Supplementary Material for "Delta Rectified Flow Sampling for Text-to-Image Editing"

A. Diffusion Models

A.1. Diffusion models background

Diffusion models define a forward process that gradually adds Gaussian noise to a clean image (or its latent) x_0 and a reverse (denoising) process that recovers x_0 from a noisy sample.

Given noise schedulers a_t and b_t, the forward process is defined as:

$$x_t = a_t x_0 + b_t \varepsilon, \quad x_0 \sim p_0,\ \varepsilon \sim \mathcal{N}(0, I). \tag{S1}$$

In practice, schedulers are chosen as

$$a_t = \sqrt{\bar{\alpha}_t}, \quad b_t = \sqrt{1 - \bar{\alpha}_t},$$

where $\bar{\alpha}_0 = 1, \bar{\alpha}_T = 0$. A neural network ε_θ is then trained to predict the noise ε from a noisy input using the loss function:

$$\mathcal{L}_{\text{DDPM}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}\{1,\dots,T\},\, x_0\sim p_0,\, \varepsilon\sim\mathcal{N}(0,I)}\Big[\|\varepsilon_\theta(a_t x_0 + b_t\varepsilon,\, t) - \varepsilon\|^2\Big]. \tag{S2}$$

Once we have a pretrained neural network ϵ_θ, clean images can be sampled in various ways. For simplicity, below, we explain a deterministic sampling process (DDIM [37]). From an initial gaussian noise $x_T \sim \mathcal{N}(0, I)$, $\{x_t\}_{t=0}^T$ are recursively defined as:

$$x_{t-1} = a_{t-1}\left(\frac{x_t - b_t\,\varepsilon_\theta(x_t, t)}{a_t}\right) + b_{t-1}\,\varepsilon_\theta(x_t, t). \tag{S3}$$

Score function. Note that the forward process Eq. (S1) induces a marginal distribution of x_t, which we denote as p_t. The score function of p_t is defined as the gradient of the log-density: $s(x_t, t) := \nabla_{x_t} \log p_t(x_t)$. The ground truth score function and the truth noise prediction exhibit the following connection:

$$s(x_t, t) = \nabla_{x_t} \log p_t(x_t) = -\frac{\mathbb{E}[\epsilon \mid x_t]}{b_t}.$$

Therefore, a diffusion model that predicts noise can be interpreted as a model that predicts the score function with the relation of $s_\theta(x_t, t) = -\frac{\varepsilon_\theta(x_t,t)}{b_t}$.

A.2. Reconstruction errors

We compare the reconstruction accuracy of (i) a diffusion model inverted with DDIM and (ii) a rectified-flow model inverted with a first-order (Euler) solver. For each method we run an inversion then reconstruction and measure the full reconstruction error at every time-step t.

Algorithm S1 DDIM inversion — reconstruction error

Require: Source image x_0; total steps T

for $t = 0$ **to** $T-1$ **do** ▷ DDIM inversion

$\quad x_{t+1} \leftarrow a_{t+1}\left(\frac{x_t - b_t\varepsilon_\theta(x_t,t)}{a_t}\right) + b_{t+1}\,\varepsilon_\theta(x_t, t)$

end for

$\tilde{x}_T \leftarrow x_T$

for $t = T-1$ **down to** 0 **do** ▷ DDIM reconstruction

$\quad \tilde{x}_t \leftarrow a_t\left(\frac{\tilde{x}_{t+1} - b_{t+1}\varepsilon_\theta(\tilde{x}_{t+1}, t+1)}{a_{t+1}}\right) + b_t\,\varepsilon_\theta(\tilde{x}_{t+1}, t+1)$

end for

$e_t \leftarrow \|\tilde{x}_t - x_t\|_2$ for all $t \in \{0, \dots, T-1\}$

return $e = (e_0, \dots, e_{T-1})$

The per-step approximation used is $\varepsilon_\theta(x_t, t) \approx \varepsilon_\theta(x_{t+1}, t)$.

