SmartShadow: Artistic Shadow Drawing Tool for Line Drawings

Lvmin Zhang
Soochow University / Style2Paints Research
lvminzhang@acm.org

Jinyue Jiang
Style2Paints Research
jinyuejiang@outlook.com

Yi Ji
Soochow University
jiyi@suda.edu.cn

Chunping Liu
Soochow University
cpliu@suda.edu.cn

Abstract

SmartShadow is a deep learning application for digital painting artists to draw shadows on line drawings, with three proposed tools. (1) Shadow brush: artists can draw scribbles to coarsely indicate the areas inside or outside their wanted shadows, and the application will generate the shadows in real-time. (2) Shadow boundary brush: this brush can precisely control the boundary of any specific shadow. (3) Global shadow generator: this tool can estimate the global shadow direction from input brush scribbles, and then consistently propagate local shadows to the entire image. These three tools can not only speed up the shadow drawing process (by 3.1× as experiments validate), but also allow for the flexibility to achieve various shadow effects and facilitate richer artistic creations. To this end, we train Convolutional Neural Networks (CNNs) with a collected large-scale dataset of both real and synthesized data, and especially, we collect 1670 shadow samples drawn by real artists. Both qualitative analysis and user study show that our approach can generate high-quality shadows that are practically usable in the daily works of digital painting artists. We present 30 additional results and 15 visual comparisons in the supplementary materiel.

1. Introduction

Shadows in artworks are essentially different from that in photography or photorealistic fields of computer vision: the artwork shadows are *drawn* by artists. These shadows depicts the mood of characters and express the emotion of artists, without being constrained by physically correct light transmission laws or geometrically precise object structures. Artists adjust the location, scale, shape, density, and many other features of shadows to achieve diverse artistic purposes, *e.g.*, amplification, exaggeration, antithesis, silhouette, *etc*.

An application that can assist artists in drawing shadows for line drawings is highly desired. This is not only because creating shadows on line drawings is one of the most frequent and time-consuming tasks in the daily work of many digital painting artists, but also because shadow drawing is the foundation of a wide variety of further artistic creations, *e.g.*, hard shadows can be smoothed into soft shadings (with techniques like joint anisotropic diffusion [46]), shadows can be stylized with hatching or drafting effects [55], sharp shadows can be used in cel-shading (see also the YouTube tutorial [27]), *etc*.

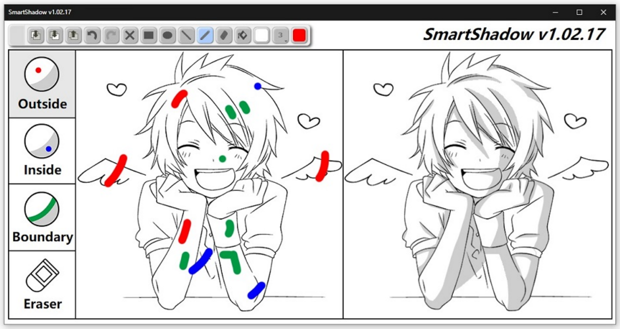


Figure 1. **Screenshot of the SmartShadow.** The user gives scribbles as shadow indications (on the left) to obtain the high-quality shadow (on the right). *Smiling boy, used with artist permission.*

Might we be able to achieve a deep learning approach that can quickly produce visually satisfying shadows given only a few user indications, saving the time and effort of digital painting artists, and simultaneously, facilitating more plentiful artistic creations? We present an interactive shadow drawing application (Fig. 1) to achieve these goals. This application consists of the following three proposed tools:

The first tool is the shadow brush. Users can draw blue or red scribbles (*e.g.*, Fig. 2-(a)) to coarsely indicate the areas inside or outside the shadows they want. This tool does not require users to have professional drawing skills, as it can "smartly" generate shadow shapes learned from

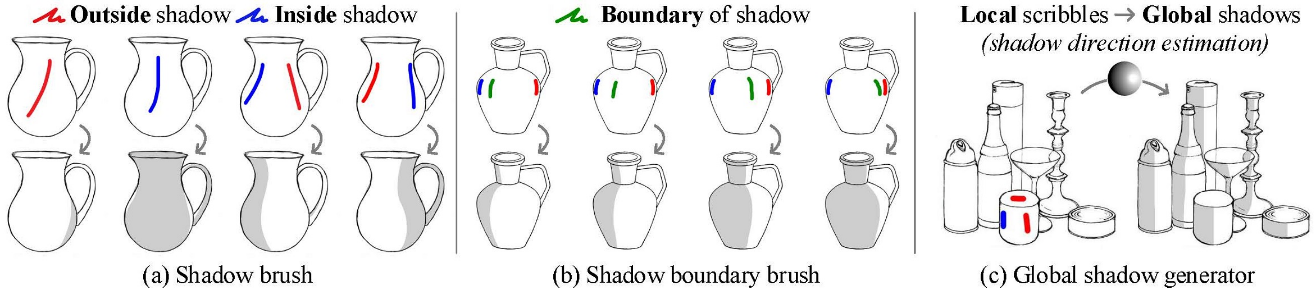

Outside shadow Inside shadow
Boundary of shadow
Local scribbles → Global shadows
(shadow direction estimation)
(a) Shadow brush
(b) Shadow boundary brush
(c) Global shadow generator

Figure 2. **Examples of our three proposed tools.** (a) The shadow brush allows users to coarsely control the areas inside or outside shadows. (b) The shadow boundary brush enables users to accurately control the shadow shapes. (c) The global shadow generator can estimate the global shadow direction and automatically produce globally consistent shadows. *Artworks used with artist permissions.*

large-scale artistic shadow data. This tool is well-suited for shadows without strict shape requirements or with low shape uncertainty, *e.g*., inconspicuous background shadow, dense shadow of gathered small objects, *etc*.

The second tool is the shadow boundary brush. Users can use this brush to precisely control the shadow boundaries. They only need to scribble a small part of their wanted boundary (*e.g*., the green scribbles in Fig. 2-(b)), and the tool will automatically estimate the boundary shape and generate the entire shadow. This tool is indispensable for professional use cases where the accurate shadow control is important, *e.g*., character face shadows, salient object shadows, close-up shadows, *etc*.

The third tool is the global shadow generator. This tool can estimate the global shadow direction from input brush scribbles, and then propagate local shadows to the entire image consistently (*e.g*., Fig. 2-(c)). This tool is user-friendly in that it is fully automatic and does not require artists to learn any extra technical knowledges, *e.g*., managing screen-space shadow direction, world-space light orientation, *etc*. This tool is especially effective for complicated artworks, *e.g*., drawings with multiple targets, artworks with complex structure, *etc*.

These three tools are designed in a data-driven way. To ensure the robustness and generalization, we learn hierarchical neural networks with a large-scale dataset of both real-artist data and synthesized data. In particular, we collect 1670 line art and shadow pairs drawn by artists manually, 25,413 pairs synthesized by rendering engine, and 291,951 shadow pairs extracted from in-the-wild internet digital paintings.

