Language-Free Generative Editing from One Visual Example

Supplementary Material

Omar Elezabi Eduard Zamfir Zongwei Wu* Radu Timofte
Computer Vision Lab, CAIDAS & IFI, University of Würzburg

In the supplementary material, we first provide further details of the compared prior works in Sec. A. Additional ablations and analyses on out-of-distribution tasks are presented in Sec. B, and Sec. E examines the sensitivity of VDC to its hyperparameters. We then analyze the computational complexity of VDC in Sec. C. Finally, Sec. F discusses the limitations of our approach, and Sec. G includes extended visual comparisons for all compared methods.

A. Further Details on Prior Works

We provide additional details on prior works used for comparison, highlighting their reliance on language.

Text-Prompt Editing Methods (T-Edit). These methods require a complete text description of the input image. We use this description as the text prompt to condition both the inversion and generation processes. To enable each edit, we manually append the visual attribute corresponding to the target task. Captions are generated using BLIP [8].

The text prompt for each degradation type is constructed as follows: *(i)* Rain: "[Text Description] + in the rain", *(ii)* Fog: "[Text Description] + in the fog", *(iii)* SR: "Low-resolution image of [Text Description]", *(iv)* Blur: "Blurry image of [Text Description]", *(v)* Noise: "Noisy image of [Text Description]", *(vi)* Colorization: "Grayscale image of [Text Description]"

- **Prompt-to-Prompt (P2P)** [6]: Manipulates cross-attention during generation to adjust visual features associated with specific prompt words. For our tests, we mask cross-attention features tied to the degradation being removed (e.g., rain, fog, noise).
- **Null-Text Optimization (Null-Opt)** [14]: Improves DDIM inversion for image editing. We apply this optimization jointly with P2P for all edits.
- **Negative Condition** [13]: Replaces standard null-text conditioning in classifier-free guidance with negative prompts describing the unwanted degradation (e.g., "fog, foggy, haze, hazy, blurry, blur" for dehazing; "noise, noisy, low quality" for denoising).

*Corresponding Author

Text-Instruction Editing (I-Edit.) These methods take an input image and a natural-language instruction describing the desired edit. Each model is trained or fine-tuned to apply edits according to the instruction. We use the default configurations provided in the authors' open-source implementations.

For text instruction, we used: for DeRain "Remove rain and water drops from the image", for DeHaze "Remove fog and haze from the image", for SR "Increase image resolution, improve quality and remove noise", for DeBlur "Increase image sharpness, improve quality and remove noise" for DeNoise "Remove noise from the image", for Colorization "Color this grayscale image".

Zero-Shot Image Restoration (Zero-IR). Zero-IR methods solve inverse problems using diffusion models as strong generative priors. They require a degradation kernel that models the corruption in the input image. These methods search the diffusion latent space for an image that degrades to an image that matches the input. We use the released code and default task-specific settings for each method. For colorization, we adopt the kernel settings from Zero-Null [20]. For the remaining tasks, we use the kernels defined in DPS [3].

Image Exemplar-based Editing (IE-Edit). These methods infer an edit from a before/after image pair and apply it to a new image. For fair comparison, we use the same reference example images employed to optimize our method.

- **VISII** [15]: Builds on the text-instruction editing model Instruct-Pix2Pix [2]. It optimizes a text instruction that reproduces the edit shown in the example pair, then applies the resulting instruction to new inputs.
- **Analogist** [5]: Uses Stable Diffusion Inpainting [17] together with a large language model that extracts the transformation between example images. It then applies this transformation to new inputs via inpainting.
- **EditClip** [19]: Fine-tunes the CLIP image encoder [16] to capture relationships between the example images. It further fine-tunes Stable Diffusion [17] to condition on these relationships. The model is trained on hundreds of thousands of edited images paired with text instructions.

Table 4. *VDC compared to fine-tuning and with different generative models.* FID (↓) and LPIPS(↓) are reported on the full RGB images. Our method highly surpasses diffusion fine-tuning methods in low data regime. Additionally, VDC can be utilized with different conditional generative models. The **best** performances are highlighted.

Type	Method	Num. Samples	SR		DeBlur		DeNoise		DeRain		DeHaze		Colorization	
			FID ↓	LPIPS ↓	FID ↓	LPIPS ↓	FID ↓	LPIPS ↓	FID ↓	LPIPS ↓	FID ↓	LPIPS ↓	FID ↓	LPIPS ↓
F-T	ControlNet [25]	200	110.62	0.4291	80.03	0.3225	145.01	0.4777	153.29	0.4803	50.00	0.2750	143.50	0.4093
	PairEdit [12]	8	98.68	0.3134	67.08	0.3959	168.68	0.4561	148.59	0.3924	85.69	0.3924	150.32	0.5089
SD	One-Shot	1	**41.41**	0.2666	35.51	0.2654	89.51	0.2801	87.12	0.2559	35.52	0.1633	107.70	0.2908
	Multi-Shot	8	45.89	0.2654	42.62	0.2651	88.58	0.2846	69.52	0.2214	34.18	0.1584	107.80	0.2744
	MS+Inv-Correc	8	45.00	**0.2624**	41.09	**0.2593**	**82.57**	**0.2768**	**66.92**	**0.2155**	**33.23**	**0.1560**	**105.26**	**0.2729**
SANA	One-Shot	1	50.20	0.2900	40.33	0.2587	82.72	0.2510	**93.61**	0.24807	29.20	0.1414	107.74	0.26254
	Multi-Shot	8	48.25	**0.2478**	33.99	0.2140	73.57	0.2485	98.80	**0.2432**	29.46	0.1403	**104.97**	**0.2596**
	MS+Inv-Correc	8	**45.81**	0.24834	**32.38**	**0.2134**	**69.65**	**0.24816**	97.54	0.2446	**28.70**	**0.1398**	105.13	0.2603