Algorithm S2 Rectified-flow inversion — reconstruction error

Require: Source image x_0; total steps T

for $i = 0$ **to** $T-1$ **do** ▷ Euler inversion

$\quad x_{t_{i+1}} \leftarrow x_{t_i} - (t_i - t_{i+1})\, v_\theta(x_{t_i}, t_i)$

end for

$\tilde{x}_{t_T} \leftarrow x_{t_T}$

for $i = T-1$ **down to** 0 **do** ▷ Euler reconstruction

$\quad \tilde{x}_{t_i} \leftarrow \tilde{x}_{t_{i+1}} + (t_i - t_{i+1})\, v_\theta(\tilde{x}_{t_{i+1}}, t_{i+1})$

end for

$e_{t_i} \leftarrow \|\tilde{x}_{t_i} - x_{t_i}\|_2$ for all $i \in \{0, \dots, T-1\}$

return $e = (e_{t_0}, \dots, e_{t_{T-1}})$

Here the per-step approximation is $v_\theta(x_{t_i}, t_i) \approx v_\theta(x_{t_{i+1}}, t_i)$.

Figure S1 plots the resulting error curves for a diffusion model (SD1.5) and a rectified-flow model (SD3).

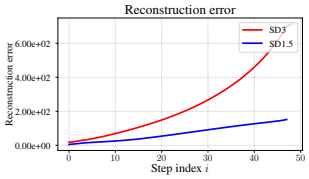

Figure S1. Reconstruction-error comparison between diffusion and rectified-flow models.

The gap can stem from two key architectural differences: diffusion models employ the scheduler $(a_t, b_t) = (\sqrt{\bar{\alpha}_t}, \sqrt{1-\bar{\alpha}_t})$, whereas rectified-flow models use the linear scheduler $(1-t,\, t)$; and diffusion models directly predict the noise ε, while rectified-flow models predict the velocity $u = \varepsilon - x_0$.

The rectified-flow reconstruction error is significantly higher across all steps, motivating our investigation of inversion-free editing methods.

A.3. Distillation Sampling paradigm

Figure (S2) represents the position of our DRFS in the Distillation Sampling paradigm.

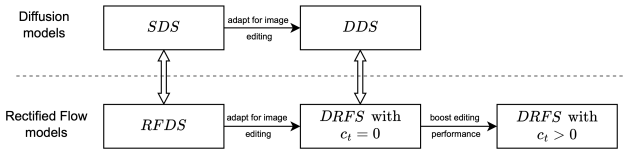

Figure S2. DRFS in the Distillation Sampling paradigm

B. Additional implementation details

An implementation of our method is available in the code appendix.

B.1. Stable Diffusion 3

We used a batch size of 1 and a unit weighting function, following [9]. We set the source and target CFG values to 6 and 16.5, respectively, and performed 50 optimization steps using the Stable Diffusion descending time-steps scheduler.

We used this setting for all figures plotted in the paper unless stated otherwise.

During the very noisy early time-steps we apply a small learning rate to avoid drifting too far from the source image while still permitting non-rigid edits. The rate is increased in the second half of optimization, where latents are cleaner and substantive edits are easier to apply, and then gently decayed at the final few steps, when further changes are unlikely to be beneficial.

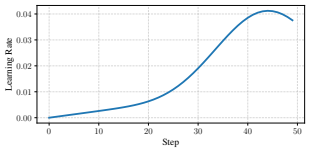

Figure S3. Learning rate used in our optimization.

B.2. Stable Diffusion 3.5

We used the same hyperparameters as SD3, except for CFG, where we set the source and target CFG values to 5.5 and 13.5 , respectively.

B.3. Efficiency and computational cost

All experiments were conducted on a single NVIDIA RTX A6000 GPU (48 GB VRAM), with 16 CPU cores and 64 GB of RAM. On average, on SD3, one edit takes 7.3 seconds using our DRFS method, compared to 4.9 seconds with FlowEdit and 2 minutes and 26 seconds with the distillation-based method iRFDS (averaged over 700 edits from the PIE benchmark).

Table S1. Efficiency on PIE (700 edits) on SD3. Mean $\pm$ std over images.

Method	NFE $\downarrow$	Peak VRAM (GB) $\downarrow$	Time (s/edit) $\downarrow$
FlowEdit (official ODE)	33	18.52 ± 0.01	4.9 ± 0.4
iRFDS	2800	37.86 ± 0.01	145.3 ± 2.7
DRFS (ours)	50	18.53 ± 0.03	7.3 ± 1.3

B.4. On the shift schedule

Under a descending timestep schedule, early iterations operate at high noise ($t \simeq 1$), where gradients are noisier and the current estimate x_0^k is least accurate. A large early shift can *reinject* these errors into future velocity queries and cause drift.