Experiments show that the SmartShadow can speed up the shadow drawing process by 3.1×. User studies demonstrate that users can use this application to effectively achieve satisfactory shadows that are practically usable in their daily jobs. Besides, even if the users do not give any input edits, our approach can still generate plausible results that are more preferable than other fully-automatic shadow generating methods. Finally, we present 30 qualitative results and 15 additional comparisons in the supplementary materiel.

In summary, our contributions are: (1) We present the SmartShadow, a digital painting application to draw shadows on line drawings, including the tools of shadow brush, shadow boundary brush, and global shadow generator. (2) We present a large-scale dataset of line drawing and shadow pairs drawn by real artists, as well as shadow data synthesized by rendering engines or extracted from in-the-wild digital paintings. (3) Perceptual user study and qualitative evaluations demonstrate that the SmartShadow is more preferable by actual end users when compared to other possible alternatives. (4) Results show that the SmartShadow can speed up the shadow drawing process by 3.1×.

2. Related Work

Artistic shadow creation. Different from photography relighting or photorealistic rendering [15, 28, 30, 31, 32, 11], the artistic creation of shadows is a perception-oriented process. ShadeSketch [55] is the current state of the art in automatic artistic shadow generating. Sketch2Normal [40] and DeepNormal [19] can generate normal maps from line drawings. Hudon *et al*. [20] also proposed a vectorgraph-based method for artistic shadow manipulation. Ink-and-Ray [43] is a typical proxy-based method for illumination effects, and Dvorožňák *et al*. [16] extended this approach to a part-based high-relief proxy structure. PaintingLight [53] is a RGB geometry framework that converts artists' brush stroke history to lighting effects. Our approach allows users to intuitively manipulate the shadow with scribbles, *i.e*., in a "what you see is what you get" manner.

Shadow synthesis and extraction. To ensure the robustness and generalization of our approach, we use shadow synthesis and extraction algorithms to increase the scale and diversity of our training data. A typical method is intrinsic imaging [4] in the field computational illumination. Optimizing-based approaches [36] solve the decomposition by optimizing an energy with specific constraints. Learning-based approaches [35, 17, 2] propose to learn the mapping between the input images and their albedo images from large amounts of data. Several in-the-wild datasets [7, 6, 8, 23] and other synthetic or annotated datasets [18, 5] make intrin-

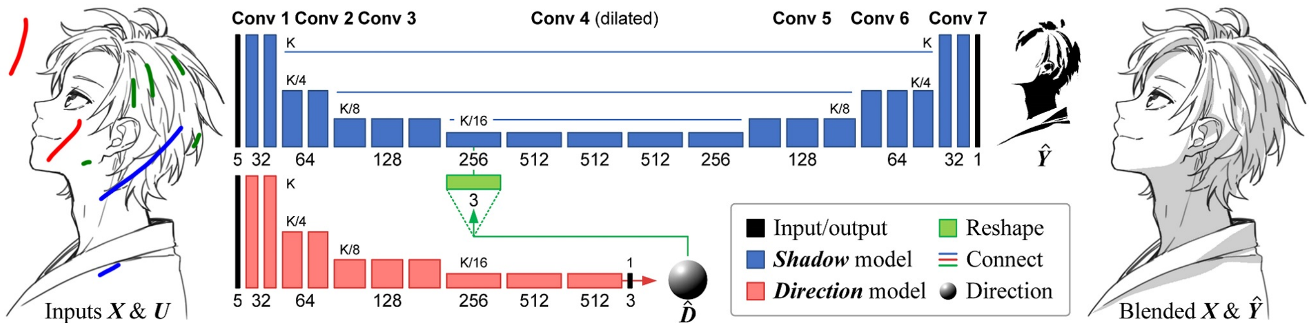


Figure 3. **Network architecture.** We train two branches of the shadow drawing network. Both branches use the blue layers for predicting the shadow. The **direction model** branch uses red layers for predicting the global shadow direction. The **shadow model** branch uses blue layers for predicting the final output shadow. All convolutional layers use 3 × 3px kernels. We do not use any normalization layers. Shortcut connections are added to upsampling convolution layers. *Boy looking upside, used with artist permission.*

sic images scalable with deep learning methods.

Interactive creation and cartoon techniques. Scribble-based interactive tools are shown to be effective in creative fields like image colorization [54] and sketch inking [38]. Another closely related field is cartoon image processing. Manga structure extraction [24], cartoon inking [39, 37, 38], and line closure [25, 26] methods analysis the lines in cartoon and digital paintings. A region-based composition method can be used in cartoon image animating [41]. Deep learning approaches [12, 45, 48, 50, 49] process artistic images or cartoon drawings in the domains of photographs and human portraits. Color filling applications [52, 44, 42] colorize sketch or line drawings with optimization-based or learning-based approaches. Our approach generates shadow from line drawings, and can be used in digital painting and related artistic creation scenarios.

3. Method

We train a deep network to draw shadows given the line drawings and user input scribbles. In Section 3.1, we describe the objective of the neural architecture and the three proposed interactive tools: shadow brush, shadow boundary brush, and global shadow generator. We then describe our presented dataset and the customized training method in Section 3.2.

3.1. Interactive tools for shadow drawing

The inputs (Fig. 3-left) of our approach are the line drawing $\boldsymbol{X} \in \mathbb{R}^{H\times W\times 1}$ along with the RGBA user scribble canvas denoted by $\boldsymbol{U}^{H\times W\times 4}$. The output $\widehat{\boldsymbol{Y}} \in \mathbb{R}^{H\times W\times 1}$ is the estimation of pixel-wise shadow probability, which is binarized (threshold is 50% gray) and blended (multiplied) to the original line drawing for shadow effects (Fig. 3-right). The mapping is learned with the neural networks $\mathcal{F}(\cdot\,;\theta)$, parameterized by θ, with the architecture specified in Fig. 3. We train with the data distribution $\mathcal{D}$ with line arts, user inputs, and desired shadows. We minimize the objective with likelihood $\mathcal{L}$ describing the distances between the estimation and ground truth as

$$\theta^* = \arg\min_{\theta} \mathbb{E}_{\boldsymbol{X},\boldsymbol{U},\boldsymbol{Y}\sim\mathcal{D}}[\mathcal{L}(\mathcal{F}(\boldsymbol{X},\boldsymbol{U};\theta),\boldsymbol{Y})]\,. \tag{1}$$

We learn two network branches: the shadow model $\mathcal{F}_s(\cdot\,;\theta_s)$ and the shadow direction model $\mathcal{F}_d(\cdot\,;\theta_d)$. In inference, the direction model estimates the global shadow direction $\widehat{\boldsymbol{D}} \in \mathbb{R}^3$ for the shadow model to predict the shadow with

$$\widehat{\boldsymbol{Y}} = \mathcal{F}_s(\boldsymbol{X},\boldsymbol{U},\widehat{\boldsymbol{D}};\theta_s) \;\text{ and }\; \widehat{\boldsymbol{D}} = \mathcal{F}_d(\boldsymbol{X},\boldsymbol{U};\theta_d)\,. \tag{2}$$

During training, the scribbles are synthesized for our tools by giving projections of the ground truth shadow $\boldsymbol{Y}$ with the projection function $\mathcal{P}_u$ as $\boldsymbol{U} = \mathcal{P}_u(\boldsymbol{Y})$. Because the training synthetically generates user inputs, our dataset only needs to contain line drawings, shadow directions, and our wanted shadows. In particular, we solve two sub-problems for the shadow model and shadow direction model with

$$\begin{aligned}\theta_d^* &= \arg\min_{\theta_d} \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y},\boldsymbol{D}\sim\mathcal{D}}[\mathcal{L}_d(\mathcal{F}_d(\boldsymbol{X},\boldsymbol{U};\theta_d),\boldsymbol{D})]\,,\\ \theta_s^* &= \arg\min_{\theta_s} \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y},\boldsymbol{D}\sim\mathcal{D}}[\mathcal{L}(\mathcal{F}_s(\boldsymbol{X},\boldsymbol{U},\boldsymbol{D};\theta_s),\boldsymbol{Y})]\,,\end{aligned} \tag{3}$$

where $\mathcal{L}_d$ is a likelihood function for the shadow direction estimation problem. The three proposed shadow drawing tools are detailed as follows.

