CLIPasso: Semantically-Aware Object Sketching

YAEL VINKER, Tel Aviv University, Israel
EHSAN PAJOUHESHGAR, Swiss Federal Institute of Technology (EPFL), Switzerland
JESSICA Y. BO, Swiss Federal Institute of Technology (EPFL), Switzerland
ROMAN CHRISTIAN BACHMANN, Swiss Federal Institute of Technology (EPFL), Switzerland
AMIT HAIM BERMANO, Tel Aviv University, Israel
DANIEL COHEN-OR, Tel Aviv University, Israel
AMIR ZAMIR, Swiss Federal Institute of Technology (EPFL), Switzerland
ARIEL SHAMIR, Reichman University, Israel

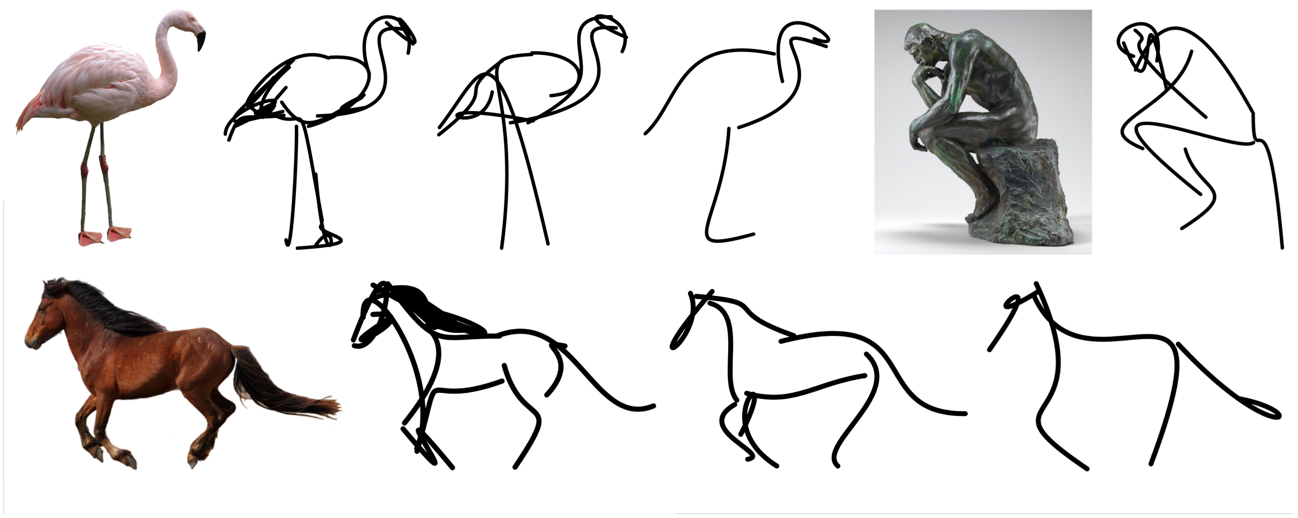


Fig. 1. Our work converts an image of an object to a sketch, allowing for varying levels of abstraction, while preserving its key visual features. Even with a very minimal representation (the rightmost flamingo and horse are drawn with only a few strokes), one can recognize both the semantics and the structure of the subject depicted. ©The Thinker (Le Penseur), model 1880 by Auguste Rodin, Gift of Mrs. John W. Simpson [Public Domain] via (https://bit.ly/3F48SxC); Flamingo from freepikpsd.com [Free for commercial use] via (https://bit.ly/3F6tipS); "Wild Horses" by firelizard5 [CC BY 2.0] via (https://bit.ly/39p4KfX).

Abstraction is at the heart of sketching due to the simple and minimal nature of line drawings. Abstraction entails identifying the essential visual properties of an object or scene, which requires semantic understanding and prior knowledge of high-level concepts. Abstract depictions are therefore challenging for artists, and even more so for machines. We present CLIPasso, an object sketching method that can achieve different levels of abstraction, guided by geometric and semantic simplifications. While sketch generation methods often rely on explicit sketch datasets for training, we utilize the remarkable ability of CLIP (Contrastive-Language-Image-Pretraining) to distill semantic concepts from sketches and images alike. We define a sketch as a set of Bézier curves and use a differentiable rasterizer to optimize the parameters of the curves directly with respect to a CLIP-based perceptual loss. The abstraction degree is controlled by varying the number of strokes. The generated sketches demonstrate multiple levels of abstraction while maintaining recognizability, underlying structure, and essential visual components of the subject drawn.

CCS Concepts: • **Computing methodologies → Computer graphics**.

Additional Key Words and Phrases: Sketch Synthesis, Image-based Rendering, Vector Line Art Generation

ACM Reference Format:
Yael Vinker, Ehsan Pajouheshgar, Jessica Y. Bo, Roman Christian Bachmann, Amit Haim Bermano, Daniel Cohen-Or, Amir Zamir, and Ariel Shamir. 2022. CLIPasso: Semantically-Aware Object Sketching. *ACM Trans. Graph.* 41, 4, Article 86 (July 2022), 11 pages. https://doi.org/10.1145/3528223.3530068

Authors' addresses: Yael Vinker, Tel Aviv University, Israel, yael.vinker@mail.huji.ac.il; Ehsan Pajouheshgar, Swiss Federal Institute of Technology (EPFL), Switzerland, e.pajouheshgar@gmail.com; Jessica Y. Bo, Swiss Federal Institute of Technology (EPFL), Switzerland, jessica7bo@gmail.com; Roman Christian Bachmann, Swiss Federal Institute of Technology (EPFL), Switzerland, roman.bachmann@epfl.ch; Amit Haim Bermano, Tel Aviv University, Israel, amberman@tauex.tau.ac.il; Daniel Cohen-Or, Tel Aviv University, Israel, dcor@taux.tau.al.il; Amir Zamir, Swiss Federal Institute of Technology (EPFL), Switzerland, amir.zamir@epfl.ch; Ariel Shamir, Reichman University, Israel, arik@idc.ac.il.

0730-0301/2022/7-ART86 $15.00
https://doi.org/10.1145/3528223.3530068

Fig. 2. "Le Taureau" by Picasso — note how the abstraction process is achieved by gradually *removing* elements while the bull's essence is preserved. ©"Pasadena, Norton Simon Museum, Picasso P. The Bull, 1946" photo by Vahe Martirosyan, [CC BY-SA 2.0] via (https://bit.ly/3MFB3pm).

1 INTRODUCTION

Free-hand sketching is a valuable visual tool for expressing ideas, concepts, and actions [Fan et al. 2018; Gryaditskaya et al. 2019; Hertzmann 2020; Tversky 2002; Xu et al. 2020]. As sketches consist of only strokes, and often only a limited number of strokes, the process of *abstraction* is central to sketching. An artist must make representational decisions to choose key visual features of the subject drawn to capture the relevant information she wishes to express, while omitting (many) others [Chamberlain and Wagemans 2016; Fan et al. 2019; Yang and Fan 2021].

For example, in the famous "Le Taureau" series (Figure 2), Picasso depicts the progressive abstraction of a bull. In this series of lithographs, the artist transforms a bull from a concrete, fully rendered, anatomical drawing, into a sketch composition of a few lines that still manages to capture the essence of the bull.

In this paper, we pose the question — can computer renderings imitate such a process of sketching abstraction, converting a photograph from a concrete depiction to an abstract one?

Today, machines can render realistic sketches simply by applying mathematical and geometric operations to an input photograph [Canny 1986; Winnemöller et al. 2012]. However, creating abstractions is more difficult for machines to achieve. The abstraction process suggests that the artist selects visual features that capture the underlying structure and semantic meaning of the object or scene, to produce a minimal, yet descriptive rendering. This demands semantic understanding of the subject, which is more complex than applying simple geometric operations to the image. To fill this semantic gap, we use CLIP [Radford et al. 2021], a neural network trained on various styles of images paired with text. CLIP is exceptional at encoding the semantic meaning of visual depictions, regardless of their style [Goh et al. 2021].