Let $\hat{x}_t(x_0)$ denote the shifted forward state used to query target velocities, let $x_0^\star$ be an ideal edited latent, and define the current optimization error $e_k := x_0^k - x_0^\star$. With rectified-flow forward $(1-t)x_0 + t\varepsilon$, introducing the shift yields

$$\hat{x}_t(x_0) = (1-t)x_0 + t\varepsilon + c_t(x_0 - x_0^{src}).$$

Define $A_t := (1-t)+c_t$. Then the shift amplifies the optimization error as

$$\hat{x}_t(x_0^k) - \hat{x}_t(x_0^\star) = A_t\, e_k,$$

so A_t **directly controls error reinjection**.

Moreover, if we assume that $v_\theta(\cdot, t, \varphi)$ is L_t-Lipschitz in x, the velocity mismatch is bounded by

$$\|v_\theta(\hat{x}_t(x_0^k), t, \varphi^{\text{tgt}}) - v_\theta(x_t^\star, t, \varphi^{\text{tgt}})\| \le L_t\big(A_t\|e_k\| + c_t\|\Delta\|\big),$$

where $\Delta := x_0^\star - x_0^{src}$ and $x_t^\star := (1-t)x_0^\star + t\varepsilon$.

Since $\|e_k\|$ is largest early, we require A_t to be small near $t \simeq 1$ while keeping a non-zero shift at mid-noise to improve target alignment.

This motivates $c_t \to 0$ as $t \to 0$ and $t \to 1$, and our simple choice $c_t = t(1-t)$. In contrast, FlowEdit uses $c_t = t$, which gives $A_t \equiv 1$ (no attenuation), while $c_t = t(1-t)$ yields $A_t = 1 - t^2 \to 0$ as $t \to 1$.

B.5. Additional details

For Fig. 1, we took the official implementation of RFDS [48], adapted to start the optimization from the source image, and slightly reduced the target CFG.

For Fig. 3a and 3b, we used 20 images, source and target prompts from PIE benchmark.

For Fig. 6, we used $c_t = 0$, 40 optimisation steps and simply a constant learning rate of 0.02, and same CFG values as before, to highlight and isolate the impact of the scheduler strategy. To try a good optimization set-up, we tested some target CFG values between between 12.5 and 18.5.

The results from the table 1 of PIE benchmark of diffusion based methods were taken from [13]. The results from FireFlow, RFSolver and RF-Inv, were taken from the FireFlow paper, and completed on the the LPIPS and MSE metrics by running the evaluation ourselves with each official implementation. For FlowEdit and iRFDS, we also used the official implementations, to run the benchmark.

When computing the metrics on our additional dataset (Fig. S5), we also took the official implementation of each model.

B.6. Additional dataset generation details

We used the Qwen2.5-VL-7B-Instruct [42] model to generate more than 300 source captions and target prompts.

- **Prompt used to caption source images:**

```
Describe simply this image in just very
few words
```

- **Prompt used to generate target prompts:**

```
Given the original description
{source_prompt}
Generate a new description by modifying
the most important object, attribute:
the description should have a completely
different meaning, by just and only
changing a (or 2 MAXIMUM) word(s). You
can also add or remove a new object,
attribute, etc. in the description.
The change should totally alter the
meaning or visual content of the
description.
All other words MUST remain the same and
in the same order. Return only the new
description.
```

C. Additional results

C.1. Effective gradient cancellation in irrelevant parts

Rewritting Eq. 8, we have

$$\begin{aligned}\nabla_\Theta \mathcal{E}_{\text{RFDS}} &= \mathbb{E}_{t,\varepsilon}\left[w_{\text{RFDS}}(t)\left(v_\theta(\hat{x}_t^{\text{tgt}}) - v_\theta(x_t^{\text{src}})\right)\right] \\ &= -\mathbb{E}_{t,\varepsilon}\left[w_{\text{RFDS}}(t)\left(\dot{\hat{x}}_t^{\text{tgt}} - \dot{x}_t^{\text{src}}\right)\right] \\ &= \mathbb{E}_{t,\varepsilon}\left[w_{\text{RFDS}}(t)\left(v_\theta(\hat{x}_t^{\text{tgt}}) - \dot{\hat{x}}_t^{\text{tgt}}\right)\right] \\ &\quad - \mathbb{E}_{t,\varepsilon}\left[w_{\text{RFDS}}(t)\left(v_\theta(x_t^{\text{src}}) - \dot{x}_t^{\text{src}}\right)\right].\end{aligned} \tag{S4}$$

We use the notation $\nabla\mathcal{E}_{\text{RFDS}}(x_t^{src}, \varphi^{\text{src}})$ and $\nabla\mathcal{E}_{\text{RFDS}}(x_t^{\text{src}}, \varphi^{\text{src}})$ for respectively the first and the second term in Equation S4, when using only one (t, ε) pair to compute the gradients.