Shadow brush. The shadow control is achieved by projecting $\mathcal{P}_u$ to sample pixels inside (*resp*., outside) the ground truth shadows in $\boldsymbol{Y}$ as blue (*resp*., red) scribbles. We observe that, unlike common pixel sampling problems (*e.g*., [54, 34, 52]) where pixels are routinely distributed and sampled uniformly, shadow images are unique in their unbalanced pixel quantity inside and outside shadows. Based on this observation, we propose to balance the pixel sampling by introducing a Bivariate Normal Distribution (BND), with a Probability Density Function (PDF) denoted by $f_b(\cdot,\cdot)$. We sample n_i pixels inside the shadows and n_o pixels outsides, subjecting to the Bivariate Normal PDF [47] as

$$f_b(n_i,n_o) = \frac{\exp(-\frac{1}{2(1-\rho^2)}p_b(n_i,n_o))}{2\pi\sigma_i\sigma_o\sqrt{1-\rho^2}}\,, \tag{4}$$

where $p_b(\cdot,\cdot)$ is a bivariate Gaussian normal term

$$p_b(n_i,n_o)=(\frac{n_i-\mu_i}{\sigma_i})^2-2\rho(\frac{n_i-\mu_i}{\sigma_i})(\frac{n_o-\mu_o}{\sigma_o})+(\frac{n_o-\mu_o}{\sigma_o})^2, \quad (5)$$

where $\{\mu_i,\mu_o,\sigma_i,\sigma_o,\rho\}$ are bivariate normal distribution values with $8,8,2,2,0.5$. Using these sampled pixels as starting positions, we synthesize small scribbles with line segments at random rotation $\theta \sim U(-\pi,\pi)$, length $l \sim U(5,15)$ pixels, and width $w \sim U(1,3)$ pixels.

Shadow boundary brush. The accurate shadow boundary control is achieved by projecting $\mathcal{P}_u$ to sample shadow edges in the ground truth $\boldsymbol{Y}$ as green scribbles. We randomly sample $n_b \sim U(0,16)$ pixels of these edges as scribble starting points, and then synthesize small solid circles at random radius of $r \sim U(5,15)$ pixels. Besides, we observe that an important characteristic of shadows drawn by artists is the smooth boundaries and sharp corners. We encourage such smoothness and sharpness by introducing an anisotropic penalty $\phi(\cdot)$ within the customized likelihood

$$\mathcal{L}(\widehat{\boldsymbol{Y}},\boldsymbol{Y})=\lambda_a\phi(\widehat{\boldsymbol{Y}})+\sum_p||\widehat{\boldsymbol{Y}}_p-\boldsymbol{Y}_p||_2^2\,, \quad (6)$$

where p is pixel position, $||\cdot||_2$ is Euclidean distance, λ_a is weighting parameter, and the penalty $\phi(\cdot)$ can be written as

$$\phi(\widehat{\boldsymbol{Y}})=\sum_p\sum_{i\in w(p)}\sum_{j\in w(p)}\left(\delta(\boldsymbol{X})_{ij}||\widehat{\boldsymbol{Y}}_i-\widehat{\boldsymbol{Y}}_j||_2^2\right)\,, \quad (7)$$

where $w(p)$ is a 3×3 window centered at pixel position p, with $\delta(\cdot)$ being a Gaussian anisotropic term

$$\delta(\boldsymbol{X})_{ij}=\exp(-||\boldsymbol{X}_i-\boldsymbol{X}_j||_2^2/\kappa^2)\,, \quad (8)$$

where κ is an anisotropic weight. This term increases and encourages smoothness when $w(p)$ is located inside shadow areas with no steep line transitions in the line drawing $\boldsymbol{X}$, while decreases and allows for sharpness when $w(p)$ comes across salient line drawing patterns like corners or contours.

Global shadow generator. The global shadow generating is guided by the shadow direction $\boldsymbol{D}=[\alpha_x\ \ \alpha_y\ \ \alpha_z]^\intercal$ with α_x and α_y being in line with the axes of image-space width (right is positive) and height (upward is positive), and α_z facing out of the image panel. We use a customized likelihood for this global shadow direction as

$$\mathcal{L}_d(\widehat{\boldsymbol{D}},\boldsymbol{D})=\sum_p(\underbrace{-\widehat{\boldsymbol{D}}_p*\boldsymbol{D}_p}_{\text{cos}}+\underbrace{\lambda_n||\widehat{\boldsymbol{D}}_p-\frac{\widehat{\boldsymbol{D}}_p}{||\widehat{\boldsymbol{D}}_p||_2}||_2^2}_{\text{norm}})\,, \quad (9)$$

where $*$ is dot product and λ_n is a penalizing weight. The "cos" term is a cosine likelihood between the estimated direction and the ground truth, and the "norm" term is a regulation to encourage the confidence — low-intensity estimation will be amplified to a norm unit scale. Note that (1) this tool is only a coarse recommendation of the shadow propagation, and more specified effects (*e.g.*, spot light, rim shadow, *etc.*) can be achieved with the other brush tools; and (2) this tool is fully automatic and does not require artists to learn any technical knowledges, *e.g.*, data structure for 3D space orientation, screen-to-world space conversion, *etc.*

