Localizing Object-level Shape Variations with Text-to-Image Diffusion Models Supplementary Materials

Or Patashnik[1] Daniel Garibi[1] Idan Azuri[2] Hadar Averbuch-Elor[1] Daniel Cohen-Or[1]

[1]Tel-Aviv University [2]Independent Researcher

A. Additional Details

A.1. Prompt-Mixing

Denoising Diffusion Process Stages Additional examples of the denoising stages are shown in Figure 1. The used prompts are "A $\langle w \rangle$ is flying in the sky" (first row), and "A $\langle w \rangle$ on the beach" (second row).

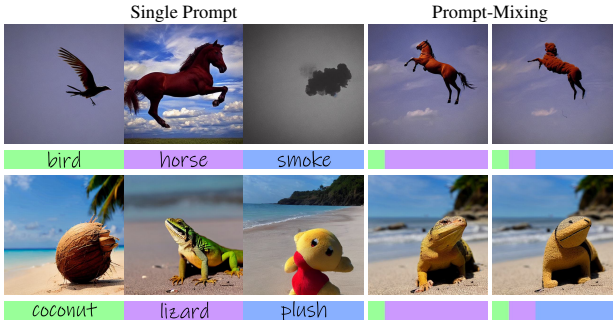


Figure 1. Prompt-mixing. The colored bars under the images on the right represent the corresponding word used along the denoising process. This figure complements Figure 4 in the main paper.

Cross-Attention Injection We validate the intuition about the role of the Keys and Values in the cross-attention layers, as mentioned in the main paper. This experiment explains the design choice of modifying only the Values with the altered prompt when applying prompt-mixing. We analyze prompt-mixing where the altered prompt is used to compute a different number of components of the cross-attention layer. In Figure 3, the first two columns correspond to the generation of an image with a single prompt $P(w)$ along the entire denoising process. In the other columns, we perform prompt-mixing where we use $P_{[T,T_3]}(w_1)$ and $P_{[T_3,0]}(w_2)$. In other words, we take the layout from w_1 ("puppies", "surfborad", "bird"), and the shape and fine visual details from w_2 ("flowers", "box", "kitten").

In the third column, we use $P(w_2)$ to compute the Values and $P(w_1)$ to compute the Keys in $[T_3, 0]$, the layouts of the obtained images are similar to these obtained by the images generated by $P(w_1)$ in the first column. For example, in the first row, a black flower is added to capture the black puppy. In the fourth column, where we use $P(w_2)$ to compute the Keys and $P(w_1)$ to compute the Values, the appearance of the objects is still of the objects generated by $P(w_1)$ in the first column. This strengthens our intuition that the Values of cross-attention layers control object shape and appearance while the Keys hardly affect it. In the fifth column, we use $P(w_2)$ to compute both the Keys and the Values in $[T_3, 0]$. As can be seen, the images in the fifth column capture the layout of the images generated by $P(w_1)$ worse than the images in the third column. For example, in the first row there are two flowers and the image looks similar to the one generated solely by $P(w_2)$ in the second column. Additionally, in the third row there are two kittens, one instead of the bird and the other at the same place as the kitten in the image generated solely by $P(w_2)$. Therefore, when applying prompt mixing, the modified prompts are used to compute only the Values of the cross-attention layers in the corresponding timestep interval.

Proxy Words We utilize proxy words as a technique for navigating through the prompt embedding space in a meaningful manner. Another potential method is to add noise to the prompt embedding or the cross-attention Values, but we have observed that this approach performs significantly worse and results in lower diversity. Our experimental findings indicate that our proxy-word selection method occasionally produces unexpected words that generate convincing variations in shape. For instance, the word "something" frequently appears as a proxy word in our examples. Additionally, we have discovered that certain words with a significant CLIP distance from the object of interest can still generate reasonable variations. However, we opt to use our simple ranking mechanism that avoids more intricate selections for words (considering for instance word concreteness) as we find that it enhances the likelihood of producing plausible variations.

In Figure 2, we show a few proxy-words examples. For the prompt "Luxury yellow *purse* on a table" we take three

Figure 2. Our method traverses the CLIP space using proxy words, enabling us to generate object-level variations. Here, we generate two different outputs using the prompt “Luxury yellow *purse* on a table” shown in the “original” column. We then apply Mix-and-Match using proxy words, where the left part shows auto-suggested proxy words using our ranking technique. In the right part, we show words from a large distance in CLIP’s space. As illustrated above, distant words tend to yield objects which can no longer be identified as the original object, while our method allows for generating convincing object-level variations.