Figure S4 visualize the DRFS gradients, and its differential nature, cancelling irrelevant gradients in irrelevant areas. This propery clearly echoes DDS [9]. However, the introduction of the c_t term in $\hat{x}_t^{\text{tgt}}$, gives straighter paths and larger gradient updates, so a much more effective editing. DDS official implementation uses 200 optimization, while we use 50 optimization steps with DRFS.

C.2. Result on our additional dataset for different CFG values

On the additional dataset described in B.6, we measured LPIPS and CLIP similarity for several methods across a range of target CFG scales (directly written on Fig. S5). As the plot shows, DRFS consistently outperforms all baselines, striking the best balance between background preservation and adherence to the target prompt.

C.3. More qualitative results

We provide additional qualitative results. First, we show editing outputs produced by our DRFS in Fig. S6. Then, in Fig. S7, we present a detailed comparison on images from the PIE benchmark between FlowEdit [18], iRFDS [48], FireFlow [5], RF-Solver [41], RF Inversion [35], and Direct Inversion + P2P [13]. Results are produced using SD3.

C.4. Structural edits

We also report challenging category-level PIE results. DRFS consistently improves both alignment and fidelity.

Table S2. Category-level PIE results on challenging edits (SD3). Lower LPIPS indicates better structural preservation, higher CLIP indicates stronger semantic alignment.

Method	Change Object Pose		Change Image Style	
	LPIPS $\times 10^3 \downarrow$	CLIP $\uparrow$	LPIPS $\times 10^3 \downarrow$	CLIP $\uparrow$
FlowEdit	102.6	22.87	116.2	27.38
DRFS (ours)	91.4	23.26	107.4	27.58

C.5. Comparison with Instruction-based methods.

We compare DRFS to prominent instruction-driven editors on PIE. DRFS achieves best overall performance *without training on large paired datasets*.

Method	Struct. dist. $\times 10^3 \downarrow$	LPIPS $\times 10^3 \downarrow$	SSIM $\times 10^2 \uparrow$	$\text{CLIP}_{\text{edit}} \uparrow$
InstructPix2Pix [4]	58.80	159.38	76.75	21.75
MagicBrush [50]	45.53	**86.59**	83.09	22.18
DRFS (ours)	**23.05**	93.81	**84.85**	**23.83**

C.6. More ablation studies

Batch size. We evaluated the effect of different batch sizes B when estimating the gradient in (8). Increasing B improves both structural integrity and background preservation, while maintaining comparable editing strength. In particular, the $B = 5$ setting further widens the gap against all baselines. However, since the computational cost scales linearly with B, we opt for $B = 1$ throughout the paper, which already outperforms the baselines.

Optimizer. We compare vanilla SGD and Adam optimizers on the PIE benchmark (Table S4). Following Hertz et al. [9], we find that vanilla SGD produces higher-quality edits. Using only 50 optimization steps is sufficient to obtain strong results (thanks to the straighter path and larger updates afforded by our added shift term) whereas DDS [9] requires 200 steps. For a fair comparison, we also evaluate (i) SGD with a constant learning rate of 0.02 and (ii) Adam with the same rate, in both cases omitting the first few noisy steps. All other hyperparameters are identical.

We found that the following explanation made in [9] was also true in rectified flow models. The difference in quality arises from Adam's adaptive normalization of gradients. For simplicity, consider the Adagrad update: $\Theta_k \leftarrow \Theta_{k-1} - \alpha \frac{g_k}{\sqrt{\sum_{i=1}^k g_i^2}}$, where g_k is the gradient at step k and α is the learning rate. Normalizing by the accumulated squared gradients can magnify outliers and downweight consistently informative gradients. This results in a an extremely low adherence to the target prompt in the ADAM case.

Figure S4. DRFS gradients. DRFS gradients cancel out in irrelevant parts of the image.

Table S3. Impact of batch size on PIE benchmark. The best is shown in bold.