Previous works that attempt to replicate human-like sketching often use sketch datasets of the desired level of abstraction to guide the form and style of the generated sketch [Berger et al. 2013; Li et al. 2015; Muhammad et al. 2018]. While such data-driven approach can imitate the final rendering of human artwork, it requires the existence and availability of relevant datasets, and it restricts the output style to match this data. In contrast, we present an optimization-based photo-to-sketch generation technique that achieves different levels of abstraction without requiring an explicit sketch dataset. Our method uses the CLIP image encoder to guide the process of converting a photograph to an abstract sketch. CLIP encoding provides the semantic understanding of the concept depicted, while the photograph itself provides the geometric grounding of the sketch to the concrete subject.

Our sketches are defined using a set of thin, black strokes (Bézier curves) placed on a white background, and the level of abstraction is dictated by the number of strokes used. Given the target image to be drawn, we use a differentiable rasterizer [Li et al. 2020] to directly optimize the strokes' parameters (control points positions) with respect to a CLIP-based loss. We combine the final and intermediate activations of a pre-trained CLIP model to achieve both geometric and semantic simplifications. For improved robustness, we propose a saliency-guided initialization process, based on the local attention maps of a pretrained vision transformer model.

The resulting sketches (see Figure 1) demonstrate a combination of the semantic and visual features that capture the essence of the input object, while still being minimal and providing good category and instance level object recognition clues.

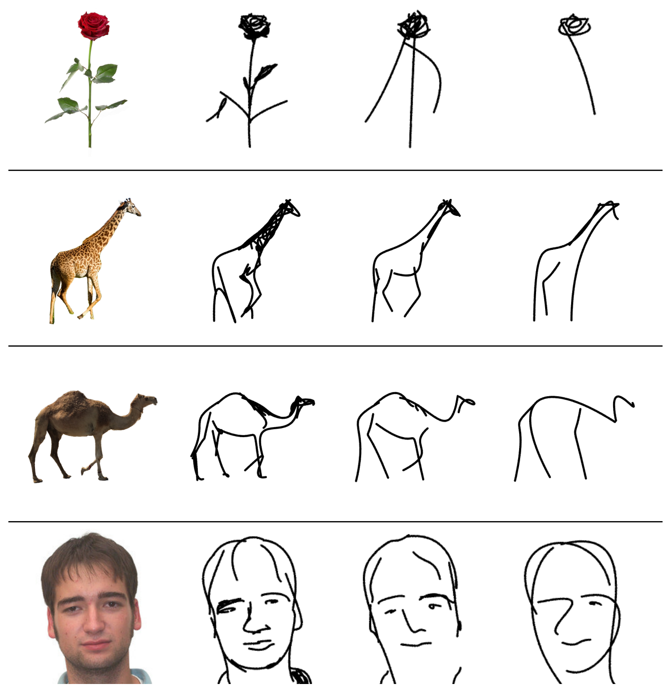


Fig. 3. Different levels of abstraction produced by our method. Left to right: input images and increased level of abstraction. The top three sketches were produced using 16, 8, and 4 strokes in columns 2, 3, and 4, respectively, and the man's sketch was produced using 32, 16, and 8 strokes. ©Rose by AndreaA - Can Stock Photo Inc. [Standard License Agreement] via (https://bit.ly/39w8egS); Giraffe [Public Domain US] via (https://bit.ly/3s61Env); Camel [Public Domain] via (https://bit.ly/3s1o1KU); Face image from [Minear and Park 2004], used with permission.

2 RELATED WORK

Unlike edge-map extraction methods [Canny 1986; Winnemöller et al. 2012] which are purely based on geometry, free-hand sketch generation aims to produce sketches that are abstract in terms of structure and semantic interpretation so as to mimic a human-like style. This high-level goal varies among different works, as there are many styles and levels of abstraction that can be produced. Consequently, existing works tend to choose the desired output style based on a given dataset: from highly abstract — guided only by a category-based text prompt [Ha and Eck 2017], to more concrete [Arbeláez et al. 2011], which is guided by contour detection. Figure 4 illustrates this spectrum. While methods that rely on sketch datasets are limited to the abstraction levels present, our method is optimization based. Hence, it is capable of producing multiple levels of abstraction without relying on the existence of suitable sketch datasets or requiring a lengthy new training phase.

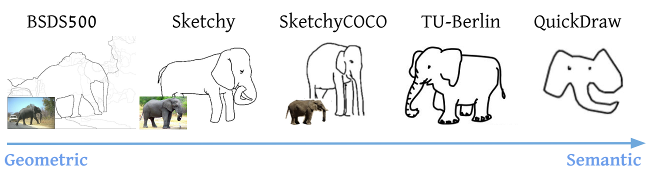


Fig. 4. Variations in style and abstraction among sketch datasets — examples are arranged from left to right by the degree of abstraction: from edge-based to category-based sketches. For datasets that have image references, the input image is placed alongside the sketch; otherwise, the input is just the category label (e.g., "elephant").

We provide a brief review of existing relevant photo-sketch synthesis works, which all rely on sketch-specific datasets. Table 1 summarizes the high-level characteristics that differentiate these methods.

Photo-Sketch Synthesis. Early methods learn explicit models to synthesize facial sketches [Berger et al. 2013; Chen et al. 2001]. To generalise to categories beyond faces, Li et al. [2015] learn a deformable stroke model based on perceptual grouping.

In the deep learning era, it is intuitive to think of photo-sketch generation as a domain translation task. However, the highly sparse and abstract nature of sketches introduces challenges for trivial methods [Isola et al. 2017; Wang et al. 2017] to adhere to the sketch domain, and therefore sketch-specific adjustments must be made.

Song et al. [2018] propose a hybrid supervised-unsupervised multi-task learning approach with a shortcut cycle consistency constraint. Li et al. [2019] present a learning-based contour generation algorithm to resolve the diversity of the human drawings in the dataset. Kampelmuhler and Pinz [2020] propose an encoder-decoder architecture, where the loss is guided by a pretrained sketch classifier network. Qi and Su et al. [2021] propose a lattice representation for sketches, employing LSTM and graph models to generate a vector sketch from points sampled from the edge map. The density of points determines the abstraction level of the sketch.

A different approach for image-sketch synthesis formulates the sketching task as a multi-agent referential game in which two reinforcement learning agents must communicate visual concepts to each other through sketches [Mihai and Hare 2021b; Qiu et al. 2021]. Similarly to these works, we were inspired by cognitive processes to identify and formulate our problem and objective. However, our primary focus is on abstractions as an integral part of sketches, and our objective is to produce sketches that are interpertable by humans. Their primary focus is on building a visual communication channel between agents, using sketches as a tool, where drawing is done in context.

Table 1. Comparison of sketch synthesis algorithms. **(A)** Is not restricted to categories from training dataset, **(B)** Can produce different levels of abstractions, **(C)** Is not limited to abstractions in the dataset, **(D)** Can produce vector sketches, **(E)** Can produce a sequential sketch **(F)** Is not directly relying on the edge map.

Method	A	B	C	D	E	F
Berger et al. [2013]	✗	✓	✗	✗	✓	✗
Li et al. [2015]	✗	✗	✗	✓	✗	✗
Muhammad et al. [2018]	✗	✓	✓	✓	✓	✗
Song et al. [2018]	✗	✗	✗	✓	✓	✓
Li et al. [2019]	✓	✗	✗	✗	✗	✓
Kampelmühler and Pinz [2020]	✗	✗	✗	✗	✗	✓
Qi and Su et al. [2021]	✗	✓	✓	✓	✓	✗
Ours	✓	✓	✓	✓	✗	✓

Vector Graphics. There is a substantial literature on stroke-based rendering, contour visualization, and feature line rendering, summarized in the surveys by Hertzmann [2003], and by Bénard and Hertzmann [2019]. Vector representations are widely used for a variety of sketching tasks and applications, employing a number of deep learning models including RNN [Ha and Eck 2017], BERT [Lin et al. 2020], Transformers [Bhunia et al. 2020; Ribeiro et al. 2020], CNNs [Chen et al. 2017], GANs [Varshaneya et al. 2021] and reinforcement learning algorithms [Ganin et al. 2018; Mellor et al. 2019; Zhou et al. 2018]. The recent development of differentiable rendering algorithms [Li et al. 2020; Mihai and Hare 2021a; Zheng et al. 2019] makes it possible to manipulate or synthesize vector content by using raster-based loss functions. We use the method of Li et al. [2020], as it can handle a wide range of curves and strokes, including Bézier curves.

