Context Diffusion: In-Context Aware Image Generation

Supplementary Material

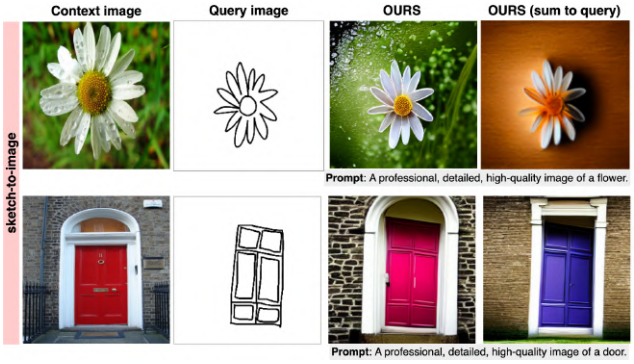

Figure A.1. **Ablation: Benefit of visual conditioning.** The images in the third column are generated by our best model, while the fourth column depicts the ones obtained with the summation approach.

A. Ablation study

Benefit of visual conditioning. The ingestion of the visual context examples is an important part of our framework because it controls the generation process as much as the text prompts. Consequently, we process the context images similarly to the prompt. Here, we compare this approach with an alternative one, implemented in Prompt Diffusion [43]. Specifically, by using ConvNets to encode the context images and then sum them to the query image. As we visualize in Figure A.1, this approach has limitations in actually learning from the context. Instead, we show that by separating the condition of the context images from the visual structure provided by the query images, our model can more effectively capture details of the context images.

Effect of different text prompt settings. In this section, we analyze the behavior of our model with several possible text prompts for the sketch tasks. We consider (i) the given prompt (ii) default prompt "A professional, detailed, high-quality image", similar to [48] (iii) empty string as a prompt. In Figure A.2 we can observe that our model is able to generate images capturing the visual cues from the context example, across all text prompts. Note that having the text prompt mentioning the name of the object in the query image helps in generating finer details (like a more detailed surface of the pumpkin), however, even without it, our model is able to generate reasonable images.

Source-target vs target-only as context. In this section, we analyze the performance of our model when trained using source-target image pairs as context examples (same as

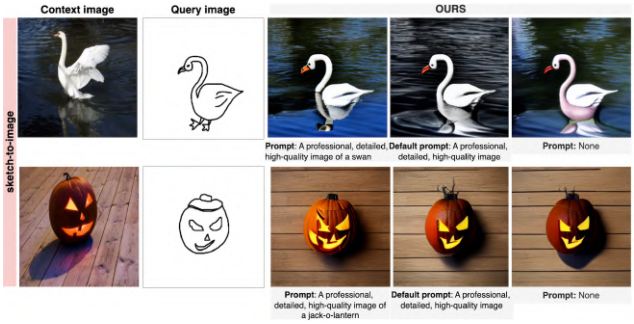

Figure A.2. **Ablation: Effect of different text prompt settings.** Our model succeeds in all three scenarios while showing it is able to capture visual characteristics from the context.

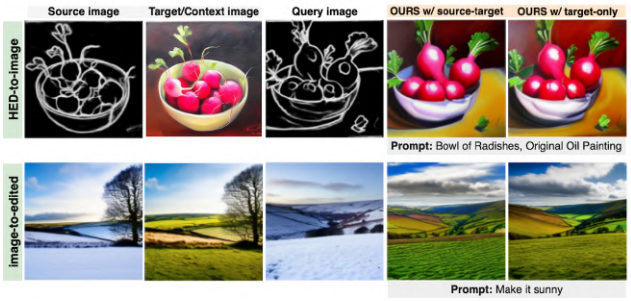

Figure A.3. **Ablation: Source-target vs target-only as context.** Adding the source image does not influence the generation process.

Prompt Diffusion [43]. As can be seen in Figure A.3, there is almost no difference when the source image is added to the training. The condition for generating the output is entirely contained in the target *i.e.* context image or the prompt, while the query image controls the structure.

B. Architecture details & comparison

In this section we compare the architecture of our model and Prompt Diffusion [43]. As can be seen from Figure B.1, the difference is in the visual conditioning using the context examples. We propose to stack the visual embeddings of the context examples - $\mathbf{h}^{\mathbf{V}}$ next to the text embeddings $\mathbf{h}^{\mathbf{c}}$. In this manner, the model is able to balance textual and visual conditioning. Moreover, it learns how to handle the structure of $\mathbf{q}$ separately from the context examples. This is different from Prompt Diffusion [43] which directly sums the examples embeddings to the query image.

Another difference is in the context examples. Different from Prompt Diffusion, we do not provide a source image, since it can be derived from the target (context) image it-

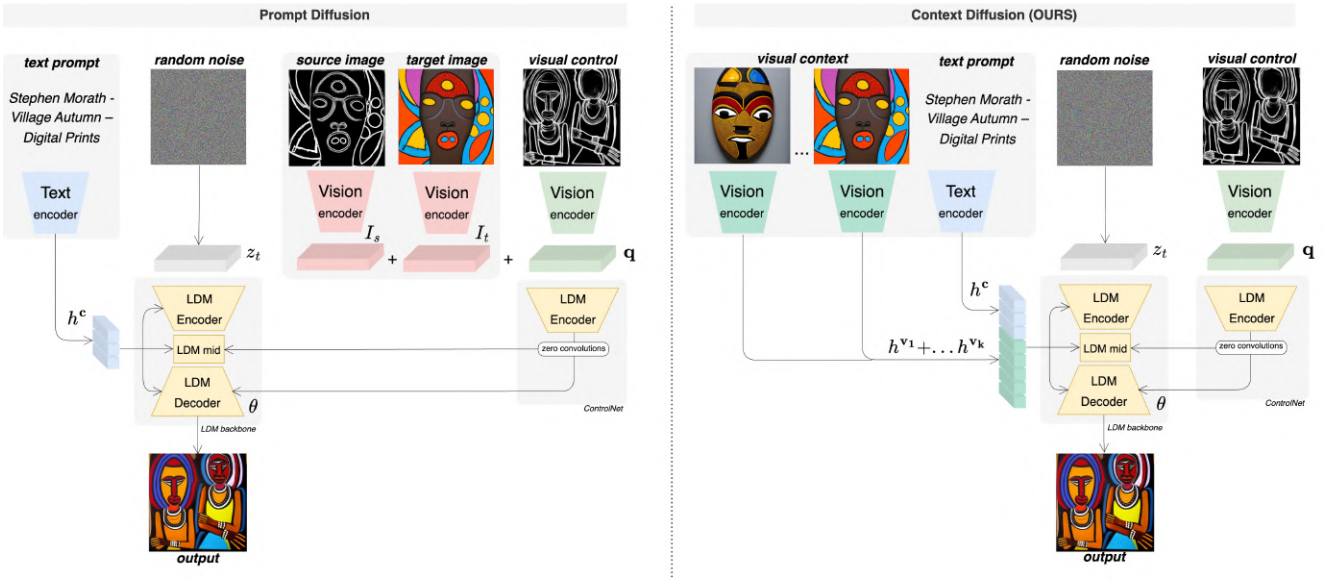

Figure B.1. **Comparison between our model architecture (right) and Prompt Diffusion (left).** We propose to use k-context examples via visual conditioning which allows the model to learn the visual characteristics of the context separated from the structure of the image. On the other hand, Prompt Diffusion is summing the examples directly to the visual control *i.e.* the query image.

self, meaning it does not provide any additional information. Note that in our early experiments, we did use a pair of source and target images, however, it showed to not bring any improvements. Furthermore, the ControlNet framework [48] is capable of controlling image generation based on image structures, making the source image unnecessary. We also provide the flexibility to include more than one context example, in order to learn stronger visual representations as conditioning, as shown in Figure B.1. Furthermore, in Figure B.2 we include the pseudo-code of our implementation showing how the cross-attention block is modified by using multiple images in the visual context.

C. Limitations

In the scenario where both the visual context and prompt are present, the current design assumes that the examples in the context are representative of the prompt. These embeddings form a stronger representation of the conditioning during the generation process. However, to build an even more flexible architecture, the visual context and prompt should ideally provide complementary information. Another limitation is the generation of images containing fine-grained details indicated in the text prompts or in the visual context. For instance, image editing is such a challenging task, especially for finer, local edits, as shown in Figure D.16.

Ethical considerations Our model is built using pre-trained models, both for the visual and textual conditioning as well as for the image generation process. This means that it inherits any biases and limitations that may exist in these pre-trained models. Therefore, a careful analysis of the risks and societal implications should be considered before building any real-world application.

```
# vision_encoder - CLIP ViT-L/14 Vision Encoder
# text_encoder - CLIP ViT-L/14 Text Encoder
# V[bs, k, H, W, C] - batch of preprocessed images
# c[bs, L] - batch of tokenized prompts
# z_t - noisy sample at time step t

# extract embeddings of the last encoder layers for each modality
h^V = vision_encoder(V).last_hidden_state   #[bs, k, N^V, d^V]
h^c = text_encoder(c).last_hidden_state     #[bs, N^c, d^c]

# sum the k-visual embeddings
h^V = torch.sum(h^V, dim=1)   #[bs, N^V, d^V]
h^V = linear_to_dc(h^V)

# stack the embeddings of both modalities
context = torch.cat((h^V, h^c), dim=1)    #[bs, N^V+N^c, d^c]

z_t = attention(norm(z_t)) + z_t   # self-attention
z_t = attention(norm(z_t), context) + z_t   # modified cross-attention
z_t = linear(norm(z_t)) + z_t
```

Figure B.2. Pseudo-code for a torch-like implementation of the modified cross-attention block in our model, by using k-images as visual context examples.

