

# Supplementary Material

## A Effect of LLM and Prompting ( $\varphi$ )

To validate the effectiveness and robustness of SplitFlow, we evaluated our model using different prompting strategies and various LLMs for prompt decomposition. The prompts used for decomposition are shown in Box 1 and Box 2. To enhance the prompt decomposition process, several illustrative examples were included in the instruction to guide the model toward more consistent semantic partitioning. As shown in Table S1, the proposed SplitFlow consistently outperforms the baseline, demonstrating its robustness. When using Qwen2 and LLaMA2 for prompt decomposition (with  $\psi_1$ ), we observe improved background preservation, albeit with a slight degradation in editability compared to Mixtral-7B. When using an alternative prompt ( $\psi_2$ ) for decomposition and employing Qwen2, we observe a notable improvement in CLIP similarity within the edited region. In conclusion, across all settings—regardless of the choice of LLM or decomposition prompt ( $\psi$ )—SplitFlow consistently improves both fidelity and editability.

Table S1: Ablation study on PIE-benchmark with different LLMs and prompts.

Baseline	LLM	$\psi$	Structure	Background Preservation				CLIP Similarity	
			Distance $\times 10^3$ ↓	PSNR ↑	LPIS $\times 10^3$ ↓	MSE $\times 10^4$ ↓	SSIM $\times 10^2$ ↑	Whole ↑	Edited ↑
FlowEdit	-	-	27.24	22.13	105.46	87.34	83.48	26.83	23.67
SplitFlow	Mixtral-7B	$\psi_1$	25.96	22.45	102.14	81.99	83.91	26.96	23.83
SplitFlow	Qwen2	$\psi_1$	25.49	22.57	100.78	79.88	84.05	26.87	23.82
SplitFlow	Llama2	$\psi_1$	25.49	22.55	101.21	80.34	84.02	26.91	23.73
SplitFlow	Mixtral-7B	$\psi_2$	25.77	22.50	101.64	81.00	83.95	26.89	23.80
SplitFlow	Qwen2	$\psi_2$	26.10	22.39	103.17	83.57	83.77	26.92	23.90
SplitFlow	Llama2	$\psi_2$	25.49	22.57	101.01	80.25	84.03	26.87	23.80

### Box1: Prompt ( $\psi_1$ ) for Target Prompt $\varphi^{tgt}$ Decomposition

Given the source caption: "*source caption*" and the target caption: "*target caption*",  
Write three semantic captions that split the target caption.  
List each as a numbered item.

### Box2: Prompt ( $\psi_2$ ) for Target Prompt $\varphi^{tgt}$ Decomposition

Given the source sentence: "*source sentence*" and the target sentence: "*target sentence*",  
Split the target sentence into three concise sentences based on step-by-step changes.  
List each as a numbered item.

## B Addition Discussions

**SplitFlow behavior without explicit sub-prompt count.** In our current implementation, we set the maximum number of sub-target prompts to three in order to avoid excessive computational overhead. Thus, the target prompt is typically decomposed into three sub-prompts. When this maximum is not explicitly enforced, the number of sub-prompts ( $N$ ) ranges from 2 to 7—generally correlating with the length and complexity of the target prompt—with an average of 4.2. The results under this unconstrained setting are provided in the Table S2. While this variant shows slightly lower performance compared to the default configuration, it still demonstrates meaningful improvements in both fidelity and editability. We attribute the performance drop to over-segmentation, where the target prompt is divided into excessively fine-grained fragments, potentially weakening the semantic coherence of each sub-prompt. Nonetheless, the overall trend remains consistent: decomposing the editing process into multiple semantically structured sub-target flows contributes positively to the quality and controllability of the final edits.

**Extremely simple cases.** In Table S3, we conducted an additional evaluation focusing specifically on cases where an existing object was changed into a *dog*, as well as cases where an existing *dog* was transformed into a different object. A total of 9 samples were tested, covering various images

Table S2: Quantitative comparison results for different prompt numbers.