Sketches Abstraction. Only two previous works propose a unified model to produce sketches of a given image at different levels of abstraction. Berger et al. [2013] collected a dataset of portraits drawn by professional artists at different levels of abstraction. For each artist, a library of strokes is created indexed by shape, curvature, and length, and these are used to replace curves extracted from the image edge map. Their method is limited to portraits and requires a new dataset for each level of abstraction.

Muhammad et al. [2018] propose a stroke-level sketch abstraction model. A reinforcement learning agent is trained to select which strokes can be removed from an edge map representation of the input image without affecting its recognizability. The recognition signal is provided by a sketch classifier trained on 9 classes from

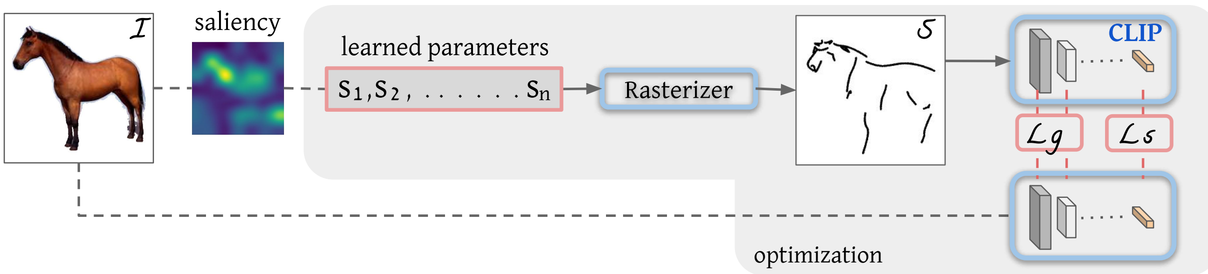


Fig. 5. Method overview – Given a target image $\mathcal{I}$, and the number of strokes n, a saliency map is used as the distribution to sample the initial strokes locations $\{s_1, ..s_n\}$. A differentiable rasterizer $\mathcal{R}$ is used to create a resterized sketch $\mathcal{S}$. Both the sketch and the image are fed into a pretrained CLIP model to evaluate the geometric distance L_g and semantic distance L_s between the two. The loss is backpropagated through $\mathcal{R}$ to optimize the strokes parameters until convergence. The learned parameters and loss terms are highlighted in red, while the blue components are frozen during the entire optimization process, solid arrows are used to mark the backpropagation path. ©Horse from cadnav.com [Personal and Commercial] via (https://bit.ly/3vzopmb).

the QuickDraw dataset [Ha and Eck 2017], and hence to operate on new classes, a fine-tuning stage is required.

CLIP-based Image Abstraction. CLIP [Radford et al. 2021] is a neural network trained on 400 million image-text pairs collected from the internet with the objective of creating a joint latent space using contrastive learning. Being trained on a wide variety of image domains along with lingual concepts, CLIP models are found to be very useful for a wide range of zero-shot tasks. The most relevant works within our context are by Frans et al. [2021] (CLIPDraw), and Tian and Ha [2021]. CLIPDraw optimizes a set of random Bezier curves to create a drawing that maximizes the CLIP similarity for a given text prompt. Likewise, we also use a differentiable rasterizer [Li et al. 2020] and a CLIP-based loss. However, while CLIPDraw is purely text-driven, we allow control over the output appearance, conditioned on the input image. For this purpose, we introduce a new geometric loss term and a saliency-guided initialization procedure.

Tian and Ha [2021] employ evolutionary algorithms combined with CLIP, to produce creative abstract concepts represented by colored triangles guided by text or shape. Their results are limited to either fully semantic (using CLIP's text encoder) or entirely geometric (using L2), whereas we are able to integrate both.

3 METHOD

We define a sketch as a set of n black strokes $\{s_1, ..s_n\}$ placed on a white background. We use a two-dimensional Bézier curve with four control points $s_i = \{p_i^j\}_{j=1}^4 = \{(x_i, y_i)^j\}_{j=1}^4$ to represent each stroke. For simplicity, we only optimize the position of control points and choose to keep the degree, width, and opacity of the strokes fixed. However, these parameters can later be used to achieve variations in style (see Figure 10a). The parameters of the strokes are fed to a differentiable rasterizer $\mathcal{R}$, which forms the rasterized sketch $\mathcal{S} = \mathcal{R}(\{p_1^j\}_{j=1}^4, ...\{p_n^j\}_{j=1}^4) = \mathcal{R}(s_1, ..s_n)$. As is often conventional [Mellor et al. 2019; Muhammad et al. 2018; Tian and Ha 2021], we vary the number of strokes n to create different levels of abstraction.

An overview of our method can be seen in Figure 5. Given a target image $\mathcal{I}$ of the desired subject, our goal is to synthesize the corresponding sketch $\mathcal{S}$ while maintaining both the semantic and geometric attributes of the subject. We begin by extracting the salient regions of the input image to define the initial locations of the strokes. Next, in each step of the optimization we feed the stroke parameters to a differentiable rasterizer $\mathcal{R}$ to produce the rasterized sketch. The resulting sketch, as well as the original image are then fed into CLIP to define a CLIP-based perceptual loss. We back-propagate the loss through the differentiable rasterizer and update the strokes' control points directly at each step until convergence of the loss function.

3.1 Loss Function

As sketches are highly sparse and abstract, pixel-wise metrics are not sufficient to measure the distance between a sketch and an image. Additionally, even though perceptual losses such as LPIPS [Zhang et al. 2018] can encode semantic information from images, they may not be suitable to encode abstract sketches, as illustrated in Figure 6 (for further analysis, please refer to the supplementary material). One solution is to train task-specific encoders to learn a shared embedding space of images and sketches under which the distance between the two modalities can be computed [Kampelmühler and Pinz 2020; Song et al. 2018]. This approach depends on the availability of such datasets, and requires additional effort for training the models.

Instead, we utilize the pretrained image encoder model of CLIP, which was trained on various image modalities so that it can encode information from both natural images and sketches without the need for further training. CLIP encodes high-level semantic attributes in the last layer since it was trained on both images and text. We therefore define the distance between the embeddings of the sketch $CLIP(\mathcal{R}(\{s_i\}_{i=1}^n)$ and image $CLIP(\mathcal{I})$ as:

$$L_{semantic} = dist\big(CLIP(\mathcal{I}), CLIP(\mathcal{R}(\{s_i\}_{i=1}^n)\big), \quad (1)$$

where $dist(x, y) = 1 - \frac{x \cdot y}{||x|| \cdot ||y||}$ is the cosine distance. However, the final encoding of the network is agnostic to low-level spatial features such as pose and structure. To measure the geometric similarity between the image and the sketch, and consequently, allow some control over the appearance of the output, we compute the $L2$ distance between intermediate level activations of CLIP:

$$L_{geometric} = \sum_l \left\| CLIP_l(\mathcal{I}) - CLIP_l(\mathcal{R}(\{s_i\}_{i=1}^n)) \right\|_2^2, \quad (2)$$

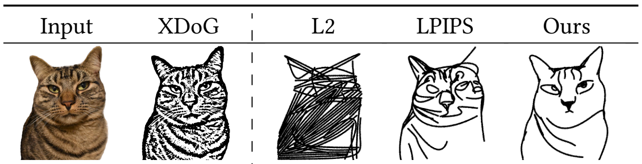


Fig. 6. Loss functions comparison — we optimize the strokes by minimizing different losses: L2 loss simply encourages the filling of colored pixels, LPIPS is more semantically aware, but the resulting sketch is still close to the edge map (see the XDog edges for comparison). In contrast, our CLIP-based loss allows better semantic depiction while preserving the morphology of the subject. ©"Cat" by Burnt Pineapple Productions [Public Domain], via (https://bit.ly/3y4IG4H).

where $CLIP_l$ is the $CLIP$ encoder activation at layer l. Specifically, we use layers 3 and 4 of the ResNet101 CLIP model. The final objective of the optimization is then defined as:

$$\min_{\{s_i\}_{i=1}^n} L_{geometry} + w_s \cdot L_{semantic}, \tag{3}$$

with $w_s = 0.1$. We analyze the contribution of different layers and weights, as well as the results of using different CLIP models in the supplementary material.

