XCI-Sketch: Extraction of Color Information from Images for Generation of Colored Outlines and Sketches

V Manushree * **Sameer Saxena*** **Parna Chowdhury*** **Manisimha Varma***
Harsh Rathod* **Ankita Ghosh** **Sahil Khose**

Manipal Institute of Technology, Manipal
{ manushree.v, sameer.saxena, parna.chowdhury, manthena.varma, harsh.rathod, ankita.ghosh1, sahil.khose }@learner.manipal.edu

Abstract

Sketches are a medium to convey a visual scene from an individual's creative perspective. The addition of color substantially enhances the overall expressivity of a sketch. This paper proposes two methods to mimic human-drawn colored sketches by utilizing the Contour Drawing Dataset. Our first approach renders colored outline sketches by applying image processing techniques aided by k-means color clustering. The second method uses a generative adversarial network to develop a model that can generate colored sketches from previously unobserved images. We assess the results obtained through quantitative and qualitative evaluations.

1 Introduction

Sketches are elementary representations of visual scenes. They are easier to draw and are not limited to artists, making them an accessible and common form of art. Sketches find application in various industries pertaining to advertisement, fashion designing, architecture, and interior designing, where illustrations play a significant role. Often while using grayscale sketches, the overall expressivity falls short, and it is challenging to derive the necessary information. Color, being an integral part of art, appeals to the human senses. It elevates the sketch and efficiently carries across its meaning, background details, and connotation.

In recent years, image processing and deep learning techniques have been extensively deployed to synthesize artwork of various forms. Previously, attempts have been made to leverage images to generate sketches, but these sketches have been limited to black and white shades. Our paper focuses on different approaches to convert photographic pictures into colored sketches. We use the Contour Drawing Dataset presented in the work by Li [20] and propose two ways to extract color information from the images and amalgamate it with the corresponding sketches. Our three-fold contribution which is explained in detail in the succeeding sections of the paper is as follows: first, we formulate a process to transfer color onto the existing black and white sketches in the dataset to produce colored outlines. Second, we propose a method to produce colored sketches by performing colorspace manipulation. Finally, we go a step further to use these sketches as the training dataset for a generative adversarial network and develop a model which can produce colored sketches from unseen images.

We discuss the related literature based on which we have modeled our approaches in Section 2. The algorithm for rendering colored outlines is explained in Section 3. Section 4 describes the image

*Authors have contributed equally to this work and share first authorship. Link of the code: https://github.com/Sampai28/GeneratedSketches

NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications

processing techniques employed to produce color sketches, followed by the methodology used for generating colored sketches using a generative model. Section 5 evaluates our results by comparing the produced sketches with existing sketch datasets and conducting a perceptual user study. We conclude by suggesting various possible applications of our work in Section 6 and the scope of improvement in Section 7.

2 Related Works

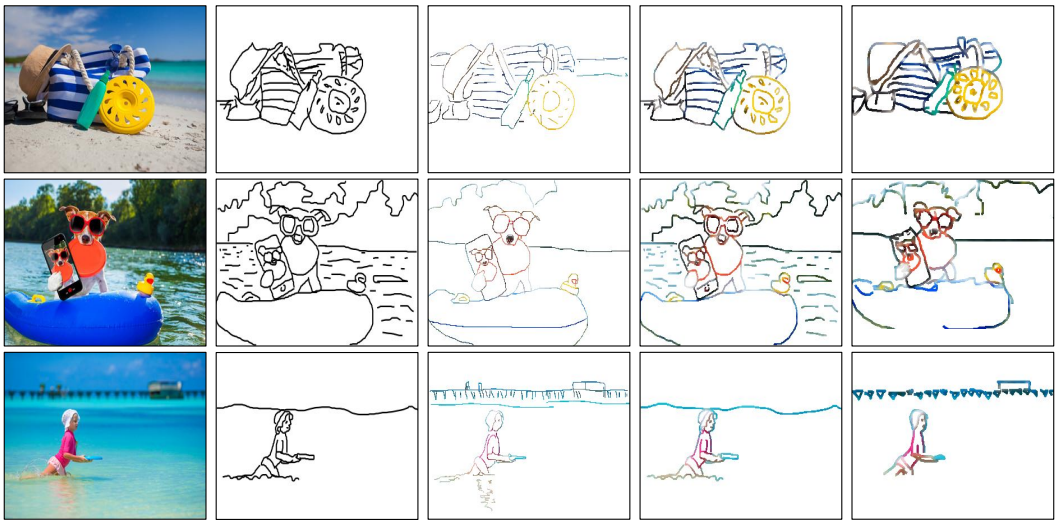
Approaches for rendering sketches: A wide range of research work has been done by the computer vision and computer graphics community in the domain of sketch synthesis. Prior techniques include rendering sketches by applying edge detection [3, 19], contour detection [20] and image segmentation [1, 24]. Many sizeable sketch datasets have been collected through crowdsourcing by designing interactive user interfaces [2, 15, 16]. The onset of deep learning substantially enhanced the results produced in this field. SketchRNN [15], a RNN-based deep variational autoencoder, was among the earlier works in deep learning that generated diverse sketches. Style transfer using neural network [10, 22] was a revolutionary step in the generation of art by producing images that preserve content and style components of the input images. Neural style transfer has since been adopted extensively to produce sketches [5, 44]. Other notable works include the use of reinforcement learning [40] to generate ink paintings from pictures and photographs and implementation of hidden Markov model [32] to transform a coarse sketch into a refined drawing. Generative adversarial networks (GANs) have found immense application in developing sketch generation models.

