SHAD3S: A model to Sketch, Shade and Shadow

Raghav Brahmadesam Venkataramaiyer, Abhishek Joshi, Saisha Narang, Vinay P. Namboodiri
{bvraghav,abhishekjoshi,saisha,vinaypn}@iitk.ac.in
INDIAN INSTITUTE OF TECHNOLOGY KANPUR

Abstract

Hatching is a common method used by artists to accentuate the third dimension of a sketch, and to illuminate the scene. Our system SHAD3S[1] *attempts to compete with a human at hatching generic three-dimensional (*3D*) shapes, and also tries to assist her in a form exploration exercise. The novelty of our approach lies in the fact that we make no assumptions about the input other than that it represents a* 3D *shape, and yet, given a contextual information of illumination and texture, we synthesise an accurate hatch pattern over the sketch, without access to* 3D *or pseudo* 3D*. In the process, we contribute towards* a) *a cheap yet effective method to synthesise a sufficiently large high fidelity dataset, pertinent to task;* b) *creating a pipeline with conditional generative adversarial network (*CGAN*); and* c) *creating an interactive utility with* GIMP*, that is a tool for artists to engage with automated hatching or a form-exploration exercise. User evaluation of the tool suggests that the model performance does generalise satisfactorily over diverse input, both in terms of style as well as shape. A simple comparison of inception scores suggest that the generated distribution is as diverse as the ground truth.*

1. Introduction

Sketches are a widely used representation for designing new objects, visualizing a 3D scene, for entertainment and various other use-cases. There have been a number of tools that have been developed to ease the workflow for the artists. These include impressive work in terms of non-photorealistic rendering (NPR) that obtain sketches given 3D models [19, 13], others that solve for obtaining 3D models given sketch as an input using interaction [16, 8], and several others that aim to ease the animation task [1, 25, 11]. However, these works still rely on the input sketch being fully generated by the artist. One of the important requirements to obtain a good realistic sketch is the need to provide the hatching that conveys the 3D and illumination information. This is a time consuming task and requires effort from the artist to generate the hatching that is consistent with the 3D shape and lighting for each sketch drawn. Towards easing this task, we propose a method that provides realistic hatching for an input line drawn sketch that is consistent with the underlying 3D and illumination.

Though recent works have tried to address problems associated with shading sketch [53], however, to the best of our study we couldn't find any prior work addressing the problem we intend to solve. Here we hope to leverage deep learning to decode and translate the underlying 3D information amongst a rasterised set of 2D lines. Deep learning has been promising to solve many interesting and challenging problems [14, 35, 24, 8]. Moreover, it is well known that deep learning algorithms are data intensive. There exist datasets [8, 53] which aim to solve 3D-related problems like predicting the illumination, and mesh inference from line drawings, but they are limited in their diversity and level of pertinence (see § 2.3). This motivates us to contribute the SHAD3S dataset for *Sketch-Shade-Shadow* like tasks (see § 4). Hereby, we attempt to help the research community to bring closer, the two domains of graphics and vision.

With the aim of creating artistic shadows with hatch-patterns for a given contour drawing of a 3D object, under a user specified illumination, we define this research problem as an attempt *to learn a function*, that maps a combination of — *a)* hand-drawn 3D objects; *b)* illumination conditions; and *c)* textures — to a space of completed sketches with shadows manifested as hatch-patterns. See § 3.

Our proposed pipeline is based on training a conditional generative adversarial network (CGAN) [35] with a data-set of 3D shape primitives and aims to model the problem as one of contextual scene completion. While this approach has been widely explored in the image domain [32, 24], the challenge is to ensure that we are able to obtain convincing results in sketch domain using a sparse set of lines. In order to solve this we train our model using a novel SHAD3S dataset explained further in § 4. These are rendered to be consistent with the illumination and 3D shape. We include the context required for solving the problem in the input. Once this is learned it is possible to automate the sketch

[1]`https://bvraghav.com/shad3s/`— The project page; hosted with further resources.

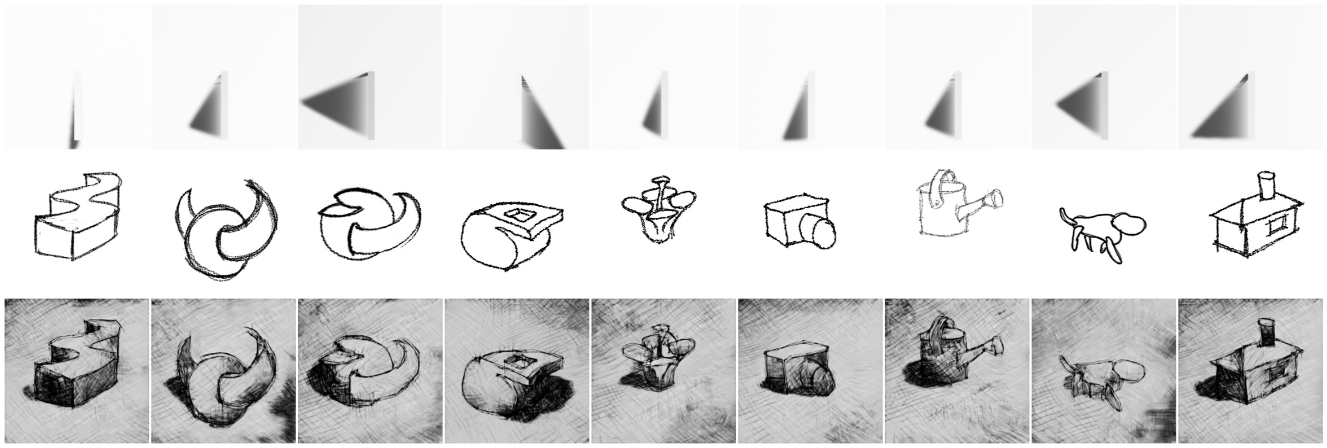
Figure 1. SHAD3S: a Model for Sketch, Shade and Shadow — is a completion framework that provides realistic hatching for an input line drawn sketch that is consistent with the underlying 3D and a user specified illumination. The figure above shows choice of illumination conditions on the top row, user sketches in the middle row, and the completion suggestions by our system. Note that the wide variety of shapes drawn by the user, as well as the wide variations in brush styles used by the user were not available in the training set.

generation using the proposed CGAN model. A glimpse of our results can be seen in Fig. 1, that once again reinforces the significance of sufficiently large dataset in the context of deep learning, and its generalizability.

