Learning to Place Objects into Scenes by Hallucinating Scenes around Objects

Lu Yuan, James Hong, Vishnu Sarukkai, and Kayvon Fatahalian
Department of Computer Science, Stanford University
{ luyuan , james.hong , sarukkai , kayvonf } @cs.stanford.edu

Abstract

The ability to modify images to add new objects into a scene stands to be a powerful image editing control. However, object insertion is not robustly supported by existing diffusion-based image editing methods. The central challenge is predicting where an object should go in a scene, given only an image of the scene. To address this challenge, we propose DreamPlace, a two-step method that inserts objects of a given class into images by 1) predicting where the object is likely to go in the image and 2) inpainting the object at this location. *We train our object placement model solely using synthetic data*, leveraging diffusion-based image outpainting to hallucinate novel images of scenes surrounding a given object. DreamPlace, using its learned placement model, can produce qualitatively more realistic object insertion edits than comparable diffusion-based baselines. Moreover, for a limited set of object categories where benchmark annotations exist, our learned object placement model, despite being trained entirely on generated data, makes up to 35% more accurate object placements than the state-of-the-art supervised method trained on a large, manually annotated dataset (>80k annotated samples).

1 Introduction

One of the central challenges of generative AI-based image synthesis involves developing mechanisms to more precisely control the output of generative models. In this work, we seek to modify an existing image by adding new objects at *plausible* locations in the scene. This form of control is useful both in image editing workflows and as a mechanism for generating synthetic datasets.

Existing methods for controlling diffusion-based image generation are limited in their ability to perform object insertion edits due to a lack of automated control over object placement locations. (see Figure 1). State-of-the-art iterative editing-based paradigms (3) also often ignore commands to add objects to an existing scene, particularly when the objects are small in the image. Other editing approaches (29; 24) perform better at inserting objects, but they assume explicit guidance on where to inpaint the object and its orientation and shape (e.g., a mask).

We introduce DreamPlace, an algorithm for modifying an image to contain a new object of a given class (e.g., add a laptop to an image of a scene containing a desk, or add a cup to an image depicting a coffee table). DreamPlace divides the problem into two phases: 1) learning a model that predicts where objects of the given class are likely to belong in an image (where the object should be “placed" in the scene); and 2) leveraging diffusion-based inpainting to generate new image pixels that depict the object at the predicted location while ensuring visual harmony with the surrounding scene context.

In this setup, the challenge is: how, given only an image of a scene, do we predict where an object can go? Our insight is that although existing text-guided diffusion models do not directly support object additional edits, they — having been trained in Internet-scale data (25) — have learned a strong prior on plausible object placements within the context of larger scenes. Given images of an object class

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

Figure 1: Prior diffusion-based methods do not provide controls for inserting objects into scenes, given only an input image with no location and scale information. From left to right: input scene, Stable Diffusion in-painting (21) with large masks, Instruct Pix2Pix (3). These methods modify the background scenes significantly, insert objects with unrealistic scale, or fail to generate the object.

Figure 2: Examples of generated training pairs. The outpainted scene, containing a laptop, is shown on the left. The red box indicates the laptop position. We remove the laptop using inpainting (12) (right) to produce the input image. The generated images may contain diffusion and inpainting artifacts as well as distortions. Despite these imperfections, we find that given sufficient quantity, we can still extract sufficient information to train a placement model.

we wish to add, we leverage this prior by using diffusion-based image outpainting to hallucinate images of plausible full scenes containing the object. We then use these generated images as a large and diverse training dataset to learn an object placement predictor via weakly supervised learning. *In other words, we learn to place objects in scenes by using off-the-shelf diffusion models to place scenes around objects.*

We compare DreamPlace to both diffusion-based and object placement baselines. Using its learned object placement guidance, DreamPlace generates qualitatively more realistic object insertion edits than prior diffusion-based approaches on a range of common tabletop, indoor, street-view, and aerial object classes (section 4.1). Moreover, for a small set of object categories (tabletop objects like cups, laptops, etc.) with labeled evaluation data, our predicted object placements (learned purely from synthetic training data) quantitatively outperform those of prior state-of-the-art models trained on a large human-annotated placement dataset (13). DreamPlace achieves an average of 25.2% increase in our Intersection over Area (IoA) metrics across tabletop object categories and a 35.1% improvement in mean displacement (μ_D) from plausible surfaces to predicted bounding boxes (section 4.2).

2 Background and Related Work

Text-guided image diffusion models (21; 20; 22) are capable of synthesizing high-quality images and recent extensions such as (2; 1; 29; 30; 24; 28) aim to control the diffusion models. These approaches are capable of adding an object to the scene *given an explicit location or layout* as input; they do so via inpainting (21; 2; 28; 29; 11) or spatial conditioning in the form of masks, layers, or segmentation (29; 1; 24; 30). However, they do not solve the problem when given a scene without explicit input on where (and how) to add the object. For example, an artist needs to draw the shape and location of the edit. This is important because requiring a human to designate locations to add objects does not scale, when the goal of object edits is to generate or augment data for other downstream tasks. (3) enables instruction-based editing of images, but the text-specified edits do poorly at inserting a new object (Figure 1). Our goal is to enable object-insertion edits using a diffusion model but to have the system decide in full where and how the object is placed.

Similar object insertion tasks have been studied in the computer vision literature. Foreground object search (32; 34) for image compositing retrieves semantically compatible and correctly posed objects given an image, specified location (e.g., placement box), and database of place-able objects. Object placement (for 2D images) is the task of predicting plausible locations in an image where

an object may go. Approaches include both supervised (33; 27; 31; 15; 35) and weakly-supervised approaches (34; 5). The former is trained using human annotated datasets such as OPA (13).