3.2 Optimization

Our goal is to optimize the set of parameters $\{s_i\}_{i=1}^n = \{\{p_i^j\}_{j=1}^4\}_{i=1}^n$ to define a sketch that closely resembles the target image $\mathcal{I}$ in terms of both geometry and semantics. At each step of the optimization, we use the Adam optimizer [Kingma and Ba 2015] to compute the gradients of the loss with respect to the strokes' parameters $\{s_i\}_{i=1}^n$. We follow the same data augmentation scheme suggested in CLIPDraw [Frans et al. 2021] and apply random affine augmentations to both the sketch and the target image before feed-forwarding into CLIP. The transformations we use are RandomPerspective and RandomResizedCrop. These augmentations prevent the generation of adversarial sketches, which minimize the objective but are not meaningful to humans. We repeat this process until convergence, (when the difference in loss between two successive evaluations is less than 0.00001). This typically takes around 2000 iterations. Figure 7 illustrates the progression of the generated sketch as the optimization evolves. The learning rate is set to 1 and we evaluate the output sketch every 10 iterations. Evaluation is done by computing the loss without random augmentations. It takes 6 minutes to run 2000 iterations on a single Tesla V100 GPU.

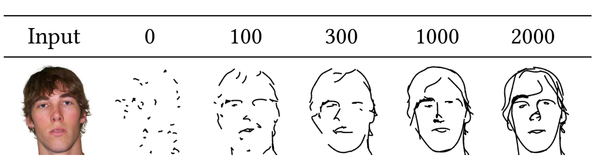
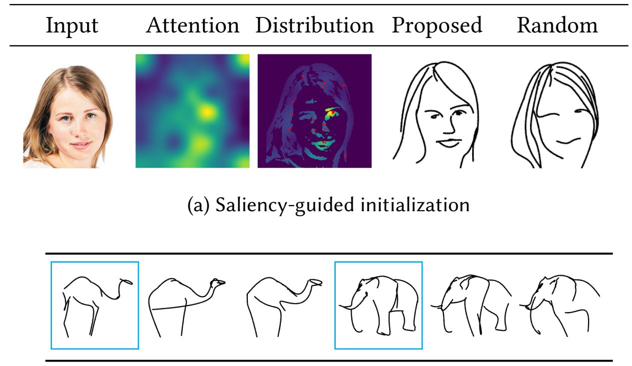


Fig. 7. The sketch appearance throughout the optimization iterations. ©Face image from [Minear and Park 2004], used with permission.



(a) Saliency-guided initialization

(b) Automatic selection procedure of the final sketch

Fig. 8. Strokes Initialization. (a) Left to right: input, the saliency map produced from CLIP ViT activations, final distribution map (adjusted to adhere to image edges) with sampled initial stroke locations (in red), the sketch produced using the proposed initialization procedure, and the sketch produced when using random initialization. (b) Results of three different initializations with the same number of strokes. The sketches marked in blue produced the lowest loss value, and would thus be used as the final output. ©Face image from [Minear and Park 2004], used with permission.

3.3 Strokes Initialization

Our objective function is highly non-convex. Therefore, the optimization process is susceptible to the initialization (i.e., the initial location of the strokes). This is especially significant at higher levels of abstraction – where very few strokes must be wisely placed to emphasize semantic components. For example, in Figure 8a, the sketches in the last two columns were produced using the same number of strokes, however, in the "Random" initialization case, more strokes were devoted to the hair while the eyes, nose, and mouth are more salient and critical features of the face.

To improve convergence towards semantic depictions, we place the initial strokes based on the salient regions of the target image. To find these regions, we use the pretrained vision transformer [Kolesnikov et al. 2021] ViT-B/32 model of CLIP, that performs global context modeling using self-attention between patches of a given image to capture meaningful features. We use the recent transformer interpretability method by Chefer et al. [2021] to extract a relevancy map from the self-attention heads, without any text supervision. Next, we multiply the relevancy map with an edge map of the image extracted using XDoG method [Winnemöller et al. 2012]. Multiplying with XDoG is used to strengthen the morphological positioning of the strokes, motivated by the hypothesis that edges are effective in predicting where people draw lines [Hertzmann 2021]. Finally, we normalize the final relevancy map using softmax and use it as a distribution map so that pixels in salient regions are assigned a higher probability. We sample n positions (pixels), and use them as the position of the first control point p_i^1 of each Bezier curve. The other 3 additional points (p_i^2, p_i^3, p_i^4) are sampled within a small radius (0.05 of image size) around p_i^1 to define the initial set of Bezier curves $\{\{p_i^j\}_{j=1}^4\}_{i=1}^n$.

Figure 8a illustrates this procedure. It can be seen that our saliency-based initialization contributes significantly to the quality of the final sketch compared to random initialization.

This sampling-based approach also lends itself to providing variability in the results. In all our examples we use 3 initializations and automatically choose the one that yields the lowest loss (see Figure 8b). Only for the highly abstract cases of Figure 3 (rightmost column) we used 5 initializations. We further analyze the initialization procedure and variability in the supplementary material.

4 RESULTS

Section 4.1 provide qualitative evaluations. In section 4.2 we compare our method with existing image-to-sketch methods, which were all trained on sketch-specific datasets. In Section 4.3, we supply a quantitative evaluation of our method's ability to produce recognizable sketches testing both category and instance recognition. For images with background, we use an automatic method (U2-Net [Qin et al. 2020]) to mask out their background. We provide further analysis of our method, extra results, and extended comparison with other methods in the supplemental file.

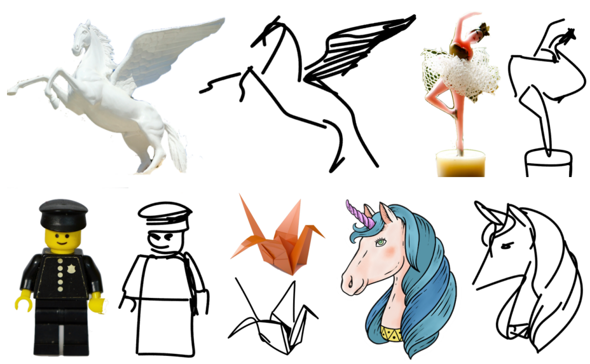
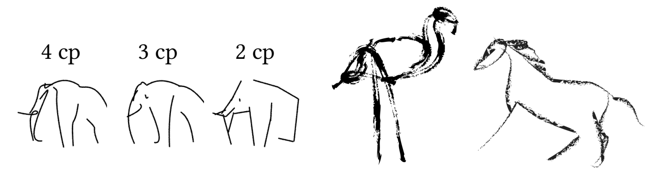


Fig. 9. Sketches produced by our method for infrequent categories. ©Pegasus from rawpixel [Public Domain], via (https://bit.ly/3w1a5lv); "Ballerina" by Gerald Pereira [CC BY 2.0], via (https://bit.ly/3KxxiAP); Unicorn from rawpixel [Commercial License (https://www.rawpixel.com/services/licenses)], via (https://bit.ly/3F9zyxi); Lego [Public Domain] from rawpixel, via (https://bit.ly/3s54LMN); Origami [Public Domain] from rawpixel, via (https://bit.ly/3vxtgUH).

4.1 Qualitative Evaluation

Our approach is different from conventional sketching methods in that it does not utilize a sketch dataset for training, rather it is optimized under the guidance of CLIP. Thus, our method is not limited to specific categories observed during training, as no category definition was introduced at any stage. This makes our method robust to various inputs, as shown in Figures 1 and 9.

In Figures 1 and 3 we demonstrate the ability of our method to produce sketches at different levels of abstraction. As the number of strokes decreases, the task of minimizing the loss becomes more challenging, forcing the strokes to capture the essence of the object. For example, in the abstraction process of the flamingo in Figure 1, the transition from 16 to 4 strokes led to the removal of details such as the eyes, feathers, and feet, while maintaining the important visual features such as the general pose, the neck and legs which are iconic characteristics of a flamingo.



(a) Changing the degree of the curves (b) Editing the brush style on SVGs

Fig. 10. Changing sketch style. (a) From left to right are the results produced by our method when using Bézier curves with 4, 3, and 2 control points (cp), respectively. We can see how this affects the style of the output sketch. (b) Using Adobe Illustrator, horse — pencil feather, flamingo — dry brush.

