CLIPDraw: Exploring Text-to-Drawing Synthesis through Language-Image Encoders

Anonymous Author(s)
Affiliation
Address
email

Abstract

CLIPDraw is an algorithm that synthesizes novel drawings from natural language input. It does not require any additional training; rather, a pre-trained CLIP language-image encoder is used as a metric for maximizing similarity between the given description and a generated drawing. Crucially, CLIPDraw operates over vector strokes rather than pixel images, which biases drawings towards simpler human-recognizable shapes. Results compare CLIPDraw with other synthesis-through-optimization methods, as well as highlight various interesting behaviors of CLIPDraw, such as satisfying ambiguous text in multiple ways, reliably producing drawings in diverse styles, and scaling from simple to complex visual representations as stroke count increases.

1 Introduction

When humans hear a description of a scene, it's easy to imagine what it may look like. Conversely, when we construct a mental image, it's easy to then describe that scene. At some level, humans have a deeply coupled representation for textual and visual structures key to understanding our world.

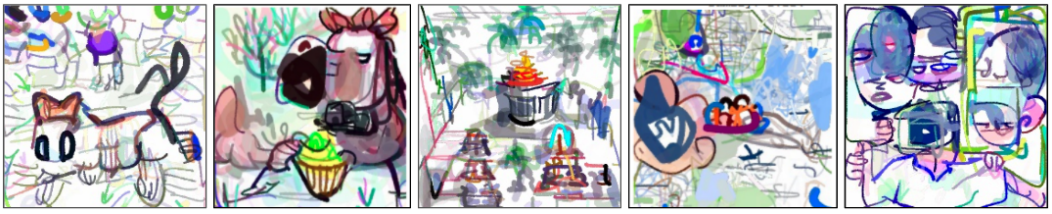
The recent introduction of CLIP (Radford et al., 2021), a dual language-image encoder, is a large step towards unifying textual and visual information. In a CLIP model, both text and images are mapped onto the same representational space, thus enabling the similarity between images and textual descriptions to be measured. When trained on large amounts of data, CLIP representations have been shown to solve a robust range of image-based recognition tasks.

This work presents *CLIPDraw*, an algorithm that synthesizes novel drawings based on natural language input. CLIPDraw does not require any training; rather a pre-trained CLIP model is used as a metric for maximizing similarity between the given description and a generated drawing. Rather than photorealistic images, CLIPDraw aims to synthesize simple drawings that nevertheless match the prompt. Thus, CLIPDraw optimizes a set of vector strokes rather than pixel images, a constraint that biases drawings towards simple human-recognizable shapes.

The aim of this work is to present CLIPDraw a testbed for exploring language-image relationships and synthesizing AI-assisted artwork, as well as to showcase various nuances of the method. Results compare between CLIPDraw and other optimization-based text-to-image methods, along with highlighting several interesting behaviors:

- By adjusting descriptive adjectives, such as "watercolor" or "3D rendering", CLIPDraw produces drawings of vastly different styles.
- CLIPDraw often matches the description prompt in creative ways, such as writing words from the prompt inside the image itself, or interpreting ambiguous nouns in multiple ways.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.



"A drawing of a cat". "Horse eating a cupcake". "A 3D rendering of a temple". "Family vacation to Walt Disney World". "Self".

Figure 1: **Various drawings synthesized by CLIPDraw**, along with the corresponding description prompts used. CLIPDraw synthesizes images from text by performing gradient descent over a set of RGBA Bézier curves, with the goal of minimizing cosine distance between the CLIP encodings of generated images and description prompts. CLIPDraw does not require learning a new model, and can generally synthesize images within a minute on a typical GPU.

- Running CLIPDraw with a low stroke count results in cartoonish drawings, while high stroke counts tend to result in realistic renderings.
- By giving CLIPDraw abstract prompts, such as "happiness" or "self", we can examine what visual concepts the CLIP model associates with them.
- CLIPDraw behavior can be further controlled through the use of negative prompts, such as "a messy drawing", to encourage the opposite behavior.

2 Related Work

Text-to-Image Synthesis. This work greatly draws from the field of text-to-image synthesis, whose primary aim is to generate images that correctly match a given textual description. In recent years, focus has been on methods that aim to learn a direct text-to-image mapping function, often in the form of a conditional GAN (Goodfellow et al., 2014a; Mirza and Osindero, 2014; Reed et al., 2016; Frolov et al., 2021). Commonly used datasets include Oxford-120 Flowers (Nilsback and Zisserman, 2008), CUB-200 Birds (Wah et al., 2011), and COCO (Lin et al., 2014), all of which contain natural images and captions describing them. While GAN-based methods have enabled considerable progress towards photorealistic image synthesis, strong autoregressive models have achieved similar quality results (Oord et al., 2017; Chen et al., 2020), with the recent DALL-E model (Ramesh et al., 2021) showcasing the benefit of scaling text-to-image synthesis networks to a large capacity. In comparison to text-to-image generative models, which require large amounts of training, this work follows the framework of *synthesis through optimization*, in which images are generated through evaluation-time optimization against a given metric.

Synthesis Through Optimization. Instead of directly learning an image generation network, an alternative method of image synthesis is to optimize towards a matching image during evaluation time. This framework is often referred to as *activation maximization* (Erhan et al., 2009; Nguyen et al., 2016; Mordvintsev et al., 2015), where a random image is optimized through backpropogation to increase certain neuron activations of a pretrained network. Activation-maximization methods have produced highly realistic images, however it is a challenge to understand the meaning of a neuron activation. CLIPDraw builds off a set of methods where rather than maximizing an activation, the objective is to minimize the distance between the produced image and a given description phrase, as defined by a powerful CLIP language-image encoder (Fernando et al., 2021; Murdock, 2021; Crowson et al., 2022; Galatolo et al., 2021). A key issue in synthesis through optimization is that the produced images often leave the space of natural images (Nguyen et al., 2015), or fool the system through adversarial means (Goodfellow et al., 2014b), thus a body of work aims to discover 'natural image priors' to constrain produced images (Nguyen et al., 2016, 2017). While a typical solution is to constrain optimization to the generative space of a GAN, this setup can be expensive to evaluate, and only allows synthesis of images producible by the GAN generator. Because CLIPDraw focuses on synthesizing drawings rather than realistic pictures, CLIPDraw instead limits optimization to a set of vector curves. This constraint results in stroke-based images, which must capture larger features such as shapes and outlines, rather than fine-grained textures.

