On-device High-Resolution Image Editing with Hallucination-Aware Loss and Adaptive Tiling

Young D. Kwon[1] **Abhinav Mehrotra**[2]* **Malcolm Chadwick**[3]*
Alberto Gil Ramos[1] **Sourav Bhattacharya**[3]*
[1]Samsung AI Center-Cambridge [2]Meta [3]NatWest
yd.kwon@samsung.com

Abstract

High-resolution (4K) image-to-image synthesis has become increasingly important for mobile applications. Existing diffusion models for image editing face significant challenges in terms of memory and image quality when deployed on resource-constrained devices. In this paper, we present *MobilePicasso*, a novel system that enables efficient image editing at high resolutions, while minimising computational cost and memory usage. *MobilePicasso* comprises three stages: (i) performing image editing at a standard resolution with hallucination-aware loss, (ii) applying latent projection to avoid going to the pixel space, and (iii) upscaling the edited image latent to a higher resolution with adaptive context-preserving tiling. Our user study with 46 participants reveals that *MobilePicasso* not only improves image quality by 18-48% but also reduces hallucinations by 14-51% over existing methods. *MobilePicasso* demonstrates significantly lower latency, e.g., up to 55.8× speed-up, yet with a small increase in runtime memory, e.g., a mere 9% increase over prior work. Surprisingly, the on-device runtime of *MobilePicasso* is faster than a server-based high-resolution image editing model running on an A100 GPU.

1 Introduction

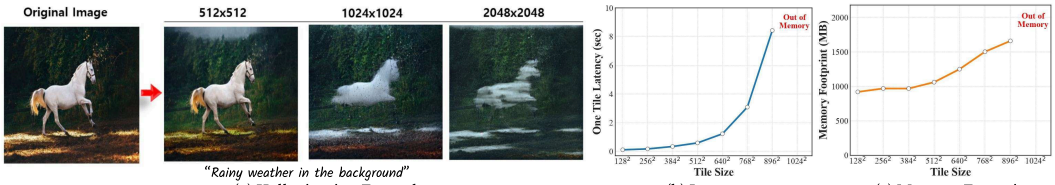
Diffusion-based image-to-image (I2I) synthesis has emerged as a powerful tool for image editing (Sheynin et al., 2023), enabling millions of users to modify images through natural language instructions (Brooks et al., 2022; Wasserman et al., 2024; Zhang et al., 2023a) such as "Rainy weather in the background" or "Make it Pixar style." However, these diffusion models (DMs) are large-scale and computationally intensive, typically requiring cloud-based solutions, so that personal photographs leave the device. Most users would rather avoid this: 69.6% of the 46 participants in our study (see Appendix F) preferred editing to happen entirely on their phone. A lightweight and trustworthy alternative must edit at native screen resolutions, avoid hallucinating[1] content the instruction never asked for, and run within the memory of a mobile NPU.

Enabling on-device image editing with DMs at native mobile resolution presents significant challenges. Firstly, the maximum resolution supported by most models remains limited. Latest models such as SDXL (Podell et al., 2023), SD3-series (Esser et al., 2024), Flux.1-series (Black Forest Labs, 2024) models support resolutions up to 1024×1024, falling short of real-world applications for phones, tablets, and TVs. Popular mobile screen resolutions range between 1080×2640 and 1440×3120, while Tablets and TVs require even higher resolutions. While recent works like DemoFusion (Du et al., 2023) can generate 4K images, its nine-minute processing time on A100 GPUs makes it completely impractical for resource-constrained devices, where such processing delays could severely impact user experience. Secondly, existing I2I generation models, such as InstructPix2Pix (IP2P) (Brooks

*Work done while at Samsung AI Center-Cambridge.

[1]Hallucinations are unrealistic objects or elements generated by DMs that were not intended by the edit instruction, such as distorted faces, floating objects, or implausible scenes (Kirstain et al., 2023; Liang et al., 2024).

40th Conference on Neural Information Processing Systems (NeurIPS 2026). Workshop: LIGHT: Deployable Small Foundation Models.



(a) Hallucination Example (b) Latency (c) Memory Footprint

Figure 1: Figure (a) shows typical examples of the effect of image resolution on I2I generation. As image resolutions get larger starting from 512×512 to 2048×2048, I2I image edit models such as IP2P are often unable to produce realistic images that align well with the edit prompt. The measurements of latency (b) and memory (c) to run U-Net on a single tile according to various tile sizes on Snapdragon 8 Gen 2 NPU.

et al., 2022), MagicBrush (Zhang et al., 2023a), and PIPE (Wasserman et al., 2024), often produce artefacts and hallucinations, i.e., distorted and implausible objects such as a floating lamp (Kirstain et al., 2023; Liang et al., 2024) (see Figure 5) even at a standard resolution (512×512), which becomes more problematic in higher resolutions (see Figure 1 for more examples). Careful human evaluations show that around 30% of the generated images using our test set (Section 4.1) contain hallucinations. Thirdly, mobile devices have limited computational and memory resources, where diffusion model deployment can be problematic. For example, measurements on Snapdragon 8 Gen 2 NPU (on Samsung Galaxy S23) reveal that system overheads increase dramatically with input image resolution (Figure 1) and resolutions over 1024×1024 often lead to an out-of-memory (OOM) error.

While a tiling approach (Song et al., 2024; von Platen et al., 2022) can address memory constraints by processing images in smaller segments or tiles (see Figures 3a and 3b for examples of tiles), it introduces significant new challenges. Processing overlapping tiles with DMs incurs substantial computational overhead that scales quadratically with the overlap ratio between successive tiles. Moreover, small overlaps compromise image quality, producing more artefacts/glitches due to a limited context from neighbouring tiles (see Figure 6).

To address these challenges, we present *MobilePicasso*, which enables high-resolution (4K) on-device I2I generation, addressing computational and memory constraints.

We introduce a **3-stage hybrid pipeline** that *formulates the challenging high-resolution image editing task into three sub-tasks that are easier to tackle*: (1) image editing at standard resolutions (512×512), (2) learnable latent projection, and (3) upscaling stages. Compared to direct image editing at high resolutions, image editing at a standard resolution natively addresses hallucination within the memory constraint of mobile devices as demonstrated in Figures 1 and 6. Also, our learnable latent projection and upscaling enable most operations happening in latent space, further reducing computation by avoiding encoding and decoding steps that commonly exist in image editing (Brooks et al., 2022) and image upscaling (Noroozi et al., 2024) (Section 3.1).

MobilePicasso establishes the first comprehensive hallucination aware training framework for mobile image editing by combining **Hallucination-Aware Loss and Data Filtering** that achieves 14-51% hallucination reduction compared to existing methods (IP2P, MagicBrush, PIPE) (Section 3.2). Through an extensive user study with 46 participants, we demonstrate that *MobilePicasso* significantly outperforms all baselines: *MobilePicasso* achieves 18-48% and 22-54% improvement in overall image quality and text alignment, respectively, compared to existing methods (see Table 2).

We develop **Adaptive Context-Preserving Tiling (ACPT)** to address computational overheads and glitches in the upscaling stage by leveraging our proposed *Adjacent Padding* that provides consistent context around tiles without overlaps (Section 3.3). Additionally, we implement a **model/system co-design approach** to further optimise on-device resource usage, enhancing latency and memory efficiency by carefully analysing tile size and overlap ratios (Section 3.4).

We implemented *MobilePicasso* on a mobile device (Samsung Galaxy S23 with a Hexagon NPU). Our extensive experiments demonstrate that *MobilePicasso* achieves (1) superior high-resolution image quality over prior works that require running the generative model on powerful GPUs, both quantitatively (see Table 5) and qualitatively (see Figure 6), (2) rapid image editing, taking only 42 seconds on Galaxy S23, achieving up to $55.8\times$ speed-up compared to baselines using tiling with overlaps. Surprisingly, *MobilePicasso* is even $4.71\times$ faster than server-based baseline running on

A100 GPU, and (3) affordable memory usage of 1.15 GB, significantly smaller (66.3×) than server-based image editing (see Table 3), throughout the execution of the image editing and upscaling stages with our efficient on-device implementation. Also, we demonstrate cross-platform generalisability across mobile NPUs and cloud GPUs, with consistent performance improvements (Section 4). *Our work paves the way for practical real-world image editing applications on mobile devices by drastically minimising resource overheads without compromising image editing quality.*

2 Related Work

Diffusion-based Image Editing. Image editing tasks have long been investigated in computer vision and graphics communities (Oh et al., 2001; Pérez et al., 2023). The emergence of text-to-image DMs (Esser et al., 2024; Karras et al., 2022; Lin et al., 2024; Lipman et al., 2022; Lu et al., 2024; Peebles and Xie, 2022; Podell et al., 2023; Rombach et al., 2022; Song et al., 2021; Stability AI, 2023b) facilitated the rapid development of text-based image editing, allowing users to seamlessly edit their images with natural language. For instance, to provide a more intuitive and user-friendly tool, IP2P pioneered instruction-based image editing by constructing a large synthetic dataset consisting of paired images generated by Prompt-to-Prompt (Hertz et al., 2022) and corresponding instruction prompts produced by GPT-3 (Brown et al., 2020) using image captions. To improve the quality of the IP2P synthetic dataset, MagicBrush (Zhang et al., 2023a) collected a manually annotated instruction-guided image editing dataset by using an online image editing tool. In addition, PIPE (Wasserman et al., 2024) introduced a large-scale realistic image-editing dataset by leveraging the insight that removing objects is simpler than adding them. This makes PIPE maintain consistency between the source (object-removed) image and the target (original/real) image. Although the edit type of PIPE is limited to only addition, its image quality is superior to all the prior works. Despite the progress in methods and datasets, state-of-the-art (SOTA) image editing methods often generate hallucinated images and struggle to closely follow edit instructions, as demonstrated in Figure 5. In this work, we improve the overall image quality of DMs by proposing hallucination-aware loss and filtering out images with artefacts from a training dataset.

On-device Deployment of Diffusion Models. To deploy large-scale DMs on-device, prior works examined various techniques such as low-precision quantization (e.g., 8-bit weights and activations) to reduce on-device requirements (Wang et al., 2024). SnapFusion (Li et al., 2023) decreased the number of iterations per sample generation and considered smaller and more efficient architectures. HierarchicalPrune (Kwon et al., 2026) compresses large-scale DMs through position-aware block pruning and distillation. Furthermore, there is a resurgence of GANs in the context of distillation of pre-trained DMs, which allow 1-step generation, e.g., SD-Turbo (Stability AI, 2023a) (based on ADD (Sauer et al., 2023)) and MobileDiffusion (Zhao et al., 2023) (based on UFOGen (Xu et al., 2023)). However, none of the prior work focuses on high-resolution (4K) due to the strict memory constraints on mobile devices. At the same time, the image editing task has not been thoroughly investigated.

3 Method

Formulation of Diffusion-based Image Editing. We briefly review diffusion-based image editing (Brooks et al., 2022) and introduce our hallucination-aware loss. DMs are formulated as:

$$\mathbb{E}_{\epsilon,t}\left[\|\epsilon - f_\theta\left(\mathbf{z}_t, t\right)\|\right],$$
$$\boldsymbol{z} \sim p_{\text{data}}(\cdot),\ \boldsymbol{\epsilon} \sim \mathcal{N}(\cdot|\mathbf{0}, \boldsymbol{I}),\ t \in [0,1],\ \alpha_t > 0,\ \sigma_t > 0,$$

where $\boldsymbol{z}_t \coloneqq \alpha_t \boldsymbol{z} + \sigma_t \boldsymbol{\epsilon}$ and noise ϵ increases over timesteps $t \mapsto \alpha_t^2/\sigma_t^2$ is strictly decreasing and $\alpha_1^2/\sigma_1^2 = 0$. $\boldsymbol{z} = \mathcal{E}(\boldsymbol{x})$ is the encoded latent of an image $\boldsymbol{x}$, where $\mathcal{E}$ is the VAE encoder.

