CoGS: Controllable Generation and Search from Sketch and Style

Cusuh Ham[1⋆], Gemma Canet Tarrés[2⋆], Tu Bui[2],
James Hays[1], Zhe Lin[3], and John Collomosse[2,3]

[1] Georgia Institute of Technology {cusuh,hays}@gatech.edu
[2] University of Surrey {g.canettarres,t.v.bui,j.collomosse}@surrey.ac.uk
[3] Adobe Inc. zlin@adobe.com

1 Pseudosketches Dataset

To build the Pseudosketches dataset, we start with the 12.5K images from Sketchy DB [6], and sample additional images for each category (or similar categories based on WordNet distances to the original category) from ImageNet [1]. After running the automated pseudosketch-extraction pipeline on all gathered images, we display the pseudosketch-image pair and its class label, and ask Amazon Mechanical Turk (AMT) workers to rate how well the pseudosketch represents its corresponding image on a scale of 1 (worst) to 5 (best). We visualize paired examples for each AMT score in Fig. 1.

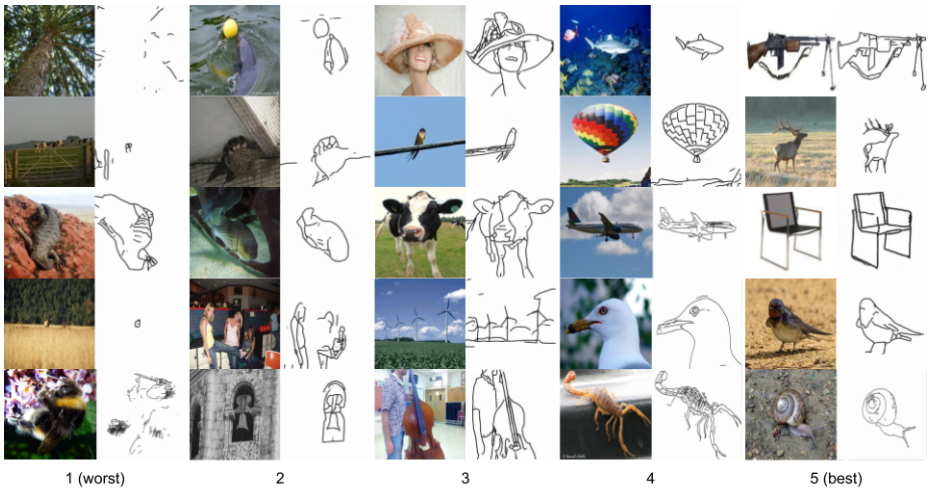

Fig. 1. Examples of image-pseudosketch pairs at each AMT score.

After removing pseudosketches with a score of 1 or 2, we are left with 113,700 pseudosketches across the original 125 categories of Sketchy DB (see Fig. 2 for

⋆ Equal contribution.

the distribution of pseudosketches per category). We note that the dataset is imbalanced due to some categories having more images than others and/or some categories producing many more poorly-scored pseudosketches than others.

2 Image synthesis using transformers

2.1 Comparison to CNN-based networks

We explore the use of various CNN-based networks instead of transformers for synthesis. The most successful alternative used a decoder adapted from StyleGAN2 [3] to learn the composite codebook representation given a set of conditional inputs, but we found the network to be susceptible to overfitting. Even when additional information was given to the CNN, such as pseudosketch of the style image, transformers proved to better interpret and learn the distribution of codebooks, leading to better control and more high frequency details (see Fig. 3).

2.2 Qualitative ablation study

In Fig. 4 we present qualitative results of the ablation study performed in the main paper to demonstrate the importance of the class label and auxiliary style loss for enabling control over the output image. Specifically, adding the class label makes the generated object more faithful to its semantic structure and texture, while the additional loss allows for a better style control over the output.

2.3 Data partitioning

We partition the categories of the Pseudosketches dataset by computing class-wise FID [2] scores with the validation images in each of the 125 categories (see Table 1). Fig. 5, Fig. 6, and Fig. 7 visualize examples from each of the "simple", "medium", and "complex" partitions, respectively. We observe common characteristics of the categories belonging to the more difficult partitions, such as less uniform textures and multiple non-related objects or humans present. However, we are still able to demonstrate the ability of CoGS to synthesize a diverse set of categories, using semantic understanding to generate appropriate textures and even applying realistic lighting, shadows, and reflections.

2.4 Style and structure controllability

We propose CoGS as an image synthesis method that provides decoupled control over the structure and style of the generated image through sketch and style image inputs, respectively. In Fig. 8 and 9 we vary the two axes of control to show that our method captures both the structure of the input sketch and the style of the input style image.