GANs for generating sketches: Generative adversarial network based models have seen major advancements in recent times, and this has motivated great research in its use as an art and image creation tool. Pencil-shading sketches have been produced by architectures like ArtPDGAN [21] which combines an image-to-image network to produce a key map that aids the generator. BP-GAN [35], and Composition Aided GANs [41] perform face sketch synthesis using methods like back projection and compositional reconstruction loss, respectively. GANs have been used for colorizing anime sketches by applying conditional generation [39], and style transfer [43]. DoodlerGAN [11] generates doodle-like sketches in a sequential drawing manner. RoboCoDraw [36] is a real-time robot-based drawing system that stylizes human face sketches interactively and generates cartoon avatar face sketch from an individual's image using AvatarGAN. CartoonGAN [7] proposes an architecture to convert real-world scenes into cartoon-style images by introducing semantic content loss and edge-promoting adversarial loss. Several GAN architectures have been used to retrieve images from sketches [6, 8, 14, 23, 31, 38], and even generate three-dimensional models from sketches [2, 27]. The ability of GANs to generate volumes of synthetic data makes it an ideal tool for expanding on existing datasets and creating new ones.

Sketch Datasets: Art generation is a broadly investigated domain in computer vision, which in turn gives rise to various applications for the dataset. Existing datasets like TU-Berlin [9] consists of 20000 unique human-drawn sketches distributed over 250 distinct object classes, which have been used to train classification models. The Sketchy Database [30] comprising of 12500 photographs and approximately 75000 human drawn sketches inspired by the same, has been used to generate more photo-sketch pairs. SketchyScene [45] contains more than 29000 scene-level sketches, and all objects in the sketches have ground-truth semantic and instance masks. This enables several applications like sketch colorization, editing, and captioning. QuickDraw [15], Sketch Me That Shoe [42] and CHUK Face Sketch Database [37] are examples of datasets that have been used for sketch generation, sketch-based image retrieval, and sketch synthesis and recognition, respectively.
The Contour Drawing [20] Dataset consists of 1000 outdoor images, which are paired with five different versions of sketches, thus making a total of 5000 human-drawn black and white sketches. These five sketches are presented in three variations of stroke width 1, 3, and 5 units. Majority of the present sketch datasets contain grayscale sketches. We extend the Contour Drawing Dataset by rendering colored outlines and color-filled versions of the available sketches, which can be utilized for various tasks.

3 Rendering Colored Outlines in Sketches

In this section, we discuss how we transfer color to the black and white sketches of the Contour Drawing Dataset. Our methodology incorporates several techniques, including k-means color clustering and bitwise operations. We also describe the method that we deploy to find the optimal number of clusters per image for the k-means algorithm.



a) Original Image | b) Original Sketch (w = 3, v = 3) | c) Colored Outline (Ours) (w = 1, v = 1) | d) Colored Outline (Ours) (w = 3, v = 3) | e) Colored Outline (Ours) (w = 5, v = 5)

Figure 1: Qualitative results of the rendered colored outlines. (b) shows the sketch presented in [20] dataset. (c), (d) and (e) display the results obtained by us after color extraction from image (a). w is the stroke width and v is the version of sketch.