D. Additional qualitative results

In the following sections, we provide additional qualitative results, spanning from in-domain tasks, such as handling HED, segmentation, and depth maps to out-of-domain tasks, such as editing and sketches, as well as examples of few-shot settings.

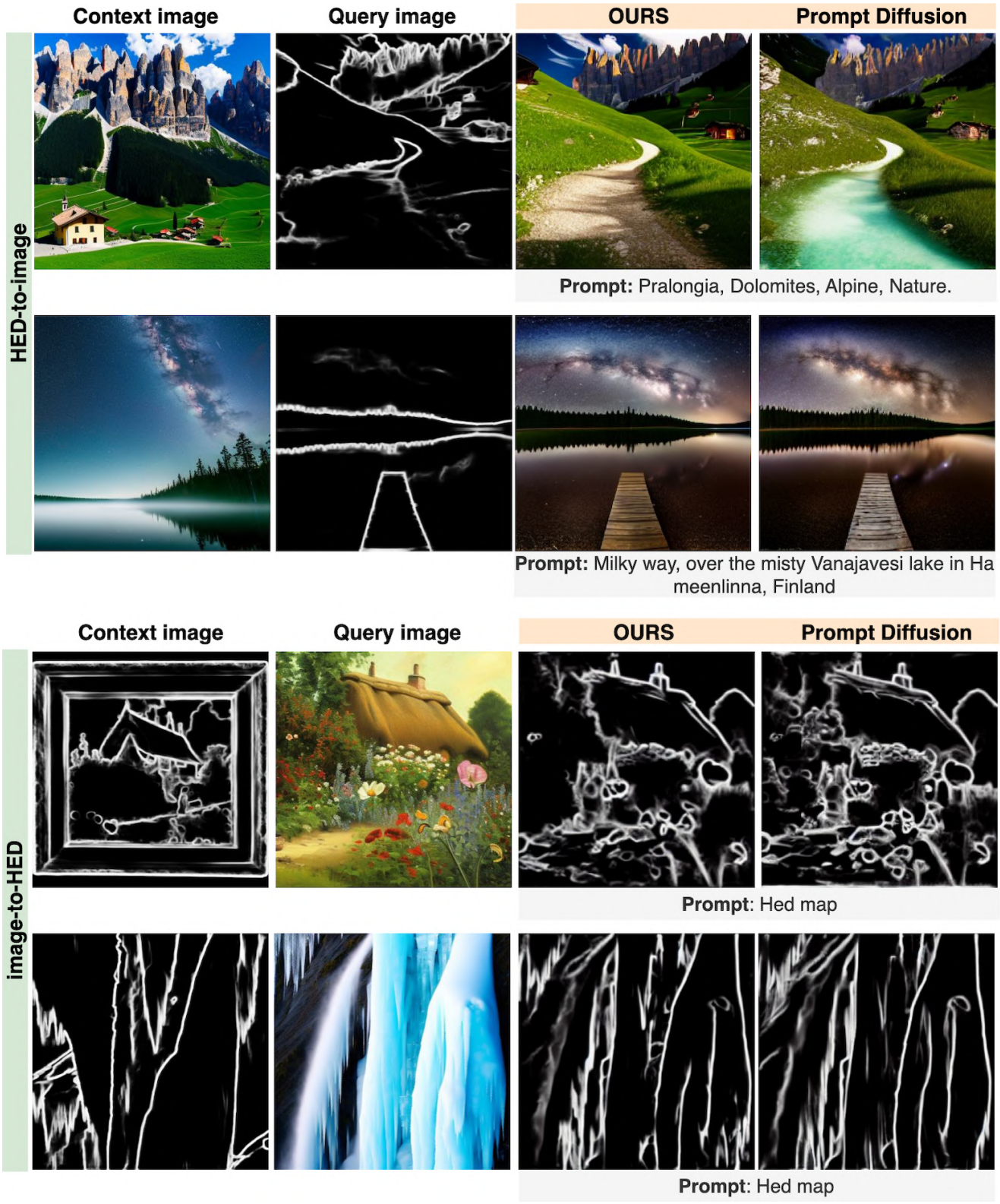

Figure D.1. HED-to-image and vice versa, with visual context and prompt as conditioning, in-domain comparison to Prompt Diffusion [43].

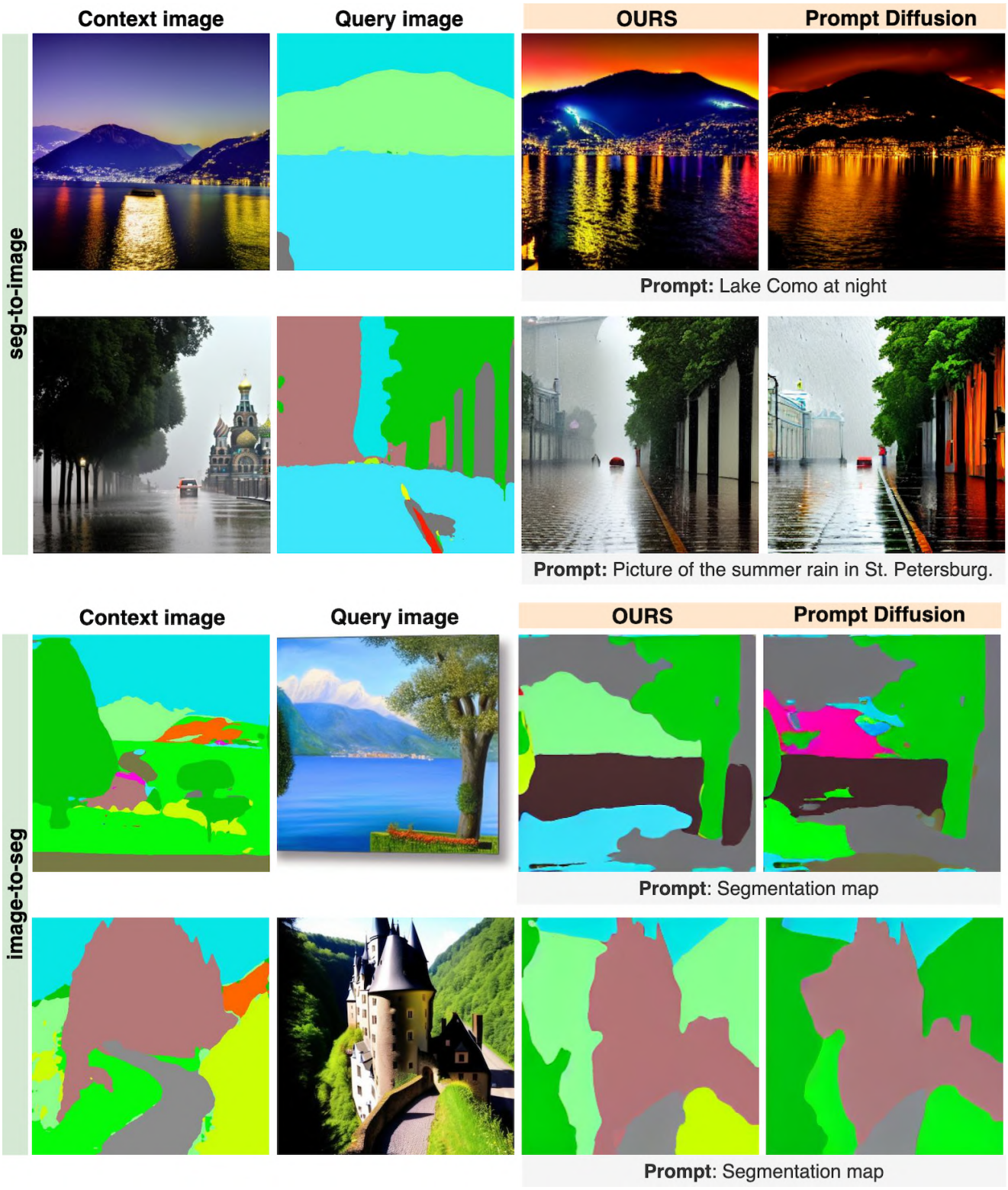

Figure D.2. Seg-to-image and vice versa, with visual context and prompt as conditioning, in-domain comparison to Prompt Diffusion [43]