In I2I editing task based on DMs, $\boldsymbol{f}$, U-Net processing usually depends on two additional inputs, namely $\boldsymbol{c}_{\text{T}}$ and $\boldsymbol{c}_{\text{I}}$, encoding text and image, respectively. In particular, inspired by IP2P (Brooks et al., 2022) for image editing use case, instead of simply adding these arguments to $\boldsymbol{f}$, we employ classifier-free guidance (CFG) (Ho and Salimans, 2022), which replaces $\boldsymbol{f}$ with $\boldsymbol{g}$. Then, CFG

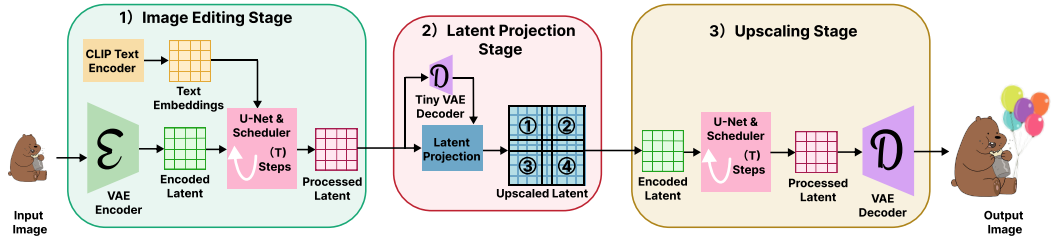


Figure 2: The overview of *MobilePicasso*'s 3-stage hybrid pipeline, which partitions the task of high-resolution image editing into three stages: (1) image editing at standard resolution (512^2), (2) learnable latent projection in latent space, and (3) upscaling to higher resolutions (4K). This modular approach allows *MobilePicasso* to solve each stage effectively and efficiently for deployment.

leverages three function evaluations to construct a new approximation via:

$$\begin{aligned} \boldsymbol{g_\theta}(\boldsymbol{z}_t, t, \boldsymbol{c}_\mathrm{I}, \boldsymbol{c}_\mathrm{T}) \coloneqq & \boldsymbol{f_\theta}(\boldsymbol{z}_t, t, \emptyset, \emptyset) \\ & + s_\mathrm{I}(\boldsymbol{f_\theta}(\boldsymbol{z}_t, t, \boldsymbol{c}_\mathrm{I}, \emptyset) - \boldsymbol{f_\theta}(\boldsymbol{z}_t, t, \emptyset, \emptyset)) \\ & + s_\mathrm{T}(\boldsymbol{f_\theta}(\boldsymbol{z}_t, t, \boldsymbol{c}_\mathrm{I}, \boldsymbol{c}_\mathrm{T}) - \boldsymbol{f_\theta}(\boldsymbol{z}_t, t, \boldsymbol{c}_\mathrm{I}, \emptyset)) \end{aligned} \tag{1}$$

where $s_\mathrm{I} > 0$ and $s_\mathrm{T} > 0$ are hyper-parameters to control image aesthetics. While increasing s_I encourages edited images to closely resemble input images, increasing s_T facilitates edited images to follow the edit prompts closely. Through this process, U-Net takes an encoded latent as input and produces a processed latent, containing an edited image feature. The processed latent is then passed to the VAE decoder $\mathcal{D}$ and converted from latent to pixel space.

3.1 3-Stage Hybrid Pipeline

High-resolution image editing faces significant challenges, including image quality degradation at larger resolutions and resource constraints on mobile devices. To address these challenges, we propose a novel *3-stage hybrid pipeline* that breaks down the complex high-resolution image editing task into three stages, as illustrated in Figure 2: (1) standard-resolution image editing (512×512), (2) learnable latent projection, and (3) upscaling to high resolutions (4K).

Image Edit Stage. *MobilePicasso* conducts image editing at a standard image resolution (e.g., 512×512) using an input image and an edit prompt. Since I2I DMs operate at an image resolution used during pretraining the model (Brooks et al., 2022; Wasserman et al., 2024), they natively produce much fewer hallucinations as demonstrated in Figures 1 and 6, without incurring OOM errors, compared to high-resolution image editing directly. With our novel *hallucination-aware loss* used during this stage, hallucinations are reduced further.

Learnable Latent Projection Stage. Given a processed latent from the image editing stage, we first need to upscale it to a higher resolution space that serves as an input for the subsequent stage. This approach significantly optimises computations by performing most operations in the latent space, eliminating the need of costly encoding and decoding steps as in (Brooks et al., 2022; Noroozi et al., 2024). However, simple linear upsampling (e.g., bilinear, bicubic, and nearest) of the processed latent proves ineffective, often generating low-quality images. To address this challenge, we propose to learn a projection between processed latents from the image edit stage and the upscaled encoded latents used as input during the upscaling stage. Inspired by prior work (Fu et al., 2024), we propose a lightweight learnable projection model that takes the processed latents from the image editing stage and upscales it to a higher-resolution space. This projection model includes a Tiny AutoEncoder (Ollin Boer Bohan, 2024) with only 1.2M parameters and convolution layers with 6K parameters. Overall, our projection model has 68$\times$ fewer parameters, and it is 280-853$\times$ faster in latency compared to simply decoding, upscaling and then encoding.

Upscaling Stage. Lastly, in the *upscaling* stage, *MobilePicasso* uses the super-resolution model (Noroozi et al., 2024) on the upscaled latents to generate high-resolution images (4K). Moreover, in this stage, *MobilePicasso* integrates our proposed *adaptive context-preserving tiling (ACPT)* and *model/system co-design* to minimise the latency for generating high-resolution images.

In summary, our 3-stage hybrid pipeline achieves fully on-device image editing for high-resolution images with significantly improved image quality and faster inference.

3.2 Hallucination-Aware Loss

Prior image editing works have contributed to enhancing the visual quality, text alignment, and fidelity of edited images (Wasserman et al., 2024; Zhang et al., 2023a). However, existing image editing approaches often fail by producing hallucinations even at a standard resolution (512×512). For instance, Figure 5 presents failure cases[2] of prior works such as IP2P, MagicBrush, and PIPE. The previous SOTA image editing approach, PIPE, enhanced image quality metrics; however, PIPE is still limited in its use cases as it primarily focuses on the addition type of prompts (Wasserman et al., 2024).

Hallucination-Aware Loss. We propose *hallucination-aware loss* to mitigate hallucinations by the image editing system at a standard resolution (512×512). Hallucination-aware loss captures the amount of the hallucinated area of the edited images and penalises the model to reduce such hallucination. Specifically, we employ the hallucination detection model (Zhang et al., 2023b) that identifies the unrealistic artefacts within a given image. We then compute the area of artefacts and add it as an additional loss term in the diffusion loss, which acts as a regulariser penalising hallucinations within an edited image. Formally, our hallucination-aware loss L is computed as follows:

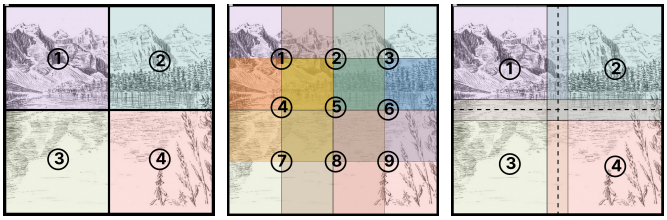
$$\begin{aligned} \hat{\boldsymbol{z}}_0 &:= \left(\boldsymbol{z}_t - \sigma_t \boldsymbol{g}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t, \boldsymbol{c}_{\mathrm{I}}, \boldsymbol{c}_{\mathrm{T}})\right) / \alpha_t \\ L_{LDM} &= \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,1),t,y}\left[\|\hat{\boldsymbol{z}}_0 - \boldsymbol{z}\|_2^2\right] \\ L_{Hallu} &= \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,1),t,y}\left[\mathcal{M}(\mathcal{D}(\hat{\boldsymbol{z}}_0))\right] \\ L &= L_{LDM} + \lambda * L_{Hallu} \end{aligned} \tag{2}$$

where $y = (\boldsymbol{x}, \boldsymbol{c}_{\mathrm{I}}, \boldsymbol{c}_{\mathrm{T}})$ is a triplet of target image, input image, and the edit prompt, $\boldsymbol{z} = \mathcal{E}(\boldsymbol{x})$ is the encoded target latent, $\hat{\boldsymbol{z}}_0$ is the predicted clean latent at timestep t obtained from the guided prediction $\boldsymbol{g}_{\boldsymbol{\theta}}$, $\mathcal{M}$ is the hallucination detection model returning the fraction of the image area predicted as artefacts, and λ controls the dominance of the hallucination-aware loss term.

Hallucination-based Data Filtering. Additionally, we observed that many images from the IP2P dataset, used for training the image editing models such as IP2P and MagicBrush, contain artefacts, similar to the findings in prior work (Liang et al., 2024). This allows the image editing models to generate implausible parts or objects. Therefore, we filter out images that exceed a certain threshold of artefact ratio, to improve the overall image editing dataset quality. We employ the hallucination detection model (Zhang et al., 2023b) to assess the ratio of artefacts in the image editing datasets and then filter out images that exceed a manually set threshold. In our main experiments, we set a threshold, filtering out around 15% of the training data.

3.3 Adaptive Context-Preserving Tiling

Our 3-stage hybrid pipeline largely alleviates the hallucination problem that occurs when directly generating high-resolution images solely based on image-editing models. Applying a tiling-based super-resolution model further requires considerable computational overheads due to the tiling overlaps. For instance, 0% overlap (Figure 3a) causes glitches/seams between neighbouring tiles as illustrated in Figure 6 (bottom row, third column).



(a) 0% Overlap (b) 50% Overlap (c) Adjacent Padding

Figure 3: The on-device tiling strategy with different overlap ratios and our proposed adjacent padding with 0% overlap.

Hence, we propose *Adaptive Context-Preserving Tiling (ACPT)* to solve the

[2]These examples were chosen to show some typical situations we came across, although we do not include every possible case.

tiling issue and improve the end-to-end latency. ACPT consists of two components: (1) adjacent padding and (2) adaptive tiling overlap mechanism.

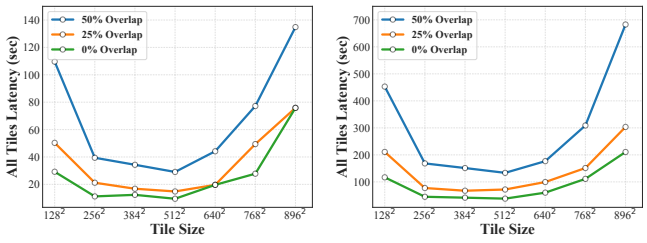
Adjacent Padding. In practice, the default padding strategies are often zero padding which fills padding with zero values or reflect padding which repeats the tile itself in the padding (von Platen et al., 2022). However, these strategies result in image quality degradation as they lack information about neighbouring tiles. Therefore, we propose a new padding strategy, *adjacent padding*, that uses surrounding tiles as padding so that it ensures that padding is naturally smooth and consistent across neighbouring tiles (see Figure 3c). As adjacent padding provides contextual information from neighbouring tiles, super-resolution models can generate smooth images consistent across connected tiles. Notably, our padding strategy enables 0% tile overlap without image quality degradation (see Figure 6) yet negligible memory and computation overheads (see Table 3) over 0% overlap, highlighting its impact on on-device efficiency.

Adaptive Tiling Overlap Mechanism. We develop an *adaptive tiling overlap mechanism* to further optimise the execution of *MobilePicasso* for diverse input resolutions by leveraging adjacent padding and tile overlap adaptively. We provide the detailed ACPT algorithm (see Algorithms 1 and 2). The adaptive strategy selection (Lines 2-21 in Algorithm 1) optimises processing approach based on input characteristics. Specifically, ACPT relies on adjacent padding when input resolution is a multiple of tile size; otherwise, it executes with small overlap ratios between tiles without any padding. Our adaptive mechanism automatically selects the strategy based on input resolutions, ensuring rapid end-to-end processing across diverse input resolutions, while making *MobilePicasso* highly versatile for real-world applications (see Table 5 for ACPT's effectiveness using super-resolution quality metrics). Furthermore, by incorporating contextual information from adjacent tiles, ACPT effectively eliminates visible seams and artefacts commonly occurring when 0% tile overlap or other tiling approaches are used (see Figure 6).

3.4 Model-System Co-design

To further optimise the computational efficiency of high-resolution I2I editing, we propose a *model/system co-design* approach that optimises on-device resource overheads. First, we conduct an extensive empirical analysis of the operational efficiency (latency) of DMs. Our analysis reveals compelling insights into the relationship between these factors and system performance. That is, in multi-tile scenarios, there exist a non-linear relationship where optimal performance is achieved with intermediate tile sizes ranging from 400 to 600 pixels. Notably, our finding challenges the conventional assumption that end-to-end latency consistently increases with tile size when processing a tile, as shown in Figure 4.

This counter-intuitive result stems from the complex interplay between several factors. Firstly, smaller tiles (e.g., 128 pixels) require more inference passes and an increased amount of additional computations due to tiling overlap. Secondly, larger tiles (e.g., 768 pixels) strain device memory and processing capabilities. Finally, intermediate tile sizes (e.g., 384 and 512 pixels) balance these competing factors while maximising NPU utilisation. Note that the performance improvements are substantial: our identified optimal tile size configuration achieves a 3-4× and 4-8× latency reduction compared to smaller (128 pixels) and larger tiles (896 pixels), respectively. We leverage these insights in our model/system co-design, tuning our DMs for image editing and upscaling to operate efficiently with these identified optimal tile sizes, while operating within mobile memory limitations.



