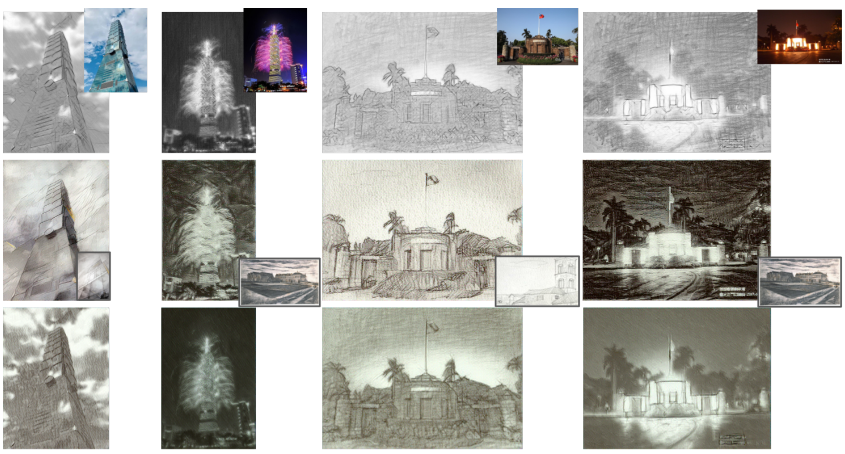

Figure 23: Compare traditional algorithm, deep style & combining on scene sketching

But deep style with GAN remains afterimage of style image, and it still does not work well on the night scene. Overall, we find out the best results in deep style occur when using Van Gogh's drawing as style images. We consider that his drawings have more intense painting without any strange texture. It makes style transfer relatively stable and natural than other style images. At last, our method learns better pencil style and makes the clear contour of the focal point (buildings) and its neighbors.

On video sketching, our method ensures variation and randomness during the deep style training process. It eliminates the artificial effect of a traditional algorithm. But it also causes the **twinkling** effects on continuous images. We try to add noise during our training process but fail at this time.

4.4 Im2pencil

Im2pencil can easily generate desired outline or shading.

On beginner sketching scenario, we could obtain eight different styles. The columns are different outline style. The rough outline learn the shaking as the human drawing; clear outline gives relative straight outlines. The rows are different shading style. The different shading style gives the different contrast and brightness of results.

On scene sketching scenario, the Im2pencil sketches more clear on shadow of the building than before. We consider that the separate outline and shading CNNs achieves this effects.

On video sketching scenario, the Im2pencil gives more robust and well results.

5 Conclusions

We compare traditional algorithm, deep style and Im2pencil.

Basically, all of these methods can achieve the pencil sketching style.

1. Traditional algorithm is relative stable on all scenarios. But it lacks of variation style and makes artificial results on video.

2. Deep style transfer generates diversity pencil sketching style. However, it spends a lot of time as we need to train every time. And it is difficult of choosing the suitable style images. Last, it could twink on the video.

3. Our combination method balance the results between traditional algorithm and deep style transfer. But we fail on video twinkling.

4. Im2pencil gives well performance on each scenarios. But we only use predefined model to predict our images. It is expensive to retrain the outline CNN as well as shading CNN when we need to collect a lot of desired style images.

In summary, our accomplishment are

- Implement several methods including traditional algorithm and deep style.
- Improve deep style by WGAN discriminator.
- Combine image processing and deep style.
- Compare with SOTA on different tasks.

We leave the video twinkling as the future work. Maybe add continuous noise term into loss is one direction. The other is make it as open source service oh the GitHub[1].

References

[1] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

[2] Ryota Okawa, Hiromi Yoshida, and Youji Iiguni. Automatic pencil sketch generation by using canny edges. In *2017 fifteenth IAPR international conference on machine vision applications (MVA)*, pages 282–285. IEEE, 2017.

[3] 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. Citeseer, 2012.

[4] Eddie Huang and Sahil Gupta. Style is a distribution of features. *arXiv preprint arXiv:2007.13010*, 2020.

[5] Yijun Li, Chen Fang, Aaron Hertzmann, Eli Shechtman, and Ming-Hsuan Yang. Im2pencil: Controllable pencil illustration from photographs. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1525–1534, 2019.

[6] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223. PMLR, 06–11 Aug 2017.

[1] https://github.com/yxliu-ntu/DIP_Final