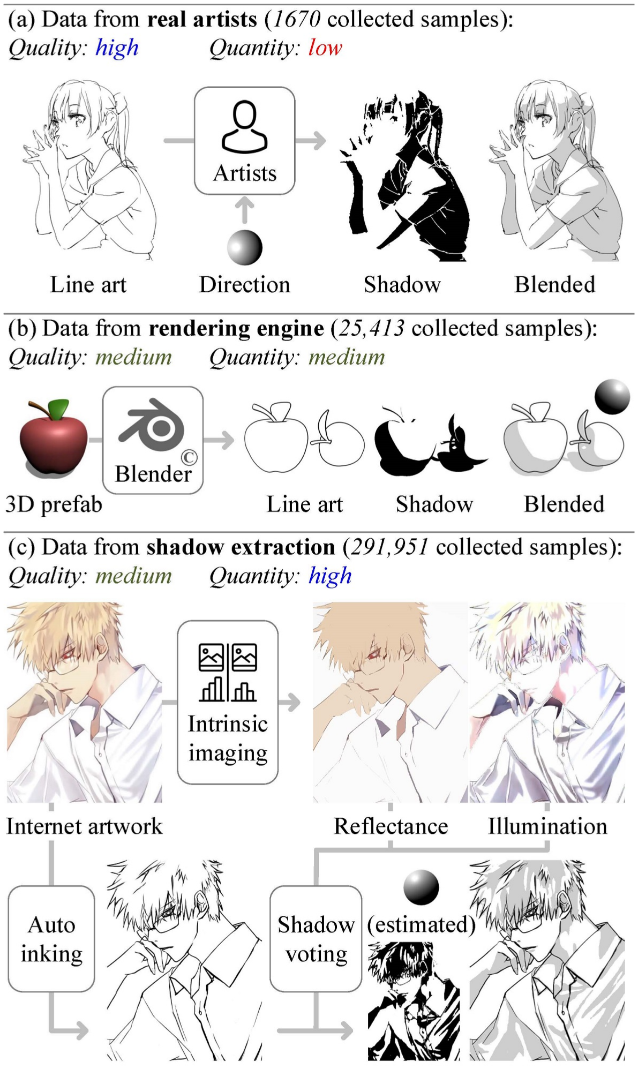


Figure 4. **Dataset preparation.** We present a large-scale dataset with both real data drawn by artists manually and synthesized data obtained from rendering engines and shadow extraction algorithms.

3.2. Data preparation and training schedule

Ideally, we may invite professional artists to manually draw a sufficient number of line drawing and shadow pairs as the training dataset so as to capture their perceptual designs and artistic understandings. Nonetheless, the highly expensive and time-consuming artistic drawing process makes large-scale annotation impractical. Another choice is to synthesize a training dataset using algorithms. Although a synthetic dataset might be larger or more diverse than real data, their shadow appearance may not match the artists' wishes and demands. We propose a customized schedule method: we *pre-train* our models with large-scale and diverse synthesized/extracted data, and then *fine-tune* the models on high-quality real data drawn by artists, to simultaneously

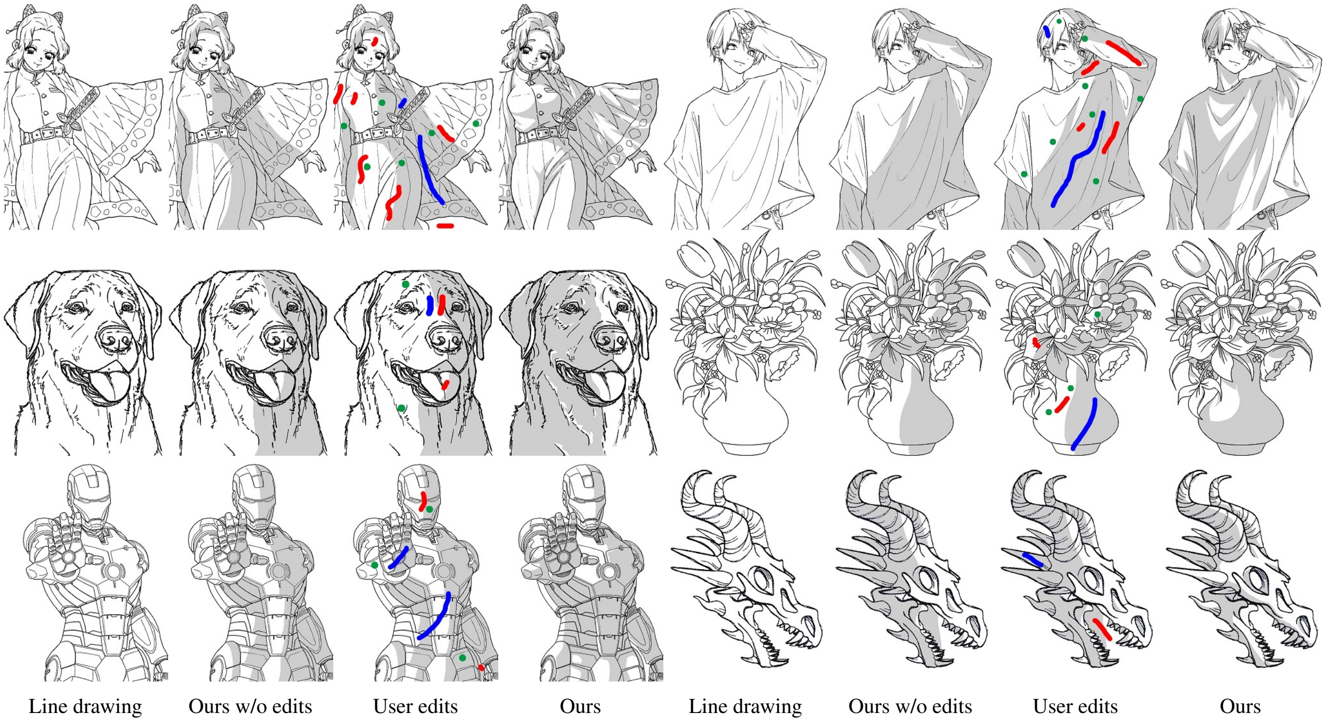


Figure 5. **Examples of interactive shadow drawing.** Zoom in to see details of the shadows and user edits. 30 more results are presented in the supplement. Notably, we have dilated the user scribbles for 3 pixels for a clearer visualization. *Artworks used with artist permissions.*

ensure the robustness and artistic faithfulness.

Data from real artists. We provide 1670 shadow samples drawn by 12 actual artists (Fig. 4-(a)). We search the key word "line drawing" in internet illustration platforms Pixiv [33] and Danbooru [14] to sample 10,000 line drawings. We then invite the 12 artists to select their interested line drawings and choose their preferred shadow directions. Afterwards, they draw the target shadows according to their artistic decisions and perceptual understandings. In this way, we collect 1670 high-quality shadow samples that captures the perceptions and designs of artists.

Data from rendering engine. We use non-photorealistic rendering (NPR) techniques to obtain line art and shadow pairs. To be specific, we search the key word "free" in *Unity 3D Assets Store* and download 471 random 3D prefabs. We import them to the rendering engine Blender [13] and write a NPR script to generate 25,413 line art and shadow pairs at random shadow directions (Fig. 4-(b)).

Data from shadow extraction. We sample 300,000 random digital paintings from Danbooru dataset [14] and Pixiv [33] (Fig. 4-(c)). We use auto inking method [39] to extract line arts, and use intrinsic imaging method [9] (enhanced with [51] and [10]) to decompose reflectance and illumination maps. We then perform a shadow voting using OTSU algorithm [29] to obtain the shadow, and use the Barron&Malik model [3] to estimate the shadow direction. After that, we manually remove 8,049 pairs with obviously low quality, and acquire the remaining 291,951 qualified pairs.

Training schedule. Our proposed training schedule consists of two phases: (1) Firstly, we pre-train the models with the extracted large-scale shadows for 20 epochs and with the rendered shadows for 15 epochs. (2) Afterwards, as a fine-tuning, we train the models with the high-quality shadows from real artists for 10 epochs. In this way, we achieve a robust model that not only generalizes to diverse inputs but also learns from real-artist data to produce shadows that are faithful to the understanding and willingness of real artists.