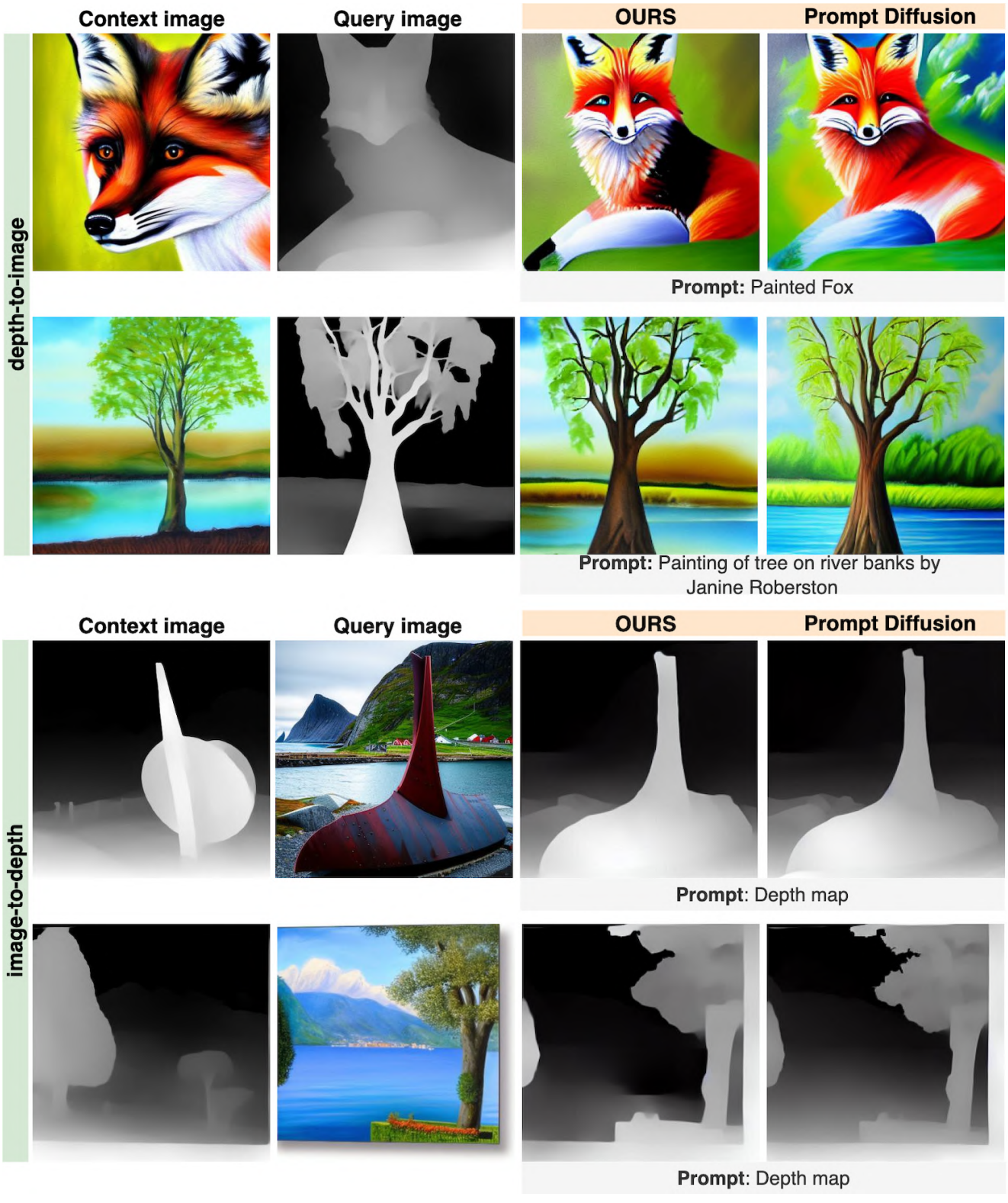

Figure D.3. Depth-to-image and vice versa, with visual context and prompt as conditioning, in-domain comparison to Prompt Diffusion [43].

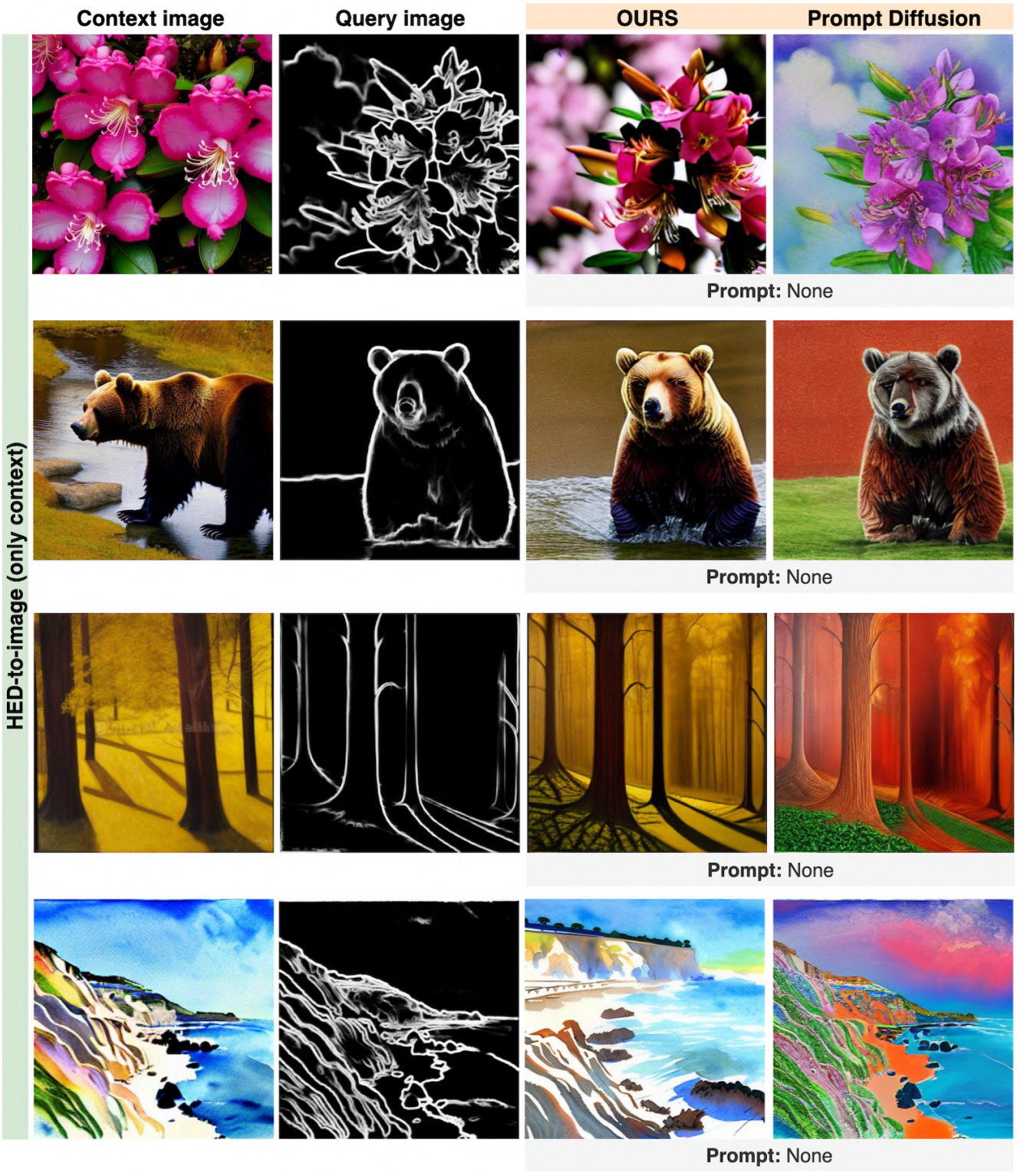

Figure D.4. HED-to-image, only with context images as conditioning, in-domain comparison to Prompt Diffusion [43].