(a) 2K Resolution (b) 4K Resolution

Figure 4: Figures (a,b) show latency results of the tiling-based approach for high-resolution images according to different tile sizes and overlap ratios. All measured on the NPU of the Snapdragon 8 Gen 2 chipset.

3.5 System Implementation

MobilePicasso is fully implemented for on-device deployment on the Qualcomm Snapdragon platform, chosen for its prevalence on mobile devices and its NPU capacity (Xu et al., 2025). At runtime, a 2K/4K input is first downsampled with Lanczos resampling so that one spatial dimension is 512 pixels before entering the pipeline of Figure 2. All components (VAE encoder and decoder, both U-Nets, and the latent projection model) are quantised post-training to W8A16 with AdaRound (Nagel et al., 2020) in AIMET (Qualcomm Innovation Center, 2025), exported to ONNX, and converted to Hexagon NPU binaries for the QNN framework (Qualcomm, 2025a), which executes them with integer arithmetic; mixed precision is outside the scope of this work. The target device is a Samsung Galaxy S23 (Snapdragon 8 Gen 2), and all on-device latency and memory figures in Section 4 are measured directly on it with Snapdragon Profiler (Qualcomm, 2025b). Details are described in Appendix E.

4 Evaluation

4.1 Experimental Setup

Datasets: We use IP2P (Brooks et al., 2022) and PIPE (Wasserman et al., 2024) datasets for finetuning the pretrained diffusion model (Rombach et al., 2022). The IP2P dataset contains synthetically generated source and target images with an edit prompt, and the PIPE dataset contains real target images with synthetic source images generated by filling the masked area with an inpainting model. We also use the MagicBrush (Zhang et al., 2023a) dataset to present quantitative evaluation results following Wasserman et al. (2024). In addition, we employ the LIU4K dataset (Liu et al., 2020) consisting of high-resolution images to further evaluate our method.

Architectures: For the image editing stage, we employ SDv1.5 (Rombach et al., 2022) following prior works (Brooks et al., 2022). For the latent projection stage, we use a tiny autoencoder (Ollin Boer Bohan, 2024) and train lightweight convolutional networks to perform projections in the latent space. For the upscaling stage, we use diffusion-based super-resolution model (Noroozi et al., 2024) to leverage its efficient upscaling capability.

Evaluation: To evaluate the performance of *MobilePicasso*, we construct the *MobilePicasso test set* using high-resolution images from LIU4K (Liu et al., 2020) consisting of five categories such as (1) animals, (2) architecture, (3) city, (4) food, and (5) landscape, as well as edit prompts. We developed edit instructions that range from simple attribute modifications to complex style transformations, accounting for 1,000 text-image pairs. We then present qualitative results with edited images using instruction-based DMs on our *MobilePicasso* test set to evaluate visual quality and text alignment. Following Sheynin et al. (2023); Wasserman et al. (2024); Zhang et al. (2023a), we employ various metrics such as L1, L2, CLIP-I, DINO, CLIP-T, and FID. On the system front, we present the latency and memory usage of running *MobilePicasso* on-device. Also, our user study evaluates each method regarding four aspects: (1) Text Alignment: How well a generated image follows the description of the text prompt, (2) Image Consistency: How closely the objects in the generated image match those in the reference images, (3) Resilience to Hallucination: Whether there exist hallucinations in the generated images, and (4) Image Quality: The overall visual quality considering clarity, colour, composition, and other factors.

Baselines: Regarding image quality, we compare *MobilePicasso* with the following image editing baselines: (1) IP2P (Brooks et al., 2022), (2) MagicBrush (Zhang et al., 2023a), and (3) PIPE (Wasserman et al., 2024). In terms of latency, we compare *MobilePicasso* with baselines using (1) direct high-resolution image editing on a GPU and (2) various tiling overlaps (e.g., 0%, 25%, and 50%).

4.2 Main Results

Image Quality at Standard Resolutions: Figure 5 shows example outputs from *MobilePicasso* and the baselines. IP2P is observed to often generate hallucinated images, this is due to their dataset containing hallucinated images derived from the synthetic data generation process. MagicBrush was fine-tuned using the MagicBrush dataset, consisting of authentic images. However, on the *MobilePicasso* test set, its output images have even more hallucinations than IP2P due to its limited diversity of images and prompts (MagicBrush training data is only 10K image-text pairs), similar to

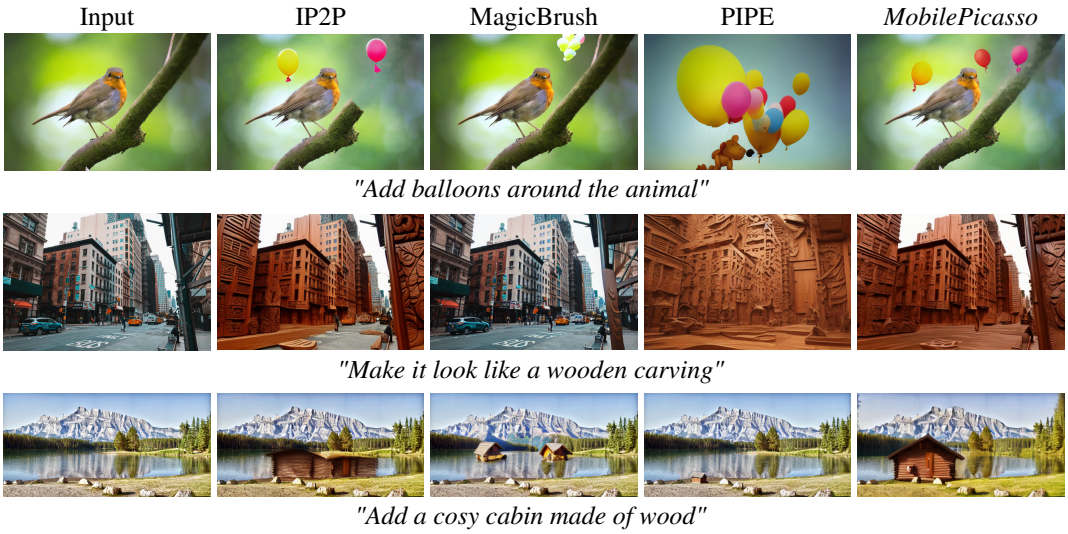


Figure 5: Qualitative comparison among image editing models and *MobilePicasso* given images at standard resolutions (e.g., 512×512).

prior work (Sheynin et al., 2023). PIPE trained on both IP2P and PIPE datasets, which contain around 800K realistic image-edit prompt pairs, improves the visual quality over IP2P and MagicBrush. *MobilePicasso* shows qualitatively better visual quality, outperforming all the baselines.

Following Sheynin et al. (2023); Wasserman et al. (2024); Zhang et al. (2023a), we perform another quantitative evaluation based on the MagicBrush test set. Table 1 shows that *MobilePicasso* achieves the best scores for the image editing task, further demonstrating the superiority of our method. *These results demonstrate the effectiveness of our proposed hallucination loss and artefact filtering process.*

Table 1: Quantitative comparison of *MobilePicasso* and image-editing baselines on the MagicBrush test set.

Method	L1 ↓	L2 ↓	CLIP-I ↑	DINO ↑	CLIP-T ↑	FID ↓
IP2P	0.100	0.031	0.897	0.725	0.269	52.8
MagicBrush	0.077	0.028	0.934	0.843	**0.274**	35.8
PIPE	0.072	0.025	0.934	0.820	0.269	78.4
Ours (Data Filtering)	0.072	0.023	0.938	0.840	0.269	35.6
Ours (+ Hallucination-aware Loss)	**0.069**	**0.021**	**0.940**	**0.845**	0.270	**34.1**

User Study: To provide more compelling and robust statistical evidence regarding hallucination reduction, we conducted an extended user study with 46 participants, and each participant rates four methods with four metrics across eight evaluation sets, resulting in 5,888 user-perceived ratings (see Appendices F and G.1 for study design and domain-specific results, respectively). Table 2 shows that *MobilePicasso* substantially improves Text Alignment (by 22-54%), Resilience to Hallucination (by 14-51%), and Image Quality (by 18-48%) compared to all the baselines except for Image Consistency (by 5-47%) where IP2P shows similar performance to *MobilePicasso*. *These human evaluation results provide much stronger evidence for our claims regarding hallucination reduction, thereby improving overall image quality, than automated metrics alone.*

Table 2: User study on perceived quality of edited images regarding four aspects with mean (standard error) for three baselines and *MobilePicasso*.

Methods	Text Alignment	Image Consistency	Resilience to Hallucination	Image Quality
IP2P	3.65 (±0.15)	4.18 (±0.12)	3.76 (±0.14)	3.73 (±0.14)
MagicBrush	2.89 (±0.19)	2.98 (±0.20)	2.84 (±0.19)	3.04 (±0.16)
PIPE	2.90 (±0.18)	3.20 (±0.20)	2.98 (±0.20)	2.98 (±0.18)
MobilePicasso	**4.45** (±0.09)	**4.39** (±0.09)	**4.29** (±0.10)	**4.40** (±0.09)

Image Quality at High Resolutions: Figure 6 shows the qualitative comparison of IP2P baselines with and without the tiling approach for high-resolution image editing. These baselines do not use our 3-stage hybrid pipeline but apply image editing directly to high-resolution images without subsequent upscaling procedures. First, IP2P without tiling, server-based high-resolution image editing, which simulates that it is free from the on-device memory constraint, often produces images

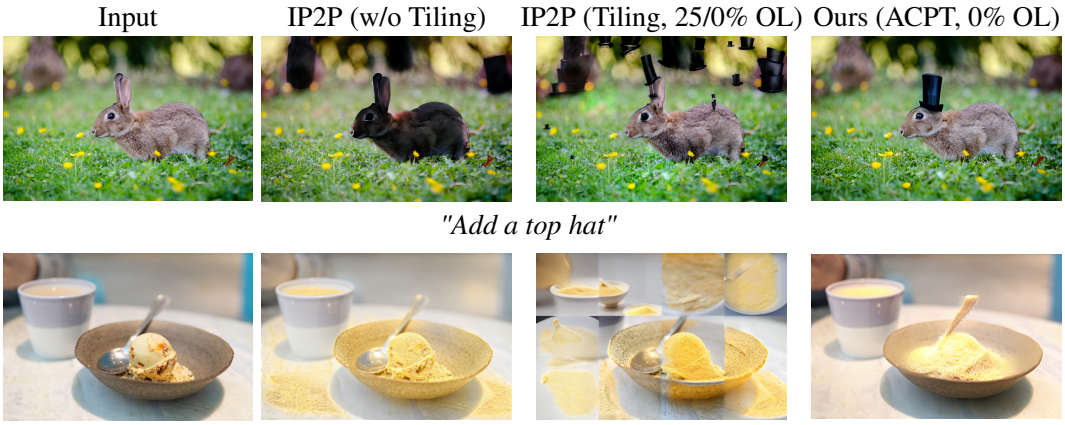


Figure 6: Qualitative comparison among instruction-based image editing models and *MobilePicasso* for high-resolution input images (2K+). (1) Note that the baselines are based on IP2P without the tiling approach to simulate server-based image editing, which serves as the upper bound. (2) The other baselines are IP2P with tiling using different tiling ratios (i.e., 25% for top row, 0% for bottom row), which serve as strong baselines with similar resource constraints to *MobilePicasso*. OL indicates an overlap between neighbouring tiles.

that do not follow the edit prompt well. Second, IP2P with tiling of 25% or 0% overlaps produces much more unrealistic artefacts and glitches/seams between neighbouring tiles, drastically degrading the overall image quality. *Yet, MobilePicasso's 3-stage hybrid pipeline equipped with the proposed hallucination-aware loss and ACPT, significantly improves the image quality by reducing unrealistic artefacts and enhancing prompt alignment while reducing the end-to-end latency, as presented next.*

End-to-end Latency: We now examine the effectiveness of our ACPT by presenting the end-to-end latency results of *MobilePicasso* with our proposed ACPT, server-based baseline without tiling, on-device baseline with tiling (25%, 50% overlaps). Table 3 demonstrates that *MobilePicasso* achieves up to 55.8× reduction in end-to-end latency compared to the baselines with better image quality. *MobilePicasso* achieves up to 55.8× speed-ups over on-device baselines with 50% overlap. Interestingly, despite on-device resource constraints of Snapdragon 8 Gen 2 NPU, *MobilePicasso*'s processing time for high-resolution image editing is even faster than server-based baselines on a powerful GPU (NVIDIA A100) by a large margin (4.71×). *This demonstrates the effectiveness of MobilePicasso's 3-stage hybrid pipeline and ACPT.*

Peak Memory: We investigate the peak memory usage of *MobilePicasso* and the baselines (Table 3). Directly applying image editing on high-resolution images (i.e. baselines without tiling) crashes due to OOM. Thus we report the peak memory consumed on a GPU for this baseline (76.2 GB). The result shows that on-device baselines with tiling and *MobilePicasso* consume significantly smaller memory of 1.06-1.15 GB. *MobilePicasso* incurs a slight memory increase by 9% (peaked at 1.15 GB). Yet, its peak memory is well under the on-device memory constraint of the mobile NPU (e.g. 2 GB).