However, unlike foreground object search and inpainting, OPA (13) does not model object geometry or pose; consequently, while a model trained on OPA may learn that tableware such as a plate often appears on a tabletop, the resulting composite is often unrealistic (Figure 7). We observe that models trained on OPA generalize poorly to more diverse and realistic scenes; the labels in OPA are often of low quality or biased (see Figure 6 for details), highlighting the difficulty of manually collecting data. Weakly-supervised approaches employ brute-force processing of large image datasets (18; 7) in order to find the relevant images containing the objects of interest and sufficient scene context (34), or they work on limited domains such as street images (31; 5). DreamPlace differs in that it generates a dataset of images for a particular set of object classes; this means that DreamPlace does not need to explicitly preprocess or filter millions of images or restrict itself to domains where wide scene images are easy to acquire. However, DreamPlace does depend on the pretrained diffusion model to generate sufficiently plausible scenes for training.

DreamPlace is loosely related to (23; 26; 6) which use diffusion models to synthesize training data for classification or image similarity tasks. Instruct Pix2Pix (3) uses a large language model (LLM) (4) and Stable Diffusion (21) to generate a paired dataset of image editing examples. DreamPlace adopts a similar approach to (3) but for the challenges of the object insertion task.

3 Methods

DreamPlace uses a two-step process to insert an object of a given category into a given input image. First, the placement network, $\mathcal{P}$, analyzes the empty scene image and proposes a coarse location and scale for the object, represented as a bounding box (section 3.1). $\mathcal{P}$ is trained using synthetic data generated by a pre-trained Stable Diffusion (21) in-painting model, which we refer to as $\mathcal{D}$. Then, $\mathcal{D}$ is used to synthesize (inpaint) the new object in the proposed bounding box (section 3.2). We describe the key details of DreamPlace and delegate full implementation details to Appendix A.

3.1 Placement Module

Given only an image depicting a scene, the goal of the placement module is to propose a bounding box in the image that represents a coarse placement instruction. Since the list of object classes is known, we generate data for each class and train independent, specialized placement models for each object class. We train the placement network, $\mathcal{P}$, to predict a plausibility score given an RGB image $\mathbf{x} \in \mathbb{R}^{128\times128\times3}$ and a binary mask $\mathbf{m} \in \{0,1\}^{128\times128}$, which encodes a bounding box for a proposed object. $\mathcal{P}$ is a ResNet-18, with 4 input channels — $\mathbf{x}$ and $\mathbf{m}$ concatenated — and is trained using positive and negative pairs; i.e., a mask encoding a plausible box vs. a randomly generated box. The loss used in training is binary cross-entropy, common to classification tasks.

To infer placements across the entire image at test time, we generate candidates for $\mathbf{m}$ using a grid and different box scales, similar to (34), and we aggregate the predicted scores into a 2D heatmap. Then, we rank the boxes according to a weighted sum of the confidence score of the box and the average value of the heatmap in this region. Boxes with the top combined scores are the predicted plausible regions. Further details and intuitions are provided in section A.1.

3.1.1 Training dataset generation

To train $\mathcal{P}$, we synthesize a large dataset of training pairs, consisting of images and corresponding object placements (locations and scales). For the model to learn non-trivial placements, the images need to have a wide field of view so that context is visible (e.g., the room around the table, in addition to the table surface, for tabletop object placement). Obtaining such images from the Internet or existing datasets is challenging (34), which motivates our generative approach.

We use a text-image diffusion in-painting model, $\mathcal{D}$, which takes an image $\in \mathbb{R}^{512\times512\times3}$, binary in-painting mask $\in \{0,1\}^{512\times512}$, and text prompt as input. As starting images, we download a small number of instance segmented seed objects (e.g., 50 laptops). These objects do not require scene context and are easily scraped from any major web search engine. We randomly paste the object and its segmentation mask in a 512×512 canvas and use $\mathcal{D}$ to outpaint the pixels outside of the mask. Descriptive text prompts are helpful for the diffusion model to generate realistic and coherent images,

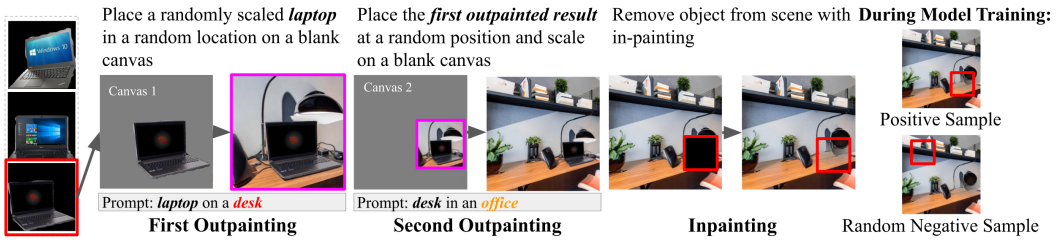

Figure 3: Data generation pipeline for object placement. We query a LLM (16) for a two-level hierarchy of locations of an object to create prompts for image generation. We use the prompts to outpaint around the object using the diffusion model, $\mathcal{D}$. Finally, we inpaint, using LaMa (12), the object from the scene to create space for placing an object. The final synthetic training example is the inpainted scene and the object placement (green box).

so we query a large-language model (LLM) (16) for text prompts that define the relationships for the object class and its context: e.g., *"a laptop on a desk"*. Given the above inputs, $\mathcal{D}$ produces an outpainted 512×512 image that contains the seed object. Finally, we use LaMa (12) to remove the seed object (via inpainting). The final inpainted image and bounding box mask of the seed object are used as a pseudo-labeled training pair, $\mathbf{x}$ and $\mathbf{m}$, for the placement network.

Figure 3 illustrates the full data generation pipeline. We repeat this process for different text prompts, seed objects, and randomly generated canvases to synthesize a large training set. A minor optimization that we find to improve generated image quality is to outpaint the context progressively (e.g., outpainting twice); the first round of out-painting generates a moderately wide context and the second round generates the remaining context in the final image. Since each round of out-painting is only required to synthesize a smaller fraction of the pixels in the final image, it avoids a common failure case for $\mathcal{D}$ to ignore the seed object and generate a new, large object when the outpainted region is too large (see examples in Figure 5 in the appendix).