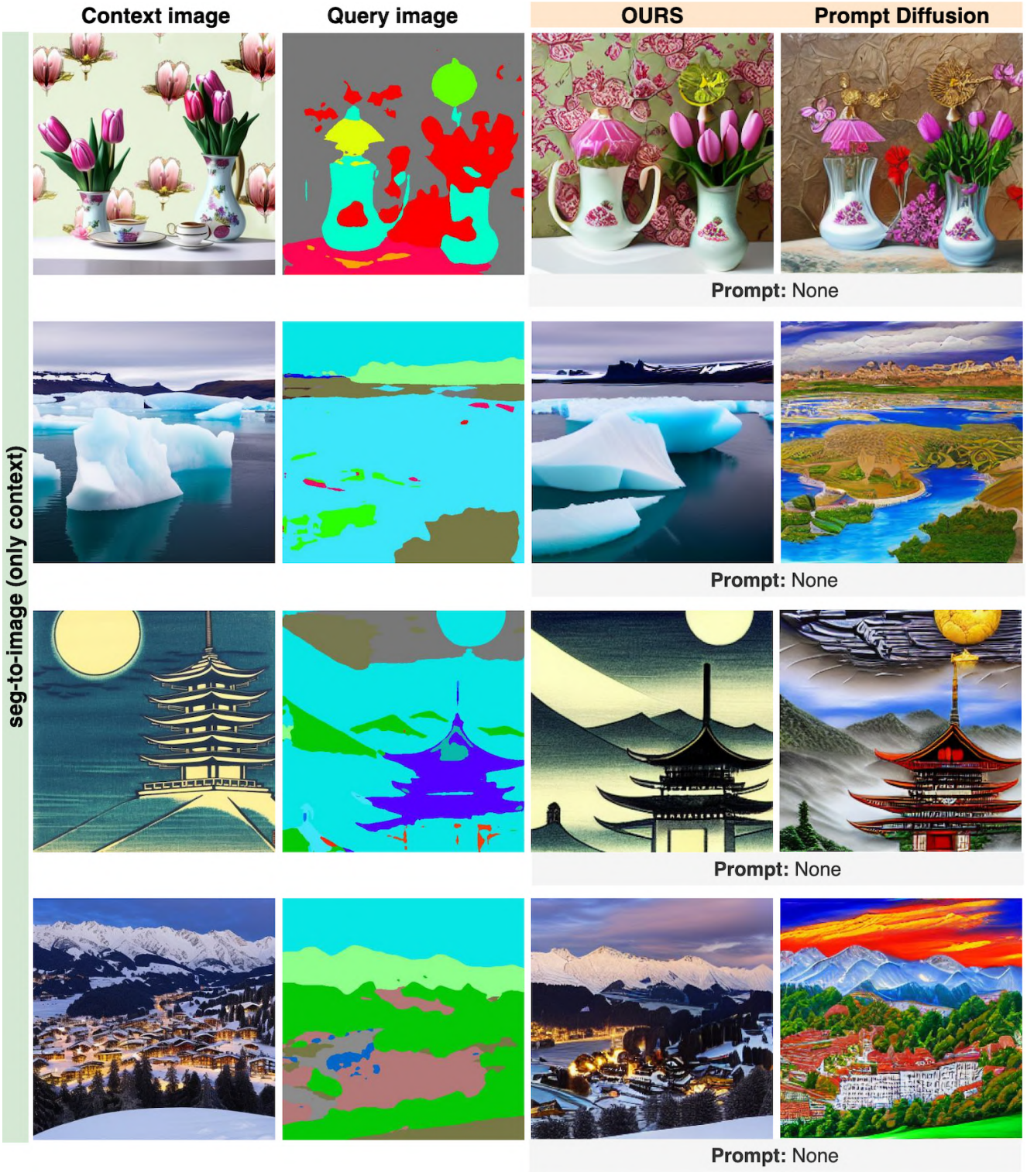

Figure D.5. Seg-to-image, only with context images as conditioning, in-domain comparison to Prompt Diffusion [43].

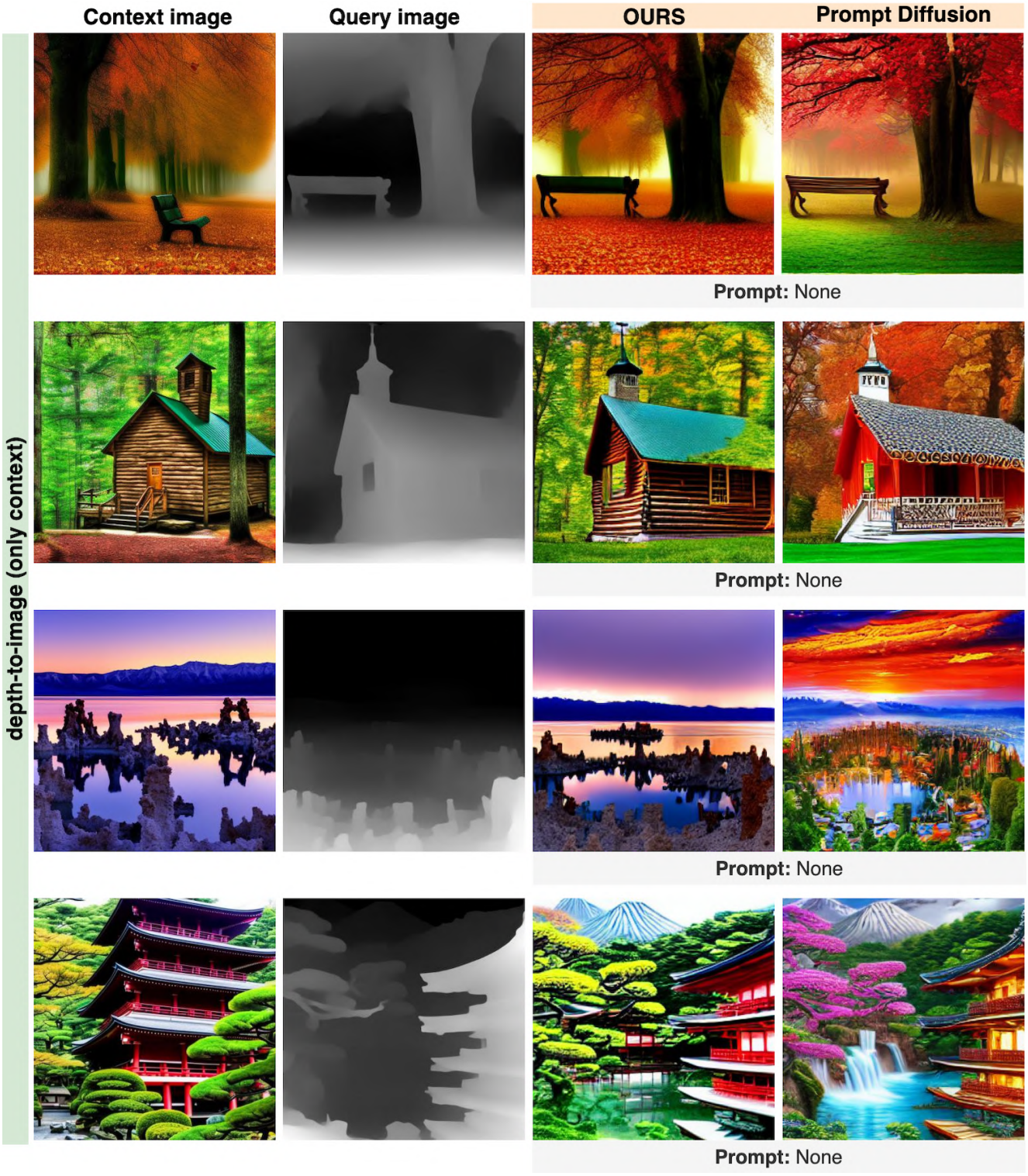

Figure D.6. Depth-to-image, only with context images as conditioning, in-domain comparison to Prompt Diffusion [43].

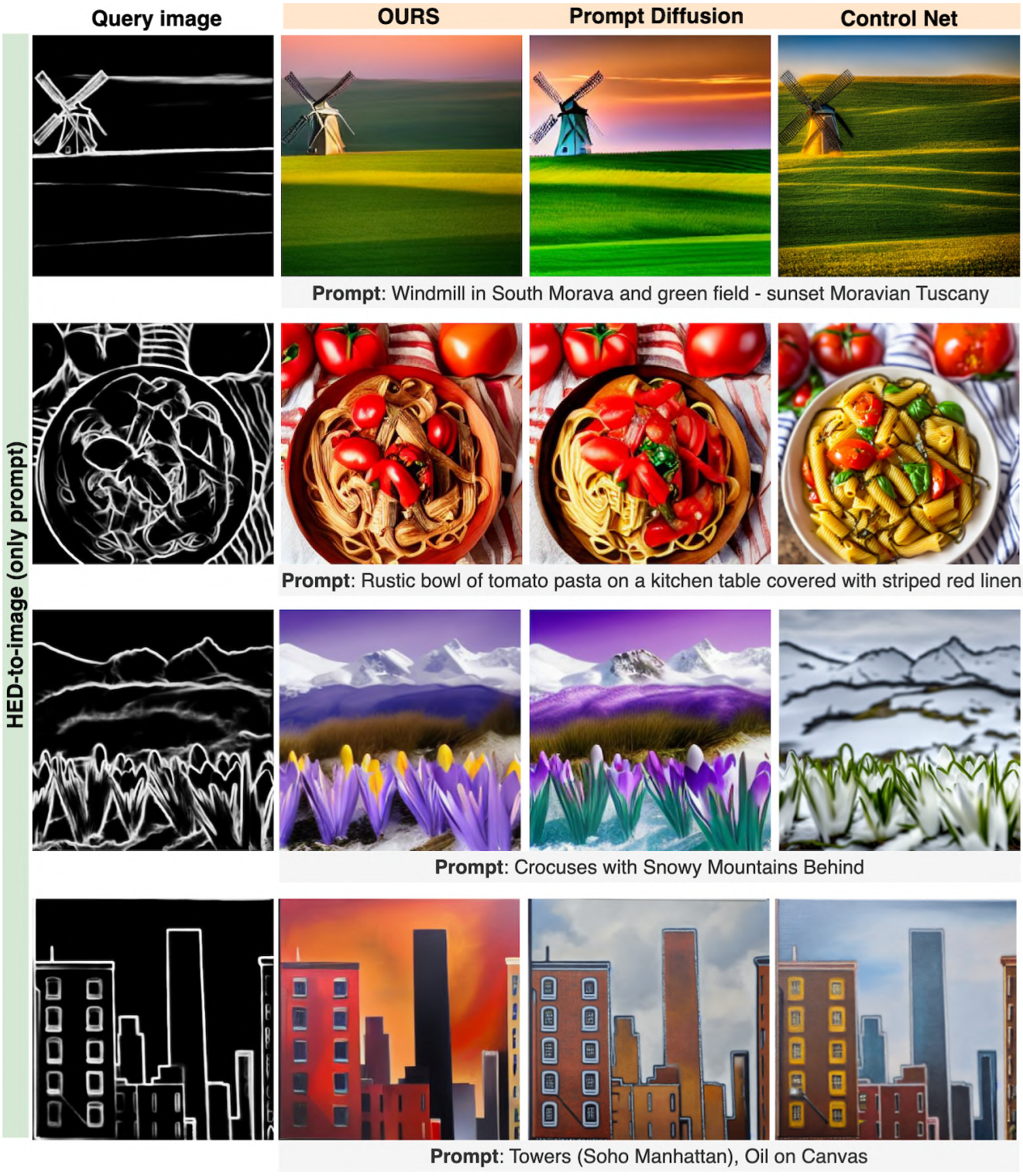

Figure D.7. HED-to-image, only with text prompt as conditioning, in-domain comparison to Prompt Diffusion [43] and Control Net [48].