Table 3: Comparison of the end-to-end latency and memory usage for editing 4K-resolution images. Note baseline (w/o tiling) crashes due to OOM on-device and thus we measured its latency and memory on a GPU simulating a server-based inference scenario.

Device	Method	Latency	Ratio	Memory	Ratio
A100 GPU	Baseline (w/o Tiling)	197s	4.71×	76.2 GB	**66.3**×
Phone	Baseline (w/ Tiling, 50% OL)	2,340s	**55.8**×	1.06 GB	0.92×
Phone	Baseline (w/ Tiling, 25% OL)	1,201s	28.6×	1.06 GB	0.92×
Phone	MobilePicasso	**42.0s**	1×	1.15 GB	1×

Cross-Platform Generalisability: To validate broader applicability, we deploy *MobilePicasso* on NVIDIA A6000 GPUs using identical model configurations, showing significant speedup and memory reduction. The result demonstrates that our method addresses fundamental algorithmic bottlenecks beyond mobile-specific optimisation (see Appendix G.2).

5 Ablation Study and Analysis

Latency Breakdown Analysis of Hybrid Pipeline: Table 4 provides a latency breakdown analysis revealing each MobilePicasso component's contribution. ACPT optimisation (3.12× speedup) eliminates redundant pixel processing from traditional 50% overlap while maintaining visual quality through adjacent padding. Hybrid Pipeline Integration (15.35× speedup) reveals that most significant gain comes from performing editing at 512×512 resolution, avoiding quadratic scaling with image dimensions. Learnable Latent Projection (1.16× speedup) replaces expensive VAE encoding/decoding operations with a single learned transformation in latent space. The multiplicative relationship ($3.12 \times 15.35 \times 1.16 = 55.8$) demonstrates that our optimisations address different computational bottlenecks, validating our systematic decomposition approach.

Table 4: Breakdown of the end-to-end latency and memory usage for editing 4K-resolution images. OL indicates an overlap ratio between neighbouring tiles.

Device	Method	Latency	Ratio	Memory	Ratio
Phone	Baseline (w/ Tiling, 50% OL)	2,340s	1×	1.06 GB	1×
Phone	+ ACPT, 0% OL	749s	**3.12**×	1.15 GB	1.09×
Phone	+ Hybrid Pipeline	48.8s	**15.35**×	1.15 GB	1.09×
Phone	+ Learnable Latent Projection	42.0s	**1.16**×	1.15 GB	1.09×

ACPT Effectiveness Analysis: We quantify ACPT's effectiveness using super-resolution quality metrics, PSNR and SSIM, that capture both global coherence and local detail preservation (Noroozi et al., 2024, 2025; Zhang et al., 2021). Table 5 shows that our ACPT approach achieves 19.73 dB PSNR, outperforming 0% overlap baseline (19.34 dB) and representing 99.7% of the upper bound quality (19.78 dB) based on 50% overlap with significantly lower computational overhead. For SSIM, ACPT achieves 98.5% of upper bound performance (0.5389 vs 0.5470), demonstrating excellent preservation of structural information and perceptual quality. Moreover, we demonstrate that ACPT is applicable across both diffusion-based and GAN-based super-resolution architectures (see Appendix G.5). Figure 8 compares padding strategies qualitatively. Reflect padding produces visible glitches between neighbouring tiles, whereas our adjacent padding provides seamless transitions by using actual neighbouring image content as contextual information.

Table 5: Comparison between ACPT and baselines with various tiling overlaps on 4K-resolution images from the LIU4K dataset. OL indicates a tile overlap.

Metric	50% OL	25% OL	0% OL	0% OL Reflect Padding	ACPT, 0% OL Adjacent Padding
PSNR	**19.78**	19.63	19.34	18.82	**19.73**
SSIM	**0.5470**	0.5386	0.5277	0.5150	**0.5389**

Hallucination-Aware Loss Ablation: We systematically evaluate our hallucination-aware loss and data filtering contributions. Table 1 shows that data filtering improves 17.8% on average across 6 quantitative metrics compared to IP2P. Also, hallucination-aware loss further improves 2.2% on average, with their combination achieving 20.0% overall improvement on average across six metrics. These results demonstrate that both components are necessary and complementary.

6 Conclusion

We have presented *MobilePicasso*, the first realistic 4K image editing framework that runs entirely on mobile devices. Its 3-stage hybrid pipeline, hallucination-aware training, adaptive tiling with adjacent padding, and model/system co-design together deliver higher perceived image quality than prior works at a fraction of their memory and compute, on both mobile NPUs and GPUs.

Limitations. We adopt SDv1.5 (Rombach et al., 2022) because it runs on the Hexagon NPU without architectural changes; SD3.5 (Esser et al., 2024), Flux.1 (Black Forest Labs, 2024) and Qwen-Image (Wu et al., 2025) (8B–20B parameters) exceed NPU memory, and even SDv1.5 exceeds the Galaxy S23 budget at 1024×1024 tiles (Figure 1c). Although we have measured only one device, older chipsets with less NPU memory and lower throughput will face tighter constraints, which we leave to future work.

References

Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. MultiDiffusion: Fusing Diffusion Paths for Controlled Image Generation. In *ICLR:tXdr5k7BfX*, 2023.

Black Forest Labs. Flux.1 Model Family. https://blackforestlabs.ai/announcing-black-forest-labs/, 2024.

Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your Latents: High-Resolution Video Synthesis with Latent Diffusion Models. In *arXiv:2304.08818*, 2023.

Tim Brooks, Aleksander Holynski, and Alexei A. Efros. InstructPix2Pix: Learning to Follow Image Editing Instructions. In *arXiv:2211.09800*, 2022.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners. In *arXiv:2005.14165*, 2020.

Yu-Hui Chen, Raman Sarokin, Juhyun Lee, Jiuqiang Tang, Chuo-Ling Chang, Andrei Kulik, and Matthias Grundmann. Speed Is All You Need: On-Device Acceleration of Large Diffusion Models via GPU-Aware Optimizations. In *arXiv:2304.11267*, 2023.

Jiwoong Choi, Minkyu Kim, Daehyun Ahn, Taesu Kim, Yulhwa Kim, Dongwon Jo, Hyesung Jeon, Jae-Joon Kim, and Hyungjun Kim. Squeezing Large-Scale Diffusion Models for Mobile. In *arXiv:2307.01193*, 2023.

Katherine Crowson, Stefan Andreas Baumann, Alex Birch, Tanishq Mathew Abraham, Daniel Z. Kaplan, and Enrico Shippole. Scalable High-Resolution Pixel-Space Image Synthesis with Hourglass Diffusion Transformers. In *arXiv:2401.11605*, 2024.

Ruoyi Du, Dongliang Chang, Timothy Hospedales, Yi-Zhe Song, and Zhanyu Ma. DemoFusion: Democratising High-Resolution Image Generation With No $$$. In *arXiv:2311.16973*, 2023.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling Rectified Flow Transformers for High-Resolution Image Synthesis. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=FPnUhsQJ5B.

Gongfan Fang, Kunjun Li, Xinyin Ma, and Xinchao Wang. TinyFusion: Diffusion Transformers Learned Shallow. *arXiv preprint arXiv:2412.01199*, 2024.

Stephanie Fu, Mark Hamilton, Laura E. Brandt, Axel Feldmann, Zhoutong Zhang, and William T. Freeman. FeatUp: A Model-Agnostic Framework for Features at Any Resolution. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=GkJiNn2QDF.

Lanqing Guo, Yingqing He, Haoxin Chen, Menghan Xia, Xiaodong Cun, Yufei Wang, Siyu Huang, Yong Zhang, Xintao Wang, Qifeng Chen, Ying Shan, and Bihan Wen. Make a Cheap Scaling: A Self-Cascade Diffusion Model for Higher-Resolution Adaptation. In *arXiv:2402.10491*, 2024.

Yingqing He, Shaoshu Yang, Haoxin Chen, Xiaodong Cun, Menghan Xia, Yong Zhang, Xintao Wang, Ran He, Qifeng Chen, and Ying Shan. ScaleCrafter: Tuning-free Higher-Resolution Visual Generation with Diffusion Models. In *ICLR:u48tHG5f66*, 2024.

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-Prompt Image Editing with Cross Attention Control. In *arXiv:2208.01626*, 2022.

Jonathan Ho and Tim Salimans. Classifier-Free Diffusion Guidance. In *arXiv:2207.12598*, 2022.

Jonathan Ho, Chitwan Saharia, William Chan, David J. Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *J. Mach. Learn. Res.*, 23(1), January 2022. ISSN 1532-4435.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the Design Space of Diffusion-Based Generative Models. In *arXiv:2206.00364*, 2022.

Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-Based Real Image Editing with Diffusion Models. In *arXiv:2210.09276*, 2022.

Bosung Kim, Kyuhwan Lee, Isu Jeong, Jungmin Cheon, Yeojin Lee, and Seulki Lee. On-device sora: Enabling training-free diffusion-based text-to-video generation for mobile devices, 2025. URL https://arxiv.org/abs/2502.04363.

Yuval Kirstain, Adam Polyak, Uriel Singer, Shahbuland Matiana, Joe Penna, and Omer Levy. Pick-a-Pic: An Open Dataset of User Preferences for Text-to-Image Generation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=G5RwHpBUv0.

Abhishek Kumar, Tristan Braud, Young D. Kwon, and Pan Hui. Aquilis: Using contextual integrity for privacy protection on mobile devices. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 4(4), December 2020. doi: 10.1145/3432205. URL https://doi.org/10.1145/3432205.

Young D. Kwon. Efficient on-device intelligence: From continual adaptation to generative ai on edge devices. In *Proceedings of the 24th Annual International Conference on Mobile Systems, Applications and Services Companion*, MobiSys Companion '26, page 86–87, New York, NY, USA, 2026. Association for Computing Machinery. ISBN 9798400727115. doi: 10.1145/3812835.3814808. URL https://doi.org/10.1145/3812835.3814808.

Young D. Kwon, Jagmohan Chauhan, Hong Jia, Stylianos I. Venieris, and Cecilia Mascolo. LifeLearner: Hardware-Aware Meta Continual Learning System for Embedded Computing Platforms. In *2023 21th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2023.

Young D. Kwon, Rui Li, Stylianos I. Venieris, Jagmohan Chauhan, Nicholas D. Lane, and Cecilia Mascolo. TinyTrain: Resource-Aware Task-Adaptive Sparse Training of DNNs at the Data-Scarce Edge. In *International Conference on Machine Learning*, 2024.

Young D. Kwon, Rui Li, Sijia Li, Da Li, Sourav Bhattacharya, and Stylianos I. Venieris. HierarchicalPrune: Position-Aware Compression for Large-Scale Diffusion Models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

Muyang Li, Tianle Cai, Jiaxin Cao, Qinsheng Zhang, Han Cai, Junjie Bai, Yangqing Jia, Ming-Yu Liu, Kai Li, and Song Han. DistriFusion: Distributed Parallel Inference for High-Resolution Diffusion Models. In *CVPR*, 2024.

Muyang Li, Yujun Lin, Zhekai Zhang, Tianle Cai, Junxian Guo, Xiuyu Li, Enze Xie, Chenlin Meng, Jun-Yan Zhu, and Song Han. SVDQuant: Absorbing Outliers by Low-Rank Component for 4-Bit Diffusion Models. In *International Conference on Learning Representations (ICLR)*, 2025.

Yanyu Li, Huan Wang, Qing Jin, Ju Hu, Pavlo Chemerys, Yun Fu, Yanzhi Wang, Sergey Tulyakov, and Jian Ren. SnapFusion: Text-to-Image Diffusion Model on Mobile Devices within Two Seconds. In *arXiv:2306.00980*, 2023.

Youwei Liang, Junfeng He, Gang Li, Peizhao Li, Arseniy Klimovskiy, Nicholas Carolan, Jiao Sun, Jordi Pont-Tuset, Sarah Young, Feng Yang, Junjie Ke, Krishnamurthy Dj Dvijotham, Katie Collins, Yiwen Luo, Yang Li, Kai J. Kohlhoff, Deepak Ramachandran, and Vidhya Navalpakkam. Rich Human Feedback for Text-to-Image Generation, April 2024. URL http://arxiv.org/abs/2312.10240. arXiv:2312.10240 [cs].