Method	Model	Structure		Background Preservation				CLIP Similarity	
		Editing	Distance $_{\times 10^3}\downarrow$	PSNR ↑	LPIPS $_{\times 10^3}\downarrow$	MSE $_{\times 10^4}\downarrow$	SSIM $_{\times 10^2}\uparrow$	Whole ↑	Edited ↑
DRFS (B=1)	SD3	-	23.05	23.38	93.81	67.49	84.85	26.90	**23.83**
DRFS (B=5)	SD3	–	**21.50**	**23.92**	**87.68**	**60.22**	**85.68**	**26.92**	23.70

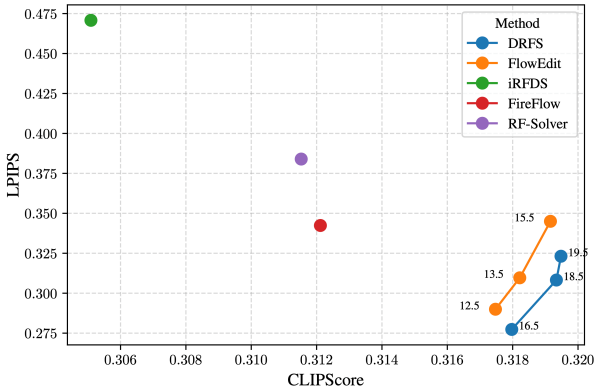

Figure S5. Comparison between LPIPS and CLIPScore on 340 source images and editing prompts. Higher CLIPScore indicates better semantic alignment with the target prompt, while lower LPIPS indicates better perceptual similarity to the reference image.

D. Connection between DRFS and DDIB

DDIB. Dual Diffusion Implicit Bridge (DDIB) [38] leverage two ODEs, and was designed for image translation. Given a source images x_0^{src}, the source ODE runs in the forward direction to convert the source to the latent noise, and the reverse the ODE with target prompts then constructs target images x_0^{tgt}. The source trajectory $(x_t^{\text{src}})_{t\in[0,1]}$ and target trajectory $(x_t^{\text{tgt}})_{t\in[0,1]}$

$$x_t^{\text{src}} = \text{ODESolve}(x_0^{\text{src}}, v(\cdot, \varphi^{\text{src}}, t), 0, t)$$
$$x_t^{\text{tgt}} = \text{ODESolve}(x_1^{\text{tgt}}, v(\cdot, \varphi^{\text{src}}, t), 1, t)$$

With these notations we can now write the edited image x_0^{tgt}:

$$\begin{aligned} x_0^{\text{tgt}} &= x_1^{\text{src}} + \int_1^0 v_\theta(x_t^{\text{tgt}}, \varphi^{\text{tgt}}, t)dt \\ &= x_0^{\text{src}} + \int_0^1 v_\theta(x_t^{\text{src}}, \varphi^{\text{src}}, t)dt + \int_1^0 v_\theta(x_t^{\text{tgt}}, \varphi^{\text{tgt}}, t)dt \\ &= x_0^{\text{src}} + \int_1^0 \big(v_\theta(x_t^{\text{tgt}}, \varphi^{\text{tgt}}, t) - (v_\theta(x_t^{\text{src}}, \varphi^{\text{src}}, t)\big)dt \end{aligned}$$

DDIB are two concatenated Schrodinger Bridges: they traverse through two one forward and one reversed (PF-ODE is special linear or degenerate Schrodinger Bridge). DDIB has the interesting Exact Cycle Consistency property.

DDS sampling. A recent work [11] proposed using diffusion weighting to define a DDS style diffusion sampling, we define $(x_{t,DDS}^{\text{tgt}})_{t\in[1,0]}$ by solving an ODE starting from $x_{1,DDS}^{\text{tgt}} = x_0^{\text{src}}$:

$$\begin{aligned} x_{0,\text{DDS}}^{\text{tgt}} &= x_0^{\text{src}} + \int_0^1 w(t)\big(\varepsilon_\theta(\tilde{x}_t^{\text{tgt}}, \varphi^{\text{tgt}}, t) - \varepsilon_\theta(\tilde{x}_t^{\text{src}}, \varphi^{\text{src}}, t)\big)dt \\ &= x_0^{\text{src}} + \int_1^0 \tilde{w}(t)\big(v_\theta(\tilde{x}_t^{\text{tgt}}, \varphi^{\text{tgt}}, t) - v_\theta(\tilde{x}_t^{\text{src}}, \varphi^{\text{src}}, t)\big)dt \end{aligned}$$

where

$$\tilde{x}_t^{\text{tgt}} = a_t x_{t,DDS}^{\text{tgt}} + b_t\varepsilon \quad \text{and} \quad \tilde{x}_t^{\text{src}} = a_t x_0^{\text{src}} + b_t\varepsilon.$$