3.1 Methodology

A digital RGB image can contain $256 \times 256 \times 256$ different colors, but generally, only a few colors are used when an individual draws a sketch. To mimic this while generating colored outlines, we perform quantization [4] of the number of colors in the images. First, we perform three iterations of Gaussian blur [12] with a kernel size of 5×5 on the images. This is done to reduce the sharp color transitions inside the images as sketches are boundary-like drawings that capture the outline of the visual scene. To achieve quantization, we perform k-means clustering [33] on the image where the pixels are categorized into k clusters according to their intensity value. We formulate an algorithm to determine this value k, which is described in Section 3.2.

Since the dimensions of the image and its corresponding sketches do not align in some cases, we slice the extraneous rows or columns to attain uniform dimensions throughout the dataset. We apply binary thresholding on the sketch and extract the black outlines to form a mask. We split the channels of the post-processed image and perform bitwise operations between the mask and each channel to obtain the color information. The resultant channels are merged to render the final colored outline sketch. We repeat this process for all 15 versions of sketches obtained from an image. Figure 1 shows our results on three variations of the black and white sketches. w denotes the width of the sketch outline, and v denotes the version of the sketch.

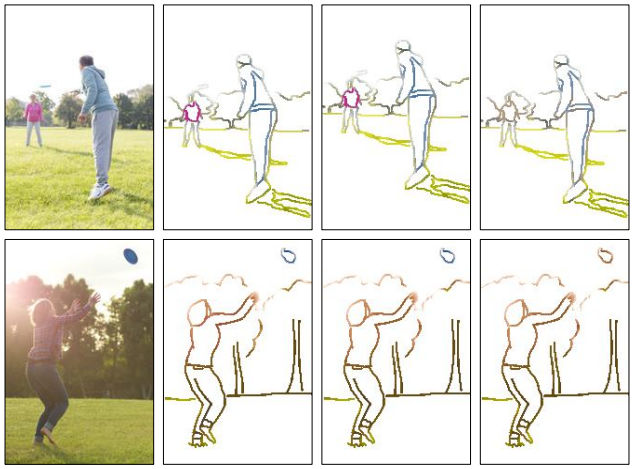
3.2 Calculating Optimal Number of Color Clusters

The appropriate number of clusters into which the data should be grouped is crucial in an unsupervised technique. In k-means clustering, the elbow method and the silhouette method [18] are among the broadly used algorithms to determine the optimal value of k. These methods prove ineffective for our objective as iterating through all k values to reach the optimal value is computationally expensive, and we desire to capture only the most prominent colors in the image. Hence we devised an alternative method. As the k-means algorithm clusters data by minimizing the sum of squared distance within the cluster, we appoint this criterion, called inertia, as a threshold value. Inertia can be calculated as:

$$\sum_{i=0}^{n} \|c_i - x_i\|^2 \tag{1}$$

where x denotes pixels, x_i denotes value of ith pixel and c_i denotes value of cluster centroid closest to x_i.

A threshold value (τ) is fixed at the beginning. We then iterate over values of k for an image at strides of 5 and calculate the inertia at each step. The value of inertia starts at infinity and reduces as



a) Image b) $\tau = 50$ c) $\tau = 70$ d) $\tau = 100$

Figure 2: Colored outlines based on different threshold values. We can observe that $\tau = 70$ is the optimal value. At higher threshold values ($\tau = 100$), we can observe the loss of color information. At lower threshold values ($\tau = 50$), the result is similar to that of the optimal value ($\tau = 70$) but is computationally expensive. τ denotes the threshold value.

the number of clusters increase. We obtain the optimal k value for an image at the point where the inertia becomes lesser than the assigned threshold value. For locating the ideal threshold value, we test various values empirically and conclude that when the threshold is 70, it gives visually pleasing results. From our experimentation, we deduce that determining an optimal value for k is essential; otherwise, it compromises the quality of the sketch or produces redundant results. This is depicted in Figure 2, which compares sketches produced when the threshold values were set to 50, 70, and 100 respectively.

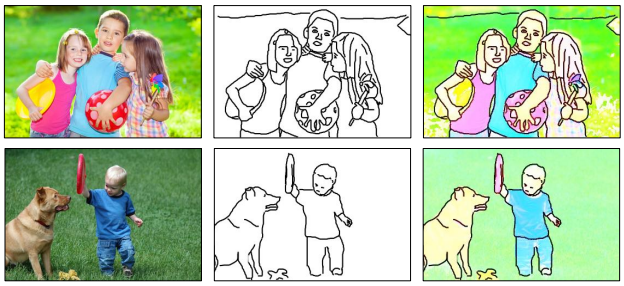
4 Generating Colored Sketches

In this section, we introduce a generative adversarial network framework that converts input images to colored sketches. We explain the colorspace manipulation administered on the dataset, illustrate the network structure, and discuss the objective functions and optimization applied on our architecture.