Besides changing the number of strokes, different sketch styles can be achieved by varying the degree of the strokes (Figure 10a) or using a brush style on top of the vector strokes (Figure 10b).

4.2 Comparison with Existing Methods

Sketches with different levels of abstraction. Only a few works have attempted to sketch objects at different levels of abstraction. In Figure 11 we compare with Muhammad et al. [2018] and Berger et al. [2013]. The results by Muhammad et al. demonstrate four levels of abstraction on two simple inputs — a shoe and a chair (in the absence of their code, the results were taken directly from the paper). We produce sketches at four levels of abstraction using 32, 16, 8, and 4 strokes. The sketches by Muhammad et al. are coherent with the geometry of the image; but to achieve higher levels of abstraction, they only remove strokes from the generated sketch without changing the remaining ones. This can result in losing class-level recognizability at higher levels of abstraction (rightmost sketches). Such an approach is sub-optimal, since a better arrangement may be possible for fewer strokes. Our method successfully produces a recognizable rendition of the subject while preserving its geometry, even in the challenging 4-stroke case (rightmost sketch).

In the right bottom part of Figure 11 we compare with the method of Berger et al. [2013]. Their results were provided by the authors and demonstrate two levels of abstraction generated based on the style of a particular artist. We use 64 and 8 strokes, respectively, to achieve two comparable levels of abstraction and place a pencil style on top of the generated sketch to better fit the artist's style. As can be seen, our approach is more geometrically coherent while still allowing abstraction. Their results fit better to a specific style, but can only work with faces and are limited to the dataset gathered.

Photo-Sketch Synthesis. In Figure 12 we present a comparison with the five works outlined in Table 1. The results by Kampelmühler and Pinz [2020] (A), Li et al. [2015] (B) and Li et al. [2019] (C) were generated based on the authors' implementation and best practice. Due to the lack of a publicly available implementation of SketchLattice [Qi et al. 2021] (E), their results are taken directly from the paper. We present the sketches of Song et al. [2018] (D) on shoe images, since their method only works with shoes and chairs.

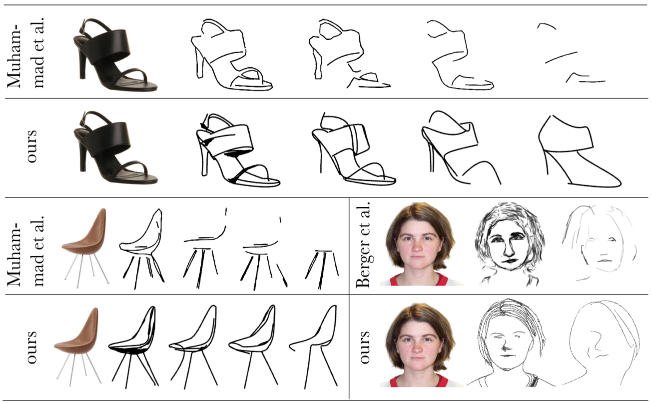


Fig. 11. Levels of Abstraction Comparison — in the top and left part are comparisons to Muhammad et al. [2018]. The leftmost column shows the input image, and the next four columns show different levels of abstraction. For the shoe and chair our results were produced using 32, 16, 8, and 4 curves (from left to right). In the right bottom part is a comparison to Berger et al. [2013], we use 64 and 8 strokes to generate our sketches. ©Face image from [Minear and Park 2004], used with permission.

Each of these methods define a specific objective which influences their dataset selection and final output style. Li et al. [2019] (C) aim for boundary-like drawings, and indeed, geometric coherence is achieved with the input image, capturing salient outlines. The other methods are designed to produce human-like sketches of non-experts, and indeed, the synthesized sketches exhibit a "doodle-like" style. Furthermore, the methods learn highly abstract concepts (such as highlighting the eyes) while maintaining some relation to the geometry of the input object.

Each method accomplishes their respective objective, but we wish to emphasize the particular benefits of our method. First, all of the above methods are sketch-data dependent, meaning they can only be used with the style and level of abstraction observed during training. With our framework, we can handle images of all categories and produce sketches of various levels of abstraction simply by changing the number of strokes. Although every method can be retrained with new datasets, this is neither convenient nor practical, and depends on the availability of such datasets. Second, while each method leans towards a more semantic or a more geometric sketching style, our method can provide both. For example, our method did not produce a perfect alignment of the legs of the horse, as in (C), but it captured the movement of the horse in a minimal way.

In Figure 13 we provide a comparison with CLIPDraw [Frans et al. 2021]. The text input for CLIPDraw is replaced with the target image. This was made possible since CLIP encodes both text and images to the same latent space. To provide a comparable visualization, we constrain the output primitives of CLIPDraw in the same manner as we defined our strokes. As can be seen, although the parts of the subject can be recognizable using CLIPDraw, since there is no geometric grounding to the image, the overall structure is destroyed. Further comparisons of CLIPDraw incorporating text and color can be found in the supplemental file.

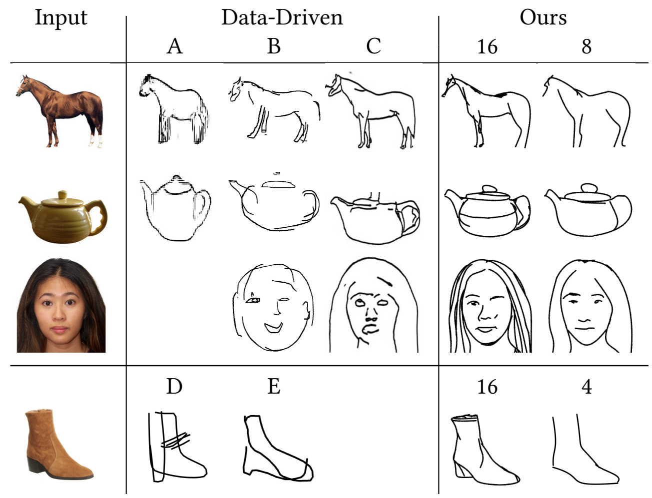


Fig. 12. Comparison to Existing Image-to-Sketch Works — the leftmost column shows the input images. The methods presented are (A) Kampelmühler and Pinz [2020], (B) Li et al. [2015], (C) Li et al. [2019] (D) Song et al. [2018], (E) SketchLattice [Qi et al. 2021]. ©"Chestnut Horse" by James Wood [CC BY-SA 2.0], via (https://bit.ly/3vzgZzc); "Bauer Teapot" by Hollie Glassner, used with permission, via (https://bit.ly/3ktIh3G); Face image from [Minear and Park 2004], used with permission.

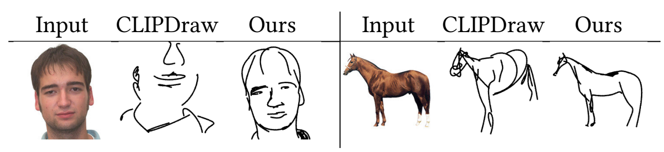


Fig. 13. Comparison to CLIPDraw [Frans et al. 2021]. All sketches were produced using 16 strokes. ©Face from the face database of the Center for Vital Longevity [Minear and Park 2004], used with permission; "Chestnut Horse" by James Wood [CC BY-SA 2.0], via (https://bit.ly/3vzgZzc).

4.3 Quantitative Evaluation

We conduct a perceptual study to assess both the category-level and instance-level recognizability of the sketches generated by our method at different levels of abstraction. Additionally, similar to previous image-to-sketch works [Kampelmühler and Pinz 2020; Muhammad et al. 2018; Song et al. 2018], we also use pretrained classifier networks to evaluate the category-level recognizability of the sketches generated by our method.

Perceptual Study. We choose five popular animal classes (cat, dog, elephant, giraffe, and horse) from the SketchyCOCO dataset [Gao et al. 2020] and randomly sample five images per class. We synthesize sketches at four levels of abstraction for each image, with 4, 8, 16, and 32 strokes. In total, we generate 100 sketches. We compare the recognition rates with the two recent photo-sketch synthesis methods by Kampelmühler and Pinz [2020] and Li et al. [2019]. The sketches were produced with one level of abstraction, suited to the capabilities of the methods. We had 121 participants in total, out of which 60 were assigned to evaluate our sketches at different levels of abstraction, 38 to the sketches by Kampelmühler and Pinz [2020], and 24 to the sketches by Li et al. [2019].