A natural approach given a dataset would be to train a regression based system that would aim to generate the exact information that is missing. This can be achieved using reconstruction based losses. The drawback of such an approach is that the resultant generation would be blurred as it averages over the multiple outputs that could be generated. In contrast, our adversarial learning based approach allows us to generate a sharp realistic generation that we show is remarkably close to the actual hatch lines.

The system has been tested for usability and deployment under casual as well as involved circumstances among more than 50 participants, the qualitative results of which are selectively displayed here (see § 6), and a larger subset has been made available as supplement. We would like to mention that the samples for which the system has been tested by the user *are very different* from the distribution of samples for which it was trained on. It is able to generalize and produce satisfactory sketches as shown in Fig. 1. Through this paper we provide, to the best of our knowledge, the first such tool that can be used by artists to automate the tedious task of hatching each sketch input. The tool and the synthetic dataset (described in § 4) are available for public use at the project page[2].

Summary. To provide an overview of our work, we summarise our main contributions as follows. *Firstly,* we define an extremely inexpensive method to generate sufficiently large number of high fidelity stroke data, and contribute the resultant novel public dataset.

Secondly, we define simple method to inject data from three different modalities, namely: *i)* sketch representing 3D information; *ii)* rendered canonical geometry representing illumination; and *iii)* a set of hand drawn textures representing multiple levels of illumination — these three may be plugged into any well established deep generative model.

Finally, we push boundaries for artists through an interactive tool to automate a tedious hatching task, and to serve as a form-exploration assistant.

2. Relevant Works

As the proposed method is the first work to address the task of automating hatch generation from sketches, there is no directly related work to the best of our knowledge. However, there are two prominent lines of approach, namely stylization and sketch-based modeling that are related to our problem. In contrast to these approaches, we aim to address the task as one of contextual information completion and therefore the proposed approach differs substantially from these approaches.

2.1. Stylization

An undercurrent, is arguably visible in a series of works following Siggraph '88, where in a panel proceedings [33], Hagen said,

> The goal of effective representational image making, whether you paint in oil or in numbers, is to select and manipulate visual information in order to direct the viewer's attention and determine the viewer's perception.

We see this in the initial works [15, 40, 6] focusing on attribute sampling from user input to recreate the image with

[2]`https://bvraghav.com/shad3s/`

arguably a different visual language and accentuation and offer interactivity as a medium of control. Thereby followed, a shift towards higher order controls, to achieve finer details, for example temporal coherence [31], modelling brush strokes [17], and space-filling [43]. Space-filling had been buttressed, on one hand with visual-fixation data and visual-acuity [7, 41], and on the other with semantic units inferred using classical vision techniques [51]. The finer levels of control, in these works, allowed for interactivity with detail in different parts of the synthesised image.

Image Analogies had popularised the machine learning framework for stylising images, with analogous reasoning $A : A' :: B : B'$, to synthesise B', using image matching [18]; to achieve temporal coherence [1]; using synthesis over retrieved best-matching feature [11, 10, 25]. The problems thus formulated, allowed use of high level information as a reference to a style, for example a Picasso's painting.

Deep Learning Techniques have recently documented success in similar context, using Conditional Generative Adversarial Networks (CGAN) [35], and its variants using sparse information [24]; using categorical data [48]; or for *cartoonization* of photographs [5].

Researchers have recently contributed to line drawings problem pertaining to sketch simplification [46, 44], line drawing colorization [12, 27, 52] and line stylization [29]. Sketch simplification [46, 44] aims to clean up rough sketches by removing redundant lines and connecting irregular lines. Researchers take a step ahead to develop a tool [45] to improve upon sketch simplification by incorporating user input. It facilitates the users to draw strokes indicating where they want to add or remove lines, and then the model will output a simplified sketch. *Tag2Pix* [27] aims to use GANs based architecture to colorize line drawing. *Im2pencil* [29] introduce a two-stream deep learning model to transform the line drawings to pencil drawings. Among the relevant literature we studied, a recent work [53] seems to be most related to our approach, where authors propose a method to generate detailed and accurate artistic shadows from pairs of line drawing sketches and lighting directions.

In general, it has been argued that models over the deep learning paradigm handle more complex variations, but they also require large training data.

2.2. Sketch-based Modeling

Interactive Modeling [50] introduced a set of rule-based shortcuts with the aim of creating *an environment for rapidly conceptualizing and editing approximate* 3D *scenes*. Sketching was integrated soon into the system with TEDDY and its relatives, [23, 22, 36], with prime focus on inflated smooth shapes as first class citizens, and sketch-based gestures for interactivity. Recently, SMARTCANVAS [54] extended the works relying on planar strokes.

Analytical Inference, for 3D reconstruction is another popular approach to modeling from sketches. With the context of vector line art, [34] inferred simple curved shapes, and [30] showed progress with complex compositions of planar surfaces. Recently, for illuminating sketches, [42, 21] had shown the effectiveness of normal-field inference through regularity cues, while [49] used isophotes to infer the normal-field.

Deep Learning Techniques, have more recently, shown substantial progress in inferring 3D models from sketches: a deep regression network was used to infer the parameters of a bilinear morph over a face mesh [16] training over the *FaceWarehouse* dataset [3]; a generative encoder-decoder framework was used to infer a fixed-size point-cloud [9] from a real world photograph using a distance metric, trained over *ShapeNet* dataset [4]; U-Net [38]-like networks were proven effective, to predict a volumetric model from a single and multiple views [8] using a diverse dataset, but this dataset *lacks pertinence to our task*.

2.3. How we stand out

Deep learning applications has seeped into many research domains. The profusion of data has helped the research community and has nicely complemented deep learning algorithms. However, there does seem to be lack of feasible, scalable, impressive in terms of both quality and quantity, and user-interpretable dataset in the graphics community specially artistic domain. There does seem to be a gap when we compare these datasets with popular datasets suitable for deep learning algorithms. For instance, ShadeSketch dataset [53] *doesn't offer much diversity* as it is limited to a specific domain and dataset has comparitively less number of images.

In our approach, we aim to solve the problem of generating hatches for sketches using our contextual image completion framework. The framework ensures that the relevant context required in terms of illumination information and textures is incorporated. We use generative adversarial networks to ensure realistic generation of hatch patterns that are indistinguishable from ground-truth hatch patterns that are generated by using accurate ground-truth 3D. Note that in our inference procedure, we make use only of the line drawings, the illumination and the texture context. We *do not* require access to 3D or psuedo-3D in order to generate the accurate hatch pattern. Further ways in which we differ are as follows.