Shanchuan Lin, Anran Wang, and Xiao Yang. SDXL-Lightning: Progressive Adversarial Diffusion Distillation. In *arXiv:2402.13929*, 2024.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision (ECCV)*, pages 740–755, 2014.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. In *arXiv:2210.02747*, 2022.

J. Liu, D. Liu, W. Yang, S. Xia, X. Zhang, and Y. Dai. A Comprehensive Benchmark for Single Image Compression Artifact Reduction. *IEEE Transactions on Image Processing*, 29:7845–7860, 2020.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow. In *arXiv:2209.03003*, 2022.

Zeyu Lu, Zidong Wang, Di Huang, Chengyue Wu, Xihui Liu, Wanli Ouyang, and Lei Bai. FiT: Flexible Vision Transformer for Diffusion Model. In *arXiv:2402.12376*, 2024.

Simian Luo, Yiqin Tan, Suraj Patil, Daniel Gu, Patrick von Platen, Apolinário Passos, Longbo Huang, Jian Li, and Hang Zhao. LCM-LoRA: A Universal Stable-Diffusion Acceleration Module. In *arXiv:2311.05556*, 2023.

Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik P. Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On Distillation of Guided Diffusion Models. In *arXiv:2210.03142*, 2022.

Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

Mehdi Noroozi, Isma Hadji, Brais Martinez, Adrian Bulat, and Georgios Tzimiropoulos. You Only Need One Step: Fast Super-Resolution with Stable Diffusion via Scale Distillation, 2024. URL https://arxiv.org/abs/2401.17258.

Mehdi Noroozi, Isma Hadji, Victor Escorcia, Anestis Zaganidis, Brais Martinez, and Georgios Tzimiropoulos. Edge-SD-SR: Low Latency and Parameter Efficient On-device Super-Resolution with Stable Diffusion via Bidirectional Conditioning, 2025. URL https://arxiv.org/abs/2412.06978.

Byong Mok Oh, Max Chen, Julie Dorsey, and Frédo Durand. Image-based modeling and photo editing. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, page 433–442, New York, NY, USA, 2001. Association for Computing Machinery. ISBN 158113374X. doi: 10.1145/383259.383310. URL https://doi.org/10.1145/383259.383310.

Ollin Boer Bohan. Tiny AutoEncoder for Stable Diffusion. https://github.com/madebyollin/taesd, 2024.

William Peebles and Saining Xie. Scalable Diffusion Models with Transformers. In *arXiv:2212.09748*, 2022.

Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 577–582. 2023.

Pablo Pernias, Dominic Rampas, Mats L. Richter, Christopher J. Pal, and Marc Aubreville. Wüerstchen: An Efficient Architecture for Large-Scale Text-to-Image Diffusion Models. In *ICLR:gU58d5QeGv*, 2024.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis. In *arXiv:2307.01952*, 2023.

Qualcomm. Qualcomm AI Engine Direct SDK (QNN). https://www.qualcomm.com/developer/software/qualcomm-ai-engine-direct-sdk, 2025a.

Qualcomm. Snapdragon Profiler. https://www.qualcomm.com/developer/software/snapdragon-profiler, 2025b.

Qualcomm Innovation Center. AI Model Efficiency Toolkit (AIMET). https://github.com/quic/aimet, 2025.

Qualcomm Research. World's fastest ControlNet demo running on a phone. https://www.youtube.com/watch?v=5WBCOIhYFFw, 2023a.

Qualcomm Research. World's fastest Stable Diffusion running on a phone. https://www.youtube.com/watch?v=R5MCj5CFReY, 2023b.

Qualcomm Research. World's first on-device demonstration of Stable Diffusion on an Android phone. https://www.youtube.com/watch?v=B-uojHRr7HE, 2023c.

Qualcomm Research. Low rank adaptation (LoRA) on an Android phone. https://www.youtube.com/watch?v=aEJW1nylYNo, 2024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

Tim Salimans and Jonathan Ho. Progressive Distillation for Fast Sampling of Diffusion Models. In *arXiv:2202.00512*, 2022.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial Diffusion Distillation. In *arXiv:2311.17042*, 2023.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5B: An open large-scale dataset for training next generation image-text models. In *NeurIPS:M3Y74vmsMcY*, 2022.

Shelly Sheynin, Adam Polyak, Uriel Singer, Yuval Kirstain, Amit Zohar, Oron Ashual, Devi Parikh, and Yaniv Taigman. Emu Edit: Precise Image Editing via Recognition and Generation Tasks, 2023. URL https://arxiv.org/abs/2311.10089.

Juyoung Song, Somin Kim, Hyeji Shin, Saemi Choi, Jungmin Kwon, Abhinav Mehrotra, Young Dae Kwon, and Sourav Bhattacharya. Sentimental and object-preserving on-device ai wallpaper. https://research.samsung.com/blog/Sentimental-and-Object-Preserving-On-Device-AI-Wallpaper, August 2024.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-Based Generative Modeling through Stochastic Differential Equations. In *ICLR:PxTIG12RRHS*, 2021.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency Models. In *arXiv:2303.01469*, 2023.

Stability AI. SD-Turbo. https://huggingface.co/stabilityai/sd-turbo, 2023a.

Stability AI. SDXL-Turbo. https://huggingface.co/stabilityai/sdxl-turbo, 2023b.

Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, Dhruv Nair, Sayak Paul, William Berman, Yiyi Xu, Steven Liu, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers, 2022.

Haoxuan Wang, Yuzhang Shang, Zhihang Yuan, Junyi Wu, and Yan Yan. QuEST: Low-bit Diffusion Model Quantization via Efficient Selective Finetuning. In *arXiv:2402.03666*, 2024.

Navve Wasserman, Noam Rotstein, Roy Ganz, and Ron Kimmel. Paint by Inpaint: Learning to Add Image Objects by Removing Them First, 2024.

Mitchell Wortsman, Jaehoon Lee, Justin Gilmer, and Simon Kornblith. Replacing softmax with ReLU in Vision Transformers. In *arXiv:2309.08586*, 2023.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-image technical report, 2025. URL https://arxiv.org/abs/2508.02324.

Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the Generative Learning Trilemma with Denoising Diffusion GANs. In *ICLR:JprM0p-q0Co*, 2021.

Daliang Xu, Hao Zhang, Liming Yang, Ruiqi Liu, Gang Huang, Mengwei Xu, and Xuanzhe Liu. Fast on-device llm inference with npus. In *Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 1*, ASPLOS '25, page 445–462, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400706981. doi: 10.1145/3669940.3707239. URL https://doi.org/10.1145/3669940.3707239.

Yanwu Xu, Yang Zhao, Zhisheng Xiao, and Tingbo Hou. UFOGen: You Forward Once Large Scale Text-to-Image Generation via Diffusion GANs. In *arXiv:2311.09257*, 2023.

Kai Zhang, Jingyun Liang, Luc Van Gool, and Radu Timofte. Designing a Practical Degradation Model for Deep Blind Image Super-Resolution. In *IEEE International Conference on Computer Vision*, pages 4791–4800, 2021.

Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. MagicBrush: A Manually Annotated Dataset for Instruction-Guided Image Editing. In *Advances in Neural Information Processing Systems*, 2023a.

Lingzhi Zhang, Zhengjie Xu, Connelly Barnes, Yuqian Zhou, Qing Liu, He Zhang, Sohrab Amirghodsi, Zhe Lin, Eli Shechtman, and Jianbo Shi. Perceptual artifacts localization for image synthesis tasks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7579–7590, October 2023b.

Yang Zhao, Yanwu Xu, Zhisheng Xiao, and Tingbo Hou. MobileDiffusion: Subsecond Text-to-Image Generation on Mobile Devices. In *arXiv:2311.16567*, 2023.

Supplementary Material
On-device High-Resolution Image Editing with Hallucination-Aware Loss and Adaptive Tiling

Table of Contents

A Detailed Experimental Setup **16**
- A.1 Datasets . . . 16
- A.2 Architectures . . . 16
- A.3 Evaluation . . . 16
- A.4 Baselines . . . 17

B Details of Learnable Latent Projection **17**
- B.1 Design Rationale . . . 17
- B.2 Training Details . . . 17

C Algorithm Specification of ACPT **17**

D Details of Handling High-Resolution Images **19**

E System Implementation Details **19**

F Detailed User Study Design **21**
- F.1 Study Objectives and Methodology . . . 21
- F.2 Experimental Interface and Protocol . . . 21
- F.3 Results Summary . . . 21

G Additional Experimental Results **22**
- G.1 Domain-specific User Study Results . . . 22
- G.2 Cross-Platform Generalisability . . . 22
- G.3 Detailed Latency Breakdown Analysis of Hybrid Pipeline . . . 23
- G.4 ACPT Effectiveness Quantification . . . 23
- G.5 Architecture-agnostic Applicability of ACPT . . . 24
- G.6 Baseline Comparison with Alternative High-Resolution Methods . . . 24

H Extended Related Work **24**
- H.1 Text-To-Image . . . 25
- H.2 (Text,Image)-To-Image . . . 26
- H.3 Text-To-Video . . . 26

I Broader Impact, Ethics, and Future Work of *MobilePicasso* **27**
- I.1 Scientific Impact . . . 27
- I.2 Societal Impact and Ethics . . . 27
- I.3 Future Work . . . 27

J Reproducibility Statement **27**

K Additional Image Editing Results **28**
- K.1 Additional High-Resolution Image Editing Results based on IP2P . . . 28
- K.2 Additional Image Editing Results Comparing *MobilePicasso* and Baselines . . . 28

A Detailed Experimental Setup

A.1 Datasets

We use IP2P (Brooks et al., 2022) and PIPE (Wasserman et al., 2024) datasets for finetuning our pre-trained diffusion model (Rombach et al., 2022). The IP2P dataset contains around 300,000 synthetic source-target image pairs with edit instructions, covering local modifications, object transformations, and style transfer. The PIPE dataset is an even larger dataset with 1M text-image pairs that combine real target images with synthetically generated source images through the inpainting pipeline.

For evaluation purposes, we employ two additional datasets. The MagicBrush (Zhang et al., 2023a) dataset serves as our primary quantitative benchmark, following established protocols in Wasserman et al. (2024). This dataset includes 1,000 diverse test cases with carefully curated edit instructions and ground-truth edited images for comparison. The standardized evaluation protocols enable consistent measurement of editing quality and fidelity across different methods. The LIU4K dataset (Liu et al., 2020) contains 4,000 images at various resolutions such as 4K or 6K. These images encompass diverse photographic styles and subjects, maintaining professional-grade image quality across multiple lighting conditions and scenes. The high resolution and variety make this dataset particularly valuable for evaluating the real-world applicability of our method.

A.2 Architectures

For the image editing stage, we employ SD1.5 (Rombach et al., 2022) following prior works (Brooks et al., 2022). The model uses a UNet backbone with 860M parameters, operating at a standard resolution of 512×512.

Moreover, for the latent projection stage, we use tiny autoencoder (Ollin Boer Bohan, 2024) and lightweight convolutional neural networks (CNNs) to perform projection in latent space. We train our lightweight CNNs to learn a mapping between the processed latent from the image edit stage and the encoded latent from the upscale stage.

Finally, for the upscale stage, we utilise a diffusion-based model (Noroozi et al., 2024) that achieves upscaling through a UNet architecture. The model is optimised for both quality and computational efficiency at inference time.

A.3 Evaluation

***MobilePicasso* Test Set.** To evaluate *MobilePicasso*, we construct the *MobilePicasso test set* using high-resolution images from LIU4K (Liu et al., 2020) across five categories:

- **Animals**: The animal category includes various species photographed in various poses, lighting conditions, and natural habitats, ensuring robust evaluation of fine detail preservation in fur and feathers, etc.
- **Architecture**: The architecture category includes historical monuments, modern buildings, and intricate architectural details.
- **City**: The city category captures urban environments across different times of day, weather conditions, and complex scene compositions.
- **Food**: The food category presents culinary items with varying textures, colours, and presentation styles.
- **Landscape**: The landscape category encompasses natural scenery across different seasons, weather conditions, and lighting scenarios.

For each category in our test set, we developed a comprehensive set of edit instructions that range from simple attribute modifications to complex style transformations. These instructions are carefully crafted to evaluate different aspects of the editing process: general edits with local modifications test precise control and style edits for detail preservation and global adjustments assess consistency in style application. In total, our *MobilePicasso* test set accounts for 1,000 text-image pairs. This diverse instruction set enables a thorough assessment of both editing capability and generalization performance. Our quantitative evaluation framework employs multiple complementary metrics to

assess different aspects of editing quality. Also, our *MobilePicasso* test set is used for qualitative evaluation with visual inspection.