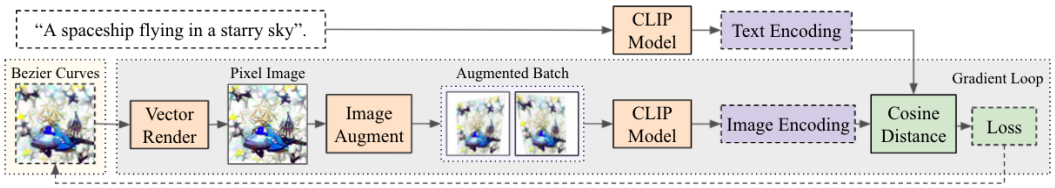


Figure 2: **CLIPDraw iteratively synthesizes images through evaluation-time gradient descent.** Starting from a random set of Bézier curves, the position and colors of the curves are optimized so that the generated drawings best match the given description prompt. Before being passed into the CLIP encoder, drawings are augmented into multiple perspective-shifted copies.

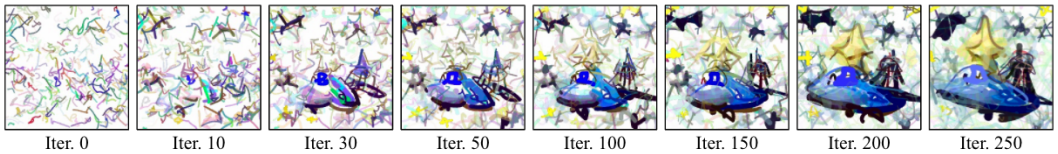


Figure 3: **A typical CLIPDraw run gradually forms curves into concrete shapes.** Here, a background of star-shaped structures develops into a large spaceship. More pronounced stars eventually appear, in addition to a Darth-Vader-like figure on the spaceship.

Vector Graphics. This work builds largely from work by Li et al. (2020), which introduces a differentiable renderer for vector graphics. Image generation methods that operate over vector images have traditionally required a vector-based dataset, however recent work has shown how differentiable renderers can be used to bypass this limitation (Reddy et al., 2021; Shen and Chen, 2021). CLIPDraw uses a differentiable renderer as a representation for generating drawings; namely a set of RGBA Bézier curves are optimized rather than pixels.

Follow Up Work to CLIPDraw. A draft of this work was previously released to the public research community, and a number of follow-up papers have since been published. StyleCLIPDraw (Schaldenbrand et al., 2021) introduces an explicit image-based style loss to the CLIPDraw framework. CLIP-CLOP (Mirowski et al., 2022) extends CLIPDraw to use image patches rather than strokes. Tian and Ha (2022) pursure optimization via evolutionary strategies rather than gradient descent. CLIPDraw has already influenced the image synthesis community in its pre-release form, and we hope a formal publication will continue to encourage collaboration.

3 Method

The objective of CLIPDraw is to synthesize a drawing that matches a given description prompt (Figure 1). Specifically, a pre-trained CLIP model is used as a judge. A CLIP model contains two networks – an image encoder and a textual encoder – which both map their respective inputs into a shared encoding space of a 512-length vector. Similarity is measured via the cosine distance between two encodings. Thus, the goal of CLIPDraw is to produce an image which, when encoded via CLIP, matches the CLIP encoding of the prompt.

Drawings in CLIPDraw are represented by a set of differentiable RGBA Bézier curves, following the method by Li et al. (2020). Each curve is parametrized by 3-5 control points, along with thickness and an RGBA color vector. Drawings initially begin with curves randomly distributed throughout the image, with a white background. During optimization, the number of curves and control points is fixed, however the positions of the points along with the thickness and color vectors can be optimized via gradient descent.

The CLIPDraw algorithm (Algorithm 1) works by running evaluation-time gradient descent, as shown in Figure 2. First, the description phrase is encoded via the CLIP model, and a random set of N Bézier curves are initialized. During each iteration, the curves are rendered to a pixel image via the differentiable renderer, and the resulting image is then duplicated D times and augmented by a random perspective shift and random crop-and-resize. The resulting batch of augmented images is passed into the CLIP image encoder, and the cosine distances to the description phrase are summed

Algorithm 1 CLIPDraw

Input: Description Phrase $desc$; Iteration Count I; Curve Count N; Augment Size D; Pre-trained CLIP model.
Begin:
Encode Description Phrase. *EncPhr = CLIP(desc)*
Initialize Curves. $Curves_{..N}$ *= RandomCurve()*
for $i = 0$ **to** I **do**
  Render Curves to Pixels. *Pixels = DiffRender(Curves)*
  Augment the Image. $AugBatch_{..D}$ *= Augment(Pixels)*
  Encode Image. *EncImg = CLIP(AugBatch)*
  Compute Loss. *Loss =* $-$*CosineSim(EncPhr, EncImg)*
  Backprop. *Curves* $\leftarrow$ *Minimize(Loss)*
end for

to form the loss value. Because all operations are differentiable, gradient descent can be run through the entire loop, optimizing the parameters of the curves to decrease loss. This procedure is repeated I times, until convergence.

The goal of the image augmentation is to force drawings to remain recognizable when viewed through various distortions. Without image augmentation, synthesis-through-optimization methods often result in adversarial images that fulfill the numerical objective but are unrecognizable to humans. This work uses the `torch.transforms.RandomPerspective` and `torch.transforms.RandomResizedCrop` functions in sequence. Note that the specific details of the augmentation were not the focus of this work, furthermore, trials with different augmentations show that augmentation choice does not influence synthesis to a noticeable degree (see Appendix).

Figure 3 showcases the gradual synthesis of a typical CLIPDraw drawing. Note that while the optimization process is largely deterministic, there is randomness in the initial curves and image augmentations, thus multiple runs of CLIPDraw can result in different drawings.

4 Results

In the following sections, various interesting behaviors of CLIPDraw are highlighted through a variety of examples. With the exception of Figure 15, example images are picked to best convey the behavior in consideration. Focus is placed on qualitative observations, unusual behavior, or recurring trends in CLIPDraw image synthesis.