4. Evaluation

4.1. Experimental setting

Implementation details. Our framework is trained using the Adam optimizer [22] with a learning rate of $\mathrm{lr} = 10^{-5}$, $\beta = 0.5$, at batch size 8. Training samples are randomly cropped to be 256×256 pixels and augmented with random left-and-right flipping. As the shadow model is fully convolutional, it receives adjustable resolutions in inference.

Hyper-parameters. The proposed and recommended configuration is $\lambda_a = 1.0$, $\kappa = 0.1$, and $\lambda_n = 0.5$.

Compared methods. We test several shadow generation methods of (1) the generic model Pix2Pix [21] trained on our dataset with the same training schedule as ours; (2) the typical data-driven normal-based method DeepNormal [19] (official implementation); (3) the interactive method

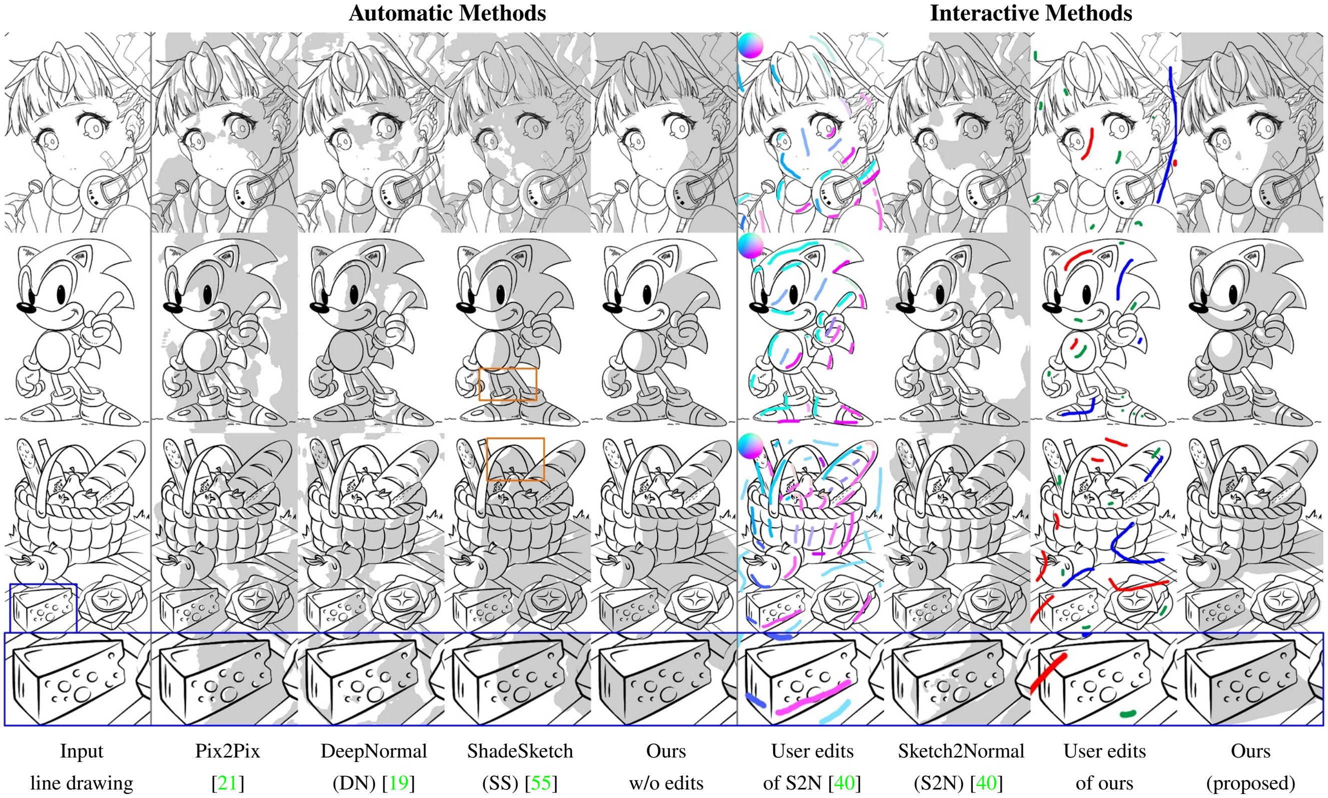


Figure 6. **Comparisons to possible alternative methods.** 15 more full-resolution comparisons are available in the supplementary material.

Sketch2Normal [40] (official method trained with the same scribble shapes as ours); (4) the state-of-the-art shadow generating method ShadeSketch [55] (official open-sourced codes); (5) our application without user edits (in this case we input same shadow directions as other methods when compared to them); and (6) our interactive application.

Testing samples. The tested images are Pixiv [33] line drawings and in-the-wild internet line arts. We make sure that all tested images are unseen from the training dataset.

4.2. Qualitative results

Interactive editing. We present examples of interactive shadow drawing in Fig. 5, and 30 additional results in the supplement. We can see that the users can work with our tools to achieve various shadow effects in diverse drawing topics, *e.g.*, human, animal, plant, robot, *etc*.

Comparison to previous methods. We present comparisons with both the automatic methods [21, 19, 55] and the interactive method [40] in Fig. 6, and 15 additional comparisons in the supplementary material. We can see that Pix2Pix [21] fails in achieving usable results, DeepNormal [19] tends to output shadows with severe distortions. The results of ShadeSketch [55] is better than [21] and [19], but it has difficulty in addressing detailed areas, *e.g.*, the mouse legs and the handrails for baskets (as marked in orange rectangles in Fig. 6). Sketch2Normal [40] yields low-quality shadows, despite the adequately given user scribbles. Our approach, regardless of whether to receive user edits or not, produces clean and practically usable shadows.