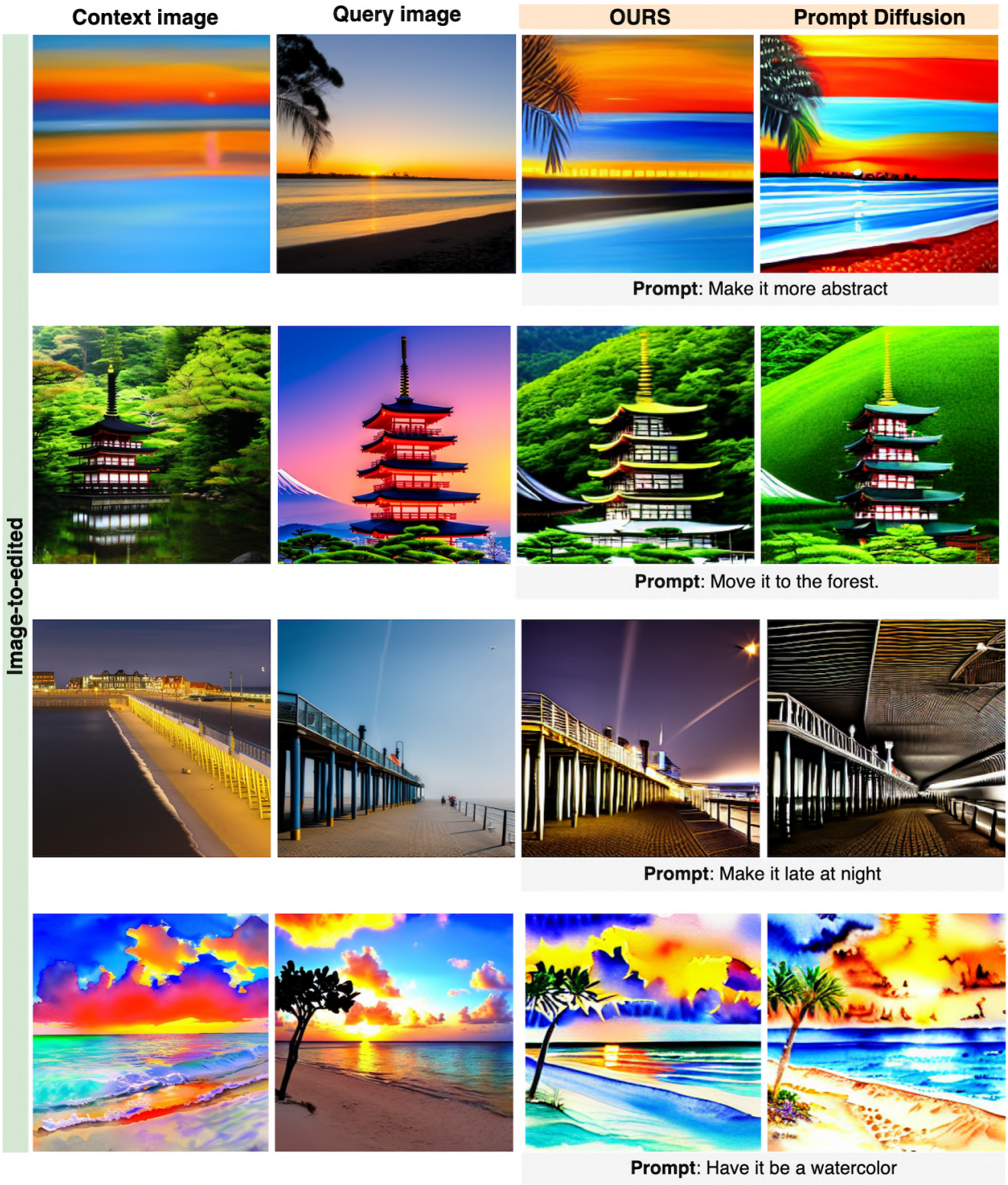

Figure D.8. Image edit, with visual context and prompt as conditioning, out-of-domain comparison to Prompt Diffusion [43].

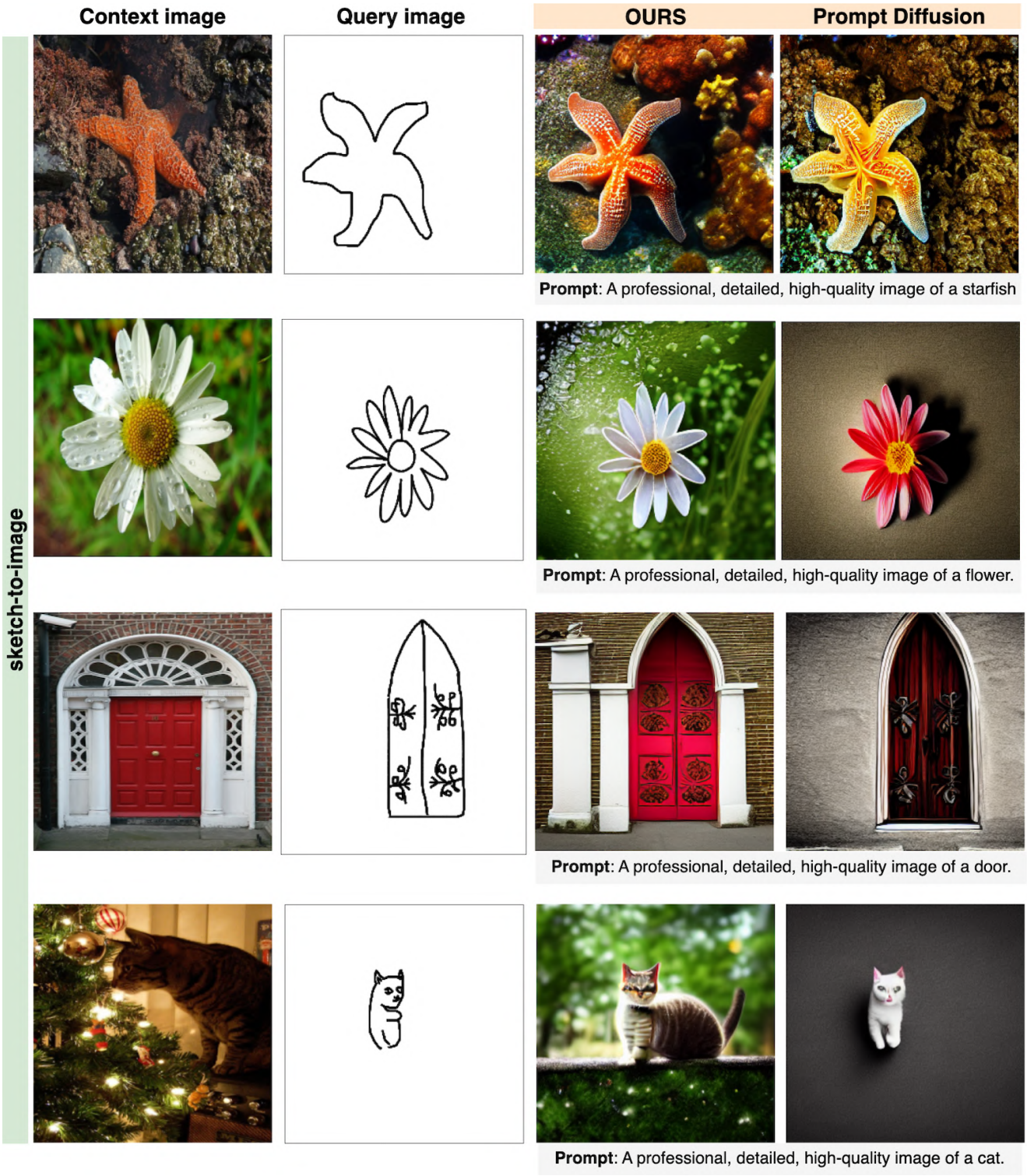

Figure D.9. Sketch-to-image, with visual context and prompt as conditioning, out-of-domain comparison to Prompt Diffusion [43].