Compared to methods that learn a direct generative model, optimization-based synthesis methods do not require prior training. Instead, images are generated through an evaluation-time optimization loop, aiming to maximize a given objective. This work specifically focuses on synthesizing images that match the CLIP encoding of a description prompt. The following methods are compared:

- **CLIPDraw**, in which drawings are produced by a set of RGBA Bézier curves. The control points, thickness, and colors of the curves can all be adjusted.
- **Pixel Optimization**, which instead optimizes a 224x224x3 matrix of RGB pixels. Otherwise, all algorithmic aspects are the same as CLIPDraw, including image augmentation.
- **BigGAN Optimization**, in which images are produced using a pre-trained BigGAN generator. The weights of the generator are frozen; only the latent Z vectors are optimized. Samples are generated as by Murdock (2021).
- **VQGAN Optimization**, in which images are created through sampling a VQGAN codebook (Esser et al., 2021). Samples are generated as by Crowson et al. (2022).
- **CLIPDraw (No Augment)**, which is identical to CLIPDraw, except no image augmentation is applied to the synthesized drawings.

These methods are run on the same CLIP matching objective for 250 steps of gradient descent (Figure 15). In CLIPDraw, stroke count is 256, and 8 duplicates are used during image augmentation. CLIPDraw tends to result in a diverse set of human-recognizable doodles. Pixel Optimization creates

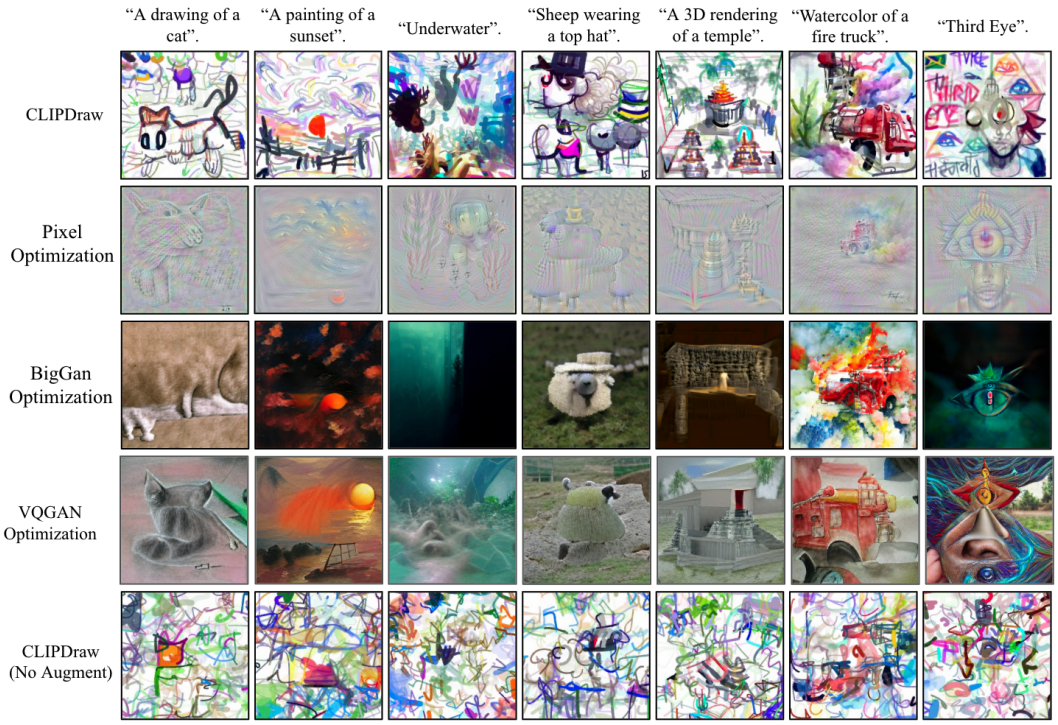


Figure 4: **Images synthesized via various synthesis-through-optimization methods**, all which share the objective of matching a given CLIP-encoded description phrase. All methods optimize for cosine similarity, as shown in Table 1. CLIPDraw can produce a diverse set of human-recognizable drawings based on simple strokes and shapes. CLIPDraw tends to result in simple drawings, often featuring multiple instances of concepts in a variety of styles. On the other hand, Pixel Optimization creates interesting textures but fails to compose colors and shapes. BigGan and VQGAN Optimization can synthesize high-resolution images, but are constrained to the set of natural images the generators are trained on, thus they tend to produce images featuring a single object, and cannot achieve techniques such as writing text. CLIPDraw without image augmentation produces images that score high during training, but fail to retain these scores when tested on augmentations, and are nonsense when viewed by humans. Cosine similarities for each prompt/method pair are shown in Table 1.

Prompt	CLIPDraw	Pixel Opt.	BigGan Opt	VQGAN Opt	(No Aug)
A drawing of a cat.	.376 ± .005	**.385 ± .009**	.325 ± .014	.377 ± .004	.240 ± .015
A paint. of a sunset.	**.390 ± .004**	.215 ± .010	.379 ± .003	.379 ± .003	.325 ± .015
Underwater.	**.413 ± .006**	.385 ± .009	.327 ± .013	.358 ± .006	.247 ± .004
Sheep wearing a top hat.	**.434 ± .010**	**.434 ± .010**	.321 ± .016	.411 ± .009	.215 ± .011
A 3D rend. of a temple.	**.467 ± .005**	.385 ± .009	.311 ± .028	.430 ± .008	.240 ± .009
Watercol. of a firetruck.	**.507 ± .007**	.385 ± .009	.318 ± .014	.457 ± .007	.233 ± .007
Third Eye.	**.436 ± .010**	.385 ± .009	.323 ± .015	.368 ± .005	.251 ± .013

Table 1: **CLIP cosine similarities for images shown above.** Scores are computed by taking the average cosine similarity between the description prompt and 64 augmentations of the generated images. Each method is run for 250 steps of gradient descent, with 10 starting seeds for every prompt, and resulting means and standard deviations are presented. In CLIPDraw results, stroke count is 256, and 8 duplicates are used during image augmentation. Images take around 1-2 minutes to synthesize on a typical Colab GPU.

"A painting of a starry night sky". "Yeti taking a selfie". "Third eye.". "自転車". (Bicycle) "Fast Food".

Figure 5: **CLIPDraw often matches the description prompt through creative techniques**, such as forming letters inside the images, or interpreting ambiguous words in multiple ways.

"A drawing of a cat". "A realistic photograph of a cat". "A cat as 3D rendered in Unreal Engine". "A Japanese woodblock print of one cat". "A watercolor painting of a cat". "A 3D wireframe model of a cat".