Firstly, the problem of stylization inherently requires an information-dense image to start with. For an artist, it is evidently quicker as well as more intuitive, to create a basic

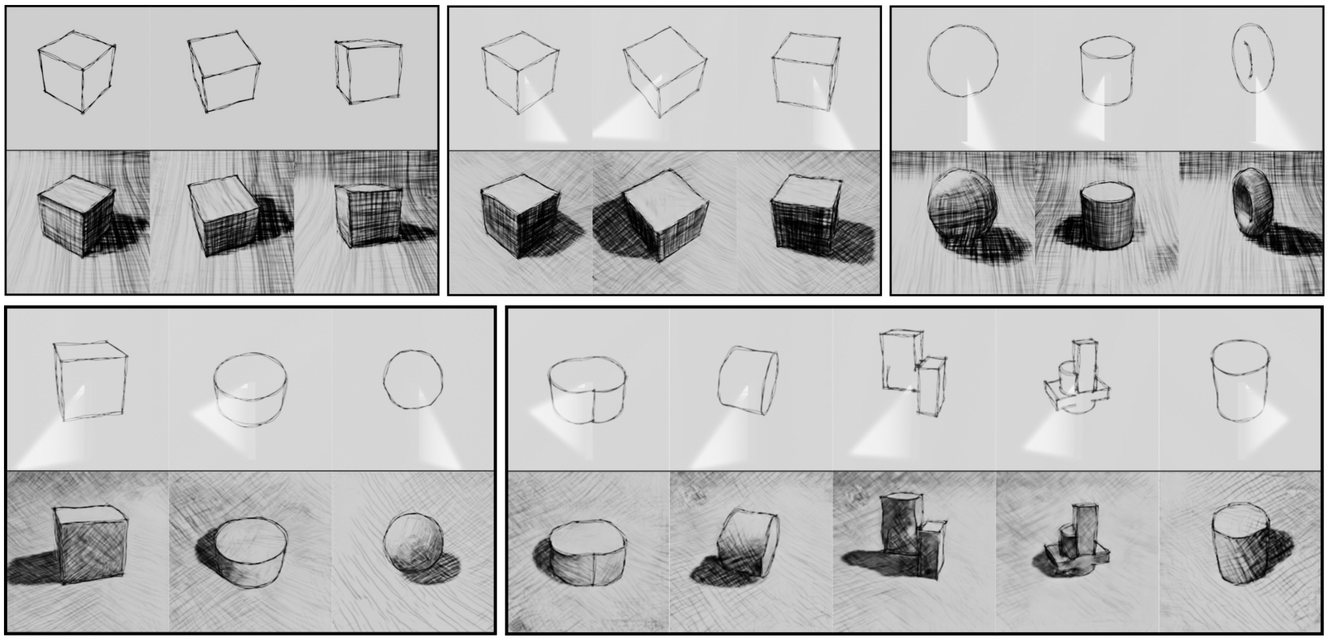

Figure 2. Progressive evaluation of DM:WB (see § 5) model varying camera pose, illumination, constituent geometry and texture. *Clockwise from top left.* POSE; POSE+LIT; POSE+LIT+SHAP; ALL; TXR. (Details at § 5.1.)

line-art, than to create a well rendered sketch; but line-art is sparse in nature.

Secondly, the heuristic based methods although coarsely capture the distribution, wherein different parameter values cater to different class of problems, they are vulnerable in fringe cases. Deep learning on the other hand has shown promise.

Finally, in decision making, it is pivotal for the designers, to control the stylization parameters of lighting and texture.

3. Problem formulation

The objective of this research is to learn a function $\mathcal{F} : C \times L \times T \rightarrow S$; where, C represents the space of hand-drawn contours for any 3D geometry; L represents an illumination analogy shown for a canonical geometry; T is a set of textures representing different shades; and S is the space of completed sketches with shadows manifested as hatch-patterns.

3.1. Model

To this end, we leverage the CGAN[14, 35] framework. If F_θ parameterised by θ, be a family of functions modelled as a deep network; $\mathbf{c} \in C$ be a line drawing, $\mathbf{l} \in L$ be the illumination analogy, $\mathbf{t} \in T$ be the set of tonal art maps (aka. textures), and $\mathbf{s} \in S$ be the completed sketch, then the reconstruction loss is defined as in Eq. (2). For diversity in generation task, F_θ is implemented with dropouts [28] as implicit noise. In order to bring closer the model-generated sketches and real data distribution, we define a discriminator D_ϕ parameterised by ϕ, as another family of functions modelled by a deep network. The adversarial loss, is thus defined as in Eq. (3). We hope the model to converge, alternating the optimisation steps for the minimax in Eq. (1). We call this formulation as **direct model**, illustrated in Fig. 3.

In the same spirit, since data generation is cheap (see § 4), we see an opportunity to make use of the illumination masks $\mathbf{m} \in M$ as an intermediate step for supervision using a **split model**, for which the losses $\mathcal{L}_{\mathrm{L}_1}$ and $\mathcal{L}_{\mathrm{adv}}$ in Eq. (1) are formulated as in Eqs. (4) and (5).

$$\theta^* = \arg\min_\theta \max_\phi \mathbb{E}_{\mathbf{c},\mathbf{l},\mathbf{t},\mathbf{s}\sim\mathrm{data}} \quad (1)$$
$$[\mathcal{L}_{\mathrm{L}_1}(\theta) + \lambda\mathcal{L}_{\mathrm{adv}}(\theta,\phi)]$$

Direct Model:

$$\widehat{\mathbf{s}} = F_\theta(\mathbf{c},\mathbf{l},\mathbf{t})$$
$$\mathcal{L}_{\mathrm{L}_1}(\theta) = \|\mathbf{s} - \widehat{\mathbf{s}}\|_1 \quad (2)$$
$$\mathcal{L}_{\mathrm{adv}}(\theta,\phi) = \log(D_\phi(\mathbf{s})) + \log(1 - D_\phi(\widehat{\mathbf{s}})) \quad (3)$$