3.2 Object Synthesis using Text-Guided Image Diffusion

Given a proposed object placement bounding box produced by $\mathcal{P}$, we use the diffusion model $\mathcal{D}$ to synthesize (via text-guided inpainting) an object of the desired category within the proposal box (x_1, y_1, x_2, y_2 coordinates). This is done by padding the proposal box by 50%, cropping, and inpainting the non-padded region using $\mathcal{D}$. The text prompt specifies the category to place; for example, to add a laptop, the prompt would be *"a laptop"*. Figure 4 shows examples of the inserted objects. We find that given coarse placement guidance, Stable Diffusion ($\mathcal{D}$) is able to inpaint reasonably posed and harmonized objects while preserving the rest of the scene.

4 Experiments

We evaluate DreamPlace's ability to place objects against diffusion-based baselines (section 4.1) as well as state-of-the-art baselines for object placement (trained on OPA (13); section 4.2).

4.1 Qualitative Comparison to Diffusion-based Approaches

We compare DreamPlace to two naive Stable Diffusion (SD) in-painting baselines and Instruct Pix2Pix (3). First, we consider using SD to inpaint a large box covering most of the image — i.e., let SD decide the placement. Second, we consider SD given a randomly chosen location and scale. These two naive baselines confirm that SD, by itself, lacks the planning ability to place objects and that random boxes are unlikely to be plausible placements. Instruct Pix2Pix (3) is a recent approach for targeted image editing using text; we show that it too is insufficient to make the object placement edits that are the focus of this paper.

We test these methods on diverse Internet images of real world scenes such as offices, living rooms, city streets, etc. As evaluation categories, we choose common objects, including tabletop objects such as laptops, bowls, cups, and keyboards; street-view objects such as cars and pedestrians; and other categories such as paintings, clocks, and birds. These categories can be generated by Stable Diffusion and often take up very small portions of a scene.

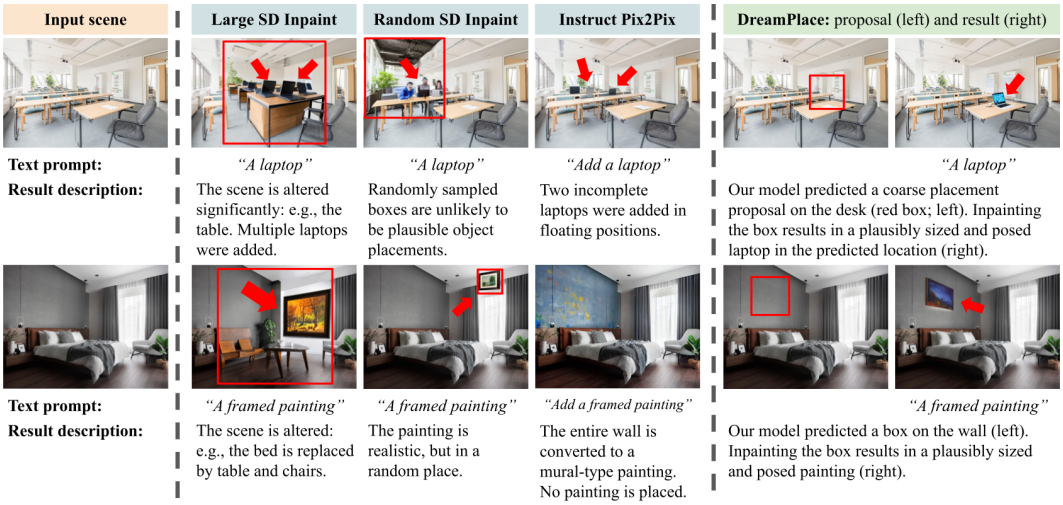

Figure 4: Qualitative results of our method compared against the baseline methods, across different categories of objects. DreamPlace avoids common failure cases of the baseline methods described in section 4.1. See appendix Figure 12 for full results with examples from more classes.

Figure 4 shows qualitative results; see Figure 12-13 in the appendix for extended results). Inpainting a large region with SD modifies the scene significantly, yielding a new image that often no longer depicts the original scene. Random placement leads to inpainted objects that are in improbable locations and incorrectly sized. Instruct Pix2Pix fails when prompts describe an object-level addition to a scene. DreamPlace-guided inpainting is able to add small-scale objects in reasonable positions while still retaining the scene context.

4.2 Quantitative Comparison to Object Placement Baselines

We also compare DreamPlace, quantitatively, to recently published object placement baselines: TERSE (27), PlaceNet (31), and GracoNet (33). These methods predict a location and scale (e.g., box) given a scene image and are trained directly on the OPA (13) dataset. Code and data for weakly-supervised methods such as GALA (34) is not publicly available.

Our evaluation dataset consists of 100 scenes for tabletop object placement and four types of objects (laptops, keyboards, cups, and bowls). To produce a ground truth label for placements for the scene by which to compute quantitative metrics, we manually annotate plausible surfaces for object placement on these images. Examples of test scenes and labels are shown in Figure 9 in the appendix. Since it is implausible to enumerate all good placement boxes in our evaluation dataset, we quantify performance using two automatic but objective metrics. While not fully informative of good placement, performing poorly on these metrics suggests bad placement.

Metric 1: Intersection over Area. We expect that a good object placement has area intersection with the feasible region — i.e., be placed on a surface. Small or no intersection would imply that the object is floating. We measure intersection over (object) area as a proxy for this heuristic; i.e, the fraction of the object that is in the plausible region. Note that intersection-over-union (IoU) is a common metric but is not applicable because the objects being placed are small compared to the surface; we are unconcerned with the area of the surface not taken up by an object.

Metric 2: Mean Displacement. Suppose an object is not within a plausible region. Then we measure its proximity to the nearest plausible region: i.e., the distance required to move a placement. We approximate this by the average distance (in fraction of image dimension) from the bottom edge of the predicted object bounding box to any plausible surface in the scene.

We compare to the baselines trained both on the entire OPA dataset and specialized by the class. DreamPlace's is most comparable to the latter since we specialize for each object class, but we include both for completeness. Results are shown in Table 1. Overall, DreamPlace generates more localized and accurate placement locations, outperforming all prior methods on both metrics. Figure 10 bounding boxes inferred by the baselines and the heatmaps produced by DreamPlace.