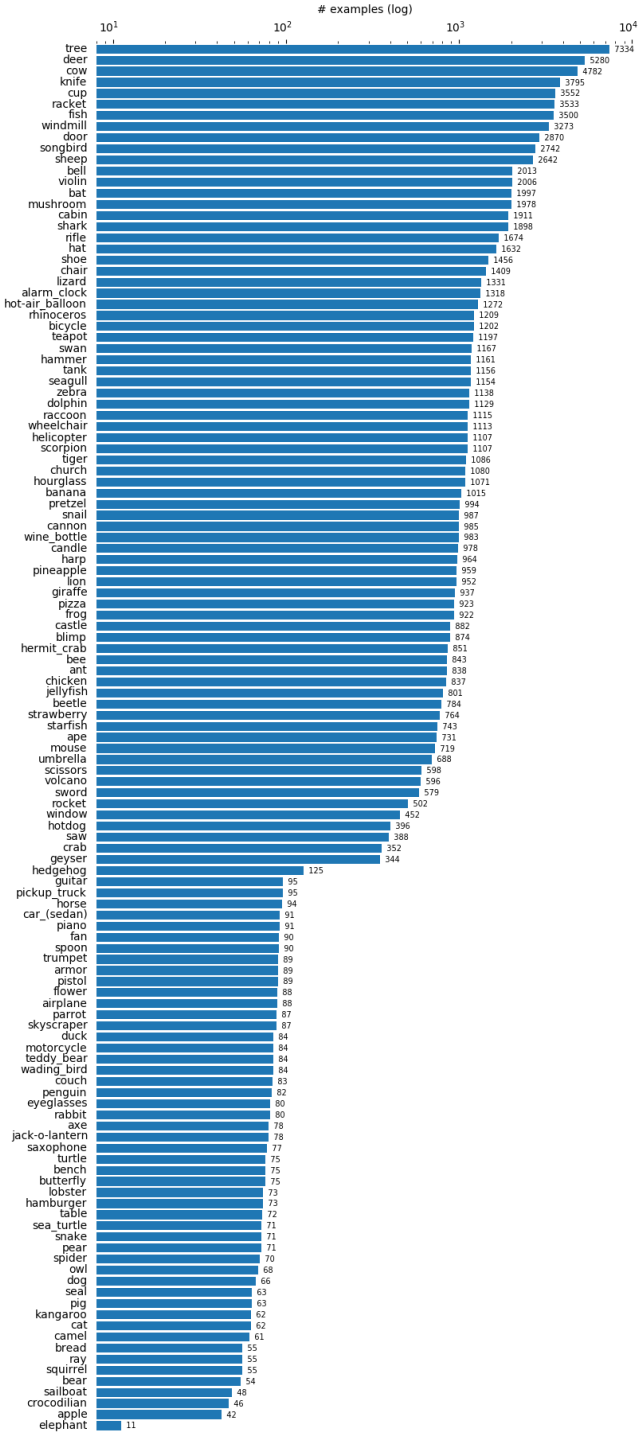

Fig. 2. Number of images per category in the Pseudosketches dataset.

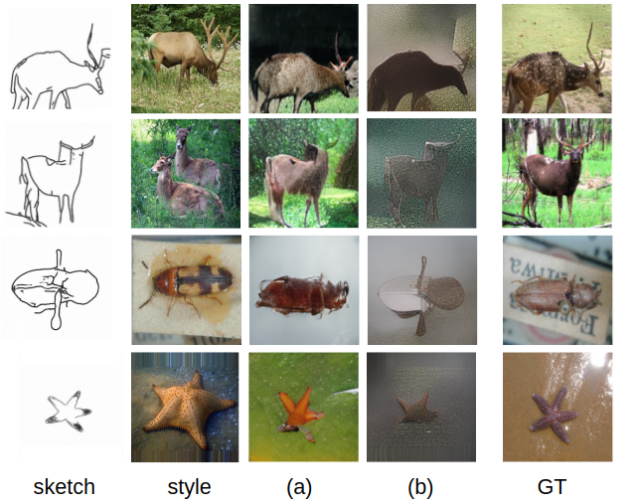

Fig. 3. For a given labeled sketch and style image, (a) is the corresponding synthesized image using transformers, and (b) represents the synthesized image using a StyleGAN2-based network. The rightmost column is the "ground truth" image.

Fig. 4. Visualization of synthesized images from the different ablated methods: (a) *inputs:* sketch, style image, *losses:* codebook, (b) *inputs:* sketch, style image, *losses:* codebook, style, (c) *inputs:* sketch, style image, class label, *losses:* codebook, (d) *inputs:* sketch, style image, class label, *losses:* codebook, style.

Partition	Categories
Simple	deer, tree, cow, zebra, songbird, windmill, door, shark, rhinoceros, cabin, cup, knife, sheep, dolphin, chair, seagull, swan, castle, pizza fish, volcano, mushroom, beetle, lion, hot-air balloon, bat, ape, tiger, helicopter, teapot, wheelchair, geyser, scissors, starfish, tank, jellyfish, rocket, raccoon, blimp, racket, wading bird
Medium	snail, church, giraffe, sword, jack-o-lantern, lizard, sailboat, car (sedan), bicycle, rifle, ant, saw, bee, window, frog, alarm clock, shoe, bell, scorpion, hermit crab, ray, hat, wine bottle, hourglass, spoon, motorcycle, penguin, sea turtle, candle, hammer, chicken, snake, kangaroo, strawberry, duck, violin, airplane, banana, cannon, crab, mouse, horse
Complex	hot dog, pineapple, owl, butterfly, pretzel, rabbit, hedgehog, pear, pistol, umbrella, hamburger, bear, bench, camel, parrot, fan, pig, pickup truck, table, apple, seal, elephant, armor, spider, flower, squirrel, piano, bread, turtle, eyeglasses, guitar, crocodilian, axe, skyscraper, couch, cat, teddy bear, trumpet, dog, saxophone, harp, lobster

Table 1. Categories belonging to each of the 3 partitions.