Figure 3. Cross-attention injection. The bars represent the word used along the denoising process. During the purple interval, we use the modified prompt (*i.e.*, with the “purple” word) to compute the cross-attention components (K, V) indicated over the purple bar. We used the prompts: “Two $\langle w \rangle$ on grass” (first row), “A $\langle w \rangle$ on sand” (second row), and “A $\langle w \rangle$ in the park” (third row).

high-ranked proxy-words and three words with a large CLIP distance. As can be seen, even words that are loosely related to the word “purse” (*e.g.*, “butterfly”) sometimes allow for generating plausible shape variations. However, closer words tend to produce more successful variations.

A.2. Attention-Based Shape Localization

Self-Attention Injection Mask Controls As we describe in Section 5.1, we selectively inject the self-attention map of the original image into the generated image, by constructing a mask that contains rows and columns corresponding to pixels of the object we aim to preserve. Specifically, we define the mask as follows:

$$M_t^{(l)}[i,j] = \begin{cases} 1 & i \in O_t^{(l)} \text{ or } j \in O_t^{(l)} \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $O_t^{(l)}$ is the set of pixels corresponding to the object we aim to preserve. Injecting only the rows (*i.e.*, setting 1 in $M_t^{(l)}$ if $i \in O_t^{(l)}$) keeps the effect of each pixel in the image on the pixels in $O_t^{(l)}$, but allows the pixels in $O_t^{(l)}$ to change their effect on other pixels in the image. When using such a “rows” mask, pixels in the image which weren’t part of the object ($O_t^{(l)}$) in the original image, can behave as part of the object in the new image. This can be seen in Figure 5, where ears were added to the dog in pixels that contained the hat in the original image. Conversely, injecting only the columns (*i.e.*, setting 1 in $M_t^{(l)}$ if $j \in O_t^{(l)}$) allows pixels in the image to affect differently on pixels of the object to preserve ($O_t^{(l)}$) than in the original image, but keeps the effect of the pixels in $O_t^{(l)}$ on the other pixels in the image. In our experiments, we found the mask that contains both the rows of the columns (Equation 1) to be the most robust.

In some cases, using both “rows” and “columns” masks to preserve an object limits the required geometric changes of the object of interest (*i.e.*, the object we aim at obtaining variations of). We define $O_t^{'(l)}$ as the set of pixels corresponding to the object of interest. In such cases, the relations between each pixel in $O_t^{(l)}$ to each pixel of $O_t^{'(l)}$ remain as they were in the original image. In those cases we suggest the following mask:

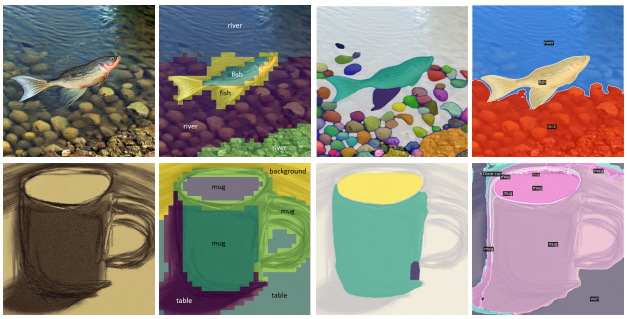
$$M_t^{(l)}[i,j] = \begin{cases} 1 & i \in O_t^{(l)} \setminus O_t^{'(l)} \text{ or } j \in O_t^{(l)} \setminus O_t^{'(l)} \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Removing the pixels of the object of interest from the

mask unleashes the object of interest from its connection to the object we are preserving. The choice of whether to remove $O_t^{'(l)}$ from the mask depends on the user's preferences.

A.3. Attention-guided Segmentation

The self-segmentation technique that we introduce leverages the generative prior of large diffusion models, rather than an external segmentation model. While large segmentation models (e.g., SAM [3], ODISE [8]) trained with a lot of supervision are strong, they also come with heavy computational costs. Our self-segmentation utilizes the rich features that are used to synthesize the given image. This allows dealing with challenging scenarios, like the fish with a transparent fin or a non-realistic image such as the sketch of the cup shown in Figure 4. Zoom in to observe the upper fin of the fish.



Original Ours SAM ODISE

Figure 4. Comparing our self-segmentation technique with state-of-the-art segmentation methods on challenging cases. Observe the transparent fin of the fish.

B. Experiments Settings

B.1. Implementation Details

In our experiments, we used $T = 50, T_3 = 44 \pm 1, T_2 = 34 \pm 1, T_1 = 15 \pm 1$. We found that the optimal T_3, T_2 which indicates the prompt-mixing range may slightly change between different prompts and seeds.

For segmenting the images using the self-attention maps, we used 5 clusters across all our experiments. Note, that since we automatically label each cluster, it is possible to increase the number of clusters without affecting the results. We used $\sigma = 0.3$ for the clusters labeling.

Our method does not require optimization, and can therefore not consume a lot of memory.