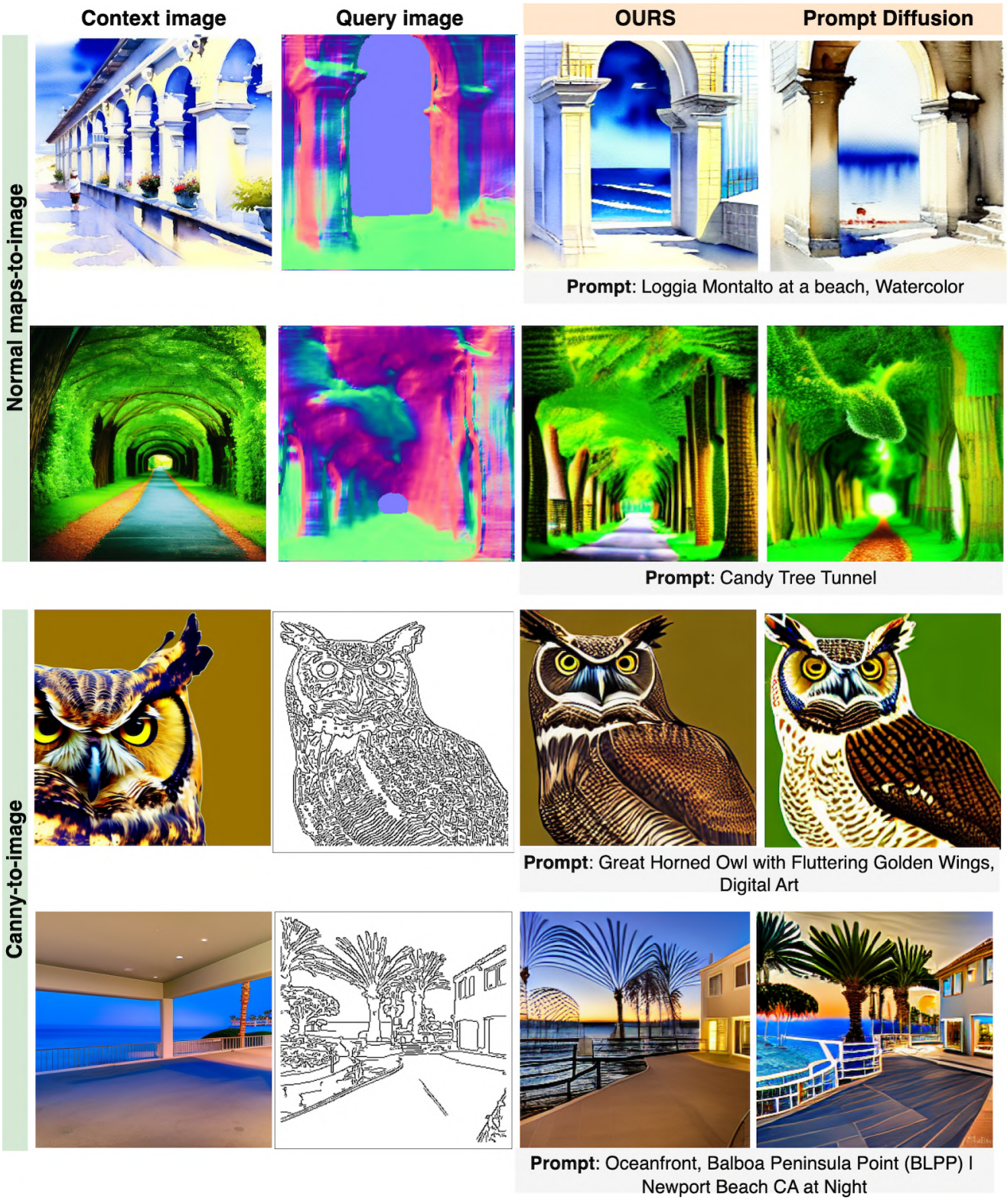

Figure D.10. Normal map-to-image (rows 1-2) and canny-to-image (rows 3-4), both with visual context and prompt as conditioning, out-of-domain comparison to Prompt Diffusion [43].

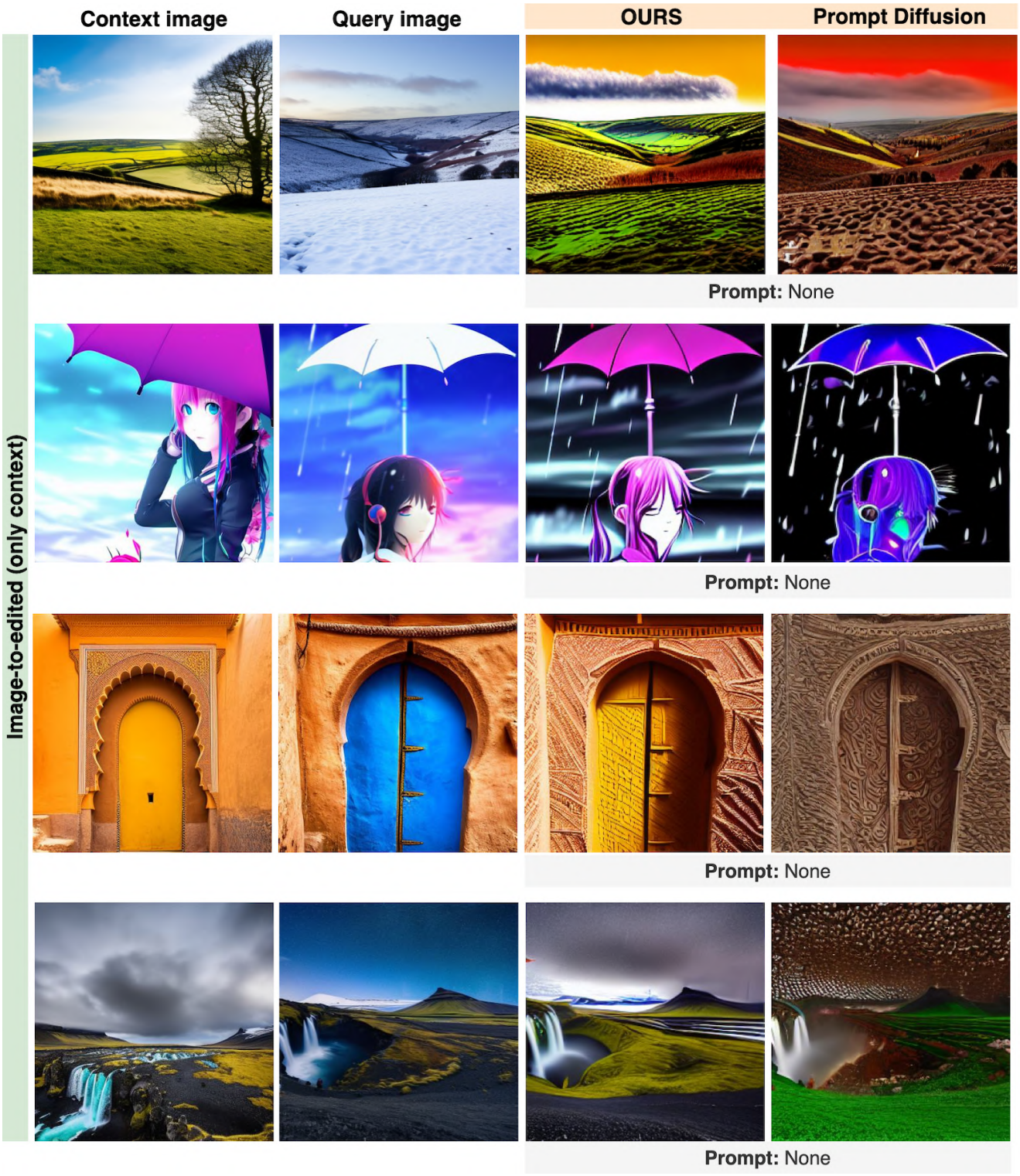

Figure D.11. Image editing, only with context images as conditioning, out-of-domain comparison to Prompt Diffusion [43].