Table 5. *OOD Generalization.* We compare our method to state-of-the-art All-in-One Image Restoration (IR) on real image De-Rain. We utilize RealRain-1k-L [10] dataset for testing. Our method is able to generalize to real data while prior works fail. **Best** results are highlighted.

Methods	Instruct-IR [4]	MoCE-IR [23]	VDC (*ours*)
FID ↓	124.82	154.94	**106.89**
LPIPS ↓	0.2553	0.3646	**0.2154**

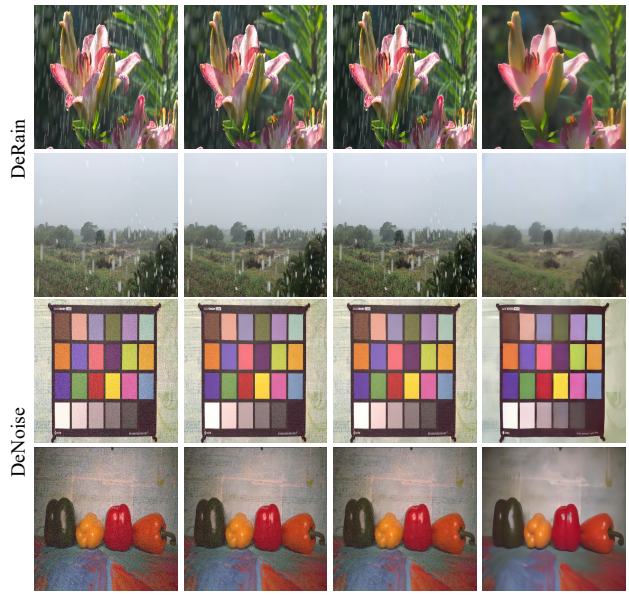

Figure 9. *Visual comparison for OOD samples.* We compare our method to SOTA restoration models on real data. Our method is able to work on real data, whereas IR methods trained on syntactic data fail to generalize. We utilize RealRain-1k-L [10] for Derain, and SIDD [1] for denoising.

B. Ablations on Generalization

B.1. VDC is model-agnostic

To demonstrate that our method is not tied to a specific generative model and can generalize to any conditional generative framework, we adapt the SANA model [22]—a conditional generative model based on Flow Matching [11]—for image editing using VDC. As shown in Tab. 4, VDC successfully enables SANA to perform image editing and restoration, and even surpasses the Stable Diffusion (SD)–based version on several tasks (DeNoise and De-Haze), benefiting from SANA's more advanced generative prior. These improvements stem from SANA's stronger prior, which yields higher-quality reconstructions of the inverted image. However, SANA's latent encoder applies a significantly higher compression rate (8× for SD versus 32× for SANA), which can limit the preservation of fine details in the latent space—an effect visible in the DeRain results. Our inversion correction module is likewise model-agnostic and can be used with Flow Matching models. As shown, it consistently improves performance, particularly on tasks that rely heavily on detail preservation. Finally, the visual comparisons in Figs. 14–19 further illustrate that VDC successfully adapts SANA for high-quality image editing and restoration.

B.2. VDC improves over Fine-Tuning

Fine-tuning (F-T) and diffusion adaptation methods like ControlNet [25] rely on massive supervision. Tab. 4 shows that fine-tuning fails in the low-data regime: ControlNet [25] trained on 200 samples suffers from severe domain shift. Even a few-shot fine-tuning method like PairEdit [12], based on LoRA and fine-tuned on 8 samples, yields poor fidelity. Additionally, it requires optimizing a new content LoRA for each inference image, requiring around 20-30 minutes of inference time. In contrast, VDC achieves strong results using only a single example and with zero inference overhead.

B.3. Out-of-distribution performance

A key advantage of adapting generative models for editing is the ability to leverage their real-data generative priors, enabling strong generalization to out-of-distribution inputs. In our framework, the generative model itself performs the edit—VDC simply provides a mechanism to communicate the desired transformation. In contrast, task-specific

restoration or editing models learn directly from training data, making their performance heavily dependent on the distribution and realism of that data. As a result, models trained on synthetic degradations often struggle to generalize to real-world scenarios. Despite using only synthetic examples to optimize the steering condition, our method generalizes effectively to real data. As shown in Tab. 5, VDC achieves strong real-world DeRain performance using just eight synthetic examples, successfully handling rain patterns that differ substantially from those in the examples. Meanwhile, specialized restoration models fail to generalize even when trained on large-scale synthetic datasets.

As shown in Fig. 9, the gap between synthetic and real rain patterns causes traditional image restoration methods to fail at detecting and removing real rain streaks. In contrast, our approach leverages the generative model's priors to correctly identify and remove these streaks, resulting in accurate edits. A similar trend is observed on real-world denoising data, where our method continues to generalize effectively while baseline restoration methods struggle.