Method	Training Data	Single-Class	Laptop		Keyboard		Cup		Bowl	
			IoA ↑	μ_D ↓	IoA ↑	μ_D ↓	IoA ↑	μ_D ↓	IoA ↑	μ_D ↓
TERSE	OPA		0.212	0.254	0.152	0.213	0.231	0.265	0.204	0.222
PlaceNet	OPA		0.262	0.137	0.269	0.131	0.240	0.227	0.244	0.184
GracoNet	OPA		0.162	0.322	0.094	0.320	0.170	0.335	0.146	0.329
TERSE	OPA	✓	0.156	0.159	0.288	0.096	0.170	0.179	0.174	0.118
PlaceNet	OPA	✓	0.301	0.120	0.347	0.086	0.290	0.163	0.305	0.123
GracoNet	OPA	✓	0.230	0.128	0.117	0.200	0.097	0.295	0.246	0.183
DreamPlace	Generated	✓	**0.389**	**0.059**	**0.432**	**0.081**	**0.313**	**0.095**	**0.424**	**0.068**

Table 1: Quantitative placement performance compared to supervised, OPA-trained models (13). The intersection-over-area (IoA) and mean displacement (μ_D) metrics are explained in section 4.2.

Method	Training Data	Single-Class	Laptop	Keyboard	Cup	Bowl
Random Box			0.30	0.25	0.24	0.13
TERSE	OPA		0.41	0.35	0.19	0.21
PlaceNet	OPA		0.47	0.46	0.20	0.31
GracoNet	OPA		0.29	0.23	0.13	0.17
TERSE	OPA	✓	0.19	0.34	0.07	0.16
PlaceNet	OPA	✓	0.42	0.40	0.30	0.26
GracoNet	OPA	✓	0.26	0.37	0.05	0.25
DreamPlace	Generated	✓	**0.55**	**0.53**	**0.49**	**0.43**

Table 2: Visual quality assessment for tabletop object classes. Refer to **Metric 3** in section 4.2 for the assessment criteria. The numbers are the percent of images that pass the criteria.

For an end-to-end evaluation, we test Stable Diffusion's ability to inpaint the final object given proposed bounding boxes from DreamPlace and the baselines; we expect that poor proposals will also hamper the ability to synthesize the object.

Metric 3: Visual Quality Assessment. We perform a qualitative assessment of images produced by SD, guided by different placement proposal methods. The metric is the percentage of output images that are plausible, defined by two criteria: (1) whether the new object interacts reasonably with a surface and (2) whether the object is of reasonable scale. Table 2 tabulates results from DreamPlace and the baselines (27; 31; 33). We also include random placement proposals as a baseline. Overall, DreamPlace produces a higher percentage of passable images. Because DreamPlace proposes more accurate edit regions, it also results in fewer scene-level changes during the object insertion. Refer to Figure 10 in the appendix for examples and analysis.

5 Discussion, Limitations, and Conclusion

DreamPlace is a proof of concept that existing text-image diffusion models can be leveraged as both weakly-labeled dataset generators and object synthesizers to perform the object placement task. We identify several improvements for future work. First, the set of objects can be expanded. Currently, DreamPlace requires an instance segmenter to seed objects for data-generation; a zero-shot model (10) would enable arbitrary objects. Second, newer approaches for object generation (29) can be used as a drop in replacement for Stable Diffusion (21). Third, DreamPlace is not optimized for speed and efficiency — heatmap generation is expensive. Optimizations from recent object placement literature (35) or training a general, multi-class DreamPlace model can provide efficiency gains.

The quality of the text-image diffusion model on which DreamPlace is based is both a limitation and a cause for optimism. Stable Diffusion (21) struggles to generate certain object classes (e.g., scissors, toothbrush), which limits the quality of both placement and object synthesis (see Figure 14 in the appendix). Diffusion models such as (17; 14; 19; 22; 9) are advancing at a rapid pace, however, and future models are likely to succeed on objects that are challenging today. Moreover, we anticipate that DreamPlace and approaches like it to leverage generative models as dataset generators will become viable for a broader palette of vision and graphics tasks.

References

[1] Omri Avrahami, Thomas Hayes, Oran Gafni, Sonal Gupta, Yaniv Taigman, Devi Parikh, Dani Lischinski, Ohad Fried, and Xi Yin. Spatext: Spatio-textual representation for controllable image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18370–18380, June 2023. 2

[2] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18208–18218, June 2022. 2

[3] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18392–18402, June 2023. 1, 2, 3, 4, 9, 14

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. 3

[5] Jui-Ting Chien, Chia-Jung Chou, Ding-Jie Chen, and Hwann-Tzong Chen. Detecting nonexistent pedestrians. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017. 3

[6] Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. Dreamsim: Learning new dimensions of human visual similarity using synthetic data, 2023. 3

[7] Google. Open images dataset, 2023. 3

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 9

[9] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P. Kingma, Ben Poole, Mohammad Norouzi, David J. Fleet, and Tim Salimans. Imagen video: High definition video generation with diffusion models, 2022. 6

[10] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023. 6

[11] Sumith Kulal, Tim Brooks, Alex Aiken, Jiajun Wu, Jimei Yang, Jingwan Lu, Alexei A. Efros, and Krishna Kumar Singh. Putting people in their place: Affordance-aware human insertion into scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17089–17099, June 2023. 2

[12] Prakhar Kulshreshtha, Brian Pugh, and Salma Jiddi. Feature refinement to improve high resolution image inpainting. *arXiv preprint arXiv:2206.13644*, 2022. 2, 4, 9

[13] Liu Liu, Zhenchen Liu, Bo Zhang, Jiangtong Li, Li Niu, Qingyang Liu, and Liqing Zhang. Opa: Object placement assessment dataset, 2022. 2, 3, 4, 5, 6, 10, 12

[14] Midjourney. Midjourney, 2023. 6

[15] Li Niu, Qingyang Liu, Zhenchen Liu, and Jiangtong Li. Fast object placement assessment. *arXiv preprint arXiv:2205.14280*, 2022. 3

[16] OpenAI. Chatgpt, 2023. 4, 9

[17] OpenAI. Dall-e 3, 2023. 6

[18] Pixabay. Pixabay.com, 2023. 3

[19] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis, 2023. 6

[20] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents, 2022. 2

[21] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, June 2022. 2, 3, 6, 9, 12, 16