Each participant was presented with randomly selected sketches of a single method. To examine category-level recognition, participants were asked to choose the correct category text description alongside four confound categories and the option 'None'. For the instance-level recognition experiment, the distractors are images from the same object category. Table 2 shows the average recognition rates attained from the perceptual study. Both the category-level and instance-level recognizability are inversely correlated with the level of abstraction. At four strokes, we can see that our sketches are hardly recognizable at the category level (36%), which illustrates a "breaking point" of our method. At eight strokes and above, both instance and class level rates are high. Li et al. [2019] demonstrate full recognition at the instance level; this is understandable given that the sketches are contour-based and not abstract. At 16 and 32 strokes, we achieve comparable rates, and even with the high level of abstraction when using only eight strokes, we achieve 95% instance level recognizability. The sketches by Kampelmühler and Pinz are more abstract, which explains their low accuracy at the instance and class level.

Table 2. Perceptual study results – average recognition rates. (A) Kampelmühler and Pinz [2020], (B) Li et al. [2019].

	A	B	Ours4	Ours8	Ours16	Ours32
Category-Level	65% ±2%	96.9% ±0.7%	36% ±3%	87% ±2%	97.9% ±0.8%	99.3% ±0.5%
Instance-Level	65% ±2%	99.1% ±0.4%	72% ±3%	95% ±1%	96% ±1%	97% ±1%

In Figure 14 we provide the confusion matrices for analyzing the sources of errors made in the category recognition task at the abstraction levels with higher error rates. Specifically, our sketches produced with 4 and 8 strokes, and those of Kampelmühler and Pinz [2020]. The three matrices show that the majority of the classification errors can be attributed to insufficient confidence (selecting 'None'). Overall, the 'dog' class achieved the lowest rates of correct answers. In the four strokes case dogs were mostly confused with elephants, whereas in the case of Kampelmühler and Pinz, the errors beside 'None' resulted from a confusion with the cat class. The 'horse' class achieved the second lowest scores. In the four strokes case, most of the errors were due to insufficient confidence, while in Kampelmühler and Pinz, besides the 'None' answers, horses were mostly miscategorized as elephants.

Sketch classifier. Two different classifiers are used for evaluating the synthesized sketches; a ResNet34 classifier from Kampelmühler and Pinz [2020] trained on the Sketchy-Database [Sangkloy et al. 2016] with 125 categories, and a CLIP ViT-B/32 zero-shot classifier using text prompts defined as "A sketch of a(n) *class-name*". Note that this is not the CLIP model we use for training.

Table 3 compares the sketch-classifier recognition accuracy of our sketches to that of Kampelmühler and Pinz [2020] and Li et al. [2019] based upon 200 randomly selected images of 10 categories from the

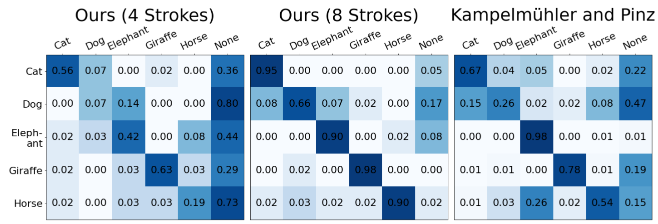


Fig. 14. Confusion matrices of category-level recognition of the perceptual study of our method with four and eight strokes (left and middle matrices) and the method of Kampelmühler and Pinz [2020].

SketchyCOCO dataset [Gao et al. 2020]. The recognition accuracy on human sketches from the SketchyCOCO dataset is also calculated as a baseline. The method by Kampelmühler and Pinz achieves the highest scores for ResNet34 classifier, possibly since they use the same model and dataset during training. Despite the distribution differences, our method still achieves good recognition rates under this classifier. With CLIP classifier, our method achieves very high accuracy levels of 78% with 16 strokes and 91% with 32 strokes. For more details and analysis please refer to the supplementary material.

Table 3. Top-1 and Top-3 sketch recognition accuracy computed with ResNet34 and CLIP ViT-B/32 on 200 sketches from 10 categories. (A) Kampelmühler and Pinz [2020], (B) Li et al. [2019]

Classifier		Human Sketches	A	B	Ours16	Ours32
ResNet34	Top1	98%	**67%**	61%	54%	63%
	Top3	99%	**82%**	78%	75%	77%
CLIP ViT-B/32	Top1	75%	49%	60%	**78%**	91%
	Top3	93%	65%	77%	**93%**	97%

Sketch diversity. We conduct an analysis to quantify the diversity of the sketches produced at different levels of abstraction. The input images for this study are identical to those used in the perceptual study (i.e., five random samples from five different animals classes). In total, we produce 1000 sketches – ten sketches per image, for each level of abstraction, using 10 different initializations. We measure the diversity of each set of sketches $\{\mathcal{S}_i\}_{i=1}^{n}, n = 10$ derived from the same image and at the same abstraction level t as the normalized average variance of the sketches:

$$D_t = \left\| \frac{1}{\|\mu_s\|_1 \cdot n} \sum_{i=1}^{n} (\mathcal{S}_i - \mu_s)^2 \right\|_1 \tag{4}$$

Where $\mu_s = \frac{1}{n} \sum_{i=1}^{n} \mathcal{S}_i$ is the pixel-wise mean sketch. To be able to compare different abstraction levels, we normalize the variance with respect to the magnitude of the average sketch $\|\mu_s\|_1$ at a specific level of abstraction. This normalization is needed because as the level of abstraction increases, the number of strokes decrease so the average magnitude is smaller. Figure 15 shows an example of a few sketches from the highest and lowest levels of abstraction along with the mean sketch at each level. Figure 16a shows the average diversity score for each class and level of abstraction. We can see that the diversity increases with the abstraction level and that similar patterns were observed among different classes.

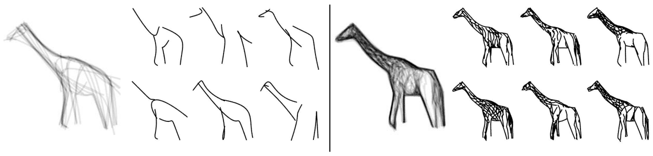


Fig. 15. A visualisation of the appearance of the mean sketch μ_s (the larger giraffes) at two levels of abstraction, next to six samples of distinct sketches from the corresponding set.

(a) Our method's diversity score increases with the level of abstraction.

(b) Diversity of human sketches produced by different artists is larger than the diversity of our method in all levels of abstraction.

Fig. 16. Diversity score as a function of the abstraction level.

We also compare the diversity of the sketches generated by our method to the ones drawn by humans. For this study, we use the dataset collected by Berger et al. [2013], which contains 672 portraits sketches drawn by 7 different artists at 4 levels of abstraction, of 24 faces from the face database of the Center for Vital Longevity [Minear and Park 2004]. The abstractions were created by limiting the amount of time available for the artists to produce the drawings, which is a common exercise in drawing classes, and therefore, a more natural way for people to produce abstractions. To compare visually similar level of abstraction, we use 8 strokes for the highest abstraction level and 64 for the lowest one. We generate 7 sketches using our method for each face and level of abstraction using 7 different seeds, each seed imitates a different artist in this case.

Figure 17 illustrates examples of the sketches produced by the seven artists as well as examples of the sketches generated by our method on a single input face at the lowest and highest abstraction levels. As can be seen, human sketches are more varied in style and semantic choices, which is understandable considering that they were created by different people, while our sketches appear to be less diverse (but still distinct). This visual observation is also expressed through our diversity measure presented in Figure 16b, showing the diversity score of sketches produced by the artists (blue) is larger than our method (orange). However, as can be seen, the graphs follow a similar pattern (diversity decreases with the level of abstraction).