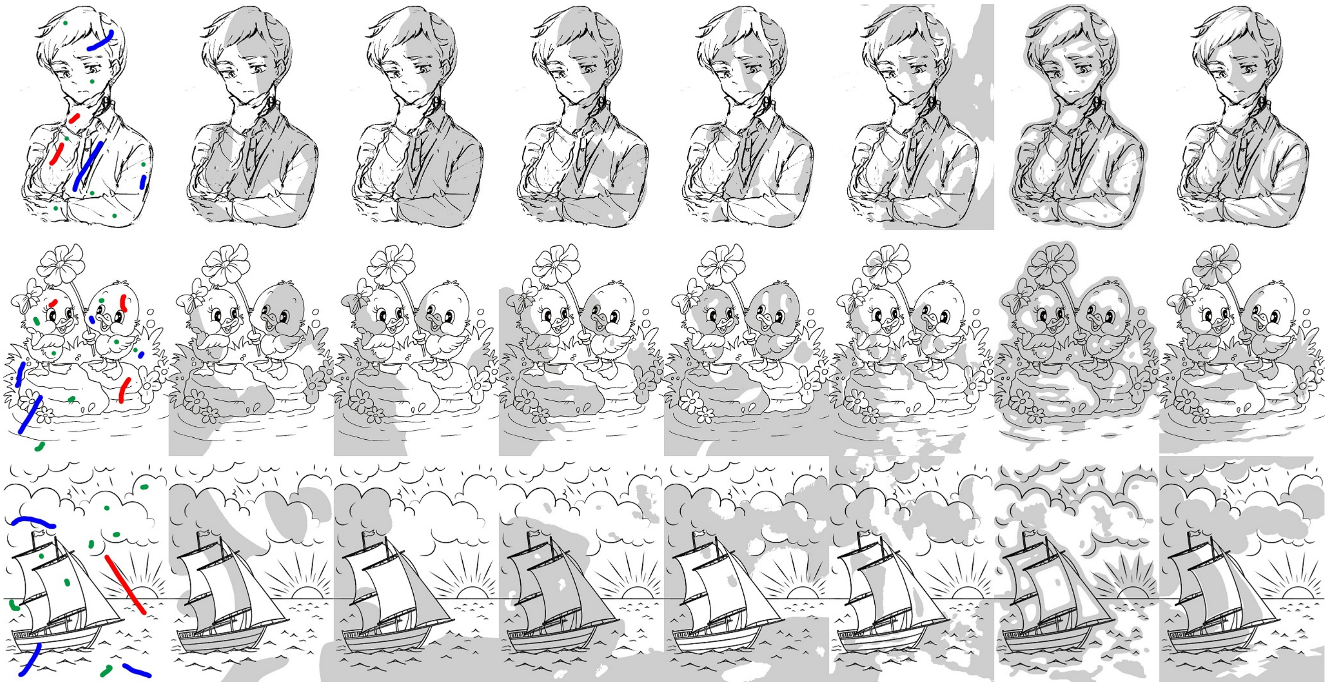
4.3. User study

Participant. The user study involves 15 persons: 10 non-artist amateurs and 5 professional artists. Each artist has at least two years of digital painting experience.

Setup. We sample 52 unseen line drawings from Pixiv [33], and then assign each line drawing to 3 random users targeted to 3 methods: a commercial tool (Adobe PhotoShop), our approach, and the baseline interactive method [40]. We also use 4 fully-automatic methods [19, 40, 21, 55] and the automatic mode of our method to generate shadows for each image. We ensure that any image is assigned to each user at most once to avoid users being trained for specific instances.

User guideline. When drawing shadows interactively, we inform the users that "your time consumption will be recorded and please draw at your normal speed". After they are finished, the users are also shuffled to rank the shadows of automatic methods [19, 21, 55] and the automatic outputs of ours. We ask users the question — "Which of the following shadow do you prefer most to use in your daily digital painting? Please rank according to your preference."

Evaluation metric. We use the Time Consumption (TC) as speed metric. We record the precise drawing minutes, and



(a) Input line drawing (b) W/o shadow brush (c) W/o shadow boundary brush (d) W/o global shadow generator (e) W/o balanced sampling $f_b(\cdot)$ (f) W/o anisotropic penalty λ_a (g) W/o norm regulation λ_n (h) Proposed full method

Figure 7. **Ablative study.** We study the impact of each individual component within our framework by removing components one-by-one.

Time t (minutes)	$t < 5$	$5 \leq t < 10$	$10 \leq t < 15$	$15 \leq t < 20$	$t \geq 20$
Commercial tool	0.00%	3.84%	28.84%	53.84%	13.46%
Ours	51.92%	46.15%	1.92%	0.00%	0.00%

Table 1. **Time Consumption (TC).** We compare the time consuming of a typical commercial tool (Adobe PhotoShop) and ours. We visualize the time consumption of 52 shadow drawing cases, *e.g.*, in "ours" row and "$t < 5$" col, the "51.92%" means that the time consumption of our method is less than 5 minutes in 51.92% cases.

Method	Pix2Pix [21]	DN [19]	S2N [40] (auto)	SS [55]	Ours (auto)
AHR ↓	4.53 ± 0.60	2.81 ± 0.76	4.19 ± 0.96	**2.44 ± 0.63**	**1.01 ± 0.13**

Table 2. **Average Human Ranking (AHR).** We present the ranking results of the user study. The arrow (↓) indicates that lower is better. Top 1 (or 2) score is marked in blue (or red).

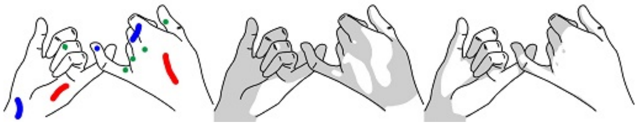
split the time consumption into intervals of five minutes. We also use the Average Human Ranking (AHR) as preference metric. For each line drawing, the users rank the results of the 5 methods from 1 to 5 (lower is better). Afterwards, we calculate the average ranking obtained by each method.

Time consumption analysis. The time data are reported in Table 1. We can see that in a dominant majority of cases, our method consumes less than 10 minutes, while in most cases the commercial tool (Adobe PhotoShop) consumes more than 15 minutes. Besides, we report that the average time consuming of ours is 5.35 minutes while the commercial tool is 16.58 minutes, indicating a $3.1\times$ speed up. See also the supplementary material for more detailed data.

Result. The user preferences are reported in Table 2. We have several interesting discoveries: (1) Our framework, even in automatic mode without any user edits, outperforms the secondly best method by a large margin of $1.43/5$. (2) Zheng's approach [55] reports the secondly best score. (3) The two normal-based methods [19, 40] reports similar perceptual quality, with [19] slightly better than [40], despite that [40] receives interactive edits while [19] is automatic.

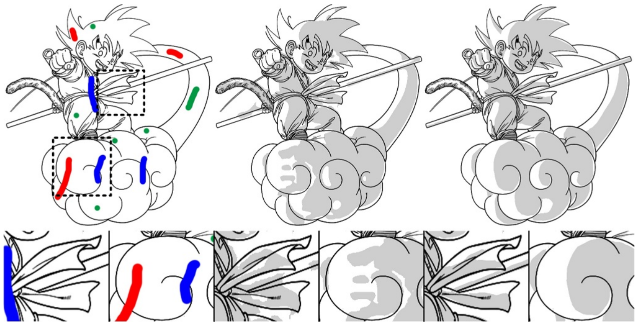
4.4. Ablative study

As shown in Fig. 7, our ablative study consists of the following experiments: (1) We remove the shadow brush and train our framework without red and blue scribbles. We can see that, in absence of the shadow brush, the shadow boundary brush cannot control the shadow locations by itself, resulting in many undesired shadows in the outputs (Fig. 7-(b)). (2) We remove the shadow boundary brush and train our framework without green scribbles. We can see that, without the help of shadow boundary brush, the shadow shape is out of control and users cannot implement their wanted shadow appearances (Fig. 7-(c)). (3) We remove the global shadow generator and train the shadow branch of our neural architecture without global shadow direction embedding. We can see that the global and local shadows becomes inconsistent and distorted (Fig. 7-(d)). (4) We train without the bivariate nor-



(a) Input (b) Uniform $f_b'(\cdot)$ (c) Proposed $f_b(\cdot)$

Figure 8. **Influence of different sampling distribution for f_b.** We compare the proposed bivariate normal distribution sampling and a common alternative uniform random sampling. *Artwork used with artist permission.*



(a) Input (b) $\lambda_a = 0.05$ (c) $\lambda_a = 1.0$

Figure 9. **Influence of the anisotropic penalty weight λ_a.** We visualize the outputs of our method with different anisotropic penalty weight λ_a. *Artwork used with artist permission.*

mal distribution sampling f_b, and instead, we simply sample random pixels as the starting position of training scribbles. We can see that the resulting shadows become severely unbalanced and defective (Fig. 7-(e)). (5) If trained without the anisotropic penalty λ_a, the neural networks fail in achieving sharp and smooth shadow boundaries, resulting in noisy outputs (Fig. 7-(f)). (6) If trained without the shadow direction norm regulation λ_n, the neural networks fail in recognizing appropriate shadow directions, and tends to output collapsed shadows surrounding input lines (Fig. 7-(g)). (7) The full framework suppresses these types of artifacts and achieves a satisfactory balance over the shadow location, shape, and appearance (Fig. 7-(h)).