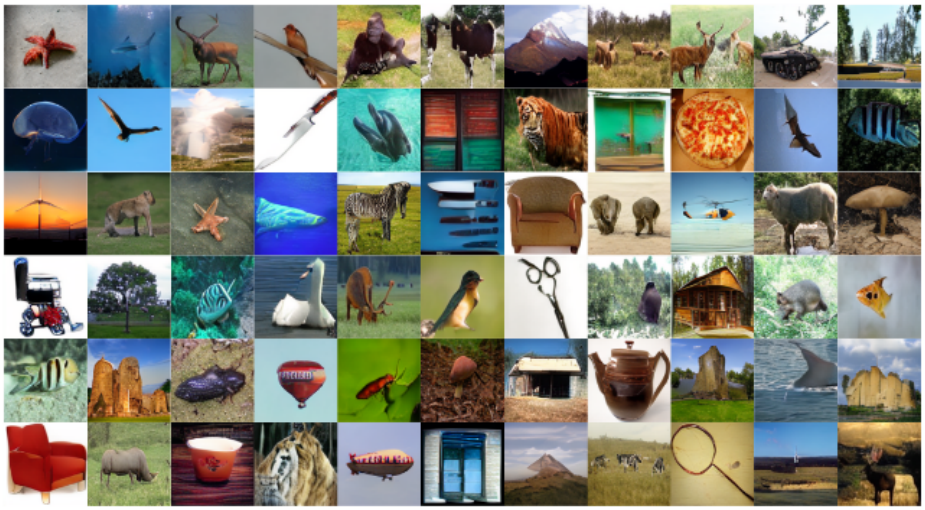

Fig. 5. Examples of synthesizes images for all classes in the "simple" partition.

Fig. 6. Examples of synthesized images for all classes in the "medium" partition.

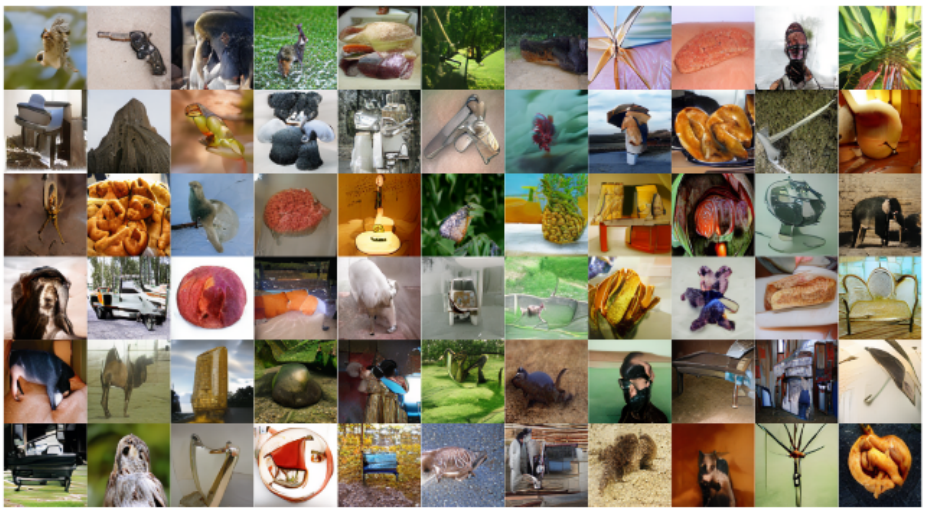

Fig. 7. Examples of synthesized images for all classes in the "complex" partition.