B.2. Evaluation Setup for Alternative Methods

As we describe in Section 6.1, we compare our method to two types of methods: (i) Non-deterministic methods that provide different results for different seeds (SDEdit[4] and

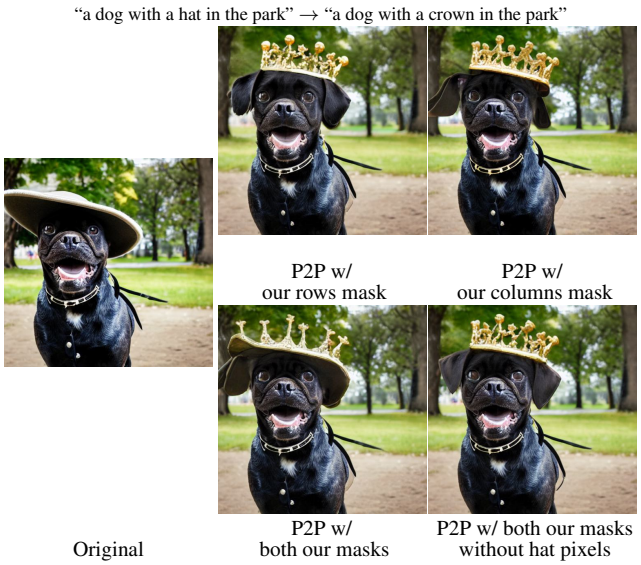


Figure 5. Comparison between different masks used for self-attention map injection, integrated with P2P.

Inpainting[6]), and (ii) Text-guided image editing methods (Prompt-to-Prompt and Instruct-Pix2Pix).

The naïve way to attain variations with non-deterministic methods is to insert the input image into the method with a different seed each time. We also compare to these methods using our auto-generated proxy words, where we use the original prompt, with the proxy word replacing the word corresponding to the object of interest. In Figure 10 in the main paper, both approaches are presented, the naïve way (indicated by the method's name, *e.g.*, SDEdit) and each method when using proxy words (*e.g.*, SDEdit-proxy).

As mentioned in Section 6.1, we used text-guided image editing methods to create variations by refining the prompt, adding adjectives to the explored object, or replacing the explored object with our proxy words. Prompt-to-Prompt[2] supports both refining a prompt and replacing a word in a prompt. To apply Plug-n-Play[7], we used prompts with proxy words. Instruct-Pix2Pix[1] receives as input a prompt that defines the required edit. To refine an object, we used the prompt "make the {object name} more {adjective} and keep the {other object}", where the object name is the object of interest, and we use a different adjective for different variations. We also added the suffix "keep the other object" to preserve other objects in the image. We used to prompt "change the {object name} to a {proxy word} and keep the {other object}" to replace the object with a proxy word. Zero-shot-Image2Image[5] supports only replacing an object, and therefore we didn't use prompt refinement with this method. In this method, 1000 sentences containing both the original and target words are required to replace an object. We used ChatGPT to create 1000 sentences for each proxy word.

B.3. Quantitative Comparison Dataset

We first created three templates of prompts: "A {mug} with w", "A {sofa} with a w on it" and "A {basket} with w". For each template, we created five prompts by replacing w with 5 different words (*e.g.*, "A {mug} with tea" and "A {mug} with hot chocolate"). For each prompt we generated 10 initial images, using different seeds, and got a dataset that contains 150 images. With each method, we created 20 variations of the object of interest for each initial image in the dataset.

C. Additional Results

C.1. Comparison to Other Methods

In Figure 7, we extend Figure 7 in the main paper by showing another example. Our method produces diverse results and remains faithful to the class of tents while preserving the rest of the image. Plug-n-Play[7] and InstructPix2Pix[1] make only minor texture changes to the tent and fail to preserve the background of the image. P2P[2] makes some noticeable texture changes when injecting self-attention during 40% for the steps, or geometric shape changes when injecting self-attention during 10% of the steps, but drifting away from the tents domain. Zero-shot Image2Image Translation struggles at changing the shape of the tent. Imagic struggles at preserving the hamster and the background and drifts away from the tents domain.

In Figure 8, we show our result for the example in Figure 2 in the main paper. As can be seen, our method produces significant shape changes while preserving the bananas and the background.

C.2. Our Method Results

In Figures 6 and 8, we show additional results of our object shape variations pipeline.

Figure 6. Object-level variations: Real image vs Synthetic. On the top, we show object variations for a real image, where the edited object is a lamp. On the bottom, we edit the lamp but in a different synthetic scene. Our method successfully generates different shape variations for both real and synthetic images.