B.4. Performance on general editing tasks

We center our benchmark on fine-detail edits, global adjustments, and image restoration tasks—categories where existing methods often struggle due to visual–text misalignment. Nonetheless, our approach is a general editing framework: it extracts the transformation from a given example and applies it to a new input.

As illustrated in Fig. 10, by simply increasing diffusion path length (60%), our method supports a wide range of edits, including semantic and object-specific modifications, compared to EditCLIP [19], which is trained for visual-instruction–guided editing. Our method more reliably interprets the edits present in the example pair, particularly for global adjustments. EditCLIP may introduce unintended artifacts as its behavior is influenced by CLIP representation abilities and common patterns in its large training corpus.

Semantic & **Non-Rigid Edits.** We clarify that VDC targets visual attribute steering (e.g., restoration, stylization) where text is ambiguous. To ensure high fidelity, we rely on pixel-space losses, which inherently prioritize structural preservation over non-rigid flexibility (e.g., pose changes). However, VDC resolves this by supporting textual control: as shown in Fig. 11, VDC handles visual patterns (DeRain) while text drives semantic shifts (e.g., bears→cats) and non-rigid edits (e.g., closing eyes).

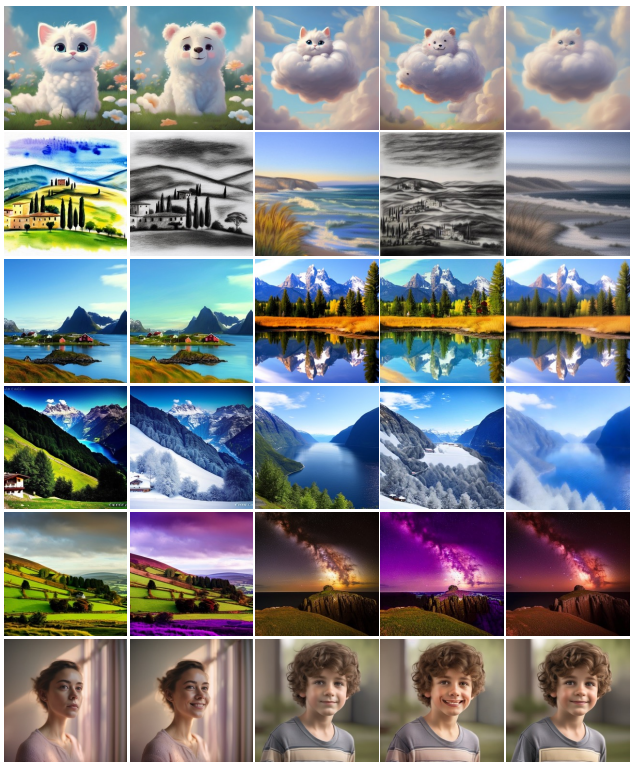

Figure 10. General Image Editing. We show the output of our method on general edits. Our method is not just limited to fine details or global edits but can also extend to semantic and object-specific edits. Images from TOP-Bench [27] Dataset.

Figure 11. *Composability of VDC.* VDC is used to steer the visual style while text independently controls the semantic content.

C. Complexity Analysis

As shown in Tab. 7, the complexity of other methods is largely determined by the inference requirements of their underlying generative models. Zero-IR methods, in particular, incur significantly higher cost due to their sampling-based search procedures. In contrast, by directly optimizing the steering condition for the chosen sampling path, our approach minimizes the number of required inference steps. This yields the highest efficiency among the compared methods, requiring only 20 total steps for editing (10 DDIM inversion steps and 10 sampling steps) while still achieving the best performance. When the inversion correction module is used, the total number of steps increases, but this module is optional and can be enabled based on the task or available computational resources.

Although our method is entirely train-free, it still requires optimizing the steering condition for each adapted task. This optimization consists of 200 full-path iterations (2000 diffusion steps) and needs to be performed only once per task. On an RTX 4090 GPU, this process takes

Table 6. *User Study.* This table represents the preferences of the participants in the user study from the compared methods' outputs across different aspects. We report the choice percentage averaged across participants. '-' represents unreported results. **Best** results are bolded.

Method	SR				DeRain			
	Perceptual %	Artifact-Free %	Preservation %	Overall %	Perceptual %	Artifact-Free %	Preservation %	Overall %
Negative-Cond [13]	1	3.5	6	1	**35**	23.5	20.5	18.5
OmniGen [21]	0.5	4.5	22	0.5	26.5	26.5	**41.5**	20
PSLD [18]	**67**	23.5	33.5	47.5	-	-	-	-
EditClip [19]	0	0	3.5	0	4.5	1	10.5	0
VDC	31.5	**68.5**	**35**	**51**	34	**47.5**	27.5	**61.5**

roughly 30 minutes. This is comparable to other train-free methods that rely on test-time optimization—such as Null-Opt [14] (500 diffusion steps) and VISII [15] (5000 diffusion steps)—but with the advantage that our optimization is performed per task rather than per inference. Overall, train-free approaches remain substantially more efficient than methods that require training or fine-tuning on hundreds of thousands of images across multiple GPUs for several days.