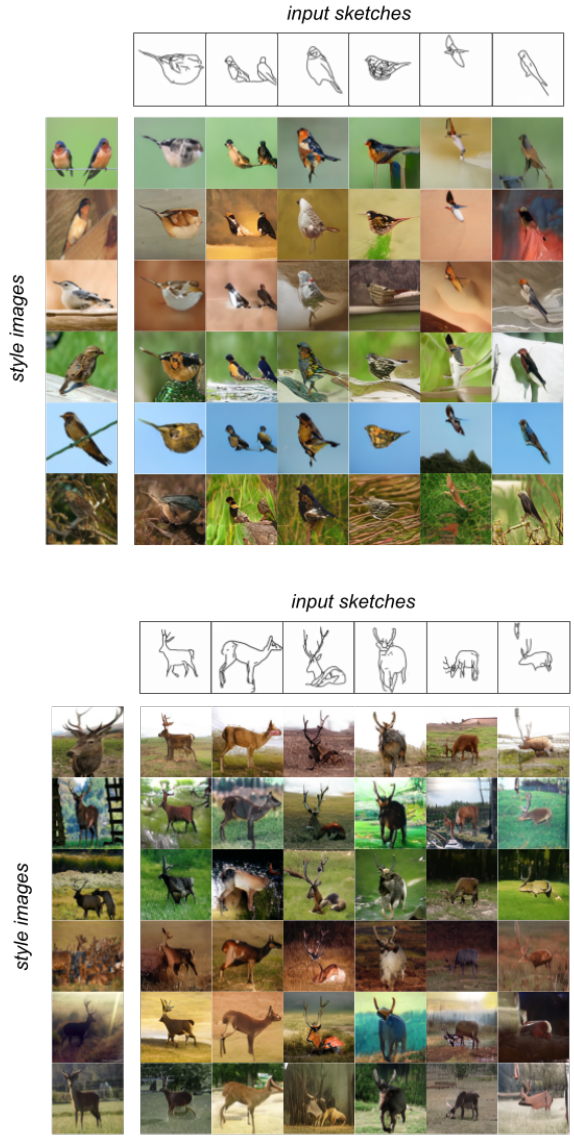

Fig. 8. Images synthesized by CoGS using the respective row and column input combination for the *songbird* and *deer* classes.

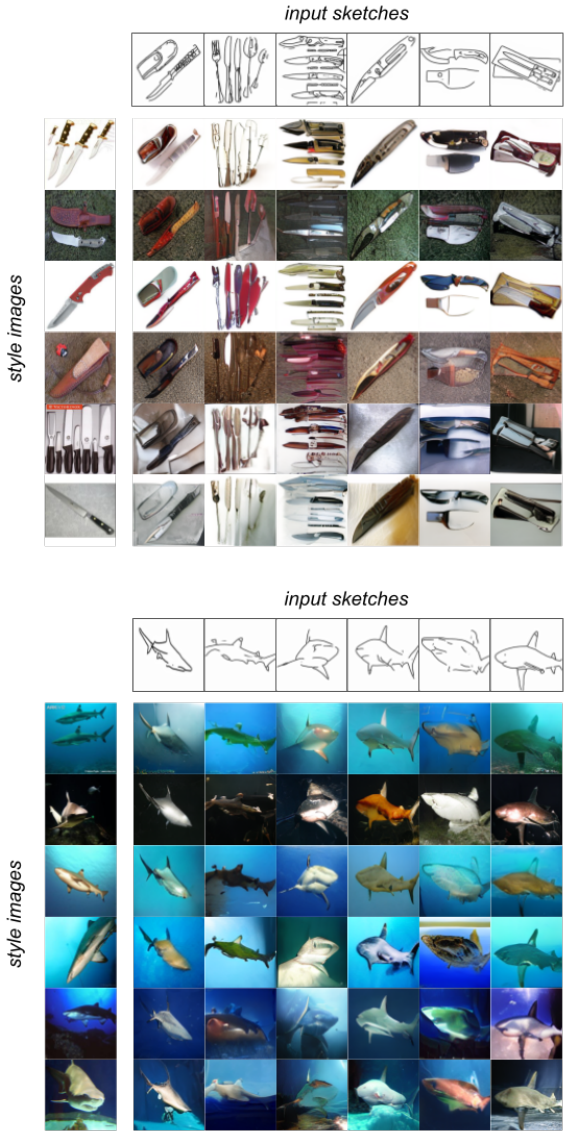

Fig. 9. Images synthesized by CoGS using the respective row and column input combination for the *knife* and *shark* classes.

Partition	$k = 1$	$k = 5$	$k = 10$	$k = 15$	$k = 20$
Simple	0.856	0.735	0.744	0.769	0.770
Medium	0.931	0.910	0.898	0.901	0.892
Complex	0.935	0.937	0.702	0.941	0.848

Table 2. Precision@k with $k = \{1, 5, 10, 15, 20\}$ for the retrieved results across various classes within each partition.

2.5 Generalization to hand-drawn sketches

We demonstrate the ability of CoGS, trained only on pseudosketches, to generalize to the higher quality human-drawn sketches from Sketchy DB [6] (see Fig. 10), which are more abstract and less faithful to the contours of their corresponding image prompt. Because Sketchy DB was collected from users of varying skill levels, we are able to see the impact of the artistry level on the outputs, highlighting that there still exists a gap between the synthesis quality of the two types of sketch inputs.

3 Image refinement using VAEs

3.1 Latent space visualization

CoGS offers an optional step to refine the generated output of the transformer through the use of variational autoencoders (VAE) [4]. We train a VAE for each class and visualize the structure of a few latent spaces using t-Distributed Stochastic Neighbor Embedding (t-SNE) [5] on the Pseudosketches validation set in Fig. 11, Fig. 12, and Fig. 13. We observe that the contrastive training paradigm yields latent spaces in which images with similar structures are closer together, and images with dissimilar structures are further apart.