Figure 6: **By adjusting adjectives, CLIPDraw can produce drawings of diverse styles.** Styles vary not only in the texture of the images, but showcase different representations of the underlying content, such as a cartoonish cat when prompted for a "drawing", versus a cat in perspective when prompted for a "3D wireframe model".

interesting textures but fails to compose colors and shapes. BigGAN and VQGAN Optimization synthesize high-resolution images, but are constrained to the set of images the generator can produce. Images from CLIPDraw without image augmentation score high numerically, but are nonsense to the human eye.

4.1 What kinds of visual techniques does CLIPDraw use to satisfy the textual description?

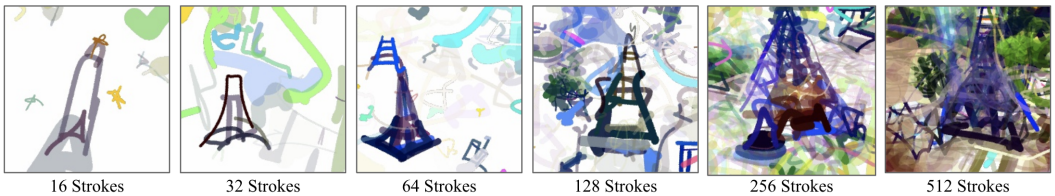
CLIPDraw often results in drawings that match their description prompts in multiple, unexpected ways, as shown in Figure 5. A prime example is the prompt for "a painting of a starry night sky". The drawing's background features a sky with a prominent moon and a few scattered stars. The drawing is rendered in a painterly style, however it also features an actual canvas and painter. Inside the canvas, black and blue swirls resemble Van Gogh's 1889 "The Starry Night".

Another interesting behavior of CLIPDraw is its tendency to write words in the drawing itself. In "Yeti taking a selfie", letters resembling "Yeti" can be seen in the top-right corner. In "Third Eye", again words resembling "third" and "eye" are scattered throughout the image. At times, the drawings contain symbols that are not literally the description, but are still associated, such as the prompt "自転車" (*bicycle* in Japanese) resembling a Google Maps screenshot with a Japanese-like character in the corner.

Prompt ambiguity also presents intriguing results. In "Fast Food", a McDonald's logo along with a set of hamburgers is shown. However, also present are two joggers in a footrace, providing another interpretation of "fast". Included in Figure 5 are the top words predicted by CLIP as closest to the image, showing that CLIP recognizes both "jogging" and "hamburger" as related to the synthesized drawing of "fast food".

4.2 Can CLIPDraw reliably produce drawings in different styles?

A useful feature of CLIPDraw is its ability to adjust not just the content of its drawings, but also the styles, based on the description prompts given. Part of this flexibility is due to the robustness of curve-based images: in comparison to methods that use a pre-trained GAN generator, CLIPDraw drawings are not limited to the space of natural images. Thus, a variety of styles can be produced, and these styles are easily explorable through text.



16 Strokes | 32 Strokes | 64 Strokes | 128 Strokes | 256 Strokes | 512 Strokes

Figure 7: **When stroke count is increased, CLIPDraw produces drawings of increasing realism.** Low-stroke drawings of "The Eiffel Tower" opt for a cartoonish representation, while high-stroke drawings capture 3D depth, background content, and shading.

"Self". | "Happiness". | "Translation". | "The space between infinity". | "What do you look like, CLIPDraw?". | "Enlightenment".

Figure 8: **Abstract prompts grant insight into how CLIP relates visual concepts.** Synthesized images often contain symbols that indirectly relate to the description prompt through a cultural connection.

As shown in Figure 6, a synthesized image of a cat can look vastly different depending on the descriptor words included. When asked for a "drawing of a cat", CLIPDraw synthesized a cartoonish depiction of a cat, comprised mostly of an outline and simple face. A "realistic photograph" features more detailed shading, while a "cat as 3D rendered in Unreal Engine" showcases complex lighting along with a depth-based blurring. Further styles feature a bias towards certain colors, such as the reds and greens of Japanese woodblock prints, or the multi-color blends of watercolors.

An interesting result is that adjusting descriptive adjectives not only changes the textures of the drawings, akin to Style Transfer methods (Gatys et al., 2015), but also changes its structural representation of the underlying content. For example, prompting for "a drawing" produces a flat cartoonish cat, while prompts like "a 3D wireframe" produce a cat in perspective, with depth and shadows.

4.3 How does the stroke count affect what drawings CLIPDraw produces?

When the stroke count is low, CLIPDraw tends to produce cartoonish or abstract representations of the given description prompt. As stroke counts increase, drawings become more detailed and incorporate additional features. Figure 7 shows "The Eiffel Tower" with various stroke counts. In the 16-stroke example, the tower is drawn as only a few straight lines. Higher stroke count images begin more details on the Eiffel Tower itself, along with additional features such as background colors and complex lighting.

A common thread in synthesis-through-optimization methods is that pure optimization leads to undesirable results; it is also necessary to constrain optimization to a suitable space of images, such as the natural images generated by a GAN, or any image made of strokes in the case of CLIPDraw. Limiting stroke count furthers this constraint. When optimizing within the space of 16-stroke images, it is hard to achieve details or textures, thus synthesized drawings will reveal the most basic forms that make up a visual concept. As a tool for AI-assisted art, the stroke-count parameter presents an easy way of adjusting between "simple" and "complex".

4.4 What happens if abstract words are given as a description prompt?

When given an abstract prompt without a literal interpretation, CLIPDraw must utilize cultural connections to come up with visual concepts that relate to the description. Often, this results in drawings that contain symbols relating to the given prompt, such as in Figure 8 with "Happiness" containing

+1.0 "Hashtag". | **+1.0** "Hashtag". **-0.3** "Words and text". **-0.3** "Many ugly, messy, drawings". | **+1.0** "A torii gate." | **+1.0** "A torii gate." **-0.3** "Purple". **-0.3** "Red". | **+1.0** "A realistic painting of a sailboat". | **+1.0** "A realistic painting of a sailboat". **-0.3** "A badly drawn sketch of many sailboats". **-0.3** "Many messy sailboats".

Figure 9: **CLIPDraw behavior adjusted through negative description prompts.** Negative prompts discourage synthesized drawings from matching with them, presenting a tool for fine-tuning.

smiling faces and fireworks, "Translation" showcasing English and Japanese-like characters, and "Enlightenment" featuring a prominent monk-like figure.