Split Model:

$$\widehat{\mathbf{m}} = F_\theta^{(1)}(\mathbf{c},\mathbf{l})$$
$$\widehat{\mathbf{s}} = F_\theta^{(2)}(\widehat{\mathbf{m}},\mathbf{t})$$
$$\mathcal{L}_{\mathrm{L}_1}(\theta) = \|\mathbf{m} - \widehat{\mathbf{m}}\|_1 + \|\mathbf{s} - \widehat{\mathbf{s}}\|_1 \quad (4)$$
$$\mathcal{L}^{(1)}_{\mathrm{adv}}(\theta,\phi) = \log(D^{(1)}_\phi(\mathbf{m})) + \log(1 - D^{(1)}_\phi(\widehat{\mathbf{m}}))$$
$$\mathcal{L}^{(2)}_{\mathrm{adv}}(\theta,\phi) = \log(D^{(2)}_\phi(\mathbf{s})) + \log(1 - D^{(2)}_\phi(\widehat{\mathbf{s}}))$$
$$\mathcal{L}_{\mathrm{adv}}(\theta,\phi) = \mathcal{L}^{(1)}_{\mathrm{adv}}(\theta,\phi) + \mathcal{L}^{(2)}_{\mathrm{adv}}(\theta,\phi) \quad (5)$$

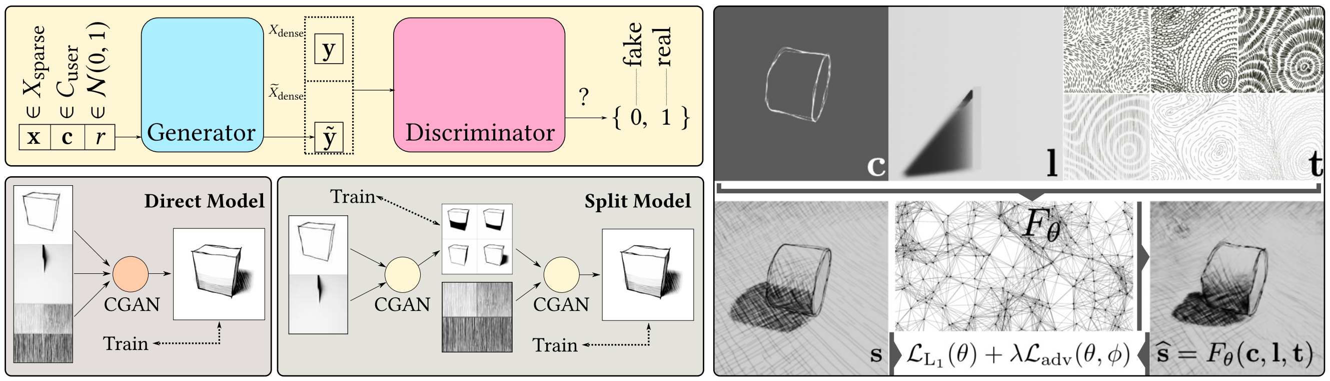

Figure 3. The overview of the our framework. *Clockwise from top*. Illustration of structure of GAN detailing injection of data from different modalities into the CGAN framework; Further detailed view of direct model; Coarse sketch of a direct model, Eqs. (2),(3),(1); Analogous sketch of a split model, Eqs. (4),(5),(1).

3.2. Architecture

Akin to earlier attempts [24], we utilise U-Net [38] with $\approx$ 12M as our base architecture. The only main difference being the use of *multi-modal bounding conditions* that are infused into the network by varying the number of input channels, namely the *sparse object outline sketch*, *illumination hint*, and *texture* (See Fig. 3). We experiment with alternate formulations of the model so that the complete sketch is predicted using a direct model Eqs. (2),(3), or by predicting an extra intermediate representation of illumination masks using a split model Eq. (4) (5). The discriminator is designed as a PatchGAN based classifier, which determines whether $N \times N$ patch in an image is real or fake.

Inspired by recent success of self-attention in adversarial nets [53], we also use a squeeze-and-excitation [20] based architecture for our model, F_θ in Eqs. (2),(3).

3.3. Remarks

We propose a conditional GAN framework to solve for the task. We adopt the baseline architecture [24], and further implement a model inspired by a recent advancement as an ablation study. However, our architecture is different and stands out from prior works in the way how muti-modal bounding conditions are being infused into the network. The model so designed can be trained in an end-to-end manner while satisfying the desired constraints.

As a general notion, deep neural networks are difficult to train and are sensitive to the choice of hyper-parameters. But our model doesn't introduce additional complex components, and thus allows us to take advantage of knowing the hyper-parameters of underlying base architecture. This also results in a conceptually clean and expressive model that relies on principled and well-justified training procedures. This is, to the best of our knowledge, the first attempt to solve the problem of generating hatches for sketches while incorporating multiple modalities as constraints.

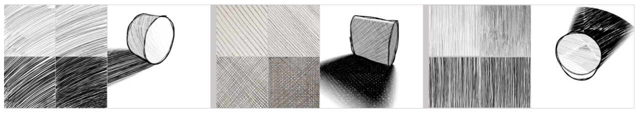

Figure 4. Three sketches rendered using three different objects with TAM's implemented using BLENDER.

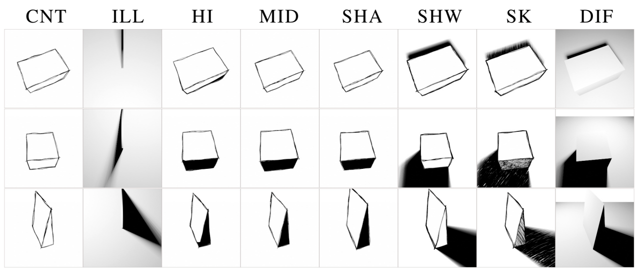

Figure 5. Few examples from the dataset. *Columns left to right*. CNT: Contour; ILL: Illumination; HI: Highlight mask; MID: Midtone mask; SHA: Shade mask (on object); SHW: Shadow mask; SK: Sketch render; DIF: Diffuse render.

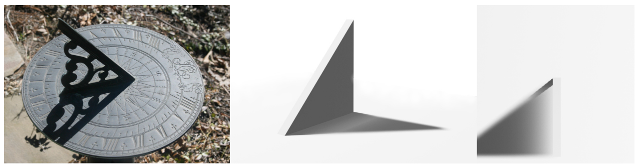

Figure 6. Gnomon in a sundial is used as canonical object to capture illumination information. *Left to right:* A sundial (*Courtesy: liz west* `https://flic.kr/p/EWBd4`); Perspective render of a gnomon; A top view of the same.