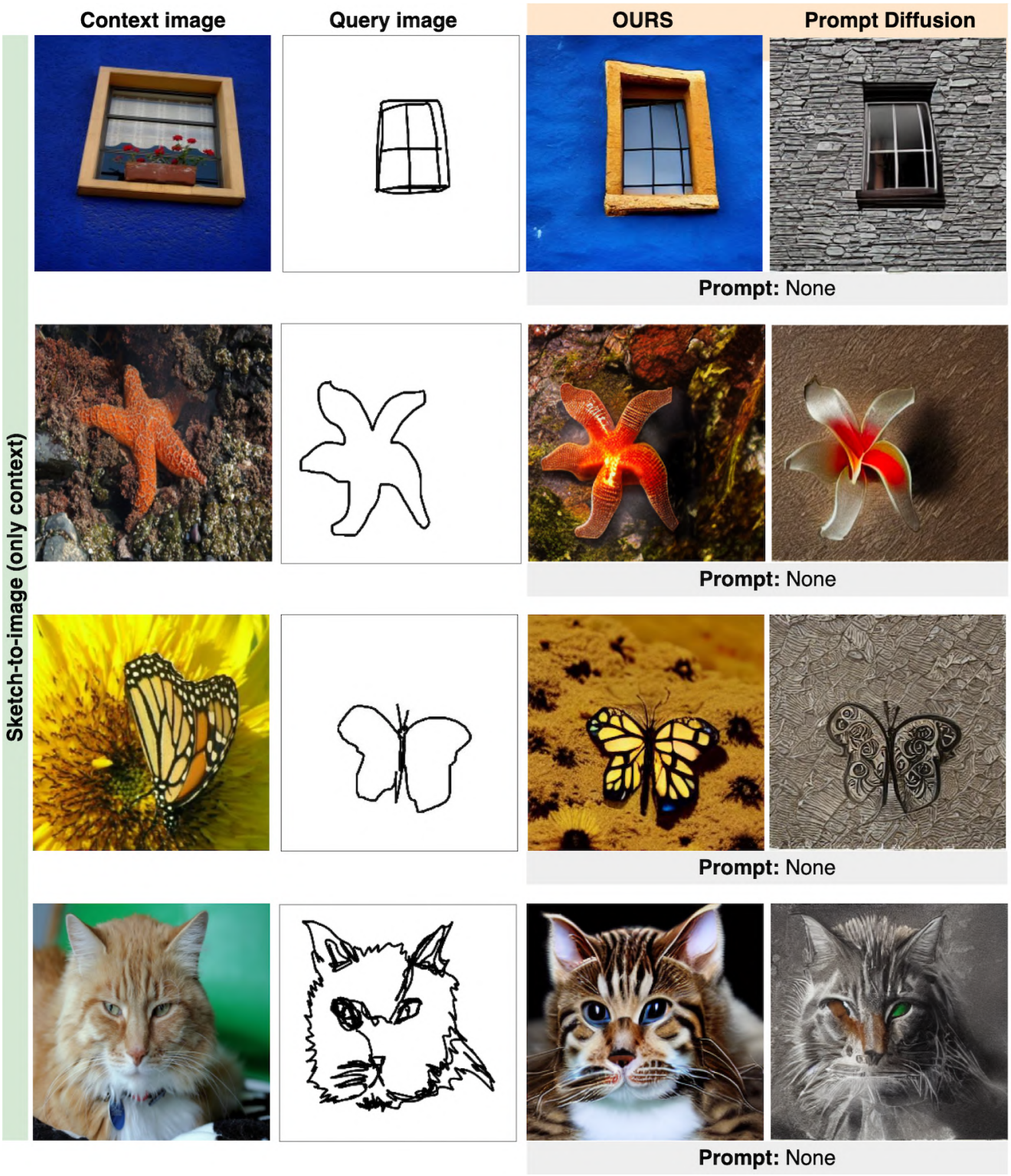

Figure D.12. Sketch-to-image, only with context images as conditioning, out-of-domain comparison to Prompt Diffusion [43].

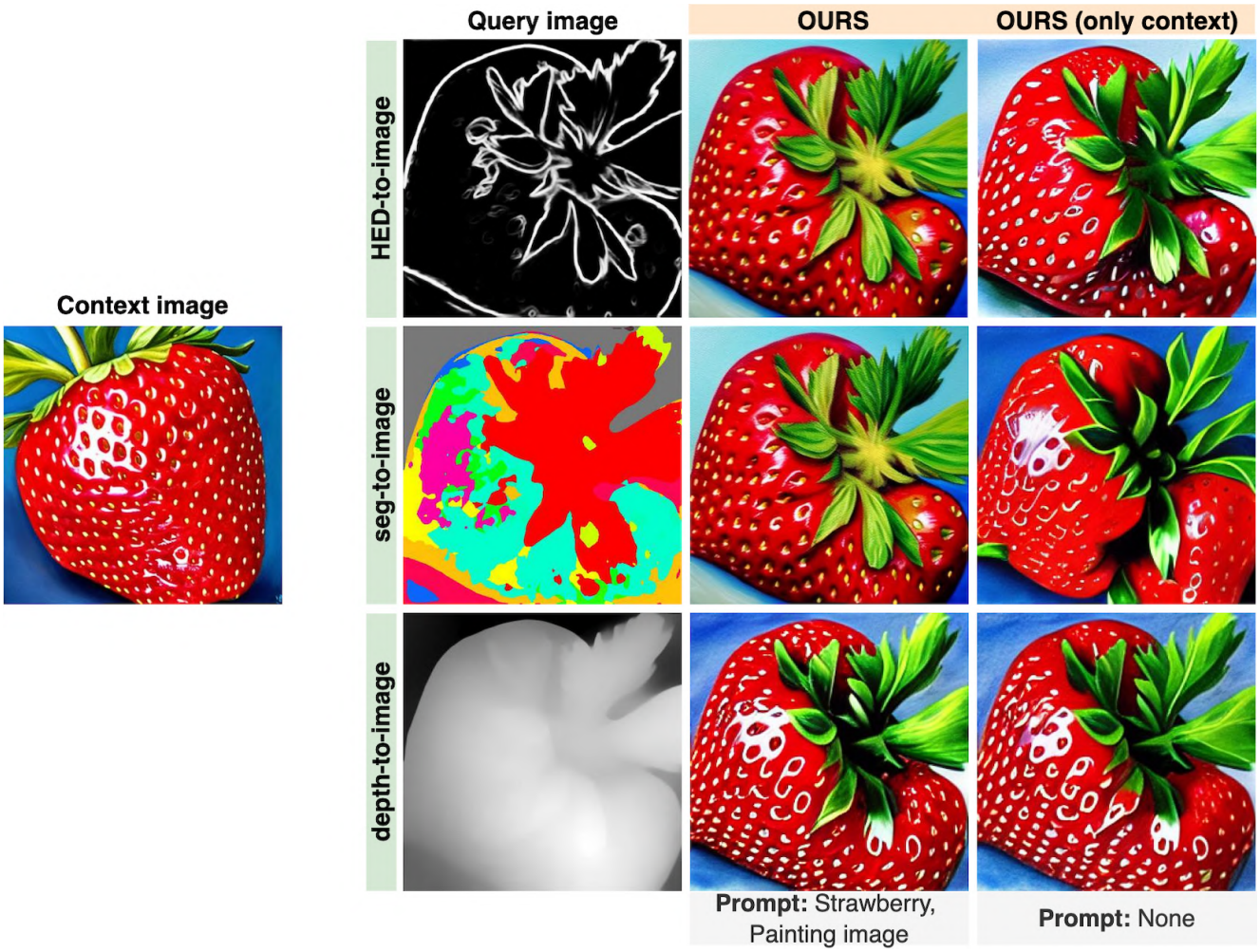

Figure D.13. Examples using different types of query images, with visual context and prompt, and only using context images as conditioning. Our model can successfully generate images that match the visual context and/or prompt, independent of the type of visual control provided by the query image.

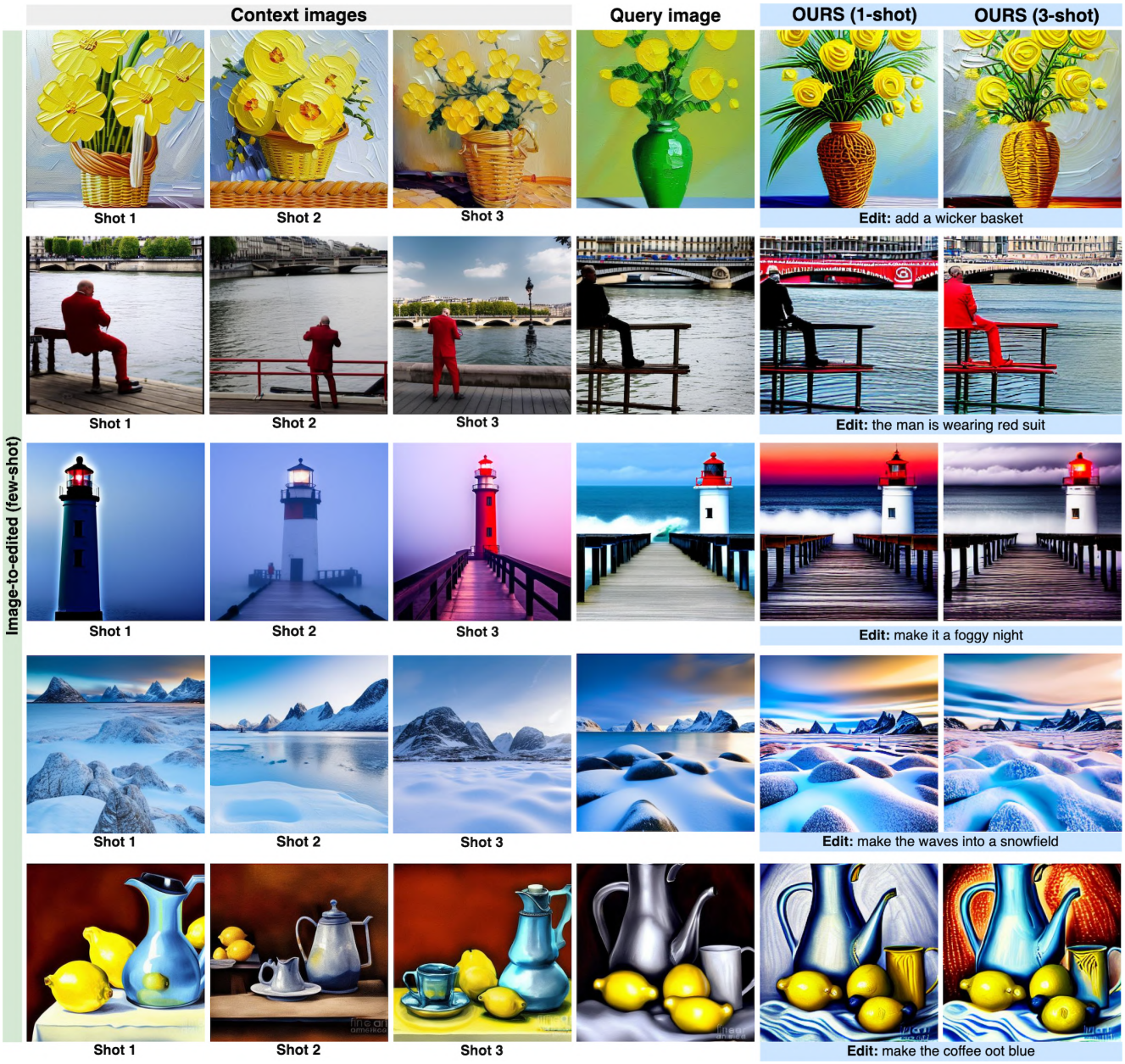

Figure D.14. Few-shot examples: image edit. Comparison using one, two, and three shots of context examples with text prompts.