Hallucination Metric. We also measure the hallucination metric. We utilise partial artefacts ratios (PAR), representing the unrealistic portion of an image (Zhang et al., 2023b), as the hallucination metric. PAR indicates undesired artefacts or modifications, providing crucial insights into editing reliability. We perform a quantitative evaluation using the hallucination metric on the *MobilePicasso* test set.

Quantitative Metric. Following established protocols from previous works (Sheynin et al., 2023; Wasserman et al., 2024; Zhang et al., 2023a), we implement a comprehensive suite of evaluation metrics. The L1 and L2 metrics provide pixel-level accuracy measurements, particularly valuable for assessing local editing precision. CLIP-I scores evaluate perceptual quality, measuring how well the edited images align with human visual expectations. DINO features assess semantic consistency, ensuring that edited images maintain appropriate high-level semantic relationships. CLIP-T scores quantify text-image alignment, measuring how accurately the edited images reflect the given instructions.

System Measurements. On the system performance front, we conduct extensive measurements of computational efficiency. Latency measurements capture end-to-end processing time. The memory footprint is measured throughout our hybrid pipeline, with particular attention to peak memory usage during different processing stages. Note that memory requirement becomes a key bottleneck in deploying large-scale DMs as it often incurs out-of-memory errors due to the excessive memory requirements of DMs. Hence, system performance measurements provide valuable insights into real-world deployment scenarios.

A.4 Baselines

Regarding image quality, we compare *MobilePicasso* with the following image editing baselines: (1) IP2P (Brooks et al., 2022), (2) MagicBrush (Zhang et al., 2023a), and (3) PIPE (Wasserman et al., 2024). In terms of latency, we compare *MobilePicasso* with IP2P using (1) direct high-resolution image editing on a GPU and (2) various tiling overlaps (e.g., 0%, 25%, and 50%).

B Details of Learnable Latent Projection

B.1 Design Rationale

Our learned projection layer effectively upscales the edited latent from standard resolution (512x512) to high resolutions (2K/4K) without image distortion, whereas linear projections, such as bilinear, bicubic, all lead to significant image distortion when upscaling from 512 to 2K. Furthermore, the learnable projection layer enables avoiding frequent execution of the VAE decoder and encoder, which incur high memory usage and latency.

B.2 Training Details

We used the MS-COCO dataset (Lin et al., 2014) as a training dataset to train the learnable latent projection component of our 3-stage hybrid pipeline. Our latent projection model includes Tiny AutoEncoder which is frozen during training and a lightweight projection model consisting of multiple blocks of convolutional layers which is trained. We experimented with different numbers of convolutional blocks, [3, 6, 12] and widths of each block, [16, 32, 64, 128]. We used ADAM optimiser and mean-squared error loss. We trained up to 10 epochs, yet we found that the performance converges quickly after 2 or 3 epochs (around 10 hours on a single A100 GPU).

C Algorithm Specification of ACPT

In this section, we provide the detailed algorithm specification of Adaptive Context-Preserving Tiling (ACPT) (see Algorithms 1 and 2), enabling memory-efficient high-resolution image synthesis through intelligent adjacent padding and adaptive tiling.

Design Rationale: The ACPT algorithm addresses the fundamental trade-off between computational efficiency and visual quality in high-resolution image processing. Traditional tiling with overlap ratio r requires processing $(1/(1-r))^2$ times more pixels. For 50% overlap, this represents $4\times$ overhead, making mobile deployment impractical.

Our adjacent padding innovation provides contextual information with minimal computational overhead by using actual neighbouring image content, providing natural contextual transitions that super-resolution models leverage effectively. This approach effectively alleviates the issue of artificial seams/glitches, while maintaining 0% overlap efficiency. Note that traditional padding strategies (e.g., reflection) create artificial glitches as demonstrated in Figure 8 qualitatively and in Table 5 quantitatively.

The adaptive strategy selection (Lines 2-21 in Algorithm 1) optimises processing approach based on input characteristics. When image dimensions are perfectly divisible by tile size, the algorithm employs adjacent padding with 0% overlap for maximum efficiency. For non-divisible dimensions, it gracefully degrades to a small overlap strategy, ensuring consistent quality across arbitrary input resolutions while minimising computational overhead.

Algorithm 1: Adaptive Context-Preserving Tiling (ACPT)

Input: $image$, $model$, $tile_size$, $padding_size$, $overlap_ratio$
Output: $output_image$

1 $height, width \leftarrow image.shape[:2]$;
2 $use_adjacent_padding \leftarrow (width \bmod tile_size = 0) \wedge (height \bmod tile_size = 0)$;
3 $output_image \leftarrow \textbf{zeros_like}(image)$;
4 **if** $use_adjacent_padding$ **then**
    `// Strategy A: Adjacent padding with 0% overlap`
5 $num_tiles_x, num_tiles_y \leftarrow \lfloor width/tile_size \rfloor, \lfloor height/tile_size \rfloor$;
6 **for** $tile_y \leftarrow 0$ **to** $num_tiles_y - 1$ **do**
7 **for** $tile_x \leftarrow 0$ **to** $num_tiles_x - 1$ **do**
            `// Extract tile with adjacent padding`
8 $padded_tile \leftarrow$ ExtractWithAdjacentPadding$(image, tile_x \times tile_size, tile_y \times tile_size, tile_size, padding_size)$;
            `// Process and place in output`
9 $processed_tile \leftarrow model(padded_tile)$;
10 $core_region \leftarrow$ RemovePadding$(processed_tile, padding_size)$;
11 $output_image[tile_y \times tile_size : (tile_y + 1) \times tile_size, tile_x \times tile_size : (tile_x + 1) \times tile_size] \leftarrow core_region$;
12 **else**
    `// Strategy B: Small overlap for non-divisible dimensions`
13 $overlap_pixels \leftarrow \lfloor overlap_ratio \times tile_size \rfloor$;
14 $stride \leftarrow tile_size - overlap_pixels$;
    `// Calculate tile positions and process with blending`
15 $tile_positions_x \leftarrow \{0, stride, 2 \times stride, \ldots\} \cap [0, width - tile_size]$;
16 $tile_positions_y \leftarrow \{0, stride, 2 \times stride, \ldots\} \cap [0, height - tile_size]$;
17 **for** $tile_y_pos \in tile_positions_y$ **do**
18 **for** $tile_x_pos \in tile_positions_x$ **do**
19 $tile \leftarrow image[tile_y_pos : tile_y_pos + tile_size, tile_x_pos : tile_x_pos + tile_size]$;
20 $processed_tile \leftarrow model(tile)$;
21 BlendTilesToOutput$(output_image, processed_tile, tile_x_pos, tile_y_pos)$;
22 **return** $output_image$

Algorithm 2: ExtractWithAdjacentPadding

Input: $image, start_x, start_y, tile_size, padding_size$
Output: $padded_tile$

```
// Extract tile with adjacent padding from neighboring regions
height, width ← image.shape[: 2];
padded_size ← tile_size + 2 × padding_size;
padded_tile ← zeros(padded_size, padded_size, image.shape[2]);
// Extract core tile
core_tile ← image[start_y : start_y + tile_size, start_x : start_x + tile_size];
padded_tile[padding_size : padding_size + tile_size, padding_size :
  padding_size + tile_size] ← core_tile;
// Add adjacent padding from neighboring tiles
if start_y ≥ padding_size then
    top_pad ← image[start_y − padding_size : start_y, start_x : start_x + tile_size];
    padded_tile[0 : padding_size, padding_size : padding_size + tile_size] ← top_pad;
if start_x ≥ padding_size then
    left_pad ← image[start_y : start_y + tile_size, start_x − padding_size : start_x];
    padded_tile[padding_size : padding_size + tile_size, 0 : padding_size] ← left_pad;
// Similar operations for bottom and right padding...
return padded_tile
```

D Details of Handling High-Resolution Images

In this section, we describe the procedure of handling high-resolution images in our 3-stage hybrid pipeline. For example, when a 2K/4K image is given as input, we perform preprocessing using Lanczos resampling to downsample the input image so that at least one spatial dimension to a standard resolution (512). If an input is provided at a standard resolution, then we use it directly to *MobilePicasso*. Then, our system performs the following three stages:

Firstly, it performs editing at a standard resolution using the image-editing model fine-tuned with our hallucination-aware loss, Secondly, it applies projection in the latent space from edited latent to 2K/4K latent space using our learnable latent projection layer, finally, it performs super-resolution on the 2K/4K latent with ACPT and adjacent padding.

Note that direct editing at 4K resolution is fundamentally impossible on mobile devices. As shown in Table 3, running the baseline architecture IP2P without tiling requires 76.2 GB of memory, far exceeding mobile capabilities. Our 3-stage decomposition makes high-resolution editing tractable within mobile constraints.

E System Implementation Details

Runtime Flow: Given a 2K/4K input, *MobilePicasso* first downsamples it with Lanczos resampling so that one spatial dimension is 512 pixels, edits at 512×512 with the model fine-tuned under our hallucination-aware loss (Section 3.2), projects the edited latent to the 2K/4K latent space with the learnable projection (Section 3.1), and runs the super-resolution model over the high-resolution latent with ACPT (Section 3.3).

Model Quantisation and Deployment: We apply post-training quantisation with AdaRound (Nagel et al., 2020) from the AIMET library (Qualcomm Innovation Center, 2025), using W8A16 precision (8-bit weights, 16-bit activations) uniformly across all components: the VAE encoder and decoder, the U-Nets of the editing and upscaling stages, and the latent projection model. Quantised models are exported to ONNX and converted to device-specific binaries for the QNN framework (Qualcomm Neural Processing SDK) (Qualcomm, 2025a), whose optimised kernels execute them with integer arithmetic on the Hexagon NPU. We did not explore alternative quantisation schemes or mixed precision across components, as they are outside the scope of this work.

Hardware Platform and Measurements: *MobilePicasso* is deployed on a Samsung Galaxy S23 with a Snapdragon 8 Gen 2 SoC. All on-device latency and memory figures reported in Section 4 are obtained directly on the device with Snapdragon Profiler (Qualcomm, 2025b).

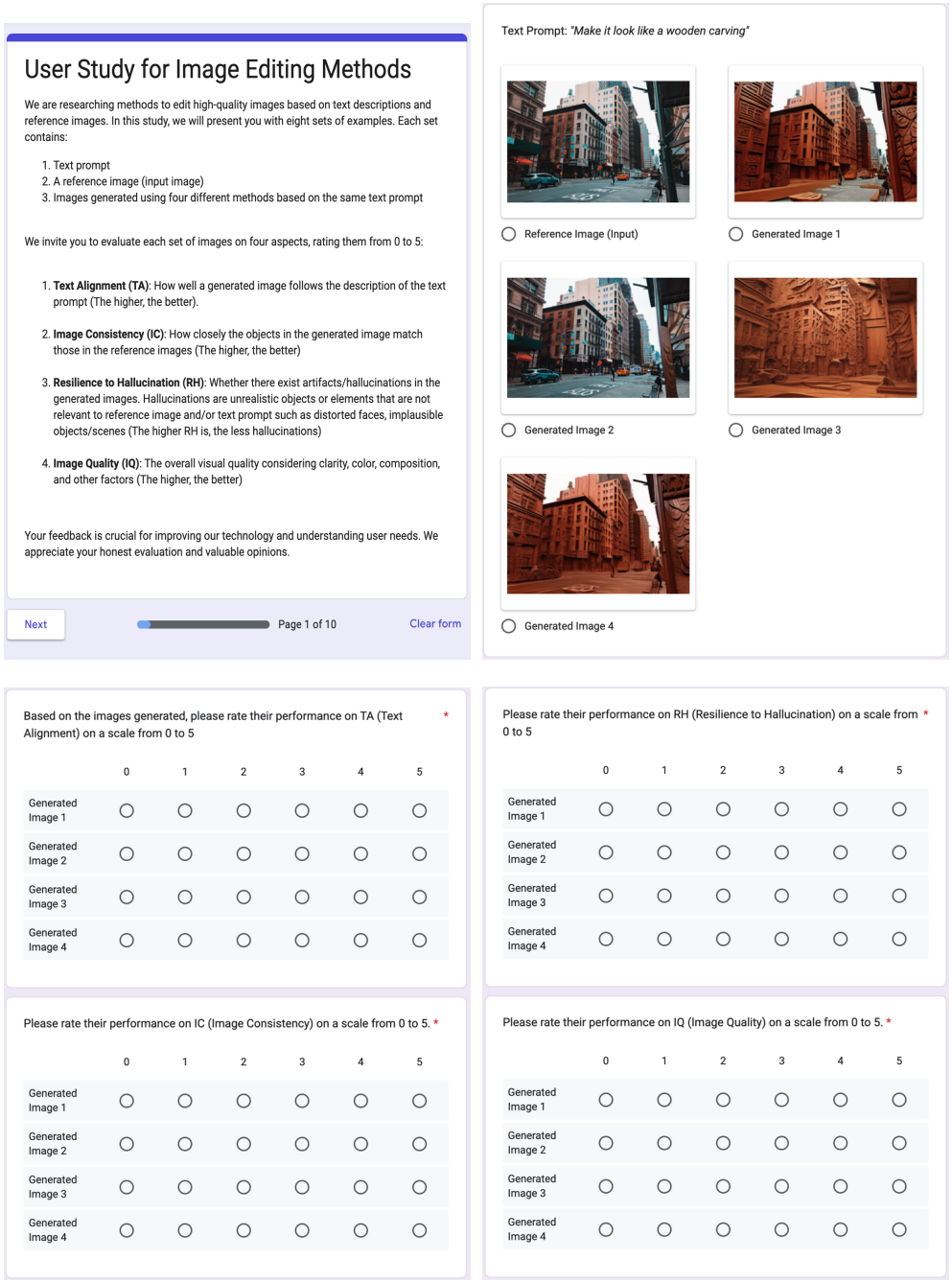


Figure 7: The user study interface that describes the objectives, instructions, evaluation criteria, as well as example image samples.