4. Dataset

We introduce the SHAD3S dataset (Fig. 4), where each data-point contains: *a)* a contour drawing of a 3D object; *b)* an illumination analogy; *c)* 3 illumination masks over *a*, namely *highlights, midtones* and *shades*; *d)* a shadow mask on the ground; *e)* a diffuse render; and *f)* a sketch render. Additionally, it contains a catalogue of 6 textures.

Geometry and render This is a fully synthetic dataset created from scratch using the following steps: *a)* Create a Constructive Solid Geometry (CSG)-graph using simple grammar over a finite set of eulidean solids; *b)* Create a mesh from CSG; *c)* Render the mesh. The illumination masks were created by thresholding a diffuse shader. Blender's [2] official freestyle tool [47] was used to render the contours. And sketch renders were created after implementing the tonal art maps [37] (see Fig. 4).

Since the geometry is also procedurally generated on the fly, we take the opportunity to progressively increase the complexity of geometry by varying the upper-bound of number of solids in the composition between 1 and 6. This assists in the progressive evaluation of the models.

Illumination and background Additionally, we also froze the information of illumination conditions in the form of a diffuse render of a canonical object resembling a gnomon (of a sundial). The rationale behind using this style of information is to provide information to image domain. We also attempted to use a background filled with hatch patterns, instead of transparent background, and analysed its effects on the model performance detailed out in § 5.

Textures as Tonal art maps For the purpose of this research, we follow the principle of tonal art maps [37], albeit in a low-fidelity version. To approximate the effect, we create an 6-tone cel-shading [26], including the pure white and pure black. For the rest of the four shades we map one texture corresponding to each tone on the object.

The similarity to tonal art maps lies in the fact that any hatch pattern map for a given tone should also be visible in all the tones darker than itself. To this effect, we collect samples of high resolution tonal art maps apriori, from artists. A random crop of these images is used by the model as input conditions (see Fig. 4)

All in all we create *six subsets*, each restricting the maximum number of solids in a CSG composition of a scene to be from *1 through 6*. Each subset is created from ~ 1024 distinct scenes. Each scene is rendered from ~ 64 different camera poses. The dataset thus contains $\sim 2^{16} \times 6$ data points with a resolution of $\sim 256 \times 256$.

5. Experimentation and results

To test our hypothesis we performed experiments on the three models, namely *a)* the direct model over U-Net architecture (DM); *b)* the split model over U-Net architecture (SP); *c)* the split model over squeeze-and-excitation architecture (SE). As an extension, we also studied the direct model trained over a dataset with background (DM:WB), and the split model trained over a dataset without shadows on ground (SP:WS). Further, the results are analysed qualitatively (§ 5.1), quantitatively (§ 5.2) and for generalizability of the model (§ 5.3).

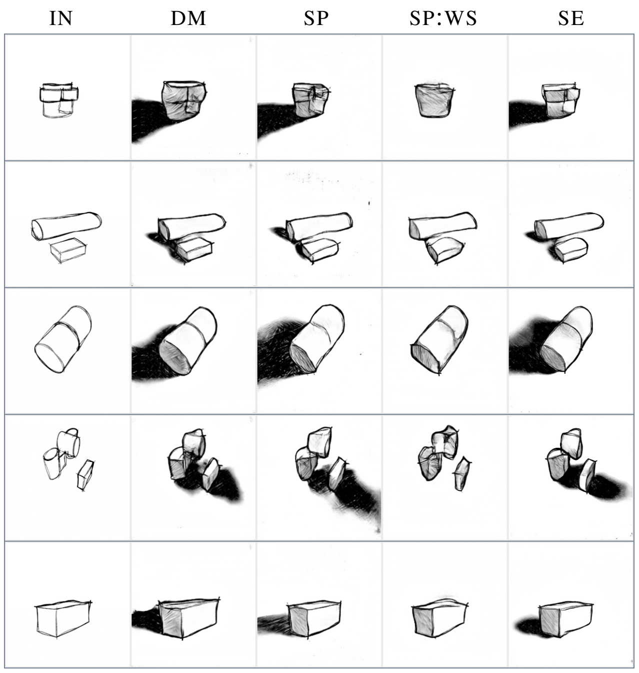

Figure 7. Comparative results for all models. *Columns left to right*. Input contour drawing; Corresponding evaluation with DM; With SP; With SP:WS; With SE.

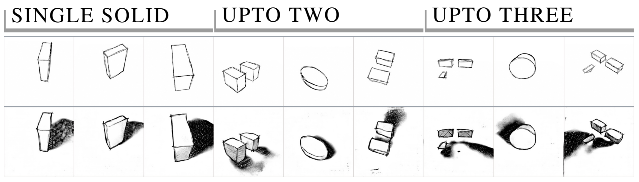

Figure 8. Progressive results for DM. Left three with single solid on scene; Middle three with upto two solids; and Rightmost three with upto three solids.

Table 1. Quantitative evaluation using PSNR, SSIM, Inference time (in ms), followed by inception scores of model predictions, juxtaposed against those of the dataset.

MODEL	PSNR	SSIM	Time	Pred. IS	GT IS
DM	55.72	0.347	13	4.25	5.51
SP	55.48	0.349	33	4.12	5.51
SE	55.90	0.376	426	4.35	5.51
SP:WS	58.92	0.287	36	3.71	5.27

Table 2. Progressive improvement in inception scores against an increase in dataset complexity

Max objects	1	2	3	4	5	6
GT IS	3.36	3.94	4.13	4.86	5.15	5.51

5.1. Qualitative Evaluation

Initially the a DM model was trained with low resolution datasets, starting with single solid in a scene, through to a combination of upto three solids in a scene. The indicative results in Fig. 8,show that with progressive increase in the complexity of scene, the model struggles to keep up with the expected response.

To illustrate the strength of our experiments, we present the qualitative results of the following progressive analysis of our DM:WB model with increasingly complex scenes, as shown in Fig. 2. The model visibly learns to shade well for given lighting conditions, and standard primitives over a canonical texture. The figure is coded as follows,
POSE: *Pose variations* of a cube, with a canonical scene.
POSE+LIT: *Variations in lighting* and pose of a cube, combined with a canonical texture.
POSE+LIT+SHAP: *Variations in lighting*, pose and shape (single primitive solid in scene), with a canonical texture.
TXR: *Variations in lighting*, pose, shape (single primitive solid in scene) and texture.
ALL: *Variations in lighting*, pose, texture and complex compositions (upto six solids combined in a scene.)