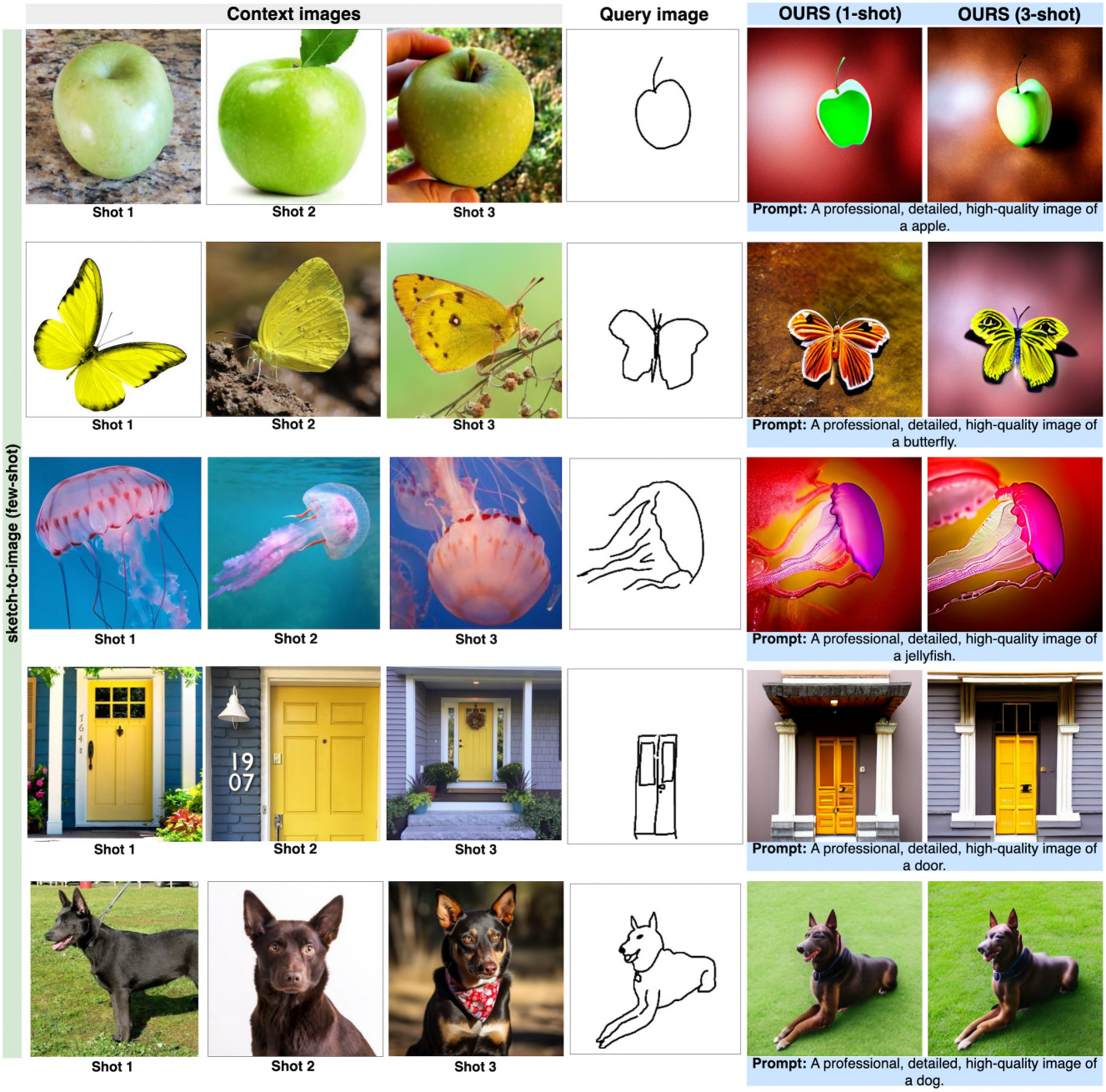

Figure D.15. Few-shot examples: sketch-to-image. Comparison by using one, two, and three shots of context examples with text prompts.

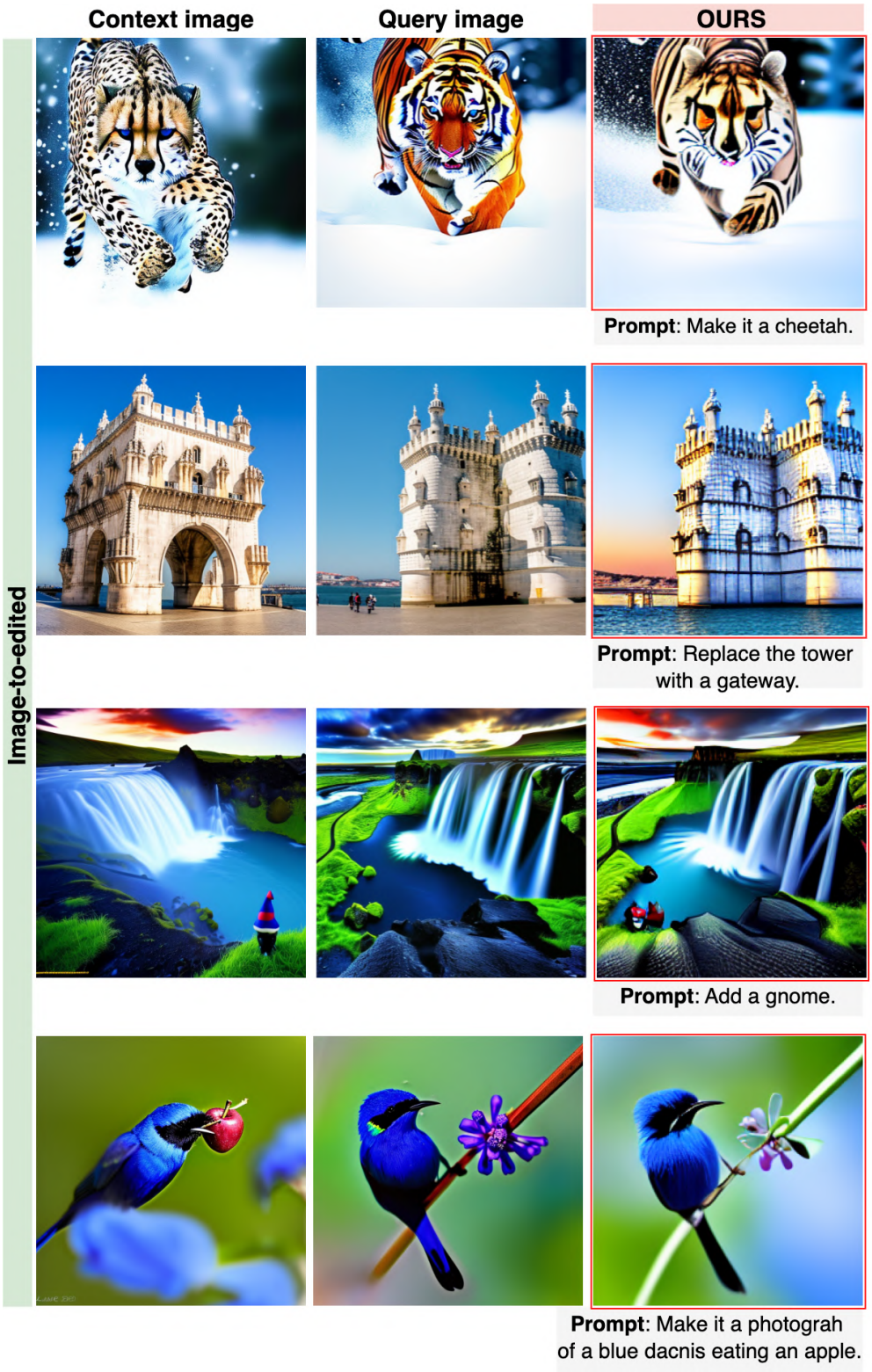

Figure D.16. Failure examples on editing tasks (local edits), using visual context and prompt as conditioning.