Hence we can clearly see that DDS is an approximation of the DDIB, with approximated $\tilde{x}_t^{\text{tgt}}$ and $\tilde{x}_t^{\text{src}}$. Taking $\tilde{x}_t^{\text{src}} = a_t x_0^{\text{src}} + b_t\varepsilon$ is a relatively valid hypothesis, whereas $\tilde{x}_t^{\text{tgt}} = a_t x_{t,DDS}^{\text{tgt}} + b_t\varepsilon \neq x_t^{\text{tgt}} = a_t x_t^{\text{tgt}} + b_t\varepsilon$ is not, because for example $x_{1,DDS}^{\text{tgt}} = x_0^{\text{src}} \neq x_0^{\text{tgt}}$. $\tilde{x}_t^{\text{tgt}}$ will always be too close from the source image than the real x_t^{tgt} from DDIB.

DRFS. To tackle this issue, we improve this approximation by adding a term $c_t(x_{t,DDS}^{\text{tgt}} - x_t^{\text{src}})$ to push more $\tilde{x}_t^{\text{tgt}}$ toward the target distribution:

$$\tilde{x}_t^{\text{tgt}} = a_t x_{t,DDS}^{\text{tgt}} + b_t\varepsilon + c_t(x_{t,DDS}^{\text{tgt}} - x_t^{\text{src}})$$

Table S4. Comparison of different optimizers on the PIE benchmark. The best and second best results are bolded and underlined, respectively.

Method	Model	Structure		Background Preservation				CLIP Similarity	
		Editing	Distance $_{\times 10^3}$ ↓	PSNR ↑	LPIPS $_{\times 10^3}$ ↓	MSE $_{\times 10^4}$ ↓	SSIM$_{\times 10^2}$ ↑	Whole ↑	Edited ↑
DRFS (SGD+dynamic lr)	SD3	-	<u>23.05</u>	<u>23.38</u>	<u>93.81</u>	<u>67.49</u>	<u>84.85</u>	**26.90**	**23.83**
DRFS (SGD+constant lr)	SD3	–	24.55	22.45	96.21	79.80	84.72	<u>26.54</u>	<u>23.21</u>
DRFS (ADAM)	SD3	–	**12.09**	**25.87**	**56.98**	**33.59**	**88.95**	24.10	21.07

DRFS can be interpreted as an enhanced approximation of DDIB, with a careful designed c_t in order to get strong editing perfomance while keeping good fidelity.

E. Limitations and Future Works

Our current formulation is limited to rectified flow models.

Extending DRFS into more models would be a promising direction. Additionally, DRFS could benefit from integration with existing inversion techniques, such as attention injection, to further enhance edit controllability. We also include and discuss DRFS failure cases.

F. Broader Impact

Our work introduces a new method for editing real images using state-of-the-art text-to-image rectified flow models. A potential societal risk of this technology includes the generation and spread of misinformation, misleading imagery, or manipulated content. To mitigate such risks, we will release our code with an appropriate licence that discourages harmful uses (offensive, or dehumanizing content, or content negatively targeting individuals, communities, cultures, or religions). Furthermore, we highlight that significant research and technological progress has recently been made towards detecting and limiting these harmful applications.

G. Failure case study

Fig. S8 shows several failure cases of DRFS. We observe that these failure cases reflect previously acknowledged common challenges in T2I editing, where a pretrained model struggles to predict an accurate velocity for out-of-distribution images.

Furthermore, in scenarios that require substantial changes, DRFS exhibits limited editing strength due to its inherent design focus on preserving details of the source image. For example, in the first row of Fig. S8, the desired transformation–from *outline of a wolf* to *outline of a man*–is both ambiguous and semantically complex. In the second row, + *with aerial view* requires extensive structural alterations, effectively amounting to the generation of a completely new image. While DRFS enhances both background preservation and alignment with target semantics, these examples underscore fundamental limitations that are prevalent across T2I approaches.