At times, synthesized drawings demonstrate concepts through more complex relationships. In the prompt "Self", the resulting drawing features a body with multiple heads, evoking e.g. the idea that a person's self may contain multiple outward personalities. When asked "What do you look like, CLIPDraw?", the synthesized drawing contains a smiling face followed by text resembling "CLIPDRAW". Finally, "The space between infinity" presents a dream-like landscape with an infinity symbol under a galaxy-filled sky.

4.5 Can synthesized drawings be fine-tuned via additional negative prompts?

A common pain point in AI-assisted image synthesis is that is hard to control what the AI will produce. One potential tool for increased control in CLIP-based methods is to introduce negative prompts; the optimization objective is to minimize cosine distance between the CLIP-encoded drawing and the description prompt, while maximizing distance between the drawing and a set of negative prompts.

Figure 9 shows pairs of drawings synthesized from the same random initialization, with the bottom row utilizing additional negative fine-tuning prompts, weighted .3:1. "Hashtag" contains many instances of the word "hashtag" written out. By penalizing "Words and text", the bottom example contains fewer words. By penalizing "Purple" and "Red", the main color of "A torii gate" switches to green. Lastly, the original drawing for "a realistic painting of a sailboat" features many sailboats on an ocean, and penalizing the phrase "many sailboats" results in only one sailboat.

While negative prompts present a richly semantic way to fine-tune image synthesis in CLIP-based methods, it remains tricky to locate prompts that consistently encourage the intended behavior. Many times, negative prompts show negligible effects. During experiments, a cure-all negative prompt such as "a low-quality drawing", with the goal of consistently improving drawing quality, was unable to be found. Further work remains on how to best influence CLIP-based synthesis-through-optimization methods through additional objectives, whether negative or positive.

5 Discussion

This work presents CLIPDraw, a text-to-drawing synthesis method based on the CLIP language-image encoder. CLIPDraw does not require any model training; rather, drawings are synthesized through iterative optimization during evaluation time. CLIPDraw is not the first method to use evaluation-time optimization for image synthesis; many recent works have also used CLIP as an objective. However, by constraining synthesis to images made of RGBA Bézier curves, CLIPDraw biases towards sketches of human-recognizable concepts.

At a higher level, CLIPDraw is work that takes advantage of an existing large pre-trained foundation model (Bommasani et al., 2021), and derives benefit from the representations provided. As foundation models are often viewed as black-box models, it is important to study their use and affect on downstream tasks. Thus, the focus of this paper is to examine the nuances of CLIPDraw behavior, and experiments focus on specific questions and observations about synthesized drawings. These observations, such as prompt ambiguity or the compositionality of styles and content, are qualitatively

distinct and interesting and likely apply to further CLIP-based work in the future, thus this paper aims to start an investigation that continues forward.

5.1 Limitations

Synthesizing high-resolution images is a challenge, and CLIPDraw will often fall short of methods that incorporate a high-functioning generative model. This problem is related to a classic pitfall in synthesis-through-optimization methods, which is that an image may very closely match the CLIP objective, while looking messy and ugly to a human. Thus it is important to introduce auxiliary objectives or constraints. In the case of CLIPDraw, drawings are constrained by the Bézier curve representation, however stricter constraints such as fooling a GAN discriminator may improve quality. However, this work shows that simple image augmentation is enough to unlock the robustness of CLIP representation to generate recognizable images.

A second limitation has to do with using CLIP encodings as a synthesis objective. While CLIP provides a rich textual representation for describing an image, in comparison to coarser neuron-activation objectives, it still remains a challenge to specify details. For example, it is hard to tell CLIPDraw to move a sailboat to the other side of the image. Preliminary experiments also explored negative prompts as a possible direction towards more fine-grained adjustments, however a consistently satisfying method was hard to locate. A promising path in future research can lie in how to correctly steer synthesized images, or introduce finer detail into description prompts via additional objectives.

5.2 Ethics and Social Biases

An important concept to keep in mind when dealing with human data is the existence of inherent social biases contained within. The pre-trained CLIP model is trained on a large corpus of online data, so its representations may include connections or biases that are undesirable. As CLIPDraw does not learn a new model, but instead optimizes based on CLIP itself, the bias studies presented in the CLIP paper (Radford et al., 2021) are highly relevant for CLIPDraw as well. In Section 4.4, a use case for CLIPDraw is mentioned as a tool for exploring visual connections in human culture. It is crucial to recognize that symbols or connections that are formed by CLIPDraw are not necessarily reflective of human culture, but rather are artifacts of the data used to train the original CLIP model. Thus, while CLIPDraw can be used to synthesize drawings that utilize cultural connections to evoke emotions or abstract concepts, it remains the duty of the user to ensure that the final product is up to desired standards.

5.3 Conclusion

Overall, the aim of this work is to introduce CLIPDraw as an easily accessible starting point to experiment with natural language image synthesis. Due to its focus on drawings rather than photorealistic rendering, CLIPDraw presents a straightforward method to examine language-image relationships without the overhead of realism. The presented CLIPDraw implementation can generally synthesize images within a minute on a typical Colab GPU.

This paper describes various interesting behaviors of CLIPDraw, however it is not exhaustive. CLIP-based text-to-image synthesis remains a field with many promising directions, and we hope others will continue to use this work for additional research into the nuances of synthesis-through-optimization, or as a practical tool for AI-assisted art and other interactive visual applications. To this end, source code is available at: **https://bit.ly/3qWTii2**.

References

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. (2021). On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Chen, M., Radford, A., Child, R., Wu, J., Jun, H., Luan, D., and Sutskever, I. (2020). Generative pretraining from pixels. In *International Conference on Machine Learning*, pages 1691–1703. PMLR.

Crowson, K., Biderman, S., Kornis, D., Stander, D., Hallahan, E., Castricato, L., and Raff, E. (2022). Vqgan-clip: Open domain image generation and editing with natural language guidance. *arXiv preprint arXiv:2204.08583*.

Erhan, D., Bengio, Y., Courville, A., and Vincent, P. (2009). Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1.

Esser, P., Rombach, R., and Ommer, B. (2021). Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12873–12883.

Fernando, C., Eslami, S., Alayrac, J.-B., Mirowski, P., Banarse, D., and Osindero, S. (2021). Generative art using neural visual grammars and dual encoders. *arXiv preprint arXiv:2105.00162*.