D. User Study

To better evaluate the tested methods, the mean opinion score (MOS) was calculated through a user study by asking participants to choose their favorite output. For a clear understanding and accurate evaluation of the underlying task, the chosen user study participants are 10 imaging experts, including professional photographers. The user study was conducted on 2 different tasks (SR and DeRain) on 20 samples from each task, selected randomly, and were fixed for all participants. We hide the names of the methods and randomly shuffle their position in the comparison grid to eliminate method bias. The comparison includes 5 different methods chosen by selecting the best overall performing method for its own type (Text Edit, Instruction Edit, etc). Our one-shot VDC version is the one included in the comparison. We asked participants to choose their favorite output for four different categories: Best Perceptual Quality, Least Artifacts, Best Content Preservation, and Best Overall for the task. MOS for all categories is represented in Tab. 6. As we see from the results, our method is the most preferred by the participants, with 51% and 61% choice as the preferred method in SR and DeRain, respectively. In SR PSLD [18] produces sharper images, which results in a higher perceptual quality; however, this method produces noticeable artifacts and noise (Fig. 14, 15), resulting in our method being chosen as the best for the task. For DeRain, we notice a similar trend as other methods tend to create artifacts and content changes to the input, resulting in our method still being preferred. Additionally, we can appreciate our method's consistency across different tasks and aspects.

E. Ablations on Hyperparameters

Performance Stability. Fig. 13 reports the variance across 10 models optimized on distinct reference pairs. Despite minor fluctuations in complex tasks (DeRain), performance remains robust regardless of the chosen example. This aligns with Fig. 8, which shows optimized conditions for the same task are closely similar regardless of the chosen visual example, proving reliable single-shot extraction.

Table 7. *Complexity Analysis.* NFEs ↓ are Neural Function Evaluations. Our method sets a new state-of-the-art while being the most inference-efficient. '-' represents unreported results. The **best** performances are highlighted.

Type	Method	Train-Free	NFEs	Deblur	DeRain
T-Edit	P2P [6]	✓	100	45.62	139.19
	Null-Opt [14]	✓	600	51.89	167.61
	Negative-Cond [13]	✓	100	43.61	96.19
I-Edit	Instruct-Pix2Pix [2]	×	100	142.91	179.93
	OmniGen [21]	×	50	46.18	119.87
	SuperEdit [9]	×	100	56.22	185.98
	ICEdit [26]	×	28	45.54	149.44
Zero-IR	PSLD [18]	✓	1000	42.89	-
	TReg[7]	✓	200	52.07	-
	DAPS[24]	✓	150	59.85	-
IE-Edit	VISII [15]	✓	40	122.63	203.83
	Analogist [5]	✓	50	75.06	158.29
	EditClip [19]	×	50	78.75	174.93
VDC	One-Shot	✓	**20**	**35.51**	87.12
	Multi-Shot	✓	**20**	42.62	69.52
	MS+Inverse-Correction	✓	220	41.09	**66.92**

Table 8. *Number of Visual Examples.* Increasing the number of visual examples can introduce more variety for a more robust optimized condition at the expense of increasing optimization time, which can affect the performance negatively. **Best** results are bolded; the final setup is highlighted.

Num Samples	SR		DeRain	
	FID ↓	LPIPS ↓	FID ↓	LPIPS ↓
1	**41.41**	0.2666	87.12	0.2559
4	46.24	0.2668	72.94	**0.2197**
8	45.89	**0.2654**	**69.52**	0.2214
16	47.30	0.2735	71.16	0.2227

Number of Visual Examples. Increasing the number of visual examples helps optimize a more robust steering condition, especially for tasks with complex and highly variable patterns such as deraining. As shown in Tab. 8, performance improves on the DeRain task as the number of examples increases. However, more examples also raise the optimiza-

Table 9. *Ablations on sampling steps and steering condition scale.* *(a)* Increasing DDIM sampling steps improves editability but also introduces more optimization constraints. *(b)* Increasing the steering condition scale allows stronger deviations from the generative path, enhancing edit strength at the cost of fidelity. **Best** results are bolded, and the final chosen configuration is highlighted.

(a) *DDIM Sampling Steps.*

Steps	SR		DeRain	
	FID ↓	LPIPS ↓	FID ↓	LPIPS ↓
50	45.43	0.2858	87.41	0.2566
100	**41.41**	**0.2666**	**87.12**	**0.2559**
200	49.79	0.2815	91.94	0.2598

(b) *Steering Condition Scale.*

Scale	SR		DeRain	
	FID ↓	LPIPS ↓	FID ↓	LPIPS ↓
5	47.17	0.2801	89.24	0.2612
7	**41.41**	**0.2666**	**87.12**	**0.2559**
9	45.73	0.2877	92.62	0.2733

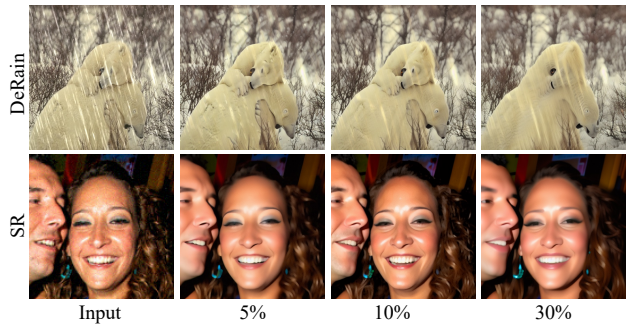

Figure 12. *Diffusion path length effect.* Extending the diffusion path increases variation, resulting in undesirable edits, while decreasing the path limits edibility.