Influence of hyper-parameters. We study different weights for the anisotropic λ_a and the norm λ_w in Fig. 8 and 9. We can see that a too small λ_a causes boundary distortions and a too small λ_n causes shadow direction defects.

Robustness and generalization. We showcase the robustness in Fig. 10 with a challenging complicated line drawing. We also present a case where our framework is generalized to another art form in Fig. 11. See also the supplementary material for results with more diverse contents and topics.

5. Conclusion

We propose a digital painting application to generate shadows on line drawings, with three tools of the shadow brush, shadow boundary brush, and global shadow generator. We train hierarchical neural networks with a collected large-scale dataset of both synthesized data and real data drawn by artists. User study shows that our tools can speed up the shadow drawing process and can achieve practically usable shadows for the daily work of artists. Our dataset will be made publicly available to facilitate related techniques.

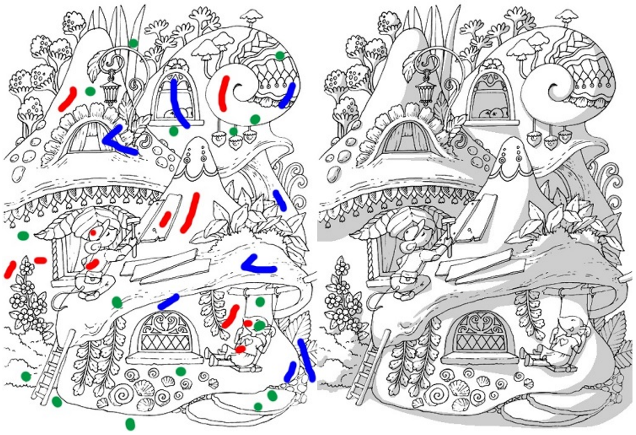
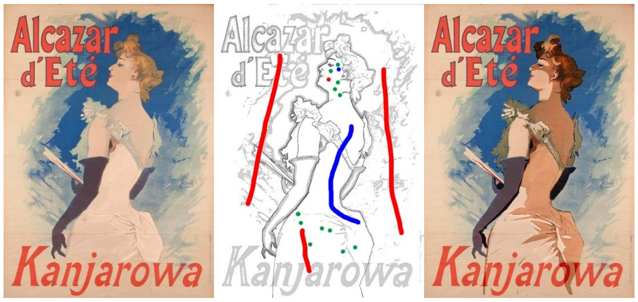


Figure 10. **Robustness to complicated line drawing.** We present a challenging case where the input line drawing is complicated and detailed. *Artwork used with artist permission.*



(a) Input (b) User edit (c) Ours

Figure 11. **Generalization to other art form.** We filter the left artwork to get the middle sketch and the user use our tools to achieve the right blended result. *Jardin de Paris, public domain.*

6. Acknowledgments

This work is conducted at Style2Paints Research (S2PR). This work is supported by National Natural Science Foundation of China Nos 61972059, 61773272, 61602332; Natural Science Foundation of the Jiangsu Higher Education Institutions of China No 19KJA230001, Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University No93K172016K08; the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).This work was partially supported by Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

[1] Maury Aaseng, Bob Berry, Jim Campbell, Dana Muise, and Joe Oesterle. *The Art of Comic Book Drawing*. Walter Foster Publishing, 1995.

[2] J.T. Barron and J. Malik. Color constancy and intrinsic images and shape estimation. *ECCV*, 2012. 2

[3] Jonathan T. Barron and Jitendra Malik. Shape, illumination, and reflectance from shading. *TPAMI*, 2015. 5

[4] H. G. Barrow and J. M. Tenenbaum. Recovering intrinsic scene characteristics from images. In A. Hanson and E. Riseman, editors, *Computer Vision Systems*, pages 3–26. Academic Press, 1978. 2

[5] S. Beigpour, M. Serra, J. van de Weijer, R. Benavente, M. Vanrell, O.Penacchio, and D.Samaras. Intrinsic image evaluation on synthetic complex scenes. *ICIP*, 2013. 2

[6] Sean Bell, Kavita Bala, and Noah Snavely. Intrinsic images in the wild. *ACM Transactions on Graphics*, 33(4), 2014. 2

[7] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. OpenSurfaces: A richly annotated catalog of surface appearance. *ACM Transactions on Graphics*, 32(4), 2013. 2

[8] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. Material recognition in the wild with the materials in context database. *CVPR*, 2015. 2

[9] Sai Bi, Xiaoguang Han, and Yizhou Yu. An l1 image transform for edge-preserving smoothing and scene-level intrinsic decomposition. *ACM Trans. Graph.*, 34(4), July 2015. 5

[10] Robert Carroll, Ravi Ramamoorthi, and Maneesh Agrawala. Illumination decomposition for material recoloring with consistent interreflections. In *ACM Transactions on Graphics*. ACM Press, 2011. 5

[11] Jiansheng Chen, Guangda Su, Jinping He, and Shenglan Ben. Face image relighting using locally constrained global optimization. *ECCV*, 2010. 2

[12] Yang Chen, Yu-Kun Lai, and Yong-Jin Liu. CartoonGAN: Generative adversarial networks for photo cartoonization. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, jun 2018. 3

[13] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 5

[14] DanbooruCommunity. Danbooru2017: A large-scale crowdsourced and tagged anime illustration dataset, 2018. 5

[15] Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. Acquiring the reflectance field of a human face. *the 27th annual conference on Computer graphics and interactive techniques*, 2000. 2

[16] Marek Dvorožňák, Saman Sepehri Nejad, Ondřej Jamriška, Alec Jacobson, Ladislav Kavan, and Daniel Sýkora. Seamless reconstruction of part-based high-relief models from hand-drawn images. In *Proceedings of International Symposium on Sketch-Based Interfaces and Modeling*, 2018. 2

[17] P.V. Gehler, C. Rother, M. Kiefel, L. Zhang, and B. Scholkopf. Recovering intrinsic images with a global sparsity prior on reflectance. *NIPS*, 2011. 2

[18] Roger Grosse, Micah K Johnson, Edward H Adelson, and William T Freeman. Ground truth dataset and baseline evaluations for intrinsic image algorithms. *International Conference on Computer Vision*, 2009. 2