Frolov, S., Hinz, T., Raue, F., Hees, J., and Dengel, A. (2021). Adversarial text-to-image synthesis: A review. *arXiv preprint arXiv:2101.09983*.

Galatolo, F. A., Cimino, M. G., and Vaglini, G. (2021). Generating images from caption and vice versa via clip-guided generative latent space search. *arXiv preprint arXiv:2102.01645*.

Gatys, L. A., Ecker, A. S., and Bethge, M. (2015). A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014a). Generative adversarial networks. *arXiv preprint arXiv:1406.2661*.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014b). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Li, T.-M., Lukáč, M., Gharbi, M., and Ragan-Kelley, J. (2020). Differentiable vector graphics rasterization for editing and learning. *ACM Transactions on Graphics (TOG)*, 39(6):1–15.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Mirowski, P., Banarse, D., Malinowski, M., Osindero, S., and Fernando, C. (2022). Clip-clop: Clip-guided collage and photomontage. *arXiv preprint arXiv:2205.03146*.

Mirza, M. and Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.

Mordvintsev, A., Olah, C., and Tyka, M. (2015). Inceptionism: Going deeper into neural networks.

Murdock, R. (2021). The big sleep: Bigganxclip.

Nguyen, A., Clune, J., Bengio, Y., Dosovitskiy, A., and Yosinski, J. (2017). Plug & play generative networks: Conditional iterative generation of images in latent space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4467–4477.

Nguyen, A., Dosovitskiy, A., Yosinski, J., Brox, T., and Clune, J. (2016). Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. *arXiv preprint arXiv:1605.09304*.

Nguyen, A., Yosinski, J., and Clune, J. (2015). Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436.

Nilsback, M.-E. and Zisserman, A. (2008). Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE.

Oord, A. v. d., Vinyals, O., and Kavukcuoglu, K. (2017). Neural discrete representation learning. *arXiv preprint arXiv:1711.00937*.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. (2021). Zero-shot text-to-image generation. *arXiv preprint arXiv:2102.12092*.

Reddy, P., Gharbi, M., Lukac, M., and Mitra, N. J. (2021). Im2vec: Synthesizing vector graphics without vector supervision. *arXiv preprint arXiv:2102.02798*.

Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., and Lee, H. (2016). Generative adversarial text to image synthesis. In *International Conference on Machine Learning*, pages 1060–1069. PMLR.

Schaldenbrand, P., Liu, Z., and Oh, J. (2021). Styleclipdraw: Coupling content and style in text-to-drawing synthesis. *arXiv preprint arXiv:2111.03133*.

Shen, I.-C. and Chen, B.-Y. (2021). Clipgen: A deep generative model for clipart vectorization and synthesis. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1.

Tian, Y. and Ha, D. (2022). Modern evolution strategies for creativity: Fitting concrete images and abstract concepts. In *International Conference on Computational Intelligence in Music, Sound, Art and Design (Part of EvoStar)*, pages 275–291. Springer.

Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. (2011). The caltech-ucsd birds-200-2011 dataset.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [N/A]

(b) Did you mention the license of the assets? [N/A]

(c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Appendix

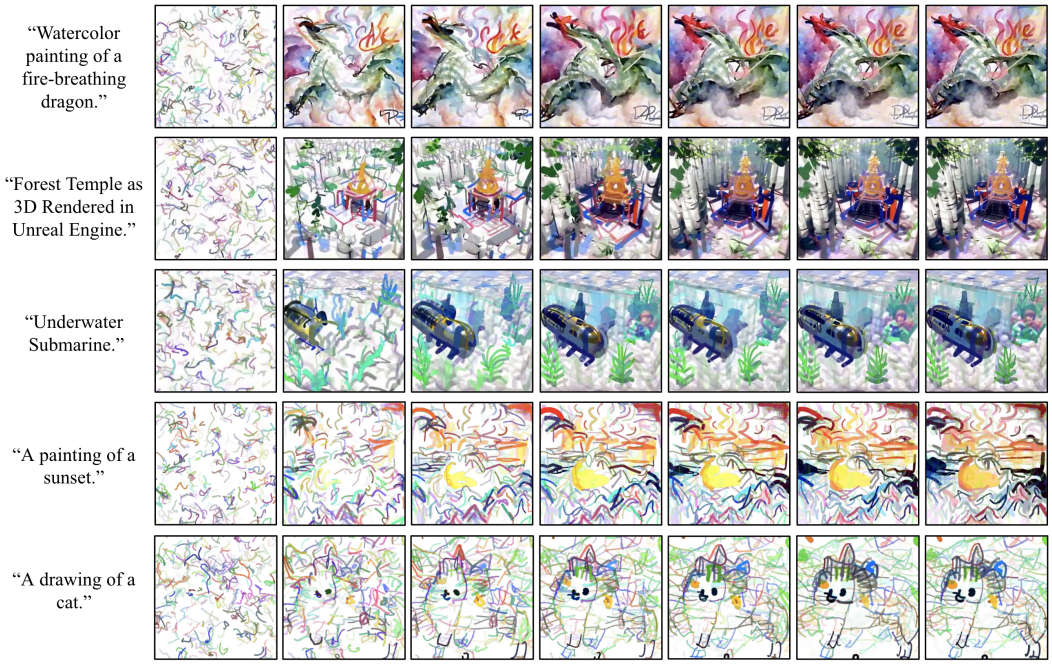


Figure 10: **Images generated over time via CLIPDraw.** On the X axis is a sample of the generated image recorded periodically during optimization. Images begin as a randomly distributed set of Bezier curves. Specifically, we iterate through the pixel space over the X and Y axes, periodically spawning curves within a constrained area. This initialization allows the curves to appear locally noisy, yet cover a uniform area globally. Since CLIPDraw depends on gradient information to adjust its image, the initial positions of the curves play a role in the final image. Notably, CLIPDraw images often approach diminishing returns in improvement around the 500-1000 iteration mark, for 256-stroke and 512-stroke drawings. At this point, the drawings begin to adjust only by wiggling their strokes, and large improvements no longer appear. To eliminate wiggling, it is possible to reduce the gradient step as optimization proceeds, thus guiding the optimization towards larger changes early on and towards finer-tuned details in later iterations.