3.2 Photorealistic image retrieval

We study the difference in retrieval performance across the three partitions of the Pseudosketches dataset by sampling images generated by CoGS and retrieving the top 20 photorealistic images within the same class from the Pseudosketches dataset. Through AMT evaluations we evaluate the precision@k for the top 20 retrieved images (see Table 2), and show coherence of the latent space and relevancy of the retrieved results. We visualize retrieval results for queries belonging to each of the partitions in Fig. 14, Fig. 15, and Fig. 16.

3.3 Latent space interpolation

In Fig. 17 we visualize images synthesized by interpolating between query images and their top retrieval results from the Pseudosketches dataset.

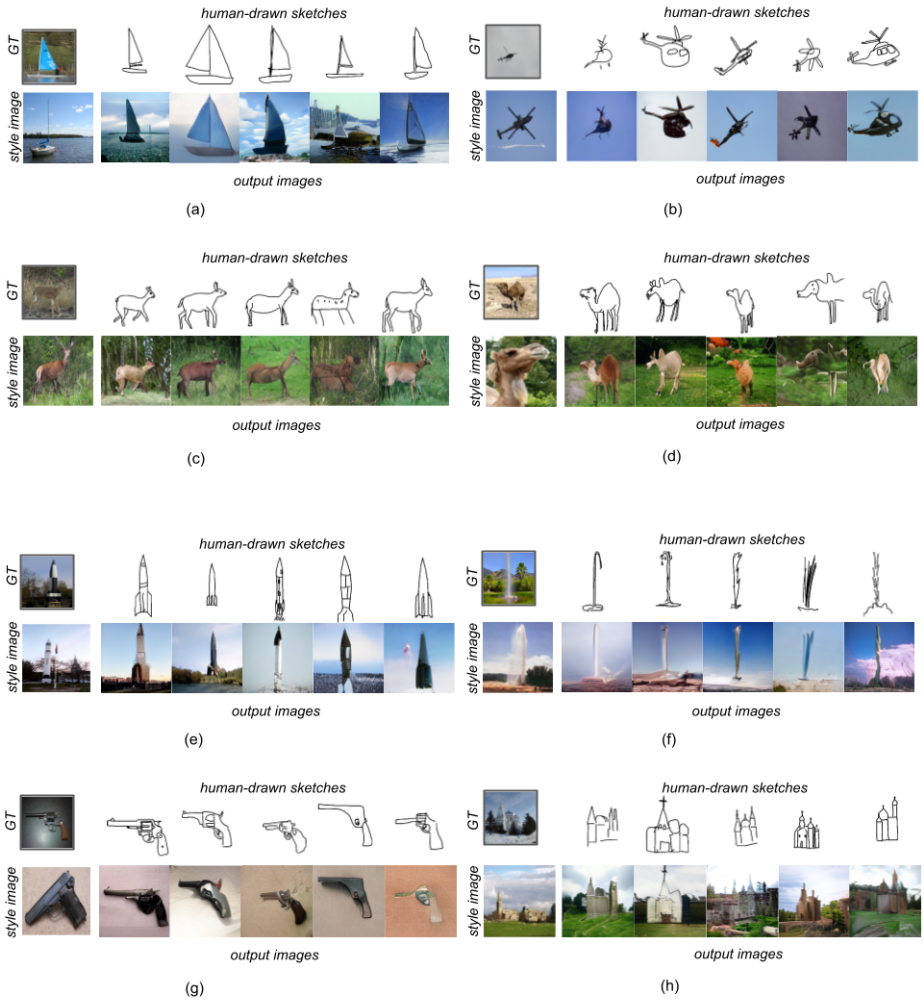

Fig. 10. For each subfigure (a-h), we generate the output using 5 hand-drawn sketches from Sketchy DB [6] corresponding to a ground truth image (framed in grey) with a given style image.

Fig. 11. t-SNE visualization of the *songbird* latent space.