References

[1] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions. November 2022. 3, 4

[2] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. 2022. 3, 4

[3] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023. 3

[4] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022. 3

[5] Gaurav Parmar, Krishna Kumar Singh, Richard Zhang, Yijun Li, Jingwan Lu, and Jun-Yan Zhu. Zero-shot image-to-image translation. *ArXiv*, abs/2302.03027, 2023. 3

[6] Robin Rombach, A. Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10674–10685, 2021. 3

[7] Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. *arXiv preprint arXiv:2211.12572*, 2022. 3, 4

[8] Jiarui Xu, Sifei Liu, Arash Vahdat, Wonmin Byeon, Xiaolong Wang, and Shalini De Mello. Open-vocabulary panoptic segmentation with text-to-image diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2955–2966, 2023. 3

Figure 7. Comparisons to text-guided editing methods. In each column, we show the results of a different word that replaces the original word "tent" in the prompt.

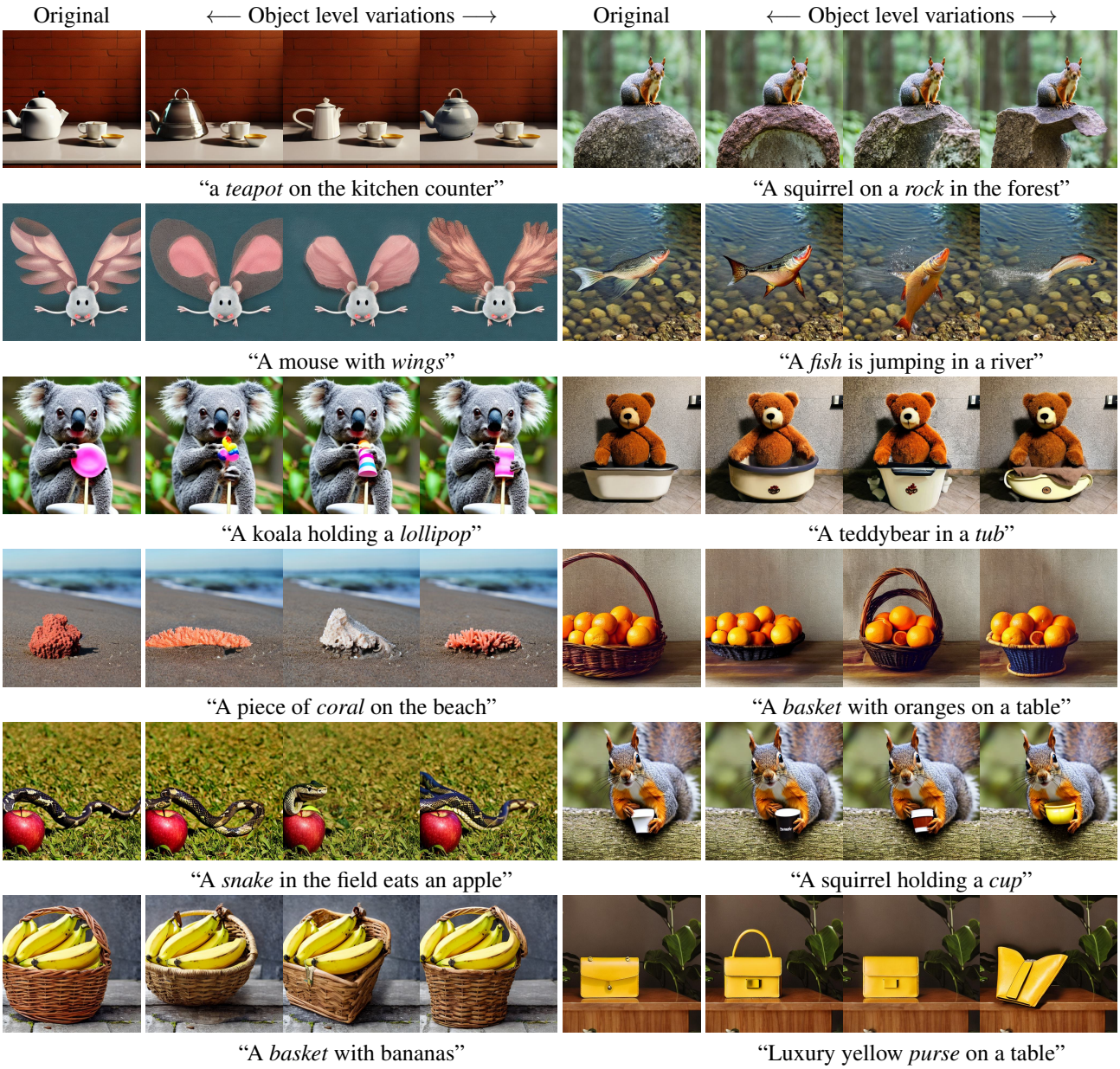


Figure 8. Object-level variations for various scenes. For each scene, the leftmost image is the original sampled one. The *emphasized* word corresponds to the explored object. As can be seen, our method generates different shape variations for each explored object.