Training Infrastructure: Training runs on a Linux server with NVIDIA A100 GPUs (80 GB). Hallucination-aware fine-tuning uses eight A100s with PyTorch distributed data-parallel training; the latent projection model is trained on a single A100 to learn the mapping from edited latents to the upscaled encoded latents expected by the upscaling stage.

F Detailed User Study Design

F.1 Study Objectives and Methodology

Our user study validates *MobilePicasso*'s effectiveness in reducing hallucinations and improving image quality through human perceptual evaluation. While automated metrics provide quantitative assessments, human evaluation serves as the golden standard for measuring end-user experience and perceptual quality that automated quality metrics ultimately aim to optimise.

The study tests two primary hypotheses: (1) *MobilePicasso* generates images with significantly fewer hallucinations compared to baseline methods, (2) *MobilePicasso* produces images with better text alignment and overall visual quality. Additionally, we validate correlations between automated metrics and human perception.

Participants: We recruited 46 participants through university networks and online platforms to evaluate the perceptual quality differences between *MobilePicasso* and baseline methods. Participants were asked to assess edited images across multiple quality dimensions without prior knowledge of the methods being evaluated.

F.2 Experimental Interface and Protocol

We conducted the evaluation using Google Forms to ensure accessibility and ease of participation. Each evaluation set presents participants with a reference image, an edit instruction, and four generated images from different methods. Each participant assesses 8 evaluation sets, accounting for 32 total tasks (8 per method: IP2P, MagicBrush, PIPE, *MobilePicasso*). We employ *MobilePicasso* Test Set as reference images (see Appendix A.3 for the dataset details). To ensure randomness and eliminate position bias, we shuffled the location of images between different evaluation sets, so that no method consistently appeared in the same position across multiple tasks. Participants rate each image on four dimensions:

Evaluation Dimensions:

- **Text Alignment:** How well the image follows the edit instruction (0=Poorly following or not following edit instruction, 5=Perfectly following edit instruction).
- **Image Consistency:** How well original elements are preserved (0=Significantly altered, 5=Perfectly preserved).
- **Resilience to Hallucination:** Whether there exist artefacts/hallucinations in the generated images (0=Major artefacts, 5=No unrealistic elements).
- **Image Quality:** Overall visual appeal and quality (0=Poor quality, 5=Superior quality)

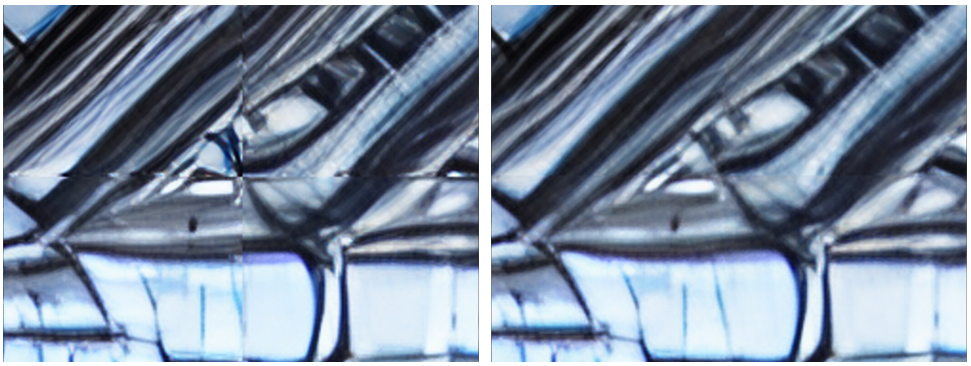
Session Structure: Participants completed the evaluation at their own pace, with clear instructions provided at the beginning. The evaluation included training examples to familiarise participants with the rating criteria, followed by the main evaluation tasks.

F.3 Results Summary

As shown in Table 2, the substantial improvements in human-perceived quality across multiple dimensions demonstrate that our algorithmic innovations address real perceptual limitations rather than merely optimising automated metrics, providing strong evidence for *MobilePicasso*'s practical value and potential for real-world deployment. Figure 7 shows our user study interface.

Also, note that each participant of our user study rates four methods (three baselines and our work) in four metrics (Text Alignment, Image Consistency, Resilience to Hallucination, Image Quality) across 8 evaluation sets, accounting for 128 ratings. In sum, we collected 5,888 user-perceived ratings in our user study, which ensures the robust statistical significance of our results.

Key Qualitative Findings: Participants consistently noted that *MobilePicasso* images "look more natural and realistic" with "better integration of new elements" and "fewer weird artefacts and floating objects." These observations directly validate our technical objectives of reducing hallucinations and improving image quality through our novel architectural and training approaches. Additionally, when participants are asked such a question, "When editing personal photos, where would you prefer the



(a) Reflect Padding (b) Adjacent Padding

Figure 8: Comparison between padding strategies: reflect padding and our proposed adjacent padding. 0% tiling overlap is used.

Table 6: Domain-specific user study results based on image quality metrics across five diverse categories (City, Animal, Landscape, Food, Architecture) using the *MobilePicasso* test set.

Methods	City	Animal	Landscape	Food	Architecture
IP2P	3.74 (±0.12)	4.31 (±0.11)	3.77 (±0.13)	2.67 (±0.15)	3.52 (±0.12)
MagicBrush	3.29 (±0.15)	2.55 (±0.18)	3.03 (±0.19)	3.64 (±0.15)	3.17 (±0.19)
PIPE	2.88 (±0.18)	1.71 (±0.23)	3.67 (±0.13)	2.95 (±0.18)	3.52 (±0.13)
MobilePicasso	**3.86** (±0.11)	**4.56** (±0.09)	**4.38** (±0.08)	**4.71** (±0.08)	**4.33** (±0.09)

processing to happen?", 69.6% of them prefer the image editing to be processed "entirely on their devices", 26.1% has "no preference", and only 4.3% favours the processing to happen "on cloud servers."

G Additional Experimental Results

In this section, we present additional experimental results and analyses that are not included in the main content of the paper due to the page limit.

G.1 Domain-specific User Study Results

Our *MobilePicasso* test set spans five diverse categories (animals, architecture, city, food, landscape) and edit instructions (from simple attribute modifications to complex style transformations), accounting for 1,000 text-image pairs in total, which covers major photographic domains. *This comprehensive coverage demonstrates MobilePicasso's domain generalisation capability within natural image distributions.*

We conducted a comprehensive cross-domain analysis showing consistent baseline gaps across all domains, as shown in Table 6. The consistent performance improvements across diverse domains indicate robust generalisation within our evaluation scope. Notably, our method maintains superiority even in challenging scenarios like city scenes with complex lighting and architectural details.

G.2 Cross-Platform Generalisability

Our evaluation on NVIDIA RTX A6000 demonstrates that the computational benefits of our hybrid pipeline extend beyond mobile deployment. Table 7 shows the significant speedup (19.14×) with and without our system, *MobilePicasso*, on A6000 GPUs validating that our approach addresses fundamental algorithmic bottlenecks rather than hardware-specific optimisation.

Memory efficiency translates well to GPU platforms. The GPU version requires only 6.89 GB compared to 37.6 GB for direct high-resolution editing showing 5.46× reduction. This demon-

Table 7: Comparison of the end-to-end latency and memory usage for editing 2K-resolution images (1340 × 2048) on A6000 GPUs.

Device	Method	Latency	Ratio	Memory	Ratio
A6000 GPU	Baseline (w/o Tiling)	156s	**19.14**×	37.6 GB	**5.46**×
A6000 GPU	MobilePicasso	**8.15s**	1×	**6.89 GB**	1×

strates that our approach enables practical high-resolution editing even on memory-constrained GPU configurations.

G.3 Detailed Latency Breakdown Analysis of Hybrid Pipeline

This subsection presents the detailed description of results shown in Table 4. It provides a latency breakdown analysis revealing the distinct performance contribution of each *MobilePicasso* component. *These results demonstrate the effectiveness of our proposed hybrid pipeline.*

ACPT Optimisation (3.12× speedup): Traditional 50% overlap requires processing more pixel data than necessary. Our ACPT eliminates this redundancy while maintaining visual quality through adjacent padding. The slight memory increase (9%) is negligible compared to the 3.12× speedup achieved.

Hybrid Pipeline Integration (15.35× speedup): The most significant gain comes from our computational decomposition through our hybrid pipeline. Direct high-resolution editing scales quadratically with image dimensions due to attention mechanisms in diffusion models. By performing editing at 512×512 resolution, we reduce computational complexity drastically.

Learnable Latent Projection (1.16× speedup): This component replaces expensive VAE decoding and bilinear upsampling, followed by VAE encoding with a single learned transformation. Maintaining latent representations throughout eliminates costly VAE encoding/decoding operations. The modest gain reflects the already optimised pipeline state. The multiplicative relationship (3.12 × 15.35 × 1.16 = 55.8) indicates that our optimisations address different computational bottlenecks, validating the systematic decomposition approach.

G.4 ACPT Effectiveness Quantification

In this subsection, we present further description of the effectiveness quantification of ACPT in alleviating tiling glitches using super-resolution quality metrics such as PSNR and SSIM that capture both global coherence and local detail preservation (Noroozi et al., 2024, 2025; Zhang et al., 2021). *These results reveal the effectiveness of our ACPT with adjacent padding during the super-resolution stage.*

PSNR Results: Our ACPT approach achieves 19.73 dB, outperforming a baseline with 0% overlap (19.34 dB), representing 99.7% of the upper bound quality (19.78 dB) based on 50% overlap with much lower computational overhead (as shown in the latency breakdown results of Table 4, the impact of ACPT on end-to-end latency is 3.12×). Then, a baseline based on 0% overlap and reflect padding shows substantial degradation of PSNR (18.82 dB), reflecting visible tile boundary glitches, as demonstrated in Figure 8.

SSIM Results: ACPT achieves 98.5% of the upper bound performance (0.5389 vs. 0. 5470) based on 50% overlap, demonstrating that our method particularly excels at preserving structural information and perceptual quality.

Qualitative Comparison of Padding Strategy: We compare our proposed adjacent padding and the default padding strategy, reflect padding. As shown in Figure 8, reflect padding produces a disjoint between neighbouring tiles, whereas our adjacent padding resolves this issue. This is because adjacent padding provides consistent, contextual information surrounding the tile on which the super-resolution model executes.

In addition, we want to highlight the novelty of ACPT. The concept and goal of using neighbouring tiles or global information shared across some prior works, such as MultiDiffusion (Bar-Tal et al., 2023), DemoFusion (Du et al., 2023) and our work, MobilePicasso. However, MultiDiffusion and

Table 8: Comparison between ACPT and baselines with various tiling overlaps on 4K-resolution images from the LIU4K dataset. OL indicates a tile overlap. ACPT with adjacent padding demonstrates consistent improvements across three different architectures.

Architecture	Metric	50% OL	25% OL	0% OL	0% OL Reflect Padding	ACPT, 0% OL Adjacent Padding
YONOS-SR (U-Net)	PSNR	**19.78**	19.63	19.34	18.82	**19.73**
YONOS-SR (U-Net)	SSIM	**0.5470**	0.5386	0.5277	0.5150	**0.5389**
Edge-SD-SR (U-Net)	PSNR	**20.12**	20.06	19.85	19.75	**20.14**
Edge-SD-SR (U-Net)	SSIM	**0.5353**	0.5326	0.5265	0.5244	**0.5352**
BSR-GAN (CNN-Based)	PSNR	**12.44**	12.44	12.43	12.47	**12.47**
BSR-GAN (CNN-Based)	SSIM	**0.3708**	0.3708	0.3704	0.3710	**0.3710**

Table 9: Latency and memory usage of the alternative high-resolution baseline for 2K-resolution image generation.