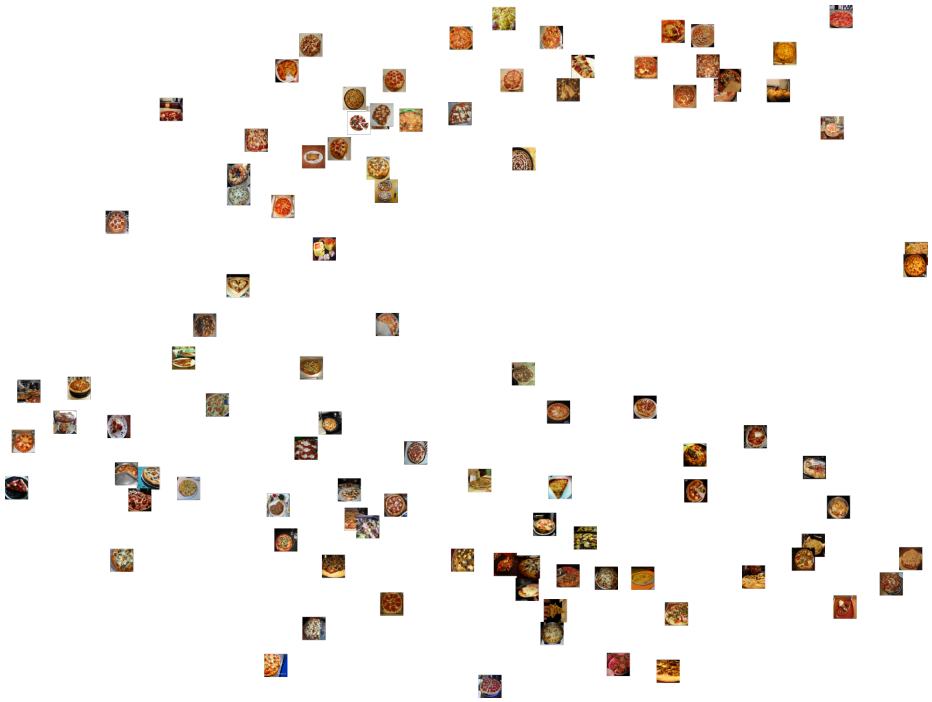

Fig. 12. t-SNE visualization of the *pizza* latent space.

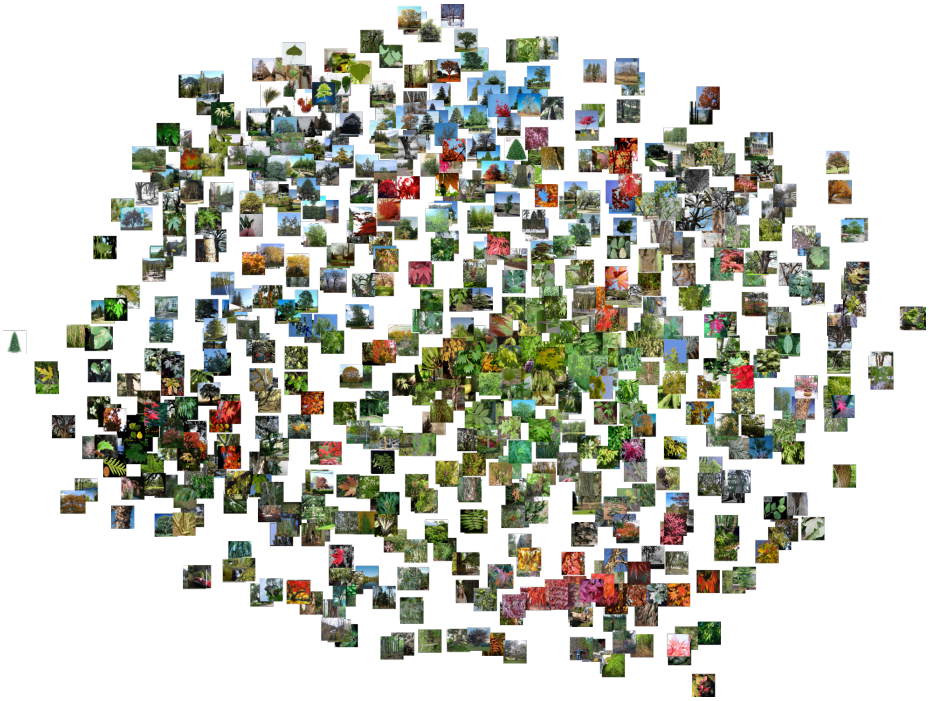

Fig. 13. t-SNE visualization of the *tree* latent space.

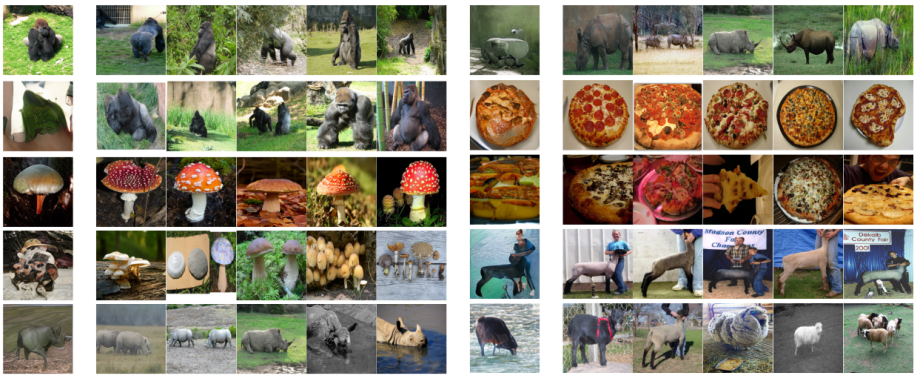

Fig. 14. Top 5 retrieval results for query images from the "simple" partition.