4.1 Data Preprocessing

The dataset consists of colored images and their corresponding black and white human-drawn sketches. To add color to the sketches, we propose a technique similar to the Gouache Color Transform stated in [44].

We convert the RGB image and its sketch to the $L^*a^*b^*$ colorspace, where L^* stands for perceptual lightness, a^* and b^* for the four colors red, blue, green, yellow. The a^* and b^* channels of the sketch



a) Original Image b) Black And White Sketch c) Post-processed Color Sketch

Figure 3: Qualitative results obtained after data preprocessing

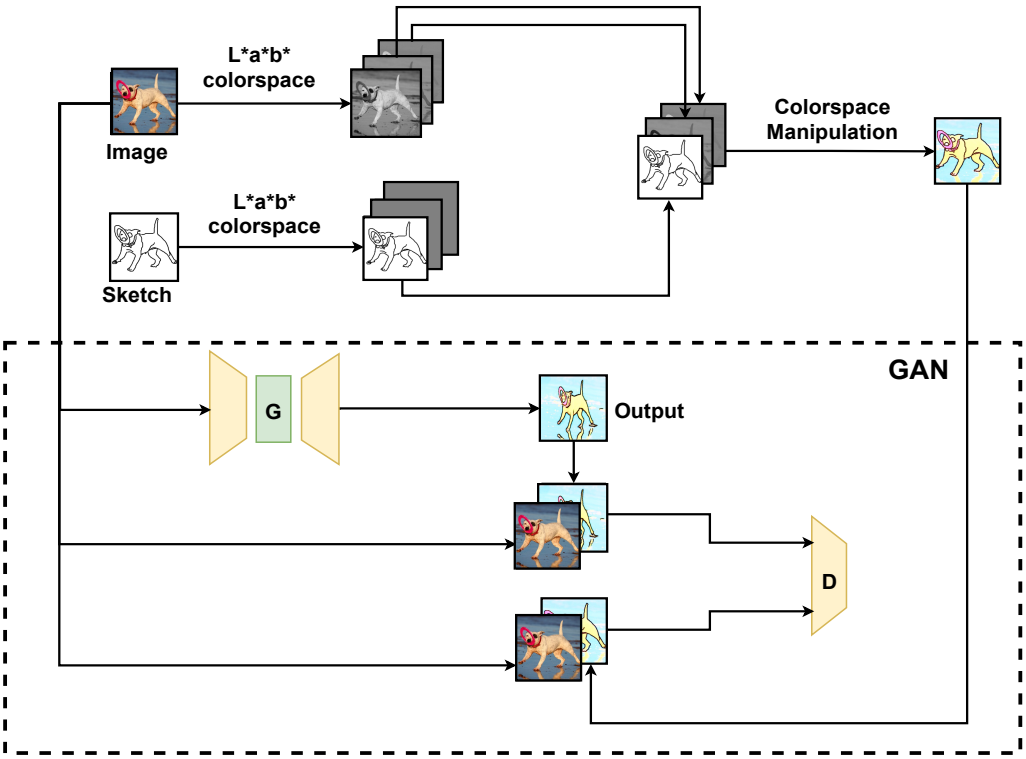
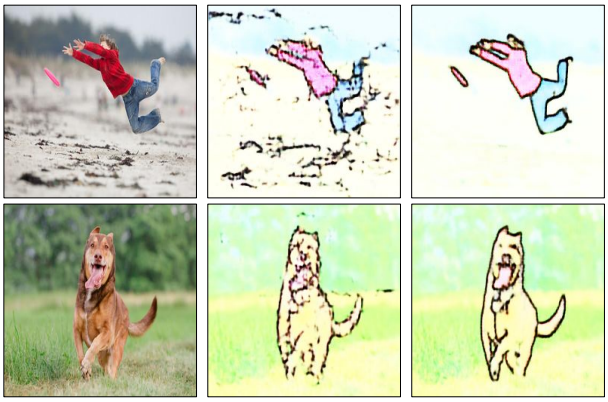


Figure 4: Pipeline of our proposed generation model

are discarded and replaced with the a^* and b^* channels of the corresponding image to transfer color without changing the content. The resultant image is converted back to RGB.
To improve the quality of the transferred colors, we increase the saturation by a factor of 1.8 in HSV colorspace. We choose this factor by empirically trying different values. A value lower than this results in poor contrast, and a value higher than this makes the brighter tones appear white. The results of the preprocessing method are depicted in Figure 3. These newly obtained colored sketches are included in the training data for our GAN model.