[22] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Raphael Gontijo-Lopes, Burcu Karagol Ayan, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. 2, 6

[23] Mert Bülent Sarıyıldız, Karteek Alahari, Diane Larlus, and Yannis Kalantidis. Fake it till you make it: Learning transferable representations from synthetic imagenet clones. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023. 3

[24] Vishnu Sarukkai, Linden Li, Arden Ma, Christopher Ré, and Kayvon Fatahalian. Collage diffusion, 2023. 1, 2

[25] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade W Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa R Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5b: An

open large-scale dataset for training next generation image-text models. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. 1

[26] Yonglong Tian, Lijie Fan, Phillip Isola, Huiwen Chang, and Dilip Krishnan. Stablerep: Synthetic images from text-to-image models make strong visual representation learners, 2023. 3

[27] Shashank Tripathi, Siddhartha Chandra, Amit Agrawal, Ambrish Tyagi, James M. Rehg, and Visesh Chari. Learning to generate synthetic data via compositing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 3, 5, 6, 9, 12, 13

[28] Su Wang, Chitwan Saharia, Ceslee Montgomery, Jordi Pont-Tuset, Shai Noy, Stefano Pellegrini, Yasumasa Onoe, Sarah Laszlo, David J. Fleet, Radu Soricut, Jason Baldridge, Mohammad Norouzi, Peter Anderson, and William Chan. Imagen editor and editbench: Advancing and evaluating text-guided image inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18359–18369, June 2023. 2

[29] Shaoan Xie, Zhifei Zhang, Zhe Lin, Tobias Hinz, and Kun Zhang. Smartbrush: Text and shape guided object inpainting with diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22428–22437, June 2023. 1, 2, 6

[30] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models, 2023. 2

[31] Lingzhi Zhang, Tarmily Wen, Jie Min, Jiancong Wang, David Han, and Jianbo Shi. Learning object placement by inpainting for compositional data augmentation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*, pages 566–581. Springer, 2020. 3, 5, 6, 9, 12, 13

[32] Hengshuang Zhao, Xiaohui Shen, Zhe Lin, Kalyan Sunkavalli, Brian Price, and Jiaya Jia. Compositing-aware image search. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 502–516, 2018. 2

[33] Siyuan Zhou, Liu Liu, Li Niu, and Liqing Zhang. Learning object placement via dual-path graph completion. In *European Conference on Computer Vision*, pages 373–389. Springer, 2022. 3, 5, 6, 9, 12, 13

[34] Sijie Zhu, Zhe Lin, Scott Cohen, Jason Kuen, Zhifei Zhang, and Chen Chen. Gala: Toward geometry-and-lighting-aware object search for compositing. In *European Conference on Computer Vision*, pages 676–692. Springer, 2022. 2, 3, 5

[35] Sijie Zhu, Zhe Lin, Scott Cohen, Jason Kuen, Zhifei Zhang, and Chen Chen. Topnet: Transformer-based object placement network for image compositing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1838–1847, June 2023. 3, 6

A Implementation Details

A.1 DreamPlace

We optimize DreamPlace's placement model, $\mathcal{P}$, with the AdamW optimizer with a learning rate of 0.0001. As mentioned in the main text, $\mathcal{P}$ is a standard ResNet (8) classifier and is trained using cross-entropy loss. During training, we apply standard data augmentations such as random cropping, random flipping, color jitter, etc. This is in addition to the already randomized object scales during dataset generation.

During inference, we divide input scenes into a 32×32 grid. We generate a set of candidate square bounding boxes at five discrete scales centered at each grid intersection point (excluding the edge coordinates). This results in a total of $31 \times 31 \times 5$ candidate boxes across the entire image. We further eliminate candidate boxes that extend beyond the borders of the scenes. Predicted confidence scores associated with each valid candidate are then aggregated to produce a heatmap. We rank the candidates according to the sum of their confidence scores and their average values of the heatmap; the combination ensures that we propose more robust candidate locations in regions with a high likelihood of reasonable placement. In cases where the input scenes are non-square, the scoring process for each candidate involves cropping a square region around the box. Our inference procedure is described in Figure 8.

To generate the final object insertion edits, given a proposed placement, we use the Stable Diffusion (21) (V2) inpainting model without fine-tuning.

A.2 Placement Dataset Generation

For each object category, we segment ≈ 50 scraped foreground objects. We query a large language model (16) to construct 6 to 20 prompts and generate ≈ 400 outpainting scenes for each foreground object, with randomly selected prompts. This procedure constructs category-specific datasets containing approximately 20k training samples. For each round of outpainting, we run the Stable Diffusion (V2) inpainting model with a guidance scale of 4.0 and 50 inference steps. Finally, to remove the seed object from the final image, we use the default settings for LaMa (12).

A.3 Baselines

The three diffusion-based baselines in section 4.1 are (1) inpainting a large box using Stable Diffusion (SD) (21), (2) inpainting a random placement using SD, and (3) using Instruct Pix2Pix (3). The text prompt for the first two SD baselines is a singular instance of the object category: e.g., *"a laptop"*. The edit instruction provided to Instruct Pix2Pix is *"add a <class>"*, where class is the object category: e.g., *"add a laptop"*.

We use publicly available code for GracoNet (33), PlaceNet (31), and TERSE (27). Refer to their official implementations for hyperparameters and training details.

Figure 5: DreamPlace adopts a hierarchical outpainting approach (2 rounds). The examples above are images with only one round of outpainting (ablation). Since the seed object covers only a small portion of the canvas (left), even with a detailed text prompt, Stable Diffusion fails to generate a coherent scene with wide visual context. A common failure mode is to ignore the seed object and to generate one or more larger instance of the object category in the outpainted region. Our hierarchical approach (used to generate the training data in Figure 2) reduces the amount of pixel area generated in each round of outpainting and produces more realistic/plausible scenes.

Figure 6: Examples in the OPA dataset (13) do not consider object geometry, pose, or whether there is space for an object. This results in poor performance by models trained on this data (Figure 7).