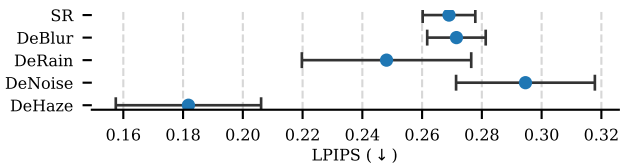

Figure 13. *Sensitivity analysis.* We assess sensitivity by optimizing 10 models on unique examples; reporting the variance per task.

tion burden. When additional examples do not introduce new visual patterns, the added complexity can negatively impact performance.

DDIM Sampling Steps. Using more DDIM sampling steps provides additional opportunities to apply edits, improving the method's editability. However, increasing the sampling length also expands the number of conditions that must be optimized, making optimization more difficult and potentially degrading results. This trade-off is evident in Tab. 9a.

Steering Condition Scale. A higher steering condition scale increases the allowed deviation from the unconditioned generative trajectory (i.e., deviation from the input), which boosts editing strength at the cost of fidelity. As shown in Tab. 9b, a small scale limits the model's ability to apply the desired edits, while an excessively large scale expands the output space too aggressively, reducing performance.

Diffusion path length effect. As discussed in Tab. 3, starting the sampling process deeper in the diffusion trajectory injects more noise into the latent, enlarging the output space but lowering fidelity. This effect is visible in Fig. 12, where a longer diffusion path introduces unwanted content changes. Conversely, using too short a path overly restricts the output space, preventing the model from reaching suitable solutions and resulting in suboptimal edits.

F. Limitations

Our method leverages strong generative priors to handle complex edits on real images, but its performance ultimately depends on the capabilities of the underlying generative model. Although we move beyond the limitations of text-based conditioning to operate entirely in the visual domain, our results still reflect the strengths and weaknesses of this visual latent space. As seen in Figs. 14–19, some fine textures may be lost due to the limited generative fidelity of Stable Diffusion [17], particularly when editing images processed through inversion. These limitations can be mitigated by adopting a more capable generative model, as demonstrated in Tab. 4.

However, latent diffusion models introduce an additional constraint: images are compressed into latent representations that may lose fine details. This affects both reconstruction quality and the ability to recognize subtle visual features. For instance, in Tab. 4, SANA-based methods underperform on the DeRain task due to SANA's higher compression ratio. Employing a latent encoder specifically optimized for detail preservation could alleviate this issue.

Additionally, VDC prioritizes structural fidelity over non-rigid flexibility to prevent hallucinations, which limits large changes. Moreover, complex patterns (e.g., generalization to real rain) can challenge one-shot alignment. However, Tab. 5 confirms that simply adding more visual examples (synthetic) effectively mitigates this.

G. Visual Results

In Figs. 14–19, we provide additional visual comparisons across all baseline methods, as well as all variants of our approach using different generative models—Stable Diffusion (SD) and SANA—and different setups: One-Shot (OS), Multi-Shot (MS), and Multi-Shot with Inversion Correction (MS+IC).

Figure 14. *Visual comparison on SR task.* Text- and example-based approaches either fail to recognize the required edits or produce undesired changes and artifacts in the output. Our one-shot (OS) VDC yields clean results, with multi-shot (MS) and inversion correction (IC) modules improving generalization and fidelity.

Figure 15. *Visual comparison on DeBlurring task.* Text- and example-based approaches either fail to recognize the required edits or produce undesired changes and artifacts in the output. Our one-shot (OS) VDC yields clean results, with multi-shot (MS) and inversion correction (IC) modules improving generalization and fidelity.