Further qualitative comparison of all the five models, are presented in Fig. 7. The nuances of shading with complex solids, and the consequent mistakes, that the model incurs, may escape an untrained eye, solely because, the hatching and shading is acceptable at a coarse level. The models trained without background also perform well but qualitatively seem far from the performance of a model trained with background. Self occlusion and self shadows pose a major challenge.

5.2. Quantitative analysis

Inception score has been a widely used evaluation metric for generative models. This metric was shown to correlate well with human scoring of the realism of generated images [39]. We see in Table 2 that as expected the metric increases with increase in scene complexity, indicating progressively more diversity through the experiment.

We evaluate the models, DM, SP, SP:WS, SE, and results are summarised under Table 1. In our case we use the NPR rendered images as a reference to the generated sketch completions. The inception scores of the images generated by our models, are comfortably close to the scores for ground truth (GT). Furthermore, the metrics PSNR, SSIM and IS exhibit mild perturbations amongst the models trained with shadow, whereas a sharp change in case of the model trained without shadows. We see a steep decline in the diversity (SSIM and IS), and a sharp ascent in PSNR, both of which can be explained by the absence of shadows, that on one hand limit the scope of expressivity and *diversity*; and on the other, limit the scope of discrepancy from GT and hence *noise*, improving the count of positive signals.

5.3. Generalizability

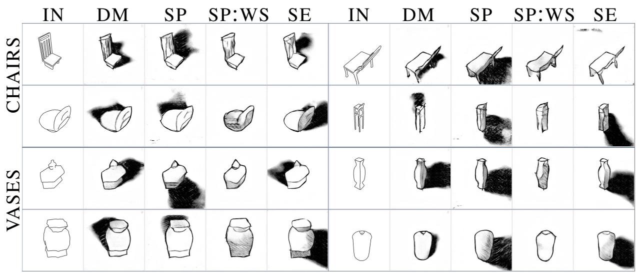

Figure 9. Generalizability results — chairs in top two rows; and vases in the next. *Set of five columns left to right.* Input contours; Evaluation with DM; With SP; With SP:WS; With SE.

We deploy two methods to verify a model's tendency to overfit, and/or its ability to generalize. One is to run on other publicly available datasets. To this end, we utilize the class of objects in chairs and vases from the ShapeNet [4]. And the other method we deploy is by user evaluation (see § 6). The representative results of qualitative evaluation can be seen in Fig. 9.

5.4. Limitations

Although the model is able to predict even for human sketches, unseen as well as totally different from the seen dataset, they are barely satisfactory. We have seen here that the model at times modifies the contours, which raises a question of fidelity in production mode. Also, to the best of our knowledge, there seems to be a lack of popularly accepted metric for measuring the levels of realism and diversity in the human sketches. Hence, we borrow the metrics of Inception score[39], PSNR, SSIM from optical domain as a proxy to examine and verify a similar trend for our dataset.

6. User Evaluation

We prepared a tool called DHI—deeply hatch it, for a user to evaluate the model in a real world scenario. The tool leverages the widely popular GIMP program over a pen-touch display to provide the *real-world* digital drawing experience. There is another TK-based applet which allows for choosing the illumination and texture. For consistency, the illumination images were computed in real-time on client-side using the blender backend. The image from the user-end *(client)* is sent to the compute server for inference triggered by a click on the result window on the TK applet, where the response is displayed.

We have classified the interactions as *a)* casual, and *b)* involved. The former was more of a rendezvous with the tool and model. In order to get familiar with the tool, the user was guided by a mentor for a couple of drawings, and later left alone to engage with the tool, hoping for her to develop familiarity and find her comfort zone .

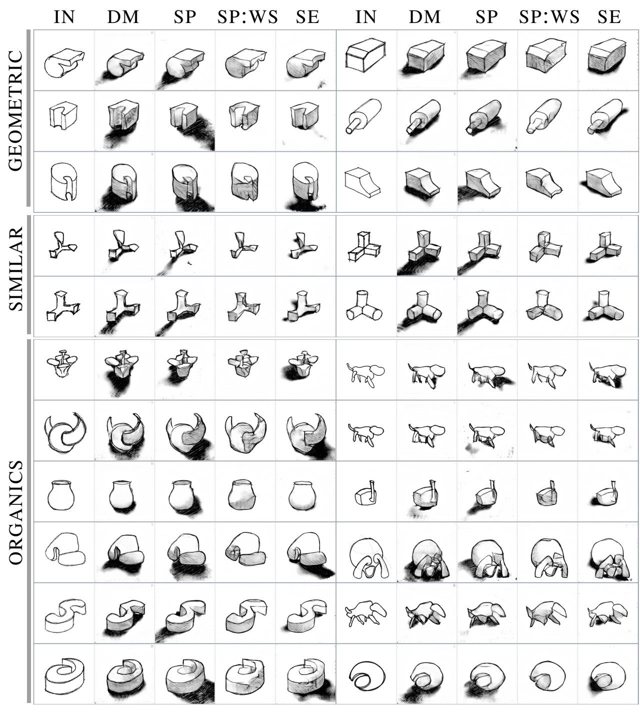

Figure 10. User evaluation results. *Blocks top to bottom.* Geometric shapes; Similar shapes with geometric and organic variations; Organic shapes. *Set of five columns left to right.* Input contours; Evaluation with DM; With SP; With SP:WS; With SE.

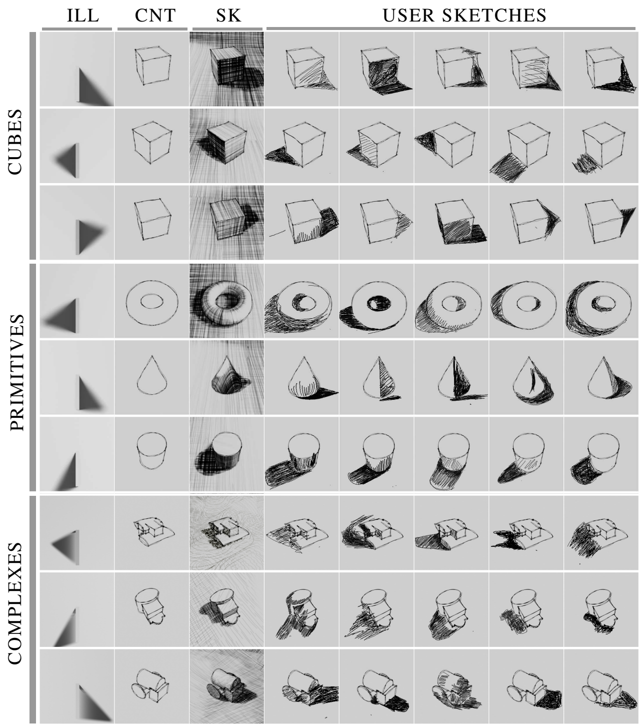

Figure 11. Selected responses from user task completion (see § 6). *Blocks top to bottom.* With only CUBES in scene; With PRIMITIVES only; With COMPLEX geometries. *Columns left to right.* ILL: Illumination hint; CNT: Contours; SK: GT sketch for reviewers reference; Next five columns show users responses.