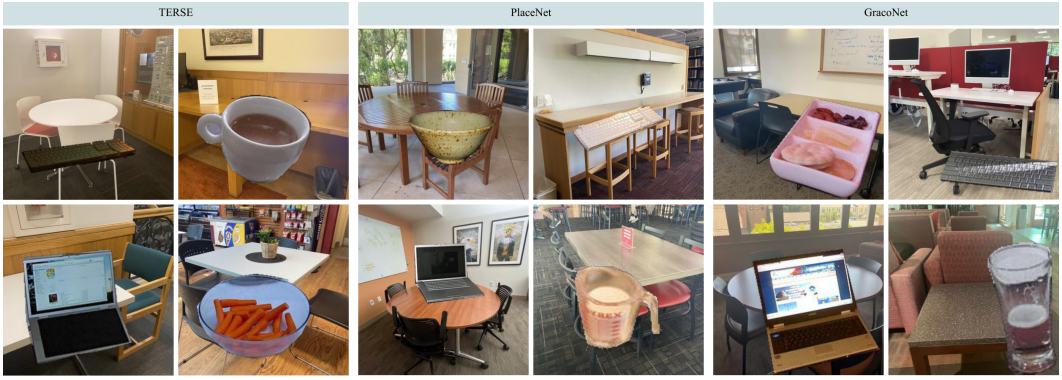

Figure 7: Models trained on the OPA dataset (13) generalize poorly to placement problems in real-world scenes, producing placements of implausible location, scale, and orientation.

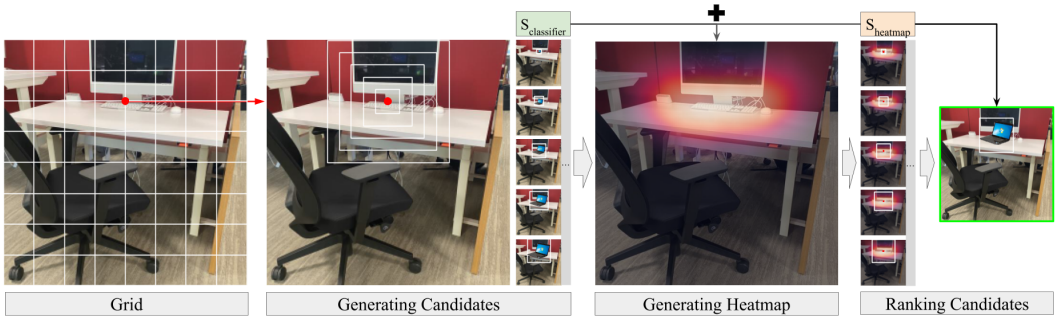

Figure 8: Inferring object placements with DreamPlace. We generate candidates of different scales at locations in a 32×32 grid; aggregate confidence scores from the placement network, $\mathcal{P}$, to form a heatmap; and rank the generated candidates by combining the confidence scores and average values of the heatmap.

Figure 9: Example images from the evaluation dataset. We annotate the surfaces that could support plausible placements. Red shows regions that are highly plausible, while blue regions are possible but less likely for objects such as laptops, keyboards, etc.