Method	# of sub-target prompt	Model	Structure	Background Preservation				CLIP Similarity	
			Distance $\times 10^3 \downarrow$	PSNR $\uparrow$	LPIPS $\times 10^3 \downarrow$	MSE $\times 10^4 \downarrow$	SSIM $\times 10^2 \uparrow$	Whole $\uparrow$	Edited $\uparrow$
FlowEdit	-	SD3	27.24	22.13	105.46	87.34	83.48	26.83	23.67
SplitFlow(Ours)	max 3	SD3	<b>25.96</b>	<b>22.45</b>	<b>102.14</b>	<b>81.99</b>	<b>83.91</b>	<b>26.96</b>	<b>23.83</b>
SplitFlow(Ours)	w/o max	SD3	26.55	22.29	104.11	84.87	83.64	26.95	23.80

and editing contexts. As shown in the results, our method consistently outperforms FlowEdit across all metrics. We believe these gains are meaningful, even when taking the associated computational burden into account.

Table S3: Quantitative comparison results on one sample from PIE benchmark. The sample is edited with simple editing prompt (cat→dog).

Method	Model	Structure		Background Preservation				CLIP Similarity	
		Editing	Distance $\times 10^3 \downarrow$	PSNR $\uparrow$	LPIPS $\times 10^3 \downarrow$	MSE $\times 10^4 \downarrow$	SSIM $\times 10^2 \uparrow$	Whole $\uparrow$	Edited $\uparrow$
FlowEdit	SD3	-	35.23	21.80	84.22	82.17	86.37	28.15	22.35
SplitFlow(Ours)	SD3	-	34.96	21.84	78.80	73.96	87.27	28.38	22.47
$\Delta$	-	-	-0.27	+0.04	-5.42	-8.21	+0.90	+0.23	+0.12

**Statistical Significance Testing.** To ensure a fair comparison, we followed the same random seed as the baseline. Additionally, we conducted three more runs with different seeds to analyze statistical variation. The table below reports the mean and standard deviation across these runs, demonstrating the consistency and robustness of our method.

Method	Model	Structure		Background Preservation				CLIP Similarity	
		Editing	Distance $\times 10^3 \downarrow$	PSNR $\uparrow$	LPIPS $\times 10^3 \downarrow$	MSE $\times 10^4 \downarrow$	SSIM $\times 10^2 \uparrow$	Whole $\uparrow$	Edited $\uparrow$
FlowEdit	SD3	-	27.11±0.15	22.18±0.05	104.94±0.48	86.54±0.72	83.54±0.05	26.88±0.04	23.72±0.06
SplitFlow(Ours)	SD3	-	25.90 ± 0.09	22.42±0.02	102.48±0.23	82.60±0.41	83.83±0.05	26.93±0.04	23.79±0.05

## C Mathematical Justification of VFA

Here, we provide a detailed mathematical justification for why VFA improves both fidelity and editability over mere averaging:

$$\langle \bar{\mathbf{g}}, \mathbf{g}_{\text{avg}} \rangle \geq \|\mathbf{g}_{\text{avg}}\|^2, \quad (15)$$

where  $\mathbf{g}_{\text{avg}} = \frac{1}{K} \sum_{i=1}^K \mathbf{g}_i$ ,  $S_{kj} = \langle \hat{\mathbf{g}}_k, \hat{\mathbf{g}}_j \rangle$  and  $a_k = \sum_j S_{kj}$ . Since the proposed Latent Trajectory Projection (LTP) is inspired by gradient conflict resolution techniques in multi-task learning, a theoretical justification for this approach can be found in Appendix A of [32].

**Recap.** For each sub-prompt  $k \in \{1, \dots, K\}$  we denote the *relative velocity field* at time  $t$  by

$$\mathbf{g}_k := \mathbf{v}^{\Delta(k)}(x_t^{\text{tgt}(k)}, x_t^{\text{src}}) = v_\theta(x_t^{\text{tgt}(k)}, \varphi^{\text{tgt}(k)}) - v_\theta(x_t^{\text{src}}, \varphi^{\text{src}}).$$

Following Eq. (12) in the main paper, we set

$$w_k = \frac{\exp(\sum_{j=1}^K \langle \hat{\mathbf{g}}_k, \hat{\mathbf{g}}_j \rangle)}{\sum_{i=1}^K \exp(\sum_{j=1}^K \langle \hat{\mathbf{g}}_i, \hat{\mathbf{g}}_j \rangle)}, \quad \bar{\mathbf{g}} = \sum_{k=1}^K w_k \mathbf{g}_k.$$