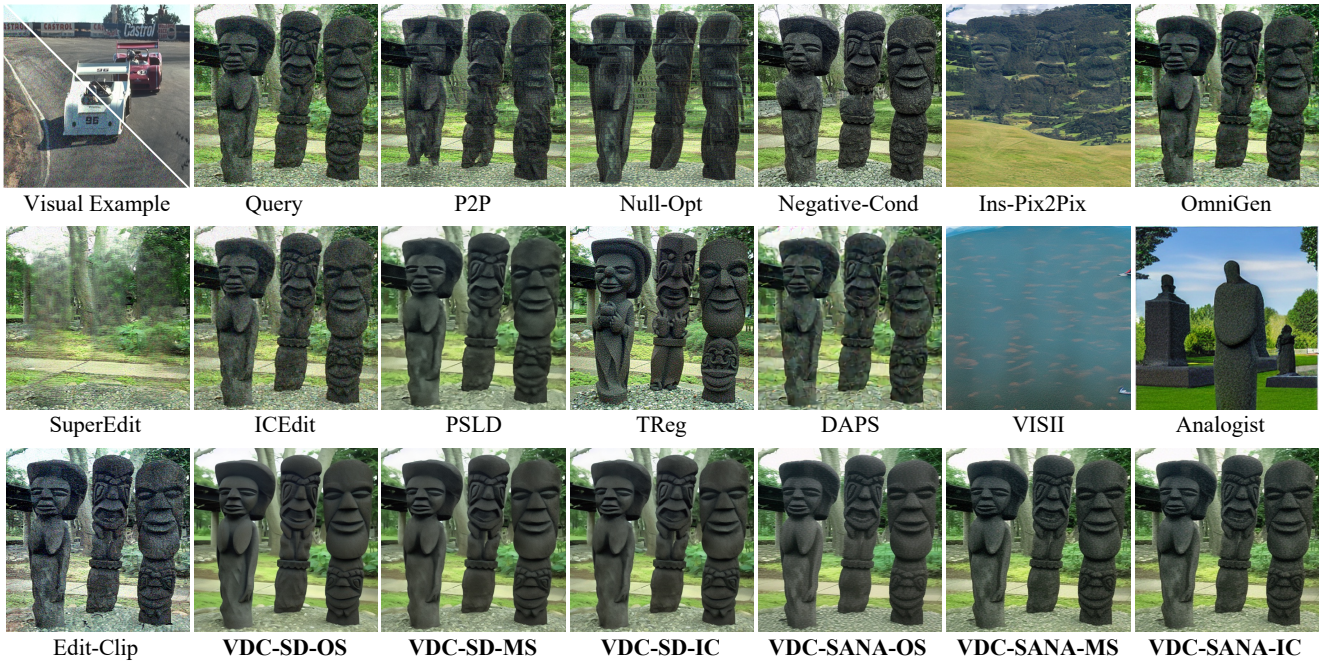

Figure 16. *Visual comparison on DeNoising task.* Text- and example-based approaches either fail to recognize the required edits or produce undesired changes and artifacts in the output. Our one-shot (OS) VDC yields clean results, with multi-shot (MS) and inversion correction (IC) modules improving generalization and fidelity.

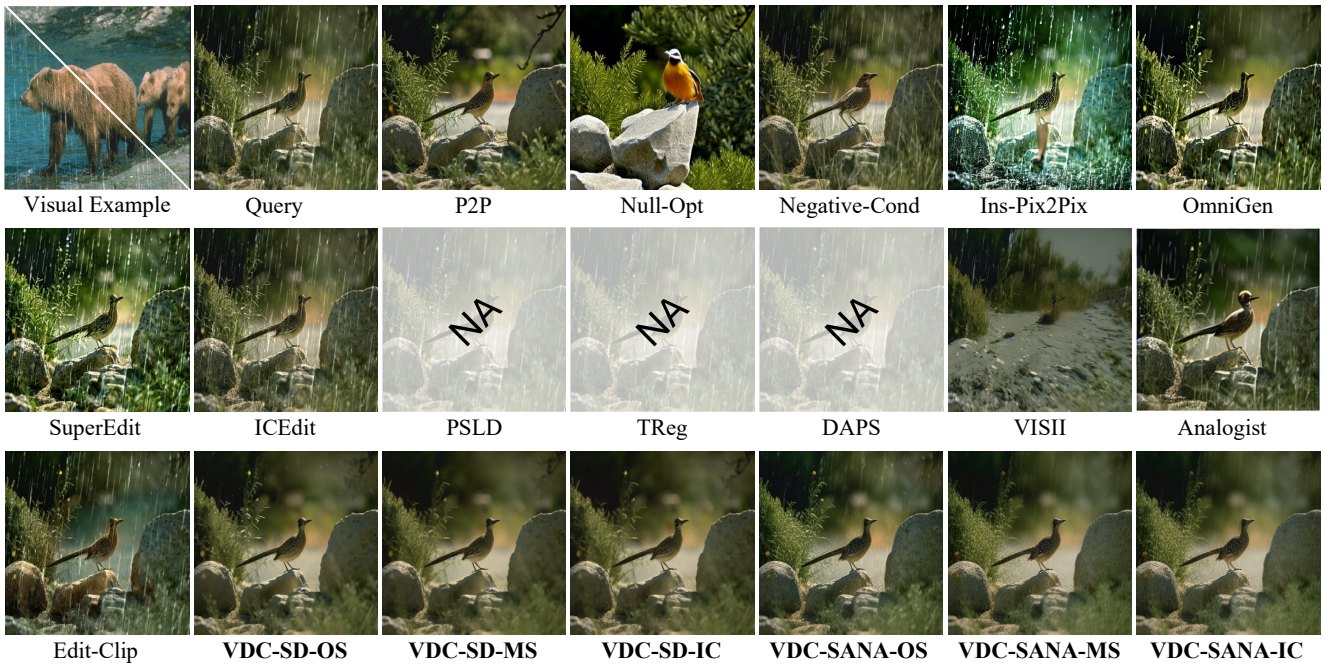

Figure 17. *Visual comparison on DeRaining task.* Text- and example-based approaches either fail to recognize the required edits or produce undesired changes and artifacts in the output. Our one-shot (OS) VDC yields clean results, with multi-shot (MS) and inversion correction (IC) modules improving generalization and fidelity.