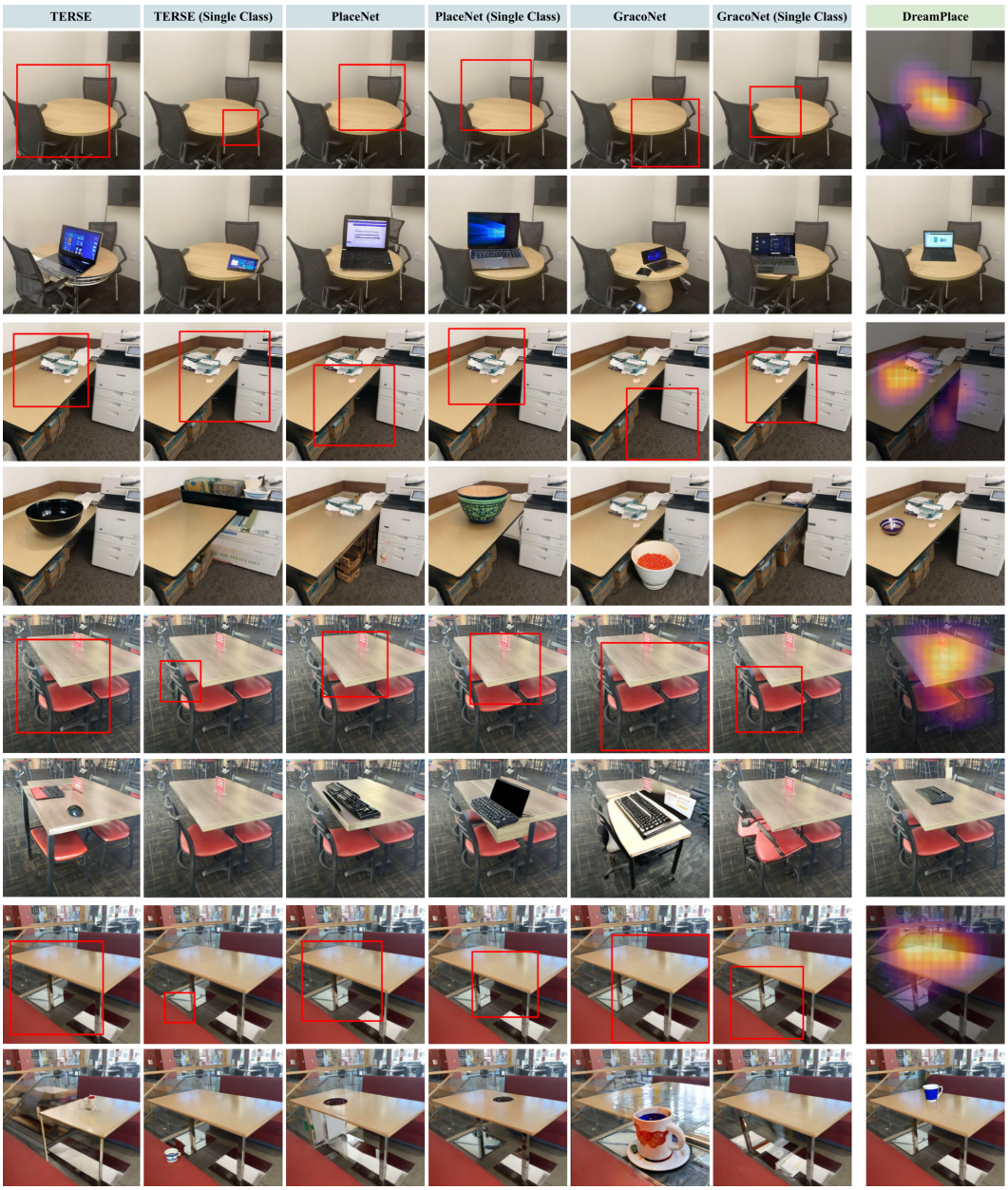

Figure 10: Qualitative placement results by DreamPlace and the OPA (13) trained baselines. The proposed placement region (red box; upper) and the final object placement generated by object inpainting (21). We show results from TERSE (27), PlaceNet (31), and GracoNet (33), trained on all classes of OPA and specialized to a single class (within OPA). DreamPlace's proposal regions are more localized and accurate in size. The baselines often predict overly large placement bounding boxes. While this provides Stable Diffusion flexibility to generate the target objects, it often results in incorrect object sizes or significant alterations the scene.

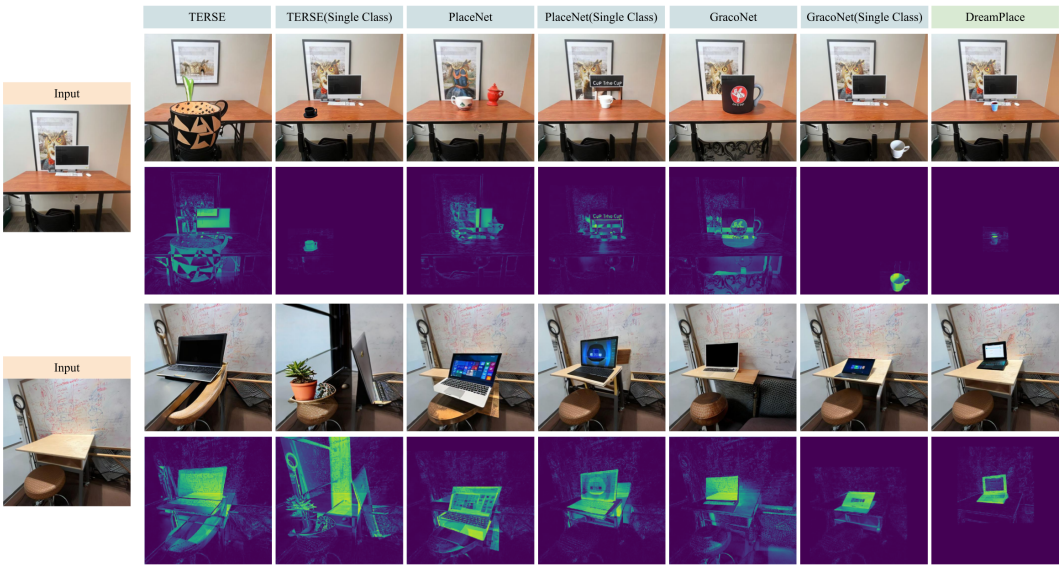

Figure 11: Unintended global scene edits produced by baseline methods. Diffusion-based inpainting models are capable of refining the location and scale of the target object given a sufficiently large inpainting region. If the inpainting mask (for object placement) is too large, however, the background may also be unintentionally changed; this is a issue for the baselines (27; 31; 33), which often propose overtly large placement masks. DreamPlace, with its more accurate proposal boxes, preserves the background more faithfully.