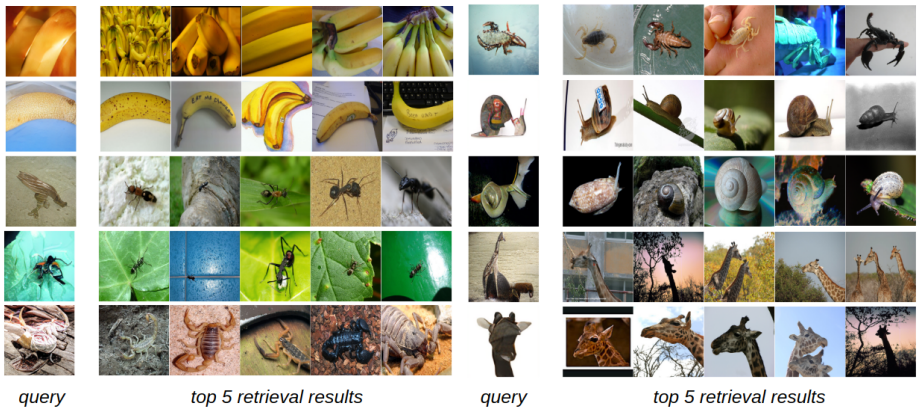

Fig. 15. Top 5 retrieval results for query images from the "medium" partition.

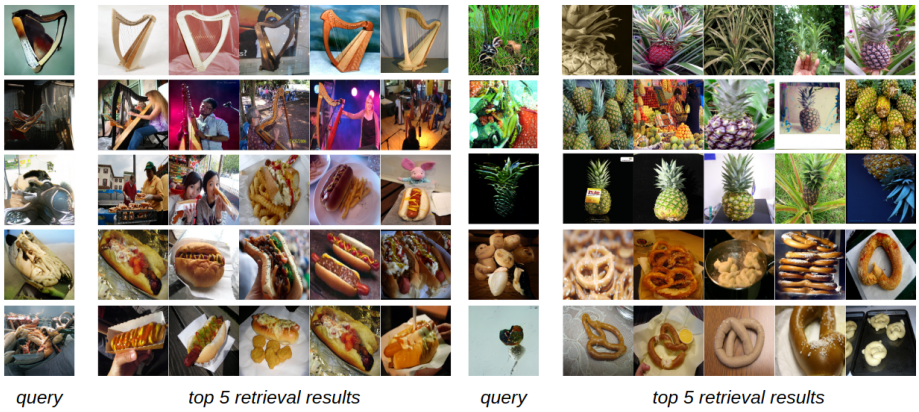

Fig. 16. Top 5 retrieval results for query images from the "complex" partition.

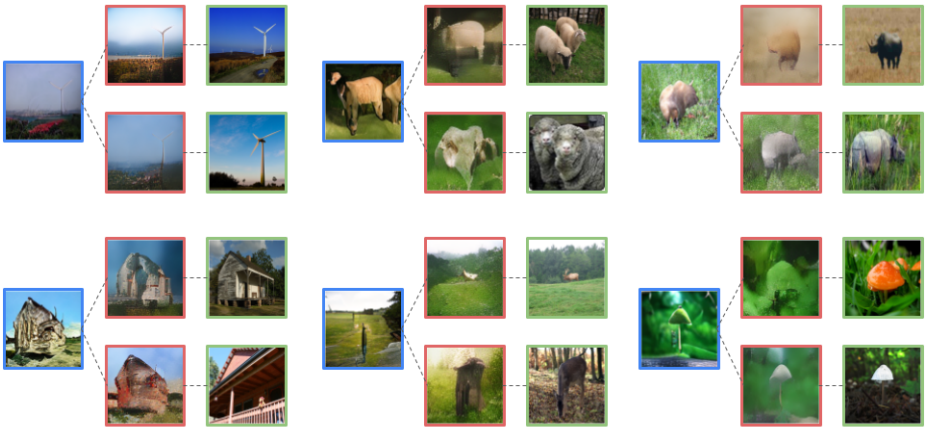

Fig. 17. For a given query image (blue box) we retrieve two of its nearest photorealistic neighbors (green) and synthesize images (red) by interpolating between the query and each retrieval result.

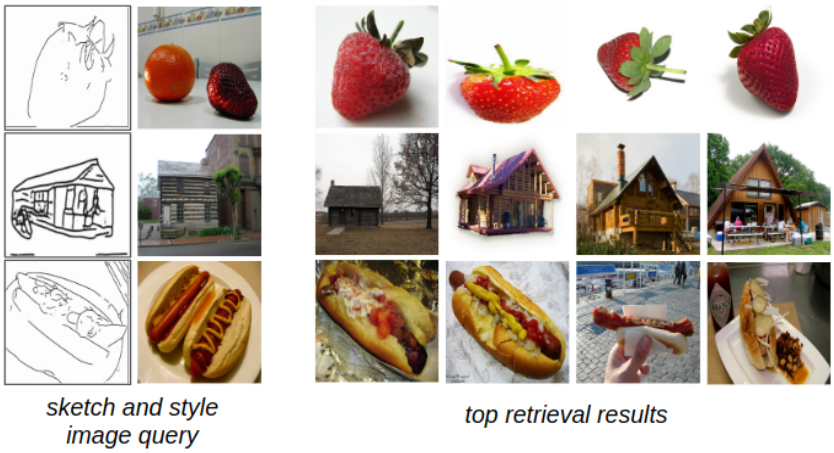

Fig. 18. Top 4 images retrieved by using CoGS as a SBIR method with (sketch, style) pairs as inputs.