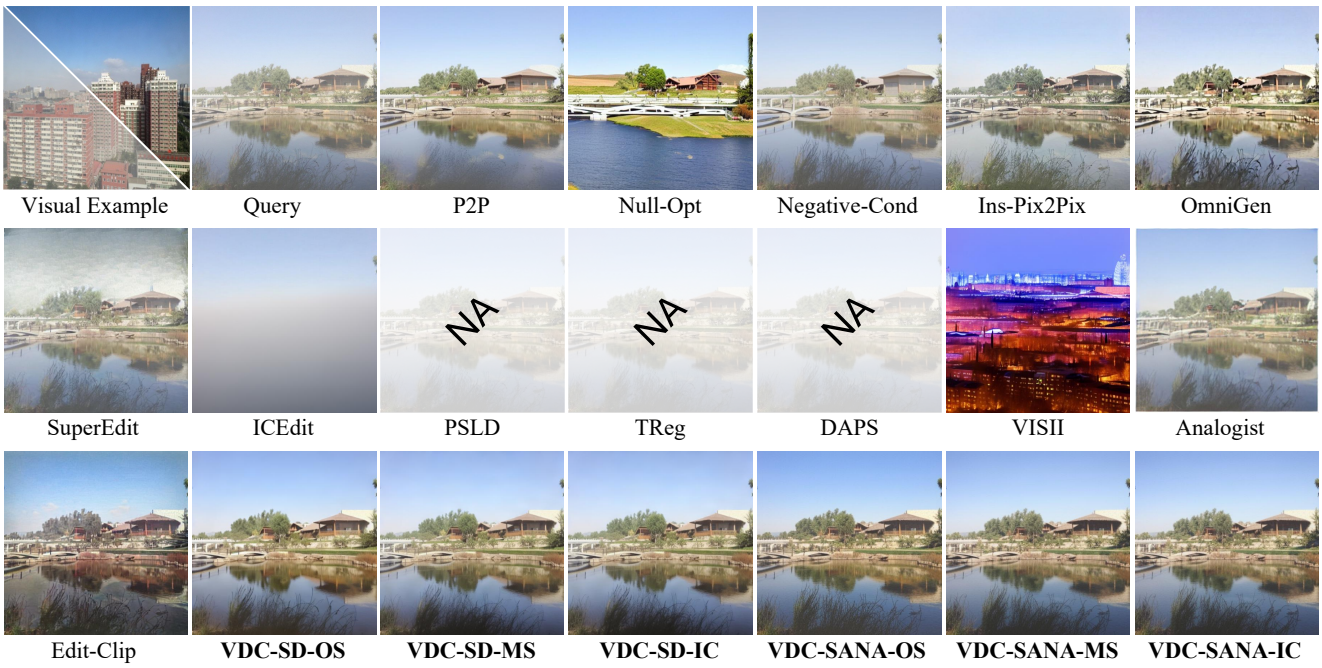

Figure 18. *Visual comparison on DeHazing task.* Text- and example-based approaches either fail to recognize the required edits or produce undesired changes and artifacts in the output. Our one-shot (OS) VDC yields clean results, with multi-shot (MS) and inversion correction (IC) modules improving generalization and fidelity.

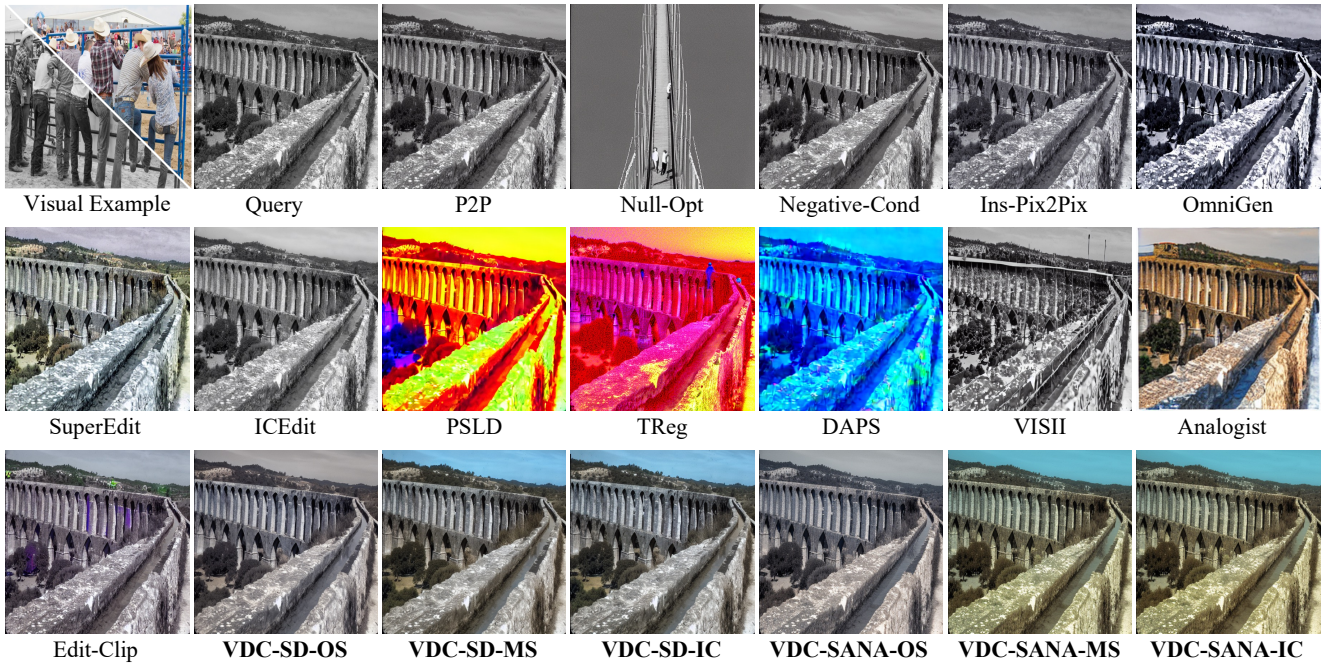

Figure 19. *Visual comparison on Colorization task.* Text- and example-based approaches either fail to recognize the required edits or produce undesired changes and artifacts in the output. Our one-shot (OS) VDC yields clean results, with multi-shot (MS) and inversion correction (IC) modules improving generalization and fidelity.