4.2 Methodology

GANs consist of two parts, a generator G and a discriminator D. Our GAN learns a mapping from an input image x to an output colored sketch y, so that $G : x \rightarrow y$. We adapt our generator and discriminator from the pix2pix work of Isola [17], where the generator resembles the architecture of U-Net [29], which consists of an encoder and decoder network with skip connections. The discriminator is a PatchGAN which penalizes the structure at the scale of local image patches. The entire pipeline, including the data preprocessing, is depicted in Figure 4.
We optimize the GAN with additional convolutional layers where we use a kernel size of 3×3, apply same padding, and set the stride to 1. This is followed by batch normalization and a leaky ReLU activation function at every downsampling layer of the encoder. The feature space is also increased so that more attributes can be extracted. We apply similar optimizations in the discriminator and introduce max-pooling layers alongside to extract necessary features which can help the discriminator differentiate between real and fake inputs. While passing input data to the discriminator, the post-processing colored sketches are considered real, while the output from the generator is regarded as fake. The original image is stacked upon the sketches before passing them to the discriminator. We train our GAN with the data resized to 256×256 dimensions followed by normalization with mean and standard deviation set to 0.5.
We compare the results obtained from pix2pix and our GAN in Figure 5. The sketches generated from our model have visibly lesser noise and well-defined outlines. Our model performs better at retaining the content of the original image. Figure 6 shows the colored sketches produced by our model from unobserved image inputs.



a) Original Image b) pix2pix GAN c) Ours

Figure 5: A visual comparision of colored sketches generated using pix2pix GAN and our GAN. Our GAN produces well-defined outlines with negligible noise.

4.3 Objective function and Optimization

In a generative adversarial network [13], a random noise vector z is given as an input to the generator, which generates an output $\hat{y} = G(z)$. The discriminator's work is to predict y as real and $\hat{y}$ as fake. The loss function is defined as

$$\begin{aligned} L_{GAN}(G, D) = \mathbb{E}_y [\log D(y)] + \\ \mathbb{E}_z [\log (1 - D(G(z)))] \end{aligned} \tag{2}$$

In a conditional GAN (cGAN), both the image x and random noise vector z are fed into the generator, which maps it to the output $\hat{y} = G(x, z)$. The generator G aims to output a sketch which resembles y, conditioned on x, while discriminator D is adversarially trained to distinguish between the label y and generated output $\hat{y}$ given x, and gives feedback to the generator on whether it is real or fake. The loss for this objective can be written as

$$\begin{aligned} L_{cGAN}(G, D) = \mathbb{E}_{x,y} [\log D(x, y)] + \\ \mathbb{E}_{x,z} [\log (1 - D(x, G(x, z)))] \end{aligned} \tag{3}$$

Usually noise vector z is ignored in the optimization as found in the previous work [17, 20, 25]. We do not include z in our loss function either.

We include the L1 loss, which suppresses the irrelevant details while generating the outlines for colored sketches and reduces the blurring of the output. The L1 loss is as follows

$$L_1(G) = \mathbb{E}_{x,y} [\|y - G(x)\|_1] \tag{4}$$

The combined loss function now becomes

$$L_c = \arg \min_G \max_D L_{cGAN}(G, D) + \lambda L_1(G) \tag{5}$$

Figure 6: Sketches generated by our GAN (bottom row) from unrevealed images (top row) spanning over a variety of data distribution.

where λ is a constant non-negative real number which controls the strength of L1 loss.
To optimize our objective function, we follow the standard approach as mentioned in [17], where we oscillate between one step of gradient descent on D and one step of gradient descent on G. The output of the discriminator is a patch of dimensions $26 \times 26 \times 1$, which is passed to the objective function. The objective function is then divided by the batch size, which is 32 in our case. We used Adam optimizer with a learning rate of 0.0005. The momentum parameters are defined as $\beta_1 = 0.5$, $\beta_2 = 0.999$, and λ is set to 1000.