**Step 1.** To prove Eq. [15](#), we first reformulate the inequality in terms of scores  $a_i$ . The left-hand side (LHS) becomes:

$$\langle \bar{\mathbf{g}}, \mathbf{g}_{\text{avg}} \rangle = \left\langle \sum_{i=1}^K w_i \mathbf{g}_i, \frac{1}{K} \sum_{j=1}^K \mathbf{g}_j \right\rangle = \frac{1}{K} \sum_{i=1}^K w_i a_i$$

The right-hand side (RHS) becomes:

$$\|\mathbf{g}_{\text{avg}}\|^2 = \left\langle \frac{1}{K} \sum_{i=1}^K \mathbf{g}_i, \frac{1}{K} \sum_{j=1}^K \mathbf{g}_j \right\rangle = \frac{1}{K^2} \sum_{i=1}^K a_i$$



Thus, the original inequality is equivalent to proving:  $\sum_{i=1}^K w_i a_i \geq \frac{1}{K} \sum_{i=1}^K a_i$ .

**Step 2.** Gibbs' inequality states that the Kullback-Leibler (KL) divergence between two probability distributions is non-negative. Let  $w = \{w_i\}$  be our softmax distribution and  $u = \{u_i = 1/K\}$  be the uniform distribution.

$$D_{KL}(w||u) = \sum_{i=1}^K w_i \log \frac{w_i}{u_i} \geq 0$$

Substituting  $w_i = e^{a_i}/Z$  (where  $Z = \sum_k e^{a_k}$ ) and  $u_i = 1/K$ :

$$\begin{aligned} \sum_{i=1}^K w_i (\log w_i - \log(1/K)) &\geq 0 \\ \left( \sum_{i=1}^K w_i a_i \right) - \log Z \left( \sum_{i=1}^K w_i \right) + \log K \left( \sum_{i=1}^K w_i \right) &\geq 0 \end{aligned}$$

Since  $\sum w_i = 1$ , this simplifies to:

$$\sum_{i=1}^K w_i a_i \geq \log Z - \log K = \log(Z/K).$$

**Step 3.** Jensen's inequality for a convex function  $f$  states that  $E[f(X)] \geq f(E[X])$ . We apply the logarithmic form,  $\log(E[e^X]) \geq E[X]$ , using the convex function  $f(x) = e^x$  and the uniform distribution over the scores  $\{a_i\}$ .

$$E[e^a] = \frac{1}{K} \sum_{i=1}^K e^{a_i} = \frac{Z}{K} \quad \text{and} \quad E[a] = \frac{1}{K} \sum_{i=1}^K a_i$$

Applying the inequality:

$$\log \left( \frac{Z}{K} \right) \geq \frac{1}{K} \sum_{i=1}^K a_i$$

**Step 4.** By chaining the inequalities from **Step 2** and **Step 3**, we get:

$$\sum_{i=1}^K w_i a_i \geq \log(Z/K) \geq \frac{1}{K} \sum_{i=1}^K a_i.$$

This yields the desired inequality from the end of **Step 1** and completes the proof:

$$\sum_{i=1}^K w_i a_i \geq \frac{1}{K} \sum_{i=1}^K a_i.$$

## D Algorithm Table of SplitFlow

For a better understanding of the overall pipeline of the proposed SplitFlow, we provide an algorithm table as shown in Alg. [S1](#) and overall pipeline in Fig. [S1](#). During the decomposition phase, from  $\eta_{max}$  to  $\eta_{dec}$ , we compute independent flows based on the decomposed sub-target prompts. Following the setting in FlowEdit, where one-third of the 50 inference steps are skipped ( $\eta_{max} = 33$ ), and we adopt the same configuration in our method. Within this phase, each sub-target prompt produces an independent latent trajectory  $x_{t_i}^{FE(k)}$ , which is aggregated at the final step of the decomposition phase. First, we apply Latent Trajectory Projection (LTP) to each trajectory to ensure global coherence with the target direction, resulting in a projected latent  $x_{t_i}^{proj(k)}$ . Next, to preserve the semantic diversity of individual flows, we perform Velocity Field Aggregation (VFA) by combining the velocity fields of the projected latents based on cosine similarity. This aggregated latent is then updated using the original target prompt in a single-flow editing process. And at the last step of editing, we obtain final edited latent  $x_0^{FE}$ .