Finally, we show outputs from a large set of diverse input classes in Figure 19. We present 108 sketches of different classes sampled

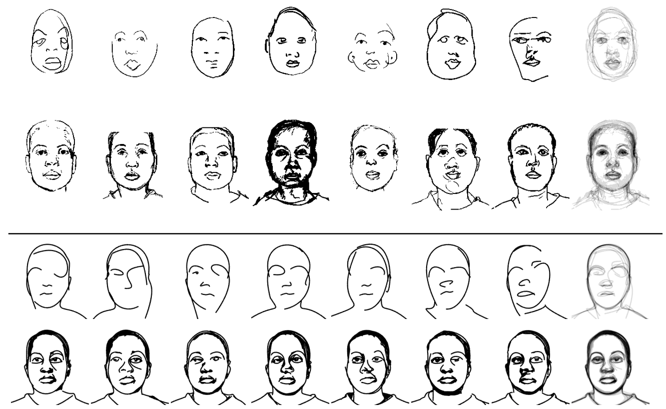


Fig. 17. An example of the sketches used in the diversity study. In each row, we show seven distinct sketches generated from a single input face, along with the average sketch that corresponds to this set in the rightmost column. The first two rows show the sketches drawn by the seven artists at 15 seconds in the first row and 270 seconds in the second row. The third and fourth rows show the sketches produced by our method with 8 and 64 strokes respectively. ©The portraits in the first two rows are from Berger et al. [2013], used with permission.

randomly from the SketchyDatabase dataset [Sangkloy et al. 2016]. We also present sketches of 100 random cats from the SketchyCOCO dataset [Gao et al. 2020] in Figure 20. We use a default number of 16 strokes to produce the sketches presented in both figures.

5 LIMITATIONS AND FUTURE WORK

For images with background, our method's performance is reduced at higher abstraction levels. This limitation can be addressed by using an automatic mask. However, a potential future development would be to include such a remedial term within the loss function. In addition, our sketches are not created sequentially and all strokes are optimized simultaneously, which differs from the conventional way of sketching. Furthermore, the number of strokes must be determined in advance to achieve the desired level of abstraction. Another possible extension could be to make this a learned parameter, as different images may require different numbers of strokes to reach similar levels of abstraction.

Future potential uses of our method could include leveraging its ability to generalize to a variety of new categories in order to build datasets containing corresponding pairs of images and sketches that could be applied to the inverse problem as well.

Finally, as our framework is based on CLIP and its latent encoding, there are limitations of CLIP that carry over to our technique. As noted by the authors of CLIP, one example is CLIP's poor performance on a simple dataset such as MNIST [LeCun and Cortes 2005], which is caused by the lack of similar images in CLIP's training dataset. Figure 18a illustrates that our approach does in fact fail to draw such a simple subject, and the semantic gap may account for this failure. Another example reported in the CLIP paper is the poor performance on several types of fine-grained classification, such as distinguishing types of cars. Figure 18b illustrates this point, as the abstraction focuses on painting the cars, whereas the semantics should focus on the brands of the cars.

Fig. 18. Limitations inherited from CLIP. Figure (a) illustrates that the semantics of the input image (e.g. the digit three) are missing. This results in our method being unable to convey this meaning, which could potentially be expressed with only four strokes. Rather, as we can see, the strokes lie close to the edges of the digit. This conclusion also applies to the 16 stroke case. As shown in figure (b), CLIP's difficulty distinguishing fine-grain attributes is also present in our method. When semantics are considered, the two cars are distinguished by their brands, and this could be shown potentially by using a few strokes in the sketch; however, the optimization focuses on class-level depictions. ©"Parked Cars" by Gosdin [Public Domain], via(https://bit.ly/3LAbKFd).

6 CONCLUSIONS

We presented a method for photo-sketch synthesis, producing sketches with different levels of abstraction, without the need to train on specific sketch datasets. Our method can generalize to various categories and cope with challenging levels of abstraction, while maintaining the semantic visual clues that allow for instance-level and class-level recognition.

ACKNOWLEDGMENTS

This work was supported in part by the Israel Science Foundation (grants no 2492/20 and 3441/21) and by the Deutsch Foundation and the Yandex Initiative in Machine Learning.

Fig. 19. Results of 108 random images from 108 classes from the Sketchy-Database [Sangkloy et al. 2016]. The class name is presented on top of each sketch.

REFERENCES

Pablo Arbeláez, Michael Maire, Charless Fowlkes, and Jitendra Malik. 2011. Contour Detection and Hierarchical Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33, 5 (2011), 898–916. https://doi.org/10.1109/TPAMI.2010.161

Pierre Bénard and Aaron Hertzmann. 2019. Line Drawings from 3D Models. *Found. Trends Comput. Graph. Vis.* 11 (2019), 1–159.

Itamar Berger, Ariel Shamir, Moshe Mahler, Elizabeth Carter, and Jessica Hodgins. 2013. Style and Abstraction in Portrait Sketching. *ACM Trans. Graph.* 32, 4, Article 55 (jul 2013), 12 pages. https://doi.org/10.1145/2461912.2461964

Ayan Kumar Bhunia, Ayan Das, Umar Riaz Muhammad, Yongxin Yang, Timothy M. Hospedales, Tao Xiang, Yulia Gryaditskaya, and Yi-Zhe Song. 2020. Pixelor: a competitive sketching AI agent. so you think you can sketch? *ACM Trans. Graph.* 39 (2020), 166:1–166:15.

John Canny. 1986. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence* 6 (1986), 679–698.

Rebecca Chamberlain and Johan Wagemans. 2016. The genesis of errors in drawing. *Neuroscience & Biobehavioral Reviews* 65 (2016), 195–207.

Hila Chefer, Shir Gur, and Lior Wolf. 2021. Generic Attention-model Explainability for Interpreting Bi-Modal and Encoder-Decoder Transformers. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), 387–396.

Hong Chen, Ying-Qing Xu, Harry Shum, Song-Chun Zhu, and Nanning Zheng. 2001. Example-based facial sketch generation with non-parametric sampling. *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001* 2 (2001), 433–438 vol.2.

Yajing Chen, Shikui Tu, Yuqi Yi, and Lei Xu. 2017. Sketch-pix2seq: a Model to Generate Sketches of Multiple Categories. *ArXiv* abs/1709.04121 (2017).

Judith E. Fan, Daniel L. K. Yamins, and Nicholas B. Turk-Browne. 2018. Common Object Representations for Visual Production and Recognition. *Cognitive science* 42 8 (2018), 2670–2698.

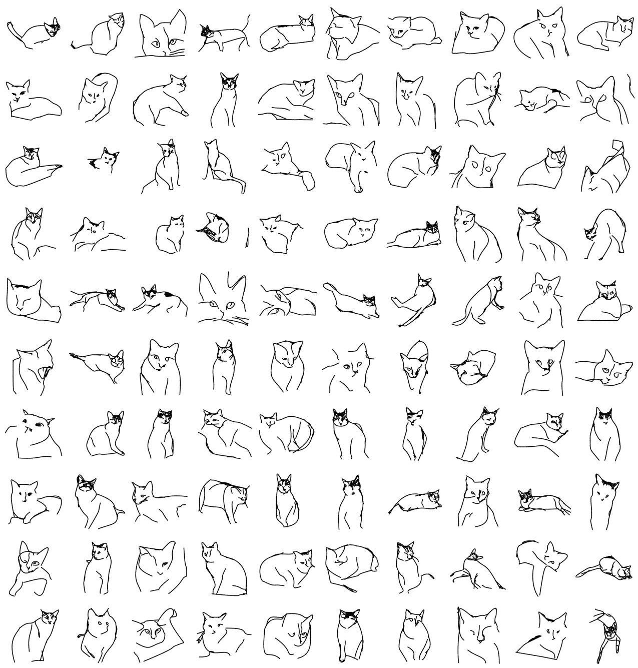


Fig. 20. Sketching "in the wild": results of 100 random images of cats from SketchyCOCO [Gao et al. 2020].

Judith W. Fan, Robert D. Hawkins, Mike Wu, and Noah D. Goodman. 2019. Pragmatic Inference and Visual Abstraction Enable Contextual Flexibility During Visual Communication. *Computational Brain & Behavior* 3 (2019), 86–101.

Kevin Frans, Lisa B. Soros, and Olaf Witkowski. 2021. CLIPDraw: Exploring Text-to-Drawing Synthesis through Language-Image Encoders. *CoRR* abs/2106.14843 (2021). arXiv:2106.14843 https://arxiv.org/abs/2106.14843

Yaroslav Ganin, Tejas D. Kulkarni, Igor Babuschkin, S. M. Ali Eslami, and Oriol Vinyals. 2018. Synthesizing Programs for Images using Reinforced Adversarial Learning. *ArXiv* abs/1804.01118 (2018).