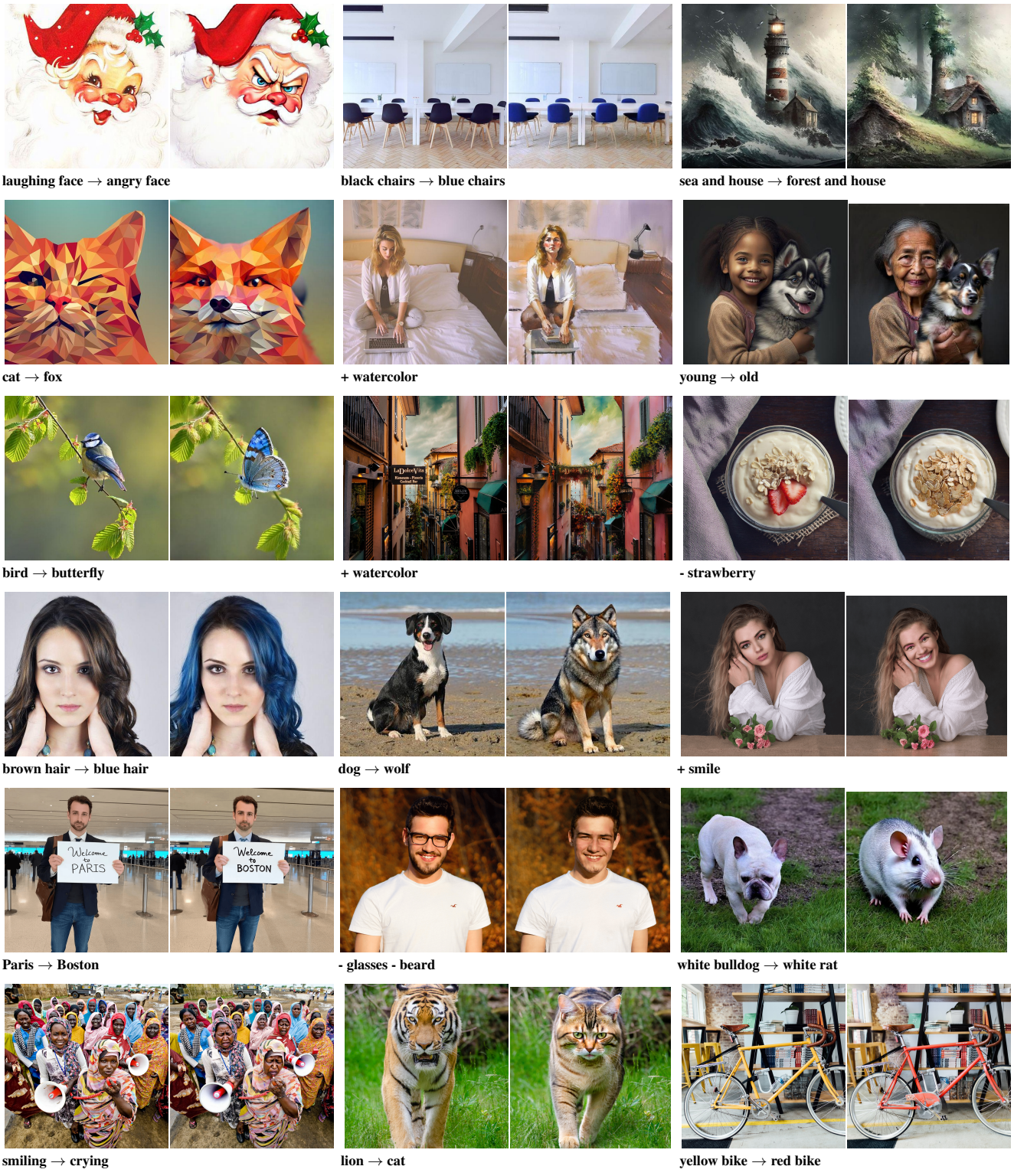

Figure S6. Qualitative edits produced by our DRFS. Each pair indicates the source image (left) and edited result (right).