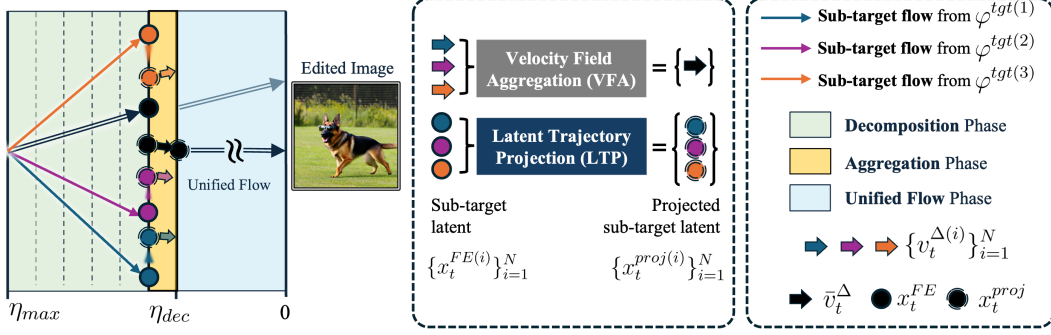


Figure S1: Detailed pipeline of SplitFlow. Given sub-target prompts, we define independent, decomposed flows. When the decomposition phase ends at timestep  $\eta_{dec}$ , we apply Latent Trajectory Projection (LTP) to obtain the projected sub-target latents. The combined velocity field  $\bar{v}_t^\Delta$ , computed via Velocity Field Aggregation (VFA), is used to update the projected target latent. The remaining flow is computed using only the target prompt.

Table S4: Computational cost on PIE-Benchmark. (+X) denotes additional cost from LLM-based prompt decomposition.

Method	GPU Spec.	Max VRAM (GB)	Inference Time (min)	# of Images
FlowEdit	NVIDIA A6000	17.9 (+14.3)	83 (+20)	700
Ours	NVIDIA A6000	17.9	57	700

## E Cost Analysis

For the experiments, we used a single NVIDIA A6000 GPU (49GB VRAM). GPU memory usage and total inference time were measured on the PIE-Benchmark, which includes 700 images. While the proposed SplitFlow requires 15 additional inference steps compared to the baseline (33 steps), we also measured the actual computational cost. As summarized in Tab. S4, the total inference time is 57 minutes for FlowEdit and 83 minutes for SplitFlow. Additionally, prompt decomposition using an LLM takes approximately 20 minutes. Although our method incurs higher computational overhead than the baseline, it remains efficient overall as it is built upon an inversion-free framework—unlike inversion-based editing methods, which are significantly more expensive. Furthermore, our method achieves substantial improvements in both fidelity and editability.

## F Additional Qualitative Results

In Fig. S2 we present additional qualitative comparisons using SplitFlow<sup>†</sup>, our fidelity-enhanced variant. Unlike all prior methods, our proposed approach is able to successfully reflect the target edits as specified by the prompts. While some changes can be observed even in regions outside the intended edit area, our method still shows promising results in maintaining background fidelity—especially when considering the inherent trade-off between editability and fidelity in image editing tasks. It is also worth noting that preserving identity-specific or fine-grained appearance details (e.g., exact person identity) remains a known limitation across nearly all existing editing methods. Our approach nonetheless pushes the boundary by balancing semantic edit success and visual consistency more effectively compared with prior works.

In Fig. S3 and Fig. S4, we provide additional qualitative comparisons in more challenging scenarios. Since SplitFlow is particularly effective when handling complex target prompts, we evaluate our method under such conditions. In the first row, FlowEdit fails to modify the text “CAFE” to “SplitFlow,” instead producing distorted text such as “SPIITFFLOW.” In contrast, our method successfully edits the text to match the target prompt. Similarly, for the instruction “add a yellow car,” FlowEdit incorrectly converts an existing van into a yellow car, whereas our method adds a new yellow car while preserving the original vehicle. In all cases, our method more faithfully reflects the complex semantics of the target prompts, outperforming FlowEdit in both fidelity and editability.