The time of participant engagement varied between 5-43 minutes ($\mu = 21.47$, $\sigma = 5.24$). The sample size was 55 users with their age varying between 12-57 years ($\mu = 26.15$, $\sigma = 7.04$). The results of this evaluation are shown in Fig. 10. More impressive results from model trained with background DM:WB have been shown in Fig. 1.

We also conducted a more involved group based task completion exercise with 19 volunteers having art/design background, split into groups of 4. This exercise was conducted in three phases and lasted a couple of hours either side of the lunch.

The first phase was an individual level task. A user was required to complete a sketch instead of the computer in a threshold of 40 sec; the threshold had been chosen by consensus. A total of 21 samples were presented, split into three levels of difficulty: *a)* the first being the nascent level where only a cube is looked at from different poses; *b)* the second consisting a simple geometry like euclidean solids and tori; and *c)* the last were combination of multiple solids generated procedurally. The results shown in Fig. 11 indicate that the users seem to have understood the problem principally. The second phase was akin to the casual interaction as described earlier. The last phase, was a group task where a group was required to utilize the tool to create a stop motion picture, one frame at a time.

We encourage the reader to view the supplementary video[3] for the results of this exercise, evaluated against different models.

7. Conclusion

We have shown the use of CGAN in our pipeline for conditional generation of hatch patterns corresponding to line-art representing 3D shapes, by synthesising a large training set, and have portrayed its generalizability over a wide variety of real world user input.

We started with a basic model called DM and experimented with architectures and models with varying complexity, assuming that model capacity was limiting to get more satisfactory results. However, that wasn't the case as suggested by our investigations and as seen from the results; there is no clear winner, in general. From a utilitarian perspective, we should hence suggest the use of simplest and most efficient model, ie. DM.

This is nascent research area, which opens up potential investigation into higher fidelity hatching patterns. Recent advances from the deep learning community have the potential to be explored to improve the results.

[3]Supplementary video is accessible on project page, `https://bvraghav.com/shad3s/`

References

[1] Pierre Bénard, Forrester Cole, Michael Kass, Igor Mordatch, James Hegarty, Martin Sebastian Senn, Kurt Fleischer, Davide Pesare, and Katherine Breeden. Stylizing Animation by Example. *ACM Trans. Graph.*, 32(4):119:1–119:12, July 2013.

[2] Blender Online Community. *Blender - a 3D Modelling and Rendering Package*. Blender Foundation, Blender Institute, Amsterdam, 2019.

[3] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou. FaceWarehouse: A 3D Facial Expression Database for Visual Computing. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):413–425, Mar. 2014.

[4] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[5] Yang Chen, Yu-Kun Lai, and Yong-Jin Liu. CartoonGAN: Generative Adversarial Networks for Photo Cartoonization. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[6] 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*, SIGGRAPH '97, pages 421–430. ACM Press/Addison-Wesley Publishing Co., 1997.

[7] Doug DeCarlo and Anthony Santella. Stylization and Abstraction of Photographs. *ACM Trans. Graph.*, 21(3):769–776, July 2002.

[8] Johanna Delanoy, Mathieu Aubry, Phillip Isola, Alexei A. Efros, and Adrien Bousseau. 3D Sketching Using Multi-View Deep Volumetric Prediction. *Proc. ACM Comput. Graph. Interact. Tech.*, 1(1):21:1–21:22, July 2018.

[9] Haoqiang Fan, Hao Su, and Leonidas J. Guibas. A Point Set Generation Network for 3D Object Reconstruction From a Single Image. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[10] Jakub Fišer, Ondřej Jamriška, Michal Lukáč, Eli Shechtman, Paul Asente, Jingwan Lu, and Daniel Sýkora. StyLit: Illumination-guided example-based stylization of 3D renderings. *ACM Trans. Graph.*, 35(4), July 2016.

[11] Jakub Fišer, Ondřej Jamriška, David Simons, Eli Shechtman, Jingwan Lu, Paul Asente, Michal Lukáč, and Daniel Sýkora. Example-based Synthesis of Stylized Facial Animations. *ACM Trans. Graph.*, 36(4):155:1–155:11, July 2017.

[12] Chie Furusawa, Kazuyuki Hiroshiba, Keisuke Ogaki, and Yuri Odagiri. Comicolorization: Semi-Automatic Manga Colorization. *arXiv:1706.06759 [cs]*, Sept. 2017.

[13] Moritz Gerl and Tobias Isenberg. Interactive Example-based Hatching. *Computers and Graphics*, 37(1-2):65–80, 2013.

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[15] Paul Haeberli. Paint by Numbers: Abstract Image Representations. *SIGGRAPH Comput. Graph.*, 24(4):207–214, Sept. 1990.

[16] Xiaoguang Han, Chang Gao, and Yizhou Yu. DeepSketch2Face: A Deep Learning Based Sketching System for 3D Face and Caricature Modeling. *ACM Trans. Graph.*, 36(4):126:1–126:12, July 2017.

[17] Aaron Hertzmann. Painterly Rendering with Curved Brush Strokes of Multiple Sizes. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 453–460. ACM, 1998.

[18] Aaron Hertzmann, Charles E. Jacobs, Nuria Oliver, Brian Curless, and David H. Salesin. Image Analogies. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 327–340. ACM, 2001.

[19] Aaron Hertzmann and Denis Zorin. Illustrating Smooth Surfaces. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 517–526. ACM Press/Addison-Wesley Publishing Co., 2000.

[20] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-Excitation Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7132–7141, 2018.

[21] Emmanuel Iarussi, David Bommes, and Adrien Bousseau. BendFields: Regularized Curvature Fields from Rough Concept Sketches. *ACM Trans. Graph.*, 34(3):24:1–24:16, May 2015.