5 Results and Evaluation

Quantitative Evaluation: Evaluating artworks is highly subjective, thus making quantitative image assessment a challenging task. In this paper, we have used Neural Image Assessment (NIMA) [34] to assess our results. NIMA consists of two convolutional network models which use pre-trained weights of ImageNet classification task and are fine-tuned on Aesthetic Visual Analysis (AVA) [26] and Tampere Image Database (TID2013) [28] datasets to predict the aesthetic and technical score, respectively. The technical score is a measure of low-level characteristics such as noise, blur, and saturation. The aesthetic score is used to assess semantic level characteristics such as style. The models provide the score on a scale of 1 to 10 where 1 is the lowest, and 10 is the perfect score.
We use the two NIMA models to predict the technical and aesthetic scores of our results: colored outlined sketches and colored sketches and compare them with the Contour Drawing Dataset. Additionally, we also calculate the scores on 1000 randomly sampled sketches from three widely used sketch datasets: TU-Berlin [9], The Sketchy Database [30] and SketchyScene [45]. The average NIMA scores of all the sampled sketches in each dataset are listed in Table 1.

Sketch Dataset	Technical Score	Aesthetic Score
TU-Berlin[9]	**5.24**	4.32
The Sketchy Database[30]	4.83	**5.15**
SketchyScene[45]	4.36	**5.15**
Contour Drawing Dataset[20]	4.26	5.12
Colored Outline (Ours)	4.34	5.02
Colored Sketch (Ours)	4.24	4.92

Table 1: Comparison of NIMA technical and aesthetic scores of our results with other sketch datasets.

The colored outlines have a marginally higher average technical score than the original black and white sketches. The average technical and aesthetic scores of the output images from both methods are very close to the scores of the original sketches and the three commonly used sketch datasets.

Perceptual Study: Besides evaluating the results based on NIMA scores, we also conduct a perceptual study. A user interface has been created which randomly generates a set of 10 photographs, their corresponding colored outline sketches, and color-filled sketches at a time. We divide the users into two categories. For our purpose, we consider painters, pencil and pen artists, digital artists, illustrators, animators, and professionals from related backgrounds to be more critical of our results and identify them as artists. Others are classified as laypeople. The users are expected to view a minimum of 50 results, and there is no maximum limit. They are then asked to rate our results on a scale of 1 to 5 based on three criteria: resemblance to real or human-drawn sketches, content retention in the sketches compared to the original image, and color retention in the sketches compared to the original image. We define a legend for the scores to remove ambiguity for the users as stated in Figure 7. A total of 100 verified users—30 artists and 70 laypeople—voluntarily participated in our user-based evaluation process. We take the weighted average of the ratings based on the number of results viewed by each user. The results have been presented in Figure 7 which confirms that both groups of users find our results to be above average rating.

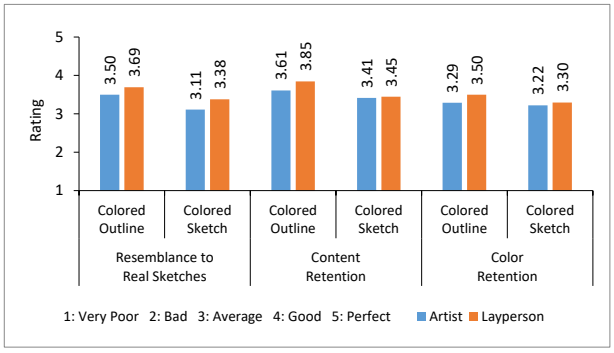


Figure 7: Results of our perceptual user study for each category rated on a scale of 1 to 5, collected from artists and laypeople.

6 Applications

There is a wide range of applications for our methods, as color plays a major role in art. Colored sketches are able to convey more information and details, while maintaining their sparsity. For example, a black and white sketch of a carrot has a higher chance of it being confused with a radish or a similar object. But the colored version will be recognized with more ease and less confusion.
Pre-primary schools could be one use for generating colored outlines and sketches. During this stage, students begin to learn how to sketch. The following is a description of the learning process. They begin by drawing black and white outlines and then progress to colored outlines and later to color-filled sketches. They may acquire the approach mentioned above from their tutor or duplicate it from photos they see. The teacher can choose colored images and draw the black and white outlines using the sketch game mentioned in the paper [20]. The teacher can feed the colored outline and black white outlines to our model to produce the colored outlines, which can assist the students by providing a reference.

7 Conclusion and Future Work

In this work, we examine the problem of producing colored sketches from images. We extend the Contour Drawing Dataset with two kinds of colored versions of the sketches, introduce a generation model, and compare our work with the existing literature. Our results hold a plethora of applications as they can be further annotated and utilized on tasks concerning sketch recognition, segmentation, text-based sketch generation, and sketch-based modeling. The generation model automates the process of drawing a sketch inspired by a visual scene or photograph, thus saving time and labor.
We make use of unsupervised learning and image processing techniques to render colored outlines. This can also be attempted with deep learning approaches like neural style transfer. The generation model can be improved by training on larger datasets that include a variety of objects. Our work can be extended to encompass these potential enhancements.