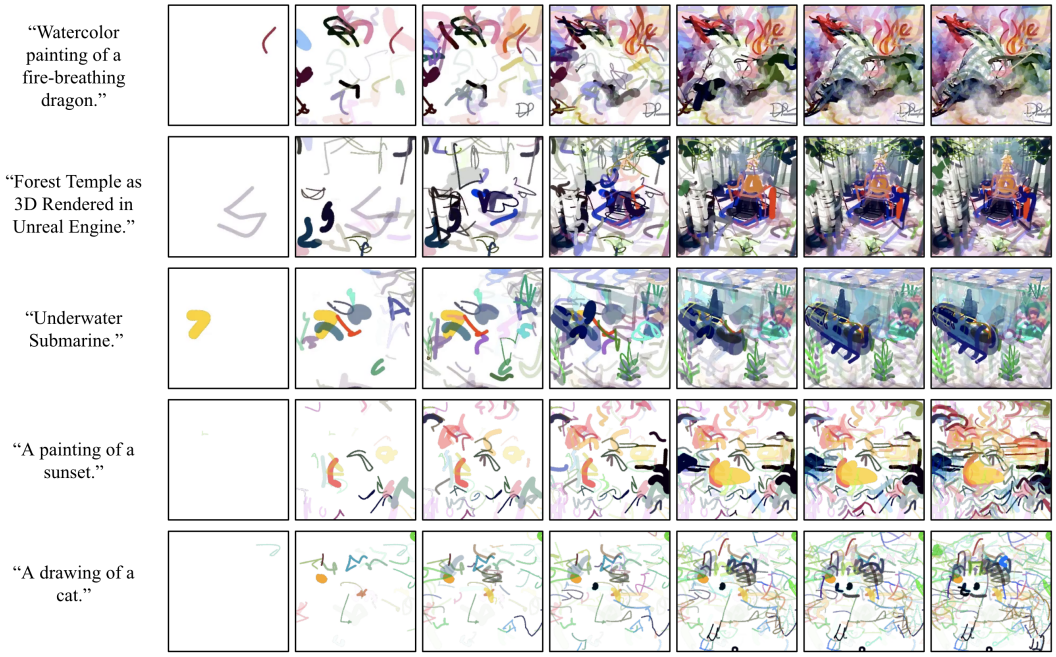


Figure 11: **Images generated via CLIPDraw, rendered stroke-by-stroke.** Because CLIPDraw represents drawings as a series of Bezier strokes, it is possible to render these images as a sequence of frames in which strokes are periodically added. The order of the strokes matters only in the sense of opacity layering, i.e. a newer stroke will overlay ontop of an older stroke. CLIPDraw only calculates gradient information using the final frame, so no pressure is applied to encourage prompt resemblance in the earlier frames, although this framework is easily possible. In cases where the drawing is comprised of a large number of strokes, such as "Watercolor painting of a fire-breathing dragon" and "Underwater submarine", a large portion of the drawing is formed with early strokes, and later strokes often have low opacity and/or a thinner length. This situation shows that additional strokes provide only marginal improvement. In contrast, drawings with lower numbers of strokes depend more on later strokes, like the body in "a drawing of a cat" only appearing in the last two frames.

Figure 12: **Images optimized from the same description phrase, with different random seed initializations.** CLIPDraw is a deterministic algorithm, with the exception of the random initialization used to seed the Bezier curve positions. Importantly, the random initialization of these curves play a large part in the positional composition of the final image. For example, in "a painting of a sunset", the position of the sun can appear in many different locations of the image. At the same time, the high-level content of the image depends greatly on the prompt, and the random initialization does not play a large factor. In "a painting of a sunset", while the sun may move around, it is always rendered as a yellow circle with an orange border, surrounded by waves. In "A 3D rendering of a temple", the architecture of the temple adjusts from seed to seed, yet the temple always features a prominent 45-degree angled tower and surrounding trees.

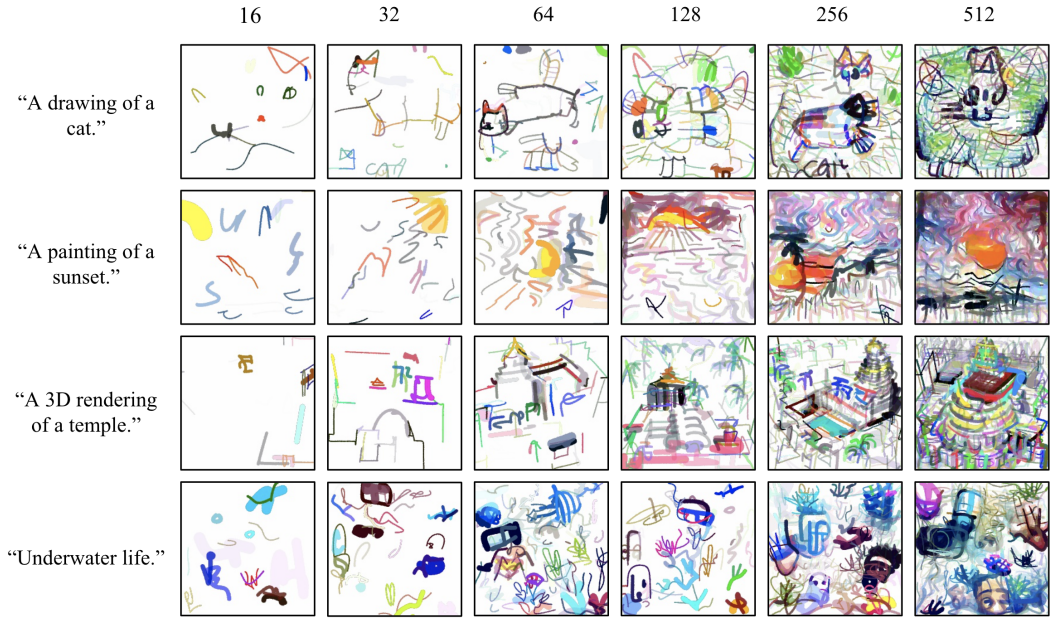


Figure 13: **Images synthesized via CLIPDraw with varying numbers of strokes.** One key parameter in the CLIPDraw algorithm is the number of Bezier strokes used in each drawing. As described in the main paper, as the number of strokes increases, the details of the synthesized drawings move from a cartoonish representation to a more realistic rendering. In this figure we showcase more examples of stroke-count adjustment. The synthesized images show that CLIPDraw generally creates a similar composition regardless of stroke count, except each object is rendered in a more simple form. For example, the cats in lower-count settings have lines for legs, whereas in higher-count settings the cats also have fingers and whiskers. This observation shows that CLIP as a metric generally has a single mode for the best way to represent a prompt in latent space, although this mode may align with many rendered images. Thus, changing the composition of an image will have a large affect on CLIP score, whereas changing the rendering style will have a lesser impact. In some cases, however, lower stroke-count results in abnormal creativity, such as the 64-stroke "A 3D rendering of a temple" discovering text that forms the word "Temple".