[22] Takeo Igarashi and John F. Hughes. Smooth Meshes for Sketch-based Freeform Modeling. In *Proceedings of the 2003 Symposium on Interactive 3D Graphics*, I3D '03, pages 139–142. ACM, 2003.

[23] Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. Teddy: A Sketching Interface for 3D Freeform Design. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pages 409–416. ACM Press/Addison-Wesley Publishing Co., 1999.

[24] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-Image Translation with Conditional Adversarial Networks. *arXiv:1611.07004 [cs]*, Nov. 2016.

[25] Ondřej Jamriška, Jakub Fišer, Paul Asente, Jingwan Lu, Eli Shechtman, and Daniel Sýkora. LazyFluids: Appearance Transfer for Fluid Animations. *ACM Trans. Graph.*, 34(4):92:1–92:10, July 2015.

[26] D. Kang, D. Kim, and K. Yoon. A study on the real-time toon rendering for 3D geometry model. In *Proceedings Fifth International Conference on Information Visualisation*, pages 391–396, July 2001.

[27] Hyunsu Kim, Ho Young Jhoo, Eunhyeok Park, and Sungjoo Yoo. Tag2Pix: Line Art Colorization Using Text Tag With SECat and Changing Loss. In *Proceedings of the IEEE/CVF*

International Conference on Computer Vision, pages 9056–9065, 2019.

[28] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, pages 1097–1105, Red Hook, NY, USA, Dec. 2012. Curran Associates Inc.

[29] Yijun Li, Chen Fang, Aaron Hertzmann, Eli Shechtman, and Ming-Hsuan Yang. Im2Pencil: Controllable Pencil Illustration From Photographs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1525–1534, 2019.

[30] H Lipson and M Shpitalni. Optimization-based reconstruction of a 3D object from a single freehand line drawing. *Computer-Aided Design*, 28(8):651–663, 1996.

[31] Peter Litwinowicz. Processing Images and Video for an Impressionist Effect. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, pages 407–414. ACM Press/Addison-Wesley Publishing Co., 1997.

[32] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully Convolutional Networks for Semantic Segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[33] Jock Mackinlay, Steven Feiner, Jim Blinn, Donald P. Greenberg, and Margaret A. Hagen. Designing Effective Pictures: Is Photographic Realism the Only Answer? In *ACM SIGGRAPH 88 Panel Proceedings*, SIGGRAPH '88, pages 12:1–12:48. ACM, 1988.

[34] J. Malik and D. Maydan. Recovering three-dimensional shape from a single image of curved objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(6):555–566, June 1989.

[35] Mehdi Mirza and Simon Osindero. Conditional Generative Adversarial Nets. *CoRR*, abs/1411.1784, 2014.

[36] Yuki Mori and Takeo Igarashi. Plushie: An Interactive Design System for Plush Toys. *ACM Trans. Graph.*, 26(3), July 2007.

[37] Emil Praun, Hugues Hoppe, Matthew Webb, and Adam Finkelstein. Real-time hatching. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '01*, page 581, Not Known, 2001. ACM Press.

[38] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. *arXiv:1505.04597 [cs]*, May 2015.

[39] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved Techniques for Training GANs. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2234–2242. Curran Associates, Inc., 2016.

[40] Michael P. Salisbury, Sean E. Anderson, Ronen Barzel, and David H. Salesin. Interactive Pen-and-ink Illustration. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '94, pages 101–108. ACM, 1994.

[41] Anthony Santella and Doug DeCarlo. Abstracted Painterly Renderings Using Eye-tracking Data. In *Proceedings of the 2nd International Symposium on Non-Photorealistic Animation and Rendering*, NPAR '02, pages 75–ff. ACM, 2002.

[42] Cloud Shao, Adrien Bousseau, Alla Sheffer, and Karan Singh. CrossShade: Shading Concept Sketches Using Cross-Section Curves. *ACM Transactions on Graphics*, 31(4), 2012.

[43] Michio Shiraishi and Yasushi Yamaguchi. An Algorithm for Automatic Painterly Rendering Based on Local Source Image Approximation. In *Proceedings of the 1st International Symposium on Non-Photorealistic Animation and Rendering*, NPAR '00, pages 53–58. ACM, 2000.

[44] Edgar Simo-Serra, Satoshi Iizuka, and Hiroshi Ishikawa. Mastering Sketching: Adversarial Augmentation for Structured Prediction. *ACM Trans. Graph.*, 37(1):11:1–11:13, Jan. 2018.

[45] Edgar Simo-Serra, Satoshi Iizuka, and Hiroshi Ishikawa. Real-time data-driven interactive rough sketch inking. *ACM Trans. Graph.*, 37(4):98:1–98:14, July 2018.

[46] Edgar Simo-Serra, Satoshi Iizuka, Kazuma Sasaki, and Hiroshi Ishikawa. Learning to simplify: Fully convolutional networks for rough sketch cleanup. *ACM Trans. Graph.*, 35(4):121:1–121:11, July 2016.

[47] The Blender Foundation. Freestyle — Blender Manual, Dec-2019.

[48] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. *arXiv:1711.11585 [cs]*, Nov. 2017.

[49] Q. Xu, Y. Gingold, and K. Singh. Inverse Toon Shading: Interactive Normal Field Modeling with Isophotes. In *Proceedings of the Workshop on Sketch-Based Interfaces and Modeling*, SBIM '15, pages 15–25. Eurographics Association, 2015.

[50] Robert C. Zeleznik, Kenneth P. Herndon, and John F. Hughes. SKETCH: An Interface for Sketching 3D Scenes. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 163–170. ACM, 1996.

[51] Kun Zeng, Mingtian Zhao, Caiming Xiong, and Song-Chun Zhu. From Image Parsing to Painterly Rendering. *ACM Trans. Graph.*, 29(1):2:1–2:11, Dec. 2009.

[52] Lvmin Zhang, Chengze Li, Tien-Tsin Wong, Yi Ji, and Chunping Liu. Two-stage sketch colorization. *ACM Trans. Graph.*, 37(6):261:1–261:14, Dec. 2018.

[53] Qingyuan Zheng, Zhuoru Li, and Adam Bargteil. Learning to Shadow Hand-drawn Sketches. In *CVPR*, page 28, 2020.

[54] Youyi Zheng, Han Liu, Julie Dorsey, and Niloy Mitra. SMART CANVAS : Context-inferred Interpretation of Sketches for Preparatory Design Studies. May 2016.