References

[1] Pablo Arbeláez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, 2011.

[2] Rahul Arora, Ishan Darolia, Vinay Namboodiri, Karan Singh, and Adrien Bousseau. Sketchsoup: Exploratory ideation using design sketches. *Computer Graphics Forum*, 36, 02 2017.

[3] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986.

[4] M. Emre Celebi. Improving the performance of k-means for color quantization. *Image and Vision Computing*, 29(4):260–271, 2011.

[5] Chaofeng Chen, Xiao Tan, and Kwan-Yee K. Wong. Face sketch synthesis with style transfer using pyramid column feature. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 485–493, 2018.

[6] W. Chen and J. Hays. Sketchygan: Towards diverse and realistic sketch to image synthesis. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9416–9425, Los Alamitos, CA, USA, jun 2018. IEEE Computer Society.

[7] 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*, pages 9465–9474, 2018.

[8] Yuanzheng Ci, Xinzhu Ma, Zhihui Wang, Haojie Li, and Zhongxuan Luo. User-guided deep anime line art colorization with conditional adversarial networks. In *Proceedings of the 26th ACM International Conference on Multimedia*, MM '18, page 1536–1544, New York, NY, USA, 2018. Association for Computing Machinery.

[9] Mathias Eitz, James Hays, and Marc Alexa. How do humans sketch objects? *ACM Trans. Graph. (Proc. SIGGRAPH)*, 31(4):44:1–44:10, 2012.

[10] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2414–2423, 2016.

[11] Songwei Ge, Vedanuj Goswami, Larry Zitnick, and Devi Parikh. Creative sketch generation. In *International Conference on Learning Representations*, 2021.

[12] Estevão S. Gedraite and Murielle Hadad. Investigation on the effect of a gaussian blur in image filtering and segmentation. In *Proceedings ELMAR-2011*, pages 393–396, 2011.

[13] 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. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[14] Longteng Guo, Jing Liu, Yuhang Wang, Zhonghua Luo, Wei Wen, and Hanqing Lu. Sketch-based image retrieval using generative adversarial networks. In *Proceedings of the 25th ACM International Conference on Multimedia*, MM '17, page 1267–1268, New York, NY, USA, 2017. Association for Computing Machinery.

[15] David Ha and Douglas Eck. A neural representation of sketch drawings. In *International Conference on Learning Representations*, 2018.

[16] Emmanuel Iarussi, A. Bousseau, and Theophanis Tsandilas. The drawing assistant: automated drawing guidance and feedback from photographs. *Proceedings of the 26th annual ACM symposium on User interface software and technology*, 2013.

[17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5967–5976, 2017.

[18] Trupti M Kodinariya and Prashant R Makwana. Review on determining number of cluster in k-means clustering. *International Journal*, 1(6):90–95, 2013.

[19] Iasonas Kokkinos. Boundary detection using f-measure-, filter- and feature- (f3) boost. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, pages 650–663, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[20] Mengtian Li, Zhe Lin, Radomír Mˇ ech, Ersin Yumer, and Deva Ramanan. Photo-sketching: Inferring contour drawings from images. In *WACV*, 2019.

[21] SuChang Li, Kan Li, Ilyes Kacher, Yuichiro Taira, Bungo Yanatori, and Imari Sato. Artpdgan: Creating artistic pencil drawing with key map using generative adversarial networks. In Valeria V. Krzhizhanovskaya, Gábor Závodszky, Michael H. Lees, Jack J. Dongarra, Peter M. A. Sloot, Sérgio Brissos, and João Teixeira, editors, *Computational Science – ICCS 2020*, pages 285–298, Cham, 2020. Springer International Publishing.

[22] Yanghao Li, Naiyan Wang, Jiaying Liu, and Xiaodi Hou. Demystifying neural style transfer. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 2230–2236. AAAI Press, 2017.

[23] Yongyi Lu, Shangzhe Wu, Yu-Wing Tai, and Chi-Keung Tang. Image generation from sketch constraint using contextual gan. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 213–228, Cham, 2018. Springer International Publishing.

[24] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 416–423 vol.2, 2001.

[25] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. January 2016. 4th International Conference on Learning Representations, ICLR 2016 ; Conference date: 02-05-2016 Through 04-05-2016.