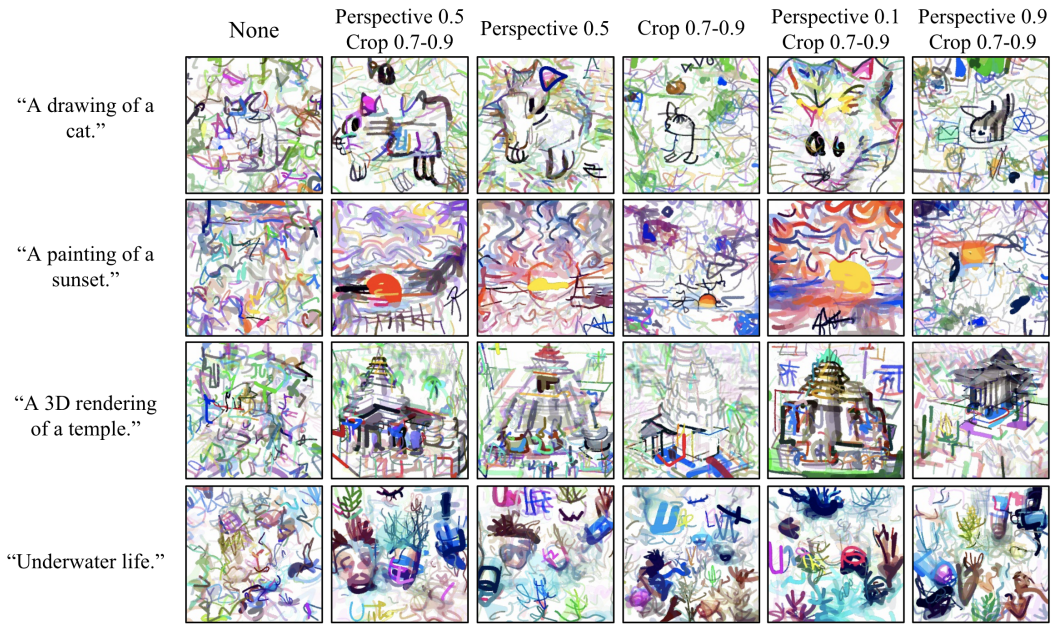


Figure 14: **Different augmentations and their affect on CLIPDraw synthesis.** In the main paper, we state that the choice of augmentation does not play a large role in the qualitative appearance of resulting images. This figure aims to provide more examples towards that point. Notably, using no image augmentations results in a mess of strokes, as the gradient signal points to an adversarial mode where the images hold a high CLIP score, yet look like nonsense to humans. Any forms of image augmentation we tried, as listed in the X-axis, improved performance greatly. The key choice of image augmentation does not seem to play a large role, rather, the important constraint is that drawings must be robust to some form of augmentation, which forces the optimization to stay away from adversarial examples.

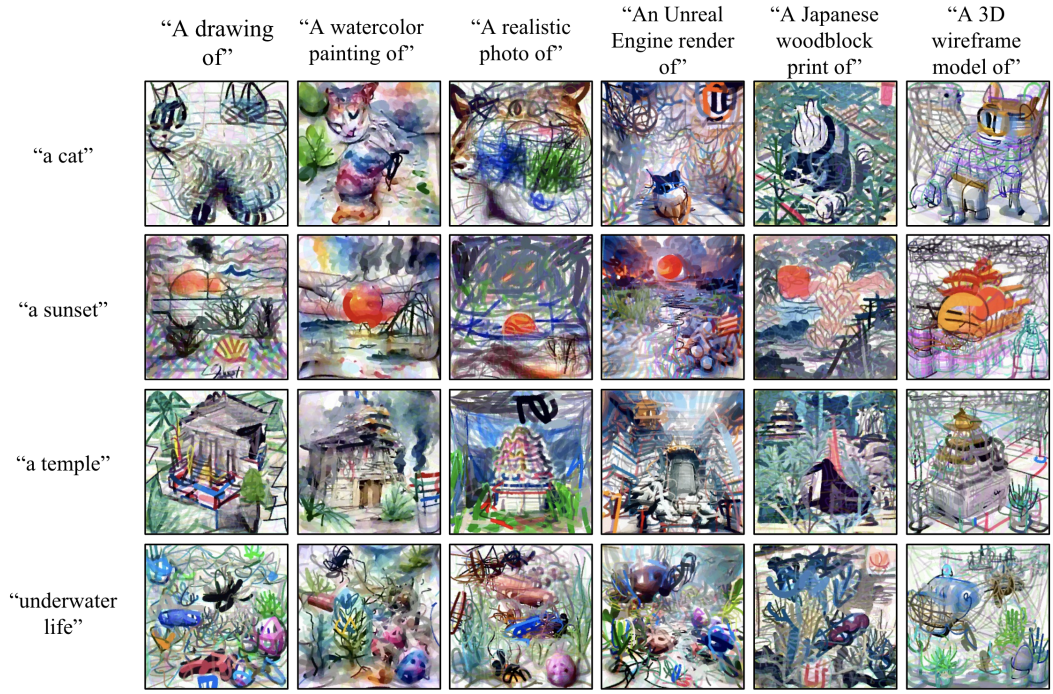


Figure 15: **Images synthesized with the same base word, using different stylistic descriptions.** In the main paper, we show that CLIPDraw can synthesize images in various artistic styles by simply describing that style using language. This figure provides more examples. Prompts are created by concatenating the style description to the base word, i.e. "A realistic photo of" + "a temple". Images are generated using 512 strokes rather than 256 to highlight fine-grained details of each style. Similarly to as seen in the random seed initialization experiment, the general composition of each drawing remains consistent across all styles. Certain exceptions exist, such as "a realistic photo of" tending to place the object close-up in the center of the frame, and "a 3D wireframe model of" tending to show objects at a 45-degree angle. In general, these results show that CLIPDraw is compositional in terms of image content and style.