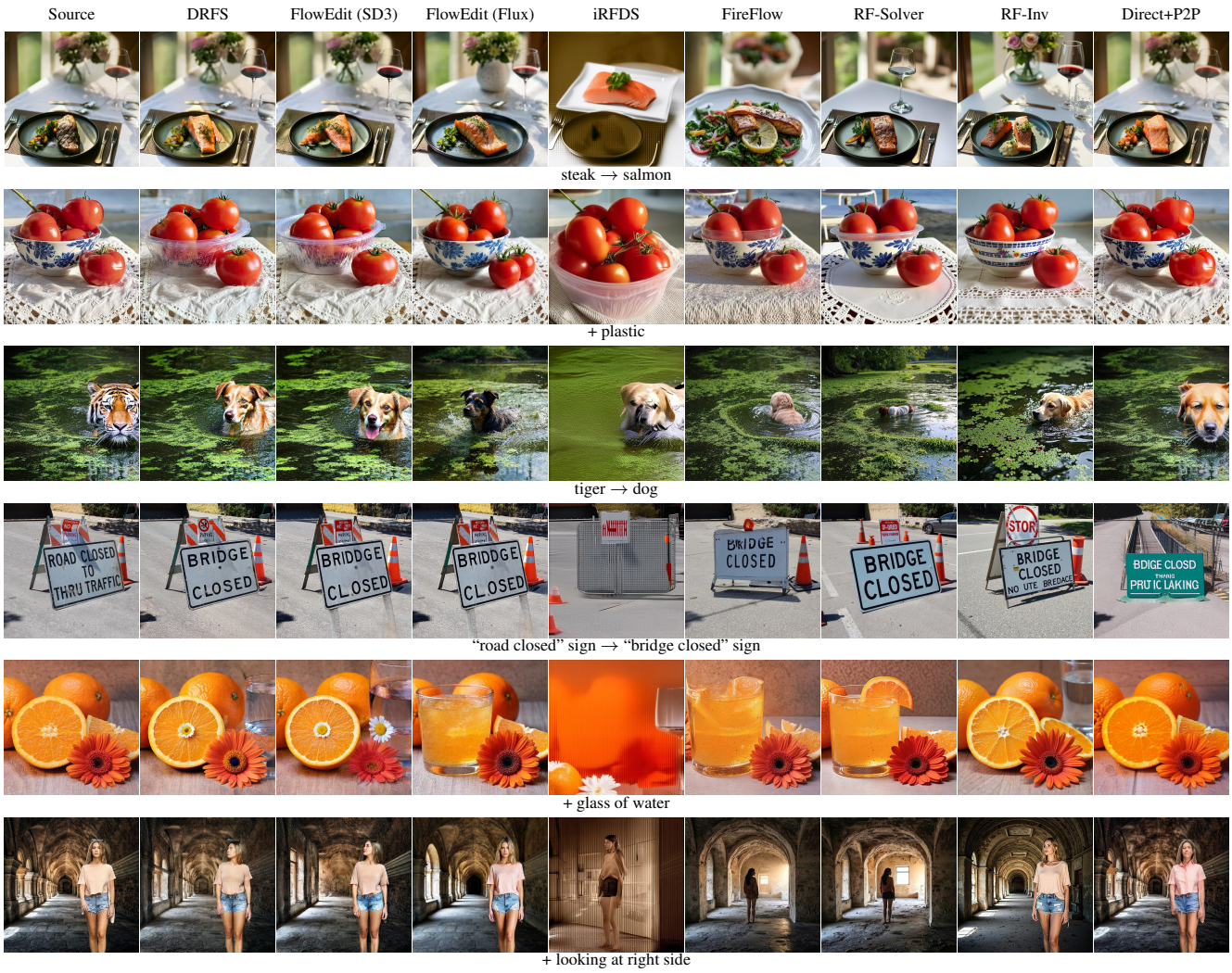

Figure S7. Qualitative comparisons on images from the PIE benchmark.

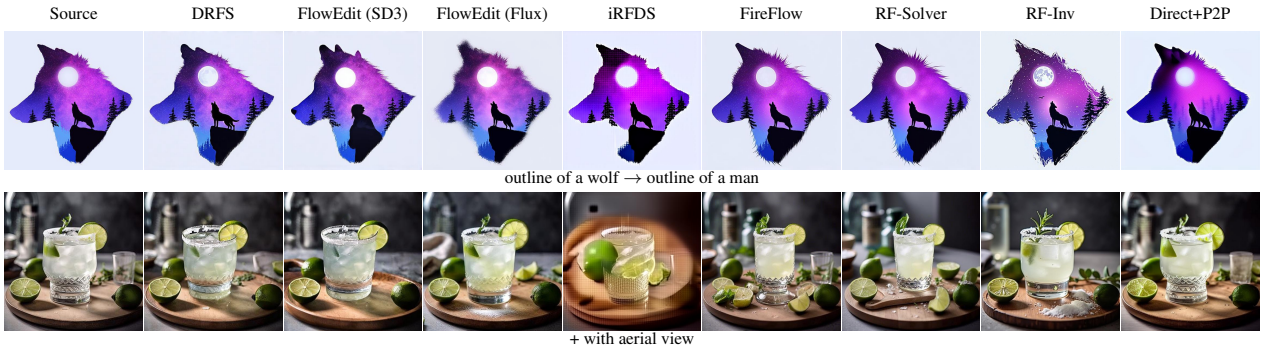

Figure S8. Examples of failure cases from our method.