[26] Naila Murray, Luca Marchesotti, and Florent Perronnin. Ava: A large-scale database for aesthetic visual analysis. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2408–2415, 2012.

[27] Naoki Nozawa, Hubert Shum, Qi Feng, Edmond Ho, and Shigeo Morishima. 3d car shape reconstruction from a contour sketch using gan and lazy learning. *The Visual Computer*, 04 2021.

[28] Nikolay Ponomarenko, Oleg Ieremeiev, Vladimir Lukin, Karen Egiazarian, Lina Jin, Jaakko Astola, Benoit Vozel, Kacem Chehdi, Marco Carli, Federica Battisti, and C.-C. Jay Kuo. Color image database tid2013: Peculiarities and preliminary results. In *European Workshop on Visual Information Processing (EUVIP)*, pages 106–111, 2013.

[29] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing.

[30] Patsorn Sangkloy, Nathan Burnell, Cusuh Ham, and James Hays. The sketchy database: Learning to retrieve badly drawn bunnies. *ACM Transactions on Graphics (proceedings of SIGGRAPH)*, 2016.

[31] Patsorn Sangkloy, Jingwan Lu, Chen Fang, Fisher Yu, and James Hays. Scribbler: Controlling deep image synthesis with sketch and color. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6836–6845, 2017.

[32] Saul Simhon and Gregory Dudek. Sketch interpretation and refinement using statistical models. In *Proceedings of the Fifteenth Eurographics Conference on Rendering Techniques*, EGSR'04, page 23–32, Goslar, DEU, 2004. Eurographics Association.

[33] Kristina P. Sinaga and Miin-Shen Yang. Unsupervised k-means clustering algorithm. *IEEE Access*, 8:80716–80727, 2020.

[34] Hossein Talebi and Peyman Milanfar. Nima: Neural image assessment. *IEEE Transactions on Image Processing*, 27(8):3998–4011, 2018.

[35] Nannan Wang, Wenjin Zha, Jie Li, and Xinbo Gao. Back projection: An effective postprocessing method for gan-based face sketch synthesis. *Pattern Recognition Letters*, 107:59–65, 2018. Video Surveillance-oriented Biometrics.

[36] Tianying Wang, Wei Qi Toh, Hao Zhang, Xiuchao Sui, Shaohua Li, Yong Liu, and Wei Jing. Robocodraw: Robotic avatar drawing with gan-based style transfer and time-efficient path optimization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(06):10402–10409, Apr. 2020.

[37] Xiaogang Wang and Xiaoou Tang. Face photo-sketch synthesis and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(11):1955–1967, 2009.

[38] Zhi-Hui Wang, Ning Wang, Jian Shi, Jian-Jun Li, and Hairui Yang. Multi-instance sketch to image synthesis with progressive generative adversarial networks. *IEEE Access*, 7:56683–56693, 2019.

[39] Ziyuan Wang. Generating anime sketches with c-GAN. *Journal of Physics: Conference Series*, 1827(1):012157, mar 2021.

[40] Ning Xie, Hirotaka Hachiya, and Masashi Sugiyama. Artist agent: A reinforcement learning approach to automatic stroke generation in oriental ink painting. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, page 1059–1066, Madison, WI, USA, 2012. Omnipress.

[41] Jun Yu, Xingxin Xu, Fei Gao, Shengjie Shi, Meng Wang, Dacheng Tao, and Qingming Huang. Toward realistic face photo-sketch synthesis via composition-aided gans. *IEEE Transactions on Cybernetics*, pages 1–13, 2020.

[42] Qian Yu, Feng Liu, Yi-Zhe Song, Tao Xiang, Timothy M. Hospedales, and Chen Change Loy. Sketch me that shoe. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 799–807, 2016.

[43] Lvmin Zhang, Yi Ji, Xin Lin, and Chunping Liu. Style transfer for anime sketches with enhanced residual u-net and auxiliary classifier gan. In *2017 4th IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 506–511, 2017.

[44] Wei Zhang, Guanbin Li, Haoyu Ma, and Yizhou Yu. Automatic color sketch generation using deep style transfer. *IEEE Computer Graphics and Applications*, 39(2):26–37, 2019.

[45] Changqing Zou, Qian Yu, Ruofei Du, Haoran Mo, Yi-Zhe Song, Tao Xiang, Chengying Gao, Baoquan Chen, and Hao Zhang. Sketchyscene: Richly-annotated scene sketches. In *ECCV*, pages 438–454. Springer International Publishing, 2018.