[19] Matis Hudon, Rafael Pages, Mairead Grogan, and Aljosa Smolic. Deep normal estimation for automatic shading of hand-drawn characters. *ECCV*, 2018. 2, 5, 6, 7

[20] Matis Hudon, Rafael Pagés, Mairéad Grogan, Jan Ondřej, and Aljoša Smolić. 2D Shading for Cel Animation. In Tunç Aydın and Daniel Sýkora, editors, *Expressive: Computational Aesthetics, Sketch-Based Interfaces and Modeling, Non-Photorealistic Animation and Rendering*. ACM, 2018. 2

[21] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017. 5, 6, 7

[22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *Computer Science*, 2014. 5

[23] Balazs Kovacs, Sean Bell, Noah Snavely, and Kavita Bala. Shading annotations in the wild. *CVPR*, 2017. 2

[24] Chengze Li, Xueting Liu, and Tien-Tsin Wong. Deep extraction of manga structural lines. *ACM Transactions on Graphics*, 36(4), 2017. 3

[25] Chenxi Liu, Enrique Rosales, and Alla Sheffer. Strokeaggregator: Consolidating raw sketches into artist-intended curve drawings. *ACM Transactions on Graphics*, 2018. 3

[26] Xueting Liu, Tien-Tsin Wong, and Pheng-Ann Heng. Closure-aware sketch simplification. *ACM Transactions on Graphics*, 34(6):168:1–168:10, November 2015. 3

[27] Whyt Maga. Drawing and coloring: A cel-shading tutorial. *https://www.youtube.com/watch?v=Xkek0JuorGE*, 2018. 1

[28] Wojciech Matusik, Matthew Loper, and Hanspeter Pfister. Progressively-refined reflectance functions from natural illumination. *Eurographics Workshop on Rendering*, 2004. 2

[29] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, jan 1979. 5

[30] Pieter Peers and Philip Dutre. Inferring reflectance functions from wavelet noise. *the Sixteenth Eurographics conference on Rendering Techniques*, 2005. 2

[31] Pieter Peers, Dhruv K Mahajan, Bruce Lamond, Abhijeet Ghosh, Wojciech Matusik, Ravi Ramamoorthi, and Paul Debevec. Compressive light transport sensing. *ACM Transactions on Graphics*, 2009. 2

[32] Pieter Peers, Naoki Tamura, Wojciech Matusik, and Paul Debevec. Post-production facial performance relighting using reflectance transfer. *ACM Transactions on Graphics*, 2007. 2

[33] pixiv.net. pixiv. *pixiv*, 2007. 5, 6

[34] Patsorn Sangkloy, Jingwan Lu, Chen Fang, Fisher Yu, and James Hays. Scribbler: Controlling deep image synthesis with sketch and color. *CVPR*, 2017. 3

[35] M. Serra, O. Penacchio, R. Benavente, and M. Vanrell. Names and shades of color for intrinsic image estimation. *CVPR*, 2012. 2

[36] Jianbing Shen, Xiaoshan Yang, Yunde Jia, and Xuelong Li. Intrinsic images using optimization. *CVPR*, 2011. 2

[37] Edgar Simo-Serra, Satoshi Iizuka, and Hiroshi Ishikawa. Mastering Sketching: Adversarial Augmentation for Structured Prediction. *ACM Transactions on Graphics*, 37(1), 2018. 3

[38] Edgar Simo-Serra, Satoshi Iizuka, and Hiroshi Ishikawa. Real-time data-driven interactive rough sketch inking. *ACM Transactions on Graphics*, 2018. 3

[39] Edgar Simo-Serra, Satoshi Iizuka, Kazuma Sasaki, and Hiroshi Ishikawa. Learning to Simplify: Fully Convolutional Networks for Rough Sketch Cleanup. *ACM Transactions on Graphics*, 35(4), 2016. 3, 5

[40] Wanchao Su, Dong Du, Xin Yang, Shizhe Zhou, and Hongbo Fu. Interactive sketch-based normal map generation with deep neural networks. *ACM on Computer Graphics and Interactive Techniques*, 1(1):1–17, jul 2018. 2, 6, 7

[41] Daniel Sýkora, Jan Buriánek, and Jiří Žára. Sketching cartoons by example. In *Proceedings of Eurographics Workshop on Sketch-Based Interfaces and Modeling*, pages 27–34, 2005. 3

[42] Daniel Sykora, John Dingliana, and Steven Collins. Lazy-Brush: Flexible painting tool for hand-drawn cartoons. *Computer Graphics Forum*, 28(2), 2009. 3

[43] Daniel Sykora, Ladislav Kavan, Martin Cadik, Ondrej Jamriska, Alec Jacobson, Brian Whited, Maryann Simmons, and Olga Sorkine-Hornung. Ink-and-ray: Bas-relief meshes for adding global illumination effects to hand-drawn characters. *ACM Transactions on Graphics*, 33(2):1–15, apr 2014. 2

[44] TaiZan. Paintschainer tanpopo. *PreferredNetwork*, 2016. 3

[45] Xinrui Wang and Jinze Yu. Learning to cartoonize using white-box cartoon representations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 3

[46] J. Weickert. *Anisotropic Diffusion in Image Processing*. Uni-saarland, 1998. 1

[47] Wikipedia. Bivariate normal distribution. *https://en.wikipedia.org/wiki/Multivariate_normal_distribution#Bivariate_case*, 2020. 3

[48] Ran Yi, Yong-Jin Liu, Yu-Kun Lai, and Paul L. Rosin. AP-DrawingGAN: Generating artistic portrait drawings from face photos with hierarchical GANs. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2019. 3

[49] Ran Yi, Yong-Jin Liu, Yu-Kun Lai, and Paul L Rosin. Unpaired portrait drawing generation via asymmetric cycle mapping. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR '20)*, 2020. 3

[50] Ran Yi, Mengfei Xia, Yong-Jin Liu, Yu-Kun Lai, and Paul L. Rosin. Line drawings for face portraits from photos using global and local structure based GANs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2020. 3

[51] Lvmin Zhang, Chengze Li, Yi JI, Chunping Liu, and Tien tsin Wong. Erasing appearance preservation in optimization-based smoothing. In *European Conference on Computer Vision (ECCV)*, 2020. 5

[52] Lvmin Zhang, Chengze Li, Tien-Tsin Wong, Yi Ji, and Chunping Liu. Two-stage sketch colorization. In *ACM Transactions on Graphics*, 2018. 3

[53] Lvmin Zhang, Edgar Simo-Serra, Yi Ji, and Chunping Liu. Generating Digital Painting Lighting Effects via RGB-space Geometry. *Transactions on Graphics (Presented at SIGGRAPH)*, 39(2), 2020. 2

[54] Richard Zhang, Jun-Yan Zhu, Phillip Isola, Xinyang Geng, Angela S Lin, Tianhe Yu, and Alexei A Efros. Real-time user-guided image colorization with learned deep priors. *ACM Transactions on Graphics*, 9(4), 2017. 3

[55] Qingyuan Zheng, Zhuoru Li, and Adam Bargteil. Learning to shadow hand-drawn sketches. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 2, 6, 7