3.4 Sketch-based image retrieval

CoGS accepts a (sketch, style, label) input for synthesis via a codebook representation, which may be used as a query for retrieval. The output images may be interpolated or used directly to refine the synthesized image. While sketch-based image retrieval (SBIR) is not the intent of this work, it is possible if a style image is provided (Fig. 18).

4 Efficiency

CoGS only requires a feed-forward inference pass for initial synthesis, and again for subsequent retrieval/interpolations for refinement (3-5s on a single Titan X).

5 AMT evaluations

We provide additional details about the three Amazon Mechanical Turk (AMT) tasks (visualized in Fig. 19) used to crowd-source human evaluations of CoGS and the baseline methods:

1. Baseline comparison (Section 4.3)
   (a) *Preference based on ground-truth.* "Look at the above 6 AI generated images. Choose which of the AI images (A-F) most closely matches the REAL image below?" (select: A-F)
   (b) *Preference based on style.* "Look at the above 6 AI generated images. Choose which of the AI images (A-F) most closely matches the style (colors and patterns/textures) of the real image below?" (select A-F)
   (c) *Preference based on structure.* "Look at the above 6 AI generated images. Choose which of the AI images (A-F) most closely matches the shape of the sketch below?" (select A-F)
   (d) *Realism.* "How realistic does the below AI generated image look?" (select 1 (very bad) - 5 (very good))
   (e) *Fidelity.* "The above photo is a real image. How close to it is the fake AI-made image below?" (select 1 (very bad) - 5 (very good))
2. Controllability experiments (Section 4.4)
   (a) *Structure.* "How close is the shape of the object in the image to the sketched shape?" (select 1 (very bad) - 5 (very good))
   (b) *Style.* "How close is the color/texture of the object above to the image below?" (select 1 (very bad) - 5 (very good))
3. Retrieval experiments (Section 4.7)
   (a) *Retrieval relevance.* "Examine this image pair. Do both the structure (the shape) and the appearance (colour and texture of the image) of the two images match?" (select yes/no)

References

1. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
2. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems **30** (2017)
3. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of StyleGAN. In: Proc. CVPR (2020)
4. Kingma, D.P., Welling, M.: Auto-Encoding Variational Bayes. ArXiv e-prints (Dec 2013)
5. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(11) (2008)
6. Sangkloy, P., Burnell, N., Ham, C., Hays, J.: The sketchy database: Learning to retrieve badly drawn bunnies. ACM Trans. Graph. **35**(4) (jul 2016). https://doi.org/10.1145/2897824.2925954, `https://doi.org/10.1145/2897824.2925954`

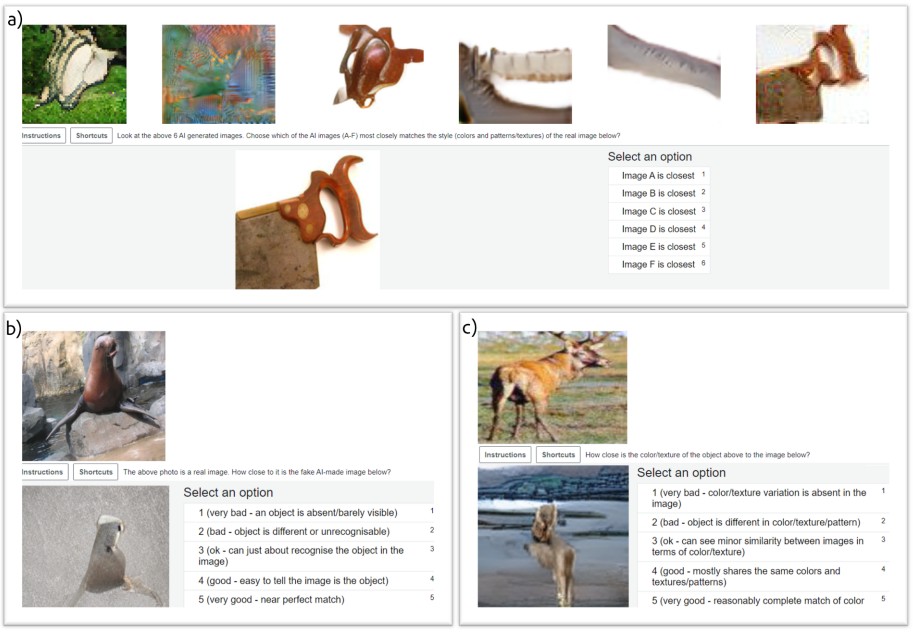

Fig. 19. Examples of AMT evaluation task prompts. (a) Prompt for selecting the preferred reconstructed image based on style. (b) Prompt for evaluating the fidelity of each generated image. (c) Prompt for evaluating style controllability on the images generated by CoGS.