References

[1] Abdelrahman Abdelhamed, Stephen Lin, and Michael S Brown. A high-quality denoising dataset for smartphone cameras. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1692–1700, 2018.

[2] 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*, pages 18392–18402, 2023.

[3] Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *The Eleventh International Conference on Learning Representations*, 2023.

[4] Marcos V Conde, Gregor Geigle, and Radu Timofte. Instructir: High-quality image restoration following human instructions. In *European Conference on Computer Vision*, pages 1–21. Springer, 2024.

[5] Zheng Gu, Shiyuan Yang, Jing Liao, Jing Huo, and Yang Gao. Analogist: Out-of-the-box visual in-context learning with image diffusion model. *ACM Transactions on Graphics (TOG)*, 43(4):1–15, 2024.

[6] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022.

[7] Jeongsol Kim, Geon Yeong Park, Hyungjin Chung, and Jong Chul Ye. Regularization by texts for latent diffusion inverse solvers. *arXiv preprint arXiv:2311.15658*, 2023.

[8] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

[9] Ming Li, Xin Gu, Fan Chen, Xiaoying Xing, Longyin Wen, Chen Chen, and Sijie Zhu. Superedit: Rectifying and facilitating supervision for instruction-based image editing. *arXiv preprint arXiv:2505.02370*, 2025.

[10] Wei Li, Qiming Zhang, Jing Zhang, Zhen Huang, Xinmei Tian, and Dacheng Tao. Toward real-world single image deraining: A new benchmark and beyond. *arXiv preprint arXiv:2206.05514*, 2022.

[11] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[12] Haoguang Lu, Jiacheng Chen, Zhenguo Yang, Aurele Tohokantche Gnanha, Fu Lee Wang, Li Qing, and Xudong Mao. Pairedit: Learning semantic variations for exemplar-based image editing. *arXiv preprint arXiv:2506.07992*, 2025.

[13] Daiki Miyake, Akihiro Iohara, Yu Saito, and Toshiyuki Tanaka. Negative-prompt inversion: Fast image inversion for editing with text-guided diffusion models. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2063–2072. IEEE, 2025.

[14] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6038–6047, 2023.

[15] Thao Nguyen, Yuheng Li, Utkarsh Ojha, and Yong Jae Lee. Visual instruction inversion: Image editing via image prompting. *Advances in Neural Information Processing Systems*, 36:9598–9613, 2023.

[16] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[17] 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*, pages 10684–10695, 2022.

[18] Litu Rout, Negin Raoof, Giannis Daras, Constantine Caramanis, Alex Dimakis, and Sanjay Shakkottai. Solving linear inverse problems provably via posterior sampling with latent diffusion models. *Advances in Neural Information Processing Systems*, 36:49960–49990, 2023.

[19] Qian Wang, Aleksandar Cvejic, Abdelrahman Eldesokey, and Peter Wonka. Editclip: Representation learning for image editing. *arXiv preprint arXiv:2503.20318*, 2025.

[20] Yinhuai Wang, Jiwen Yu, and Jian Zhang. Zero-shot image restoration using denoising diffusion null-space model. *arXiv preprint arXiv:2212.00490*, 2022.

[21] Shitao Xiao, Yueze Wang, Junjie Zhou, Huaying Yuan, Xingrun Xing, Ruiran Yan, Chaofan Li, Shuting Wang, Tiejun Huang, and Zheng Liu. Omnigen: Unified image generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 13294–13304, 2025.

[22] Enze Xie, Junsong Chen, Junyu Chen, Han Cai, Haotian Tang, Yujun Lin, Zhekai Zhang, Muyang Li, Ligeng Zhu, Yao Lu, et al. Sana: Efficient high-resolution image synthesis with linear diffusion transformers. *arXiv preprint arXiv:2410.10629*, 2024.

[23] Eduard Zamfir, Zongwei Wu, Nancy Mehta, Yuedong Tan, Danda Pani Paudel, Yulun Zhang, and Radu Timofte. Complexity experts are task-discriminative learners for any image restoration. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 12753–12763, 2025.

[24] Bingliang Zhang, Wenda Chu, Julius Berner, Chenlin Meng, Anima Anandkumar, and Yang Song. Improving diffusion inverse problem solving with decoupled noise annealing. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 20895–20905, 2025.

[25] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3836–3847, 2023.

[26] Zechuan Zhang, Ji Xie, Yu Lu, Zongxin Yang, and Yi Yang. In-context edit: Enabling instructional image editing with in-

context generation in large scale diffusion transformer. *arXiv preprint arXiv:2504.20690*, 2025.

[27] Ruoyu Zhao, Qingnan Fan, Fei Kou, Shuai Qin, Hong Gu, Wei Wu, Pengcheng Xu, Mingrui Zhu, Nannan Wang, and Xinbo Gao. Instructbrush: Learning attention-based instruction optimization for image editing. *arXiv preprint arXiv:2403.18660*, 2024.