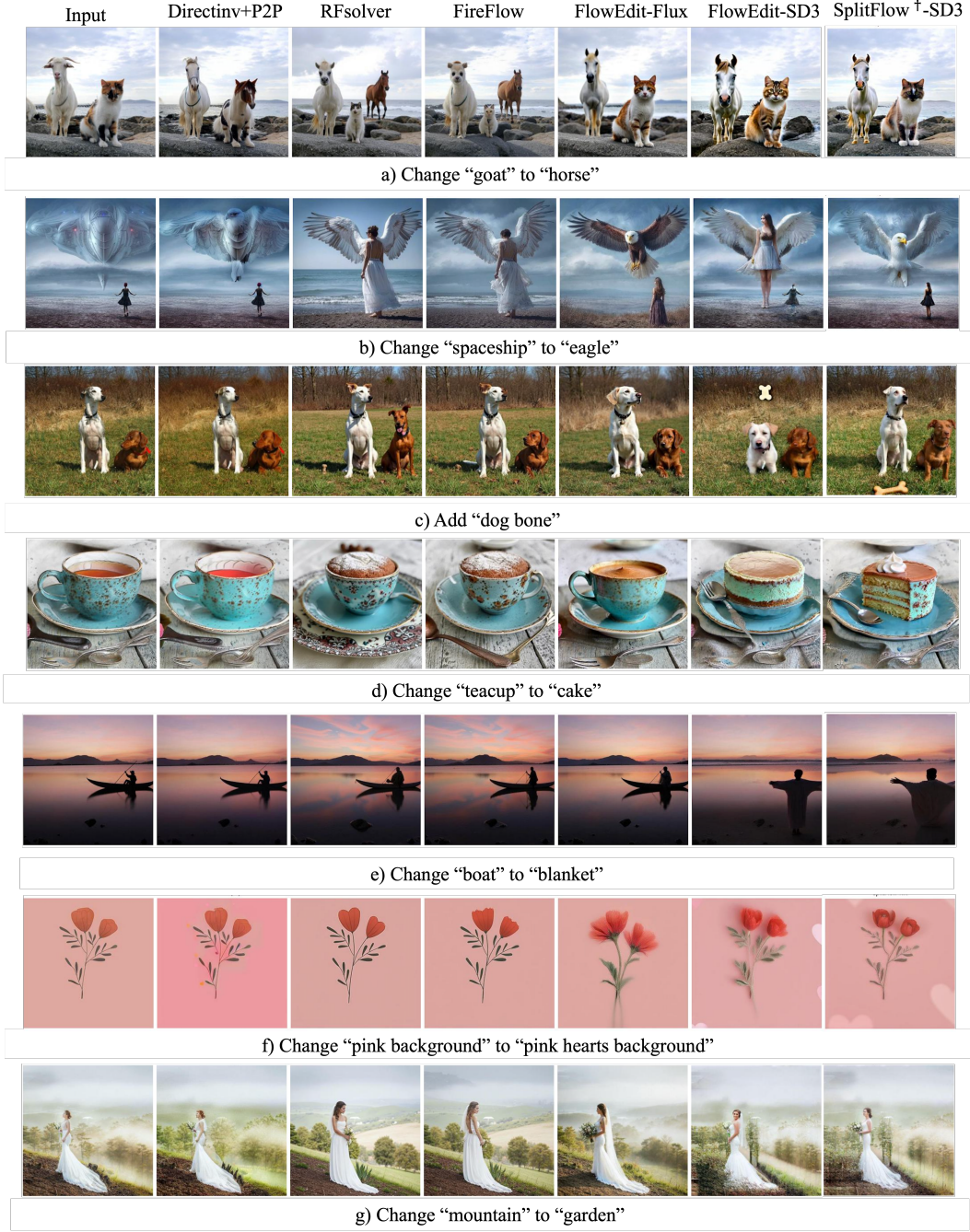


Figure S2: Qualitative comparison of prompt-based image editing methods. Each row corresponds to a specific editing instruction, where the source prompt is modified into a target prompt. From top to bottom, the tasks are: (a) *change* “goat” to “horse”, (b) *change* “spaceship” to “eagle”, (c) *add* “dog bone”, (d) *change* “teacup” to “cake”, (e) *change* “boat” to “blanket”, (f) *change* “pink background” to “pink hearts background”, (g) *change* “mountain” to “garden”. The columns show the input image and the results generated by different models, including Directinv+P2P, RFsolver, FireFlow, FlowEdit-Flux, FlowEdit-SD3, and SplitFlow<sup>†</sup>.





Figure S3: Qualitative comparison results with baseline in more complex editing scenarios.



Figure S4: Qualitative comparison results with baseline in more complex editing scenarios.

---

**Algorithm S1 SplitFlow**


---

**Require:** Source latent  $x_0^{src}$ , source prompt  $\varphi^{src}$ , sub-target prompts  $\{\varphi^{tgt(k)}\}_{k=1}^N$ , target prompt  $\varphi^{tgt}$ , scheduler timestep  $\{t_i\}_{i=0}^T$ ,  $\eta_{max}$ ,  $\eta_{dec}$ .

- 1: Initialize  $x_{t_{max}}^{FE} \leftarrow x_0^{src}$ ,  $x_{t_{max}}^{FE(i)} \leftarrow x_0^{src}$
- 2: **for**  $i = \eta_{max}$  **to**  $\eta_{dec}$  **do** ▷ Decomposition Phase
- 3:    $x_{t_i}^{src} = t_i x_0^{src} + (1 - t_i)\varepsilon$ ,    $\varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  ▷ Interpolate  $x_t^{src}$