Chengying Gao, Qi Liu, Qi Xu, Limin Wang, Jianzhuang Liu, and Changqing Zou. 2020. SketchyCOCO: Image Generation From Freehand Scene Sketches. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 5174–5183.

Gabriel Goh, Nick Cammarata †, Chelsea Voss †, Shan Carter, Michael Petrov, Ludwig Schubert, Alec Radford, and Chris Olah. 2021. Multimodal Neurons in Artificial Neural Networks. *Distill* (2021). https://doi.org/10.23915/distill.00030 https://distill.pub/2021/multimodal-neurons.

Yulia Gryaditskaya, Mark Sypesteyn, Jan Willem Hoftijzer, Sylvia C. Pont, Frédo Durand, and Adrien Bousseau. 2019. OpenSketch: a richly-annotated dataset of product design sketches. *ACM Trans. Graph.* 38 (2019), 232:1–232:16.

David Ha and Douglas Eck. 2017. A Neural Representation of Sketch Drawings. *CoRR* abs/1704.03477 (2017). arXiv:1704.03477 http://arxiv.org/abs/1704.03477

A. Hertzmann. 2003. A survey of stroke-based rendering. *IEEE Computer Graphics and Applications* 23, 4 (2003), 70–81. https://doi.org/10.1109/MCG.2003.1210867

Aaron Hertzmann. 2020. Why Do Line Drawings Work? A Realism Hypothesis. *Perception* 49 (2020), 439 – 451.

Aaron Hertzmann. 2021. The Role of Edges in Line Drawing Perception. *Perception* 50 (2021), 266 – 275.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-Image Translation with Conditional Adversarial Networks. In *CVPR*.

Moritz Kampelmühler and Axel Pinz. 2020. Synthesizing human-like sketches from natural images using a conditional convolutional decoder. *CoRR* abs/2003.07101 (2020). arXiv:2003.07101 https://arxiv.org/abs/2003.07101

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2015).

Alexander Kolesnikov, Alexey Dosovitskiy, Dirk Weissenborn, Georg Heigold, Jakob Uszkoreit, Lucas Beyer, Matthias Minderer, Mostafa Dehghani, Neil Houlsby, Sylvain Gelly, Thomas Unterthiner, and Xiaohua Zhai. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ICLR '21*.

Yann LeCun and Corinna Cortes. 2005. The mnist database of handwritten digits.

Mengtian Li, Zhe Lin, Radomir Mech, Ersin Yumer, and Deva Ramanan. 2019. Photo-Sketching: Inferring Contour Drawings from Images. arXiv:1901.00542 [cs.CV]

Tzu-Mao Li, Michal Lukáč, Gharbi Michaël, and Jonathan Ragan-Kelley. 2020. Differentiable Vector Graphics Rasterization for Editing and Learning. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* 39, 6 (2020), 193:1–193:15.

Yi Li, Yi-Zhe Song, Timothy M. Hospedales, and Shaogang Gong. 2015. Free-hand Sketch Synthesis with Deformable Stroke Models. *CoRR* abs/1510.02644 (2015). arXiv:1510.02644 http://arxiv.org/abs/1510.02644

Hangyu Lin, Yanwei Fu, Yu-Gang Jiang, and X. Xue. 2020. Sketch-BERT: Learning Sketch Bidirectional Encoder Representation From Transformers by Self-Supervised Learning of Sketch Gestalt. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020), 6757–6766.

John F. J. Mellor, Eunbyung Park, Yaroslav Ganin, Igor Babuschkin, Tejas Kulkarni, Dan Rosenbaum, Andy Ballard, Theophane Weber, Oriol Vinyals, and S. M. Ali Eslami. 2019. Unsupervised Doodling and Painting with Improved SPIRAL. *CoRR* abs/1910.01007 (2019). arXiv:1910.01007 http://arxiv.org/abs/1910.01007

Daniela Mihai and Jonathon S. Hare. 2021a. Differentiable Drawing and Sketching. *ArXiv* abs/2103.16194 (2021).

Daniela Mihai and Jonathon S. Hare. 2021b. Learning to Draw: Emergent Communication through Sketching. *ArXiv* abs/2106.02067 (2021).

Meredith Minear and Denise C. Park. 2004. A lifespan database of adult facial stimuli. *Behavior Research Methods, Instruments, & Computers* 36 (2004), 630–633.

Umar Riaz Muhammad, Yongxin Yang, Yi-Zhe Song, Tao Xiang, and Timothy M. Hospedales. 2018. Learning Deep Sketch Abstraction. *CoRR* abs/1804.04804 (2018). arXiv:1804.04804 http://arxiv.org/abs/1804.04804

Yonggang Qi, Guoyao Su, Pinaki Nath Chowdhury, Mingkang Li, and Yi-Zhe Song. 2021. SketchLattice: Latticed Representation for Sketch Manipulation. *ArXiv* abs/2108.11636 (2021).

Xuebin Qin, Zichen Zhang, Chenyang Huang, Masood Dehghan, Osmar Zaiane, and Martin Jagersand. 2020. U2-Net: Going Deeper with Nested U-Structure for Salient Object Detection. *Pattern Recognition* 106, 107404.

Shuwen Qiu, Sirui Xie, Lifeng Fan, Tao Gao, Song-Chun Zhu, and Yixin Zhu. 2021. Emergent Graphical Conventions in a Visual Communication Game.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. *CoRR* abs/2103.00020 (2021). arXiv:2103.00020 https://arxiv.org/abs/2103.00020

Leo Sampaio Ferraz Ribeiro, Tu Bui, John P. Collomosse, and Moacir Antonelli Ponti. 2020. Sketchformer: Transformer-Based Representation for Sketched Structure. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020), 14141–14150.

Patsorn Sangkloy, Nathan Burnell, Cusuh Ham, and James Hays. 2016. The Sketchy Database: Learning to Retrieve Badly Drawn Bunnies. *ACM Trans. Graph.* 35, 4, Article 119 (jul 2016), 12 pages. https://doi.org/10.1145/2897824.2925954

Jifei Song, Kaiyue Pang, Yi-Zhe Song, Tao Xiang, and Timothy Hospedales. 2018. Learning to Sketch with Shortcut Cycle Consistency. arXiv:1805.00247 [cs.CV]

Yingtao Tian and David Ha. 2021. Modern Evolution Strategies for Creativity: Fitting Concrete Images and Abstract Concepts. *CoRR* abs/2109.08857 (2021). arXiv:2109.08857 https://arxiv.org/abs/2109.08857

Barbara Tversky. 2002. What do Sketches Say about Thinking.

V Varshaneya, Sangeetha Balasubramanian, and Vineeth N. Balasubramanian. 2021. Teaching GANs to sketch in vector format. *Proceedings of the Twelfth Indian Conference on Computer Vision, Graphics and Image Processing* (2021).

Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. 2017. High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. *CoRR* abs/1711.11585 (2017). arXiv:1711.11585 http://arxiv.org/abs/1711.11585

Holger Winnemöller, Jan Eric Kyprianidis, and Sven C. Olsen. 2012. XDoG: An eXtended difference-of-Gaussians compendium including advanced image stylization. *Comput. Graph.* 36 (2012), 740–753.

Peng Xu, Timothy M. Hospedales, Qiyue Yin, Yi-Zhe Song, Tao Xiang, and Liang Wang. 2020. Deep Learning for Free-Hand Sketch: A Survey and A Toolbox. arXiv:2001.02600 [cs.CV]

Justin Yang and Judith E. Fan. 2021. Visual communication of object concepts at different levels of abstraction. *ArXiv* abs/2106.02775 (2021).

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. 2018. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), 586–595.

N. Zheng, Yf Jiang, and Ding jiang Huang. 2019. StrokeNet: A Neural Painting Environment. In *ICLR*.

Tao Zhou, Chen Fang, Zhaowen Wang, Jimei Yang, Byungmoon Kim, Zhili Chen, Jonathan Brandt, and Demetri Terzopoulos. 2018. Learning to Sketch with Deep Q Networks and Demonstrated Strokes. *ArXiv* abs/1810.05977 (2018).