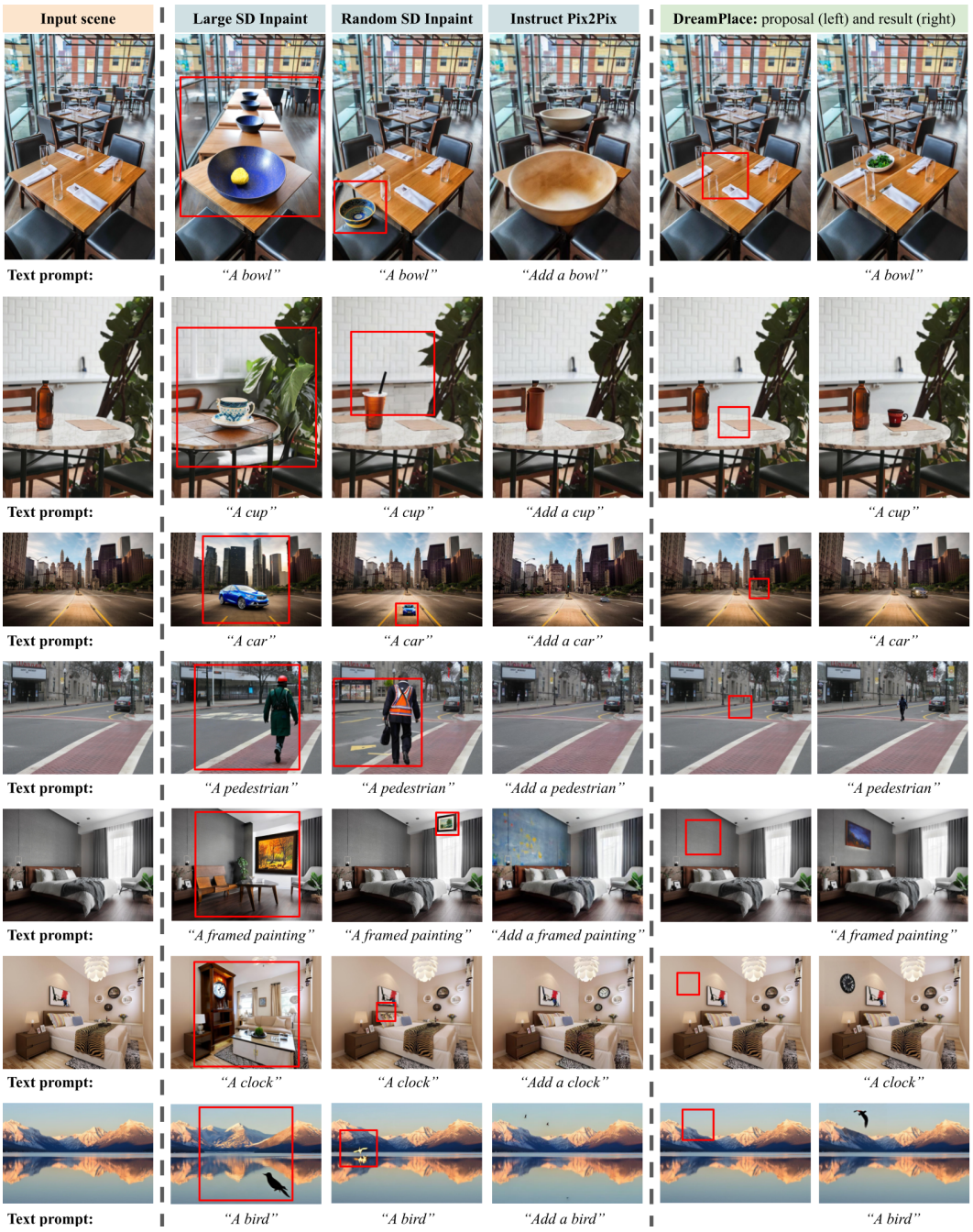

Figure 12: Qualitative object insertion results for different object categories. DreamPlace achieves coherent visual results more consistently than the baselines. "SD inpaint with large masks" often alters the scenes significantly. "SD inpaint with random masks" frequently fails due to poor location guidance. Instruct Pix2Pix (3) is distracted by objects already present in the scenes, rather than adding new objects. As a concrete example, in the clock row, "SD inpaint with large masks" changes the room layout completely, "SD inpaint with a random mask" chooses a poor location, and Instruct Pix2Pix changes the decorations on the wall into clocks. DreamPlace is able to select an empty location on the wall and guide Stable Diffusion to produce a convincing output image.

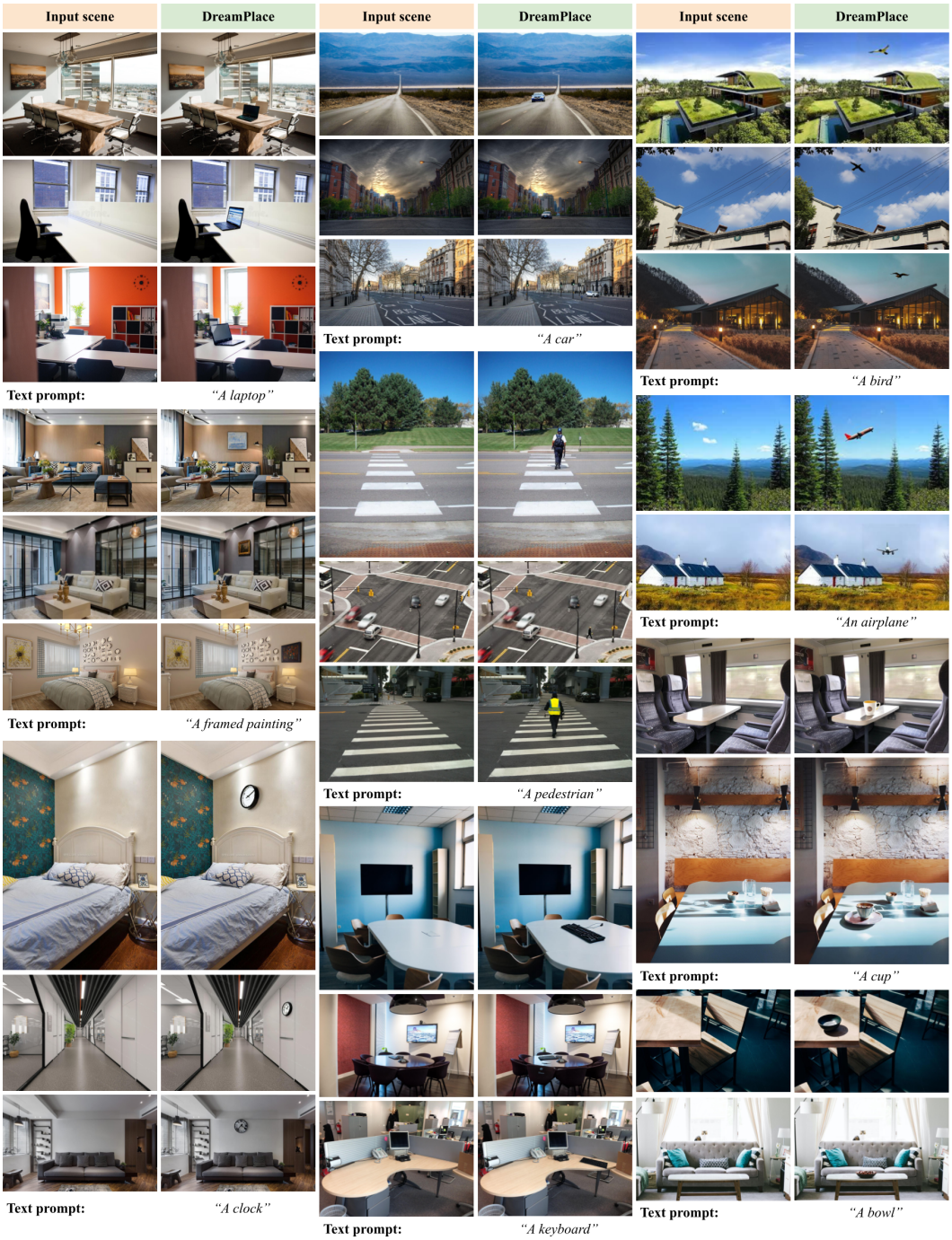

Figure 13: Additional qualitative results by DreamPlace. The list of classes is given in section 4.1.

Figure 14: Stable Diffusion (21) struggles to generate certain categories of objects: for example, small thin objects such as scissors and toothbrushes. This is a limitation for DreamPlace and its current ability to generate placement data for these classes.