- 4:   **if**  $i \geq \eta_{dec}$  **then**
- 5:     **for**  $k = 1$  **to**  $N$  **do**
- 6:        $x_{t_i}^{FE(k)} = x_0^{src} + x_{t_i}^{tgt(k)} - x_{t_i}^{src}$  ▷ Compute independent flow, Eq. (7)
- 7:     **end for**
- 8:     **if**  $i = \eta_{dec}$  **then**
- 9:       Latent Trajectory Projection (LTP)
- 10:        $x_{t_i}^{proj(k)} = \left( \langle x_{t_i}^{FE(k)}, \hat{x}_{t_i}^{FE} \rangle \right) \hat{x}_{t_i}^{FE}$ ,    $x_{t_i}^{proj} = \frac{1}{N} \sum_{k=1}^N x_{t_i}^{proj(k)}$  ▷ LTP, Eq. (8)-(9)
- 11:       Velocity Field Aggregation
- 12:        $v_{t_i}^\Delta(x_{t_i}^{proj(k)}, x_{t_i}^{src}) = v_\theta(x_{t_i}^{proj(k)}, t_i, \varphi^{tgt(k)}) - v_\theta(x_{t_i}^{src}, t_i, \varphi^{src})$  ▷ VFA, Eq. (10)
- 13:        $w_a = softmax(\langle \hat{v}_{t_i}^{\Delta(a)}, \hat{v}_{t_i}^{\Delta(b)} \rangle)$ ,  $\bar{v}_{t_i}^\Delta = \sum_{j=1}^N w_j \cdot v_{t_i}^{\Delta(j)}$  ▷ VFA, Eq. (12)
- 14:       Compute Unified Latent
- 15:        $x_{t_{i-1}}^{FE} = x_{t_i}^{proj} + \bar{v}_{t_i}^\Delta \cdot dt_i$ . ▷ Eq. (13)
- 16:     **end if**
- 17:   **end if**
- 18: **end for**
- 19: **for**  $i = \eta_{dec} - 1$  **to**  $1$  **do** ▷ Unified Flow Phase
- 20:   Compute final flow
- 21:    $x_{t_{i-1}}^{FE} = x_{t_i}^{FE} + \Delta t_i \cdot (v_{t_i}^{tgt} - v_{t_i}^{src})$
- 22: **end for**
- 23: **return** Final edited latent  $x_0^{FE}$

---