Device	Method	Latency	Memory
GPU	Original Text-to-Image	90.1s	41.7 GB
GPU	Make a Cheap Scaling	27.9s	**53.0 GB**

DemoFusion use 87.5% and 50% overlaps between tiles to achieve consistency. In contrast, our ACPT aims to reduce overlaps (we achieved 0% overlap) to minimise computational cost while removing glitches with simple yet effective adjacent padding (padding size is merely 6% of the tile), making ACPT a novel solution suitable for efficient and effective mobile deployment of high-resolution image editing.

G.5 Architecture-agnostic Applicability of ACPT

In this subsection, we demonstrate that ACPT is a generalisable technique applicable to both diffusion-based super-resolution models and conventional super-resolution approaches. We demonstrate this by applying ACPT across two substantially different architectures: (1) Edge-SD-SR (Noroozi et al., 2025), a U-Net based diffusion super-resolution model. Note that it is significantly smaller than YONOS-SR (Noroozi et al., 2024) adopted in the upscaling stage of our *MobilePicasso* 3-stage pipeline; (2) BSR-GAN (Zhang et al., 2021), a widely used GAN-based super-resolution model. Note that its architecture is primarily based on convolutional layers, fundamentally different from U-Net based diffusion super-resolution architecture.

As shown in Table 8, our ACPT with adjacent padding consistently achieves over 99.7% of the PSNR performance and 98.5% of the SSIM performance compared to the 50% overlap upper bound baseline across all the evaluated architectures.

G.6 Baseline Comparison with Alternative High-Resolution Methods

In this subsection, we investigated alternative high-resolution image generation methods that rely on the idea of latent projection and late-stage scaling. Among various methods employing the late-stage upscaling approach such as SD-Cascade (Pernias et al., 2024), CDM (Ho et al., 2022), and "Make a cheap scaling" (Guo et al., 2024), we conducted experiments with "Make a cheap scaling" since it enhances inference computational efficiency.

While this alternative high-resolution baseline achieves some inference speedup, its memory usage increases by 27% reaching 53 GB, indicating this approach is not suitable for mobile devices. For 4K resolution, memory exceeds 80 GB, causing OOM errors even on A100 GPUs with 80 GB VRAM.

H Extended Related Work

This paper puts forth a first of its kind diffusion approach via tiling, which allows higher resolution (above 2048×2048) and privacy preserving low-latency on-device deployment for image generation conditioned on text and image inputs. Our work naturally touches upon adjacent fields of text-to-

image and (text,image)-to-image applications, under settings of higher and lower resolutions as well as cloud or on-device deployments.

H.1 Text-To-Image

Lower resolution cloud deployment The holy grail of image generation is to simultaneously achieve fast diverse high-quality sampling. Variational Autoencoders (VAEs), Generative Adversarial Networks (GANs) and Denoising Diffusion Models (DMs) have traditionally only achieved two of the three (Xiao et al., 2021). However, the diverse high-quality samples generated by recent DMs (Karras et al., 2022; Lipman et al., 2022; Lu et al., 2024; Peebles and Xie, 2022; Rombach et al., 2022; Song et al., 2021) captured the imagination of the general public. This sparked a wave of innovation which has decreased DMs sampling time from minutes to seconds across a range of GPU hardware running on desktops or the cloud (Liu et al., 2022; Meng et al., 2022; Salimans and Ho, 2022; Song et al., 2023).

Lower resolution on-device deployment Researchers were quick to notice that many of the latency improvements offered by bleeding edge research work were mostly orthogonal and could be combined to decrease the number of iterations per sample generation as well as to consider smaller and more efficient architectures (Li et al., 2023). Concurrently, GPU-aware optimisations offered additional latency improvements (Chen et al., 2023; Choi et al., 2023). This decreased memory and latency requirements allowing mobile on-device GPU and NPU deployment of lower resolution text-to-image diffusion models, quickly demonstrated with on-device demos (Qualcomm Research, 2023a,c).

Notable recent work considers low bit weights and activations quantisations (e.g., w8a8, w4a8, w4a4) to further reduce on-device requirements (Li et al., 2025; Wang et al., 2024). Furthermore, there is a resurgence of GANs in the context of distillation of pre-trained DMs (Fang et al., 2024), which allow 1-step generation, e.g., SD-Turbo (Stability AI, 2023a) (based on ADD (Sauer et al., 2023)) and MobileDiffusion (Zhao et al., 2023) (based on UFOGen (Xu et al., 2023)). Current state-of-the-art on-device text-to-image generation for 512×512 is 0.6sec per text prompt (Qualcomm Research, 2023b).

Higher resolution cloud deployment Scaling text-to-image DMs to higher resolution is challenging. High-resolution datasets are limited to LAION5B (Schuhmann et al., 2022), and state-of-the-art models such as Stable Diffusion 2.1 require 200,000 NVIDIA A100 GPU hours with training resolution curricula starting at 256×256 for hundreds of steps and moving to 512×512 for thousands of additional optimisation steps. Given the encoder and decoder VAEs robustness to various resolutions as well as the specific convolutional down- and up-scaling architectures of Latent Diffusion Models (LDMs) (Rombach et al., 2022), it has been observed that models trained at 256×256 or 512×512 can run inference at higher 512×512, 512×1024 and 1024×1024 resolutions, albeit with noticeable loss of fine grained detail. This has encouraged higher resolution research in adapting pre-trained lower resolution models to higher resolution settings and multiple aspect ratios.

MultiDiffusion (Bar-Tal et al., 2023) demonstrated the effectiveness of leveraging lower resolution pre-trained DMs to enable higher resolution applications. It trains forward and backward overlapped and consistent patch-based denoising models, enabling panoramic high resolution with arbitrary aspect ratios. The generated images have in general high quality with smooth transitions across the overlapping regions. One key limitation is revealed for text prompts requiring the generation of a single object, often resulting in undesired repetition of the object across the generated panorama. ScaleCrafter (He et al., 2024) and DemoFusion (Du et al., 2023) address the aforementioned repetition issues, in a training-free manner. ScaleCrafter (He et al., 2024) achieves higher resolution generation by dilating certain convolutional layer's kernels of the original pre-trained lower resolution model. DemoFusion (Du et al., 2023) does so via an up-sampling, diffusing, and denoising loop with up-scaling skip connections between diffusion processes at different resolutions, leading to higher latency.

Concurrently, fine-tuning to higher resolution was spearheaded by SDXL (Podell et al., 2023), SDXL-Turbo (Stability AI, 2023b), and SDXL-Lightning (Lin et al., 2024) leveraging multi-aspect training with average pixel count around 1024×1024.

Previous works demonstrated inherent scaling limitations of the first generation of DMs architectures. Initial works curbed latency and memory issues by moving expensive self-attention and cross-attention

blocks towards smaller bottleneck layers, as well as replacing expensive gelu operations with cheaper swish alternatives (Choi et al., 2023) and finetuning softmax into element-wise relu (Wortsman et al., 2023). However, significant gains were only demonstrated with more significant changes such as those introduced in HDiT (Crowson et al., 2024) and Würstchen (Pernias et al., 2024). Notably, Würstchen (Pernias et al., 2024) architectural, training and inference improvements decreased the aforementioned 200,000 GPU hours training cost to 24,602 A100 hours.

Linking together the previous works of leveraging pre-trained models, but with adaptor layers for higher resolution at lower training costs Make a Cheap Scaling (Guo et al., 2024) introduces a self-cascade DM with a two stage approach: a pivot guided noise reschedule and a trainable time-aware feature up-sampler. The first approach is reminiscent of DemoFusion (Du et al., 2023) but denoising only at higher Signal to Noise Ratios (SNRs), thereby bypassing the need for skip connections. Training only the time dependent up-sampler layers yields an impressive 5x training speed up with only thousands of parameters. As a trade-off, however, sampling trajectory length and latency increase at inference.

Finally, Li et al. (2024) introduced displaced patch parallelism for leveraging multiple GPUs to decrease the latency of text-to-image generation distributed across multiple GPUs.

H.2 (Text,Image)-To-Image

Lower resolution cloud deployment InstructPix2Pix (Brooks et al., 2022) pioneered the use of image editing based on text prompts. It did so via a two-stage process. Firstly, it generated (text, original image, edited image) tuple datasets via GPT-3 (Brown et al., 2020) and Prompt-to-Prompt (Hertz et al., 2022). Secondly, it modified the architecture of a pre-trained stable diffusion model by doubling the number of channels of the first convolutional layer and feeding the text as well as the stacked original and edited image to the new model for training. Consequently, (text,image)-to-image models memory and computational requirements are higher but nevertheless resemble their text-to-image counterparts, and the body of work mentioned earlier for the latter can often be applied verbatim to the former. This has spurred a range of cloud offerings for lower resolutions synthesis.

Lower resolution on-device deployment As an example of the relative cost of image generation based on text and image as opposed to only on text. Current state-of-the-art on-device deployment of (text,image)-to-image (Qualcomm Research, 2024) (via Luo et al. 2023) solutions is around 7sec compared to text-to-image (Qualcomm Research, 2023b) at around 0.6sec.

High-Resolution Mobile Editing Challenges: The memory requirements for high-resolution image editing on mobile devices present unique challenges. Tiling-based approaches (Song et al., 2024) have been explored. While such an approach reduces peak memory, it suffers from significant computational overhead due to tile overlaps. Our work addresses these fundamental limitations through the novel ACPT approach and 3-stage hybrid pipeline, achieving better quality-efficiency trade-offs.

Higher resolution cloud deployment At much higher latency and computational requirements per image generation, Imagic (Kawar et al., 2022) introduced the ability to generate truly complex image edits from demanding text prompts including but not limited to the ability to change the posture and composition of multiple objects in an image. At 1024×1024 image resolution, their inference process takes up to 8 minutes per image on two TPUv4 accelerators.

H.3 Text-To-Video

Lower resolution cloud deployment In the context of leveraging pre-trained diffusion models to cater for new applications via adaptors, e.g., for higher resolution (Guo et al., 2024) or additional conditioning (Brooks et al., 2022), VideoLDM (Blattmann et al., 2023) shows that introducing and training temporal and conv3d layers allows higher resolution improvements for text-to-image to translate to higher fidelity video generation.

Recent mobile video generation: On-device Sora (Kim et al., 2025) shows the feasibility of generating short video clips on high-end mobile devices. This work leverages many of the same optimisation strategies developed for image generation while facing additional challenges from temporal consistency requirements and dramatically increased memory demands.

I Broader Impact, Ethics, and Future Work of *MobilePicasso*

Our work builds upon these advances while addressing the specific challenge of high-resolution image editing on mobile devices. Unlike previous works focusing primarily on text-to-image generation at standard resolutions, *MobilePicasso* tackles the significantly more challenging problem of 4K image editing within strict mobile constraints.

I.1 Scientific Impact

Novel Algorithmic Contributions: Our 3-stage hybrid pipeline can be viewed as a novel form of computational decomposition that makes high-resolution editing tractable on mobile devices. The 3-stage hybrid pipeline and ACPT techniques are complementary to existing acceleration methods and could be combined with recent quantisation advances to achieve even greater mobile efficiency.

Broader Scientific Impact: The techniques developed in *MobilePicasso* extend beyond image editing to inform future work in mobile deployment of large-scale AI models (Kwon, 2026). Our approach to cross-domain latent operations, adaptive resource management, and quality-preserving decomposition provides a framework for deploying computationally intensive AI applications on resource-constrained devices (Kwon et al., 2023, 2024).

I.2 Societal Impact and Ethics

Societal Impact: The widespread adoption of cloud-based image editing solutions has raised significant societal concerns regarding privacy (Kumar et al., 2020). Such systems require users to upload personal photos to remote servers, creating privacy vulnerabilities. Our work, *MobilePicasso*, directly addresses these societal concerns by enabling high-quality image editing directly on mobile devices and eliminating the need to transmit sensitive data to external servers. While on-device image editing technology could potentially enable misuse through easier access to image manipulation capabilities on personal devices, the democratisation of image editing tools also provides substantial positive benefits, including privacy protection, offline accessibility, and reduced computational infrastructure dependence.

Ethics: Our user study involving 46 participants was conducted in accordance with standard research ethics practices. Participation was voluntary, and no personally identifiable information was collected. All participants were informed about the study's purpose and provided consent for their participation. Furthermore, this work does not involve sensitive personal data beyond the voluntary user study. The datasets used in our experiments (IP2P, PIPE, MagicBrush, LIU4K) are publicly available benchmarks that have been widely adopted in the research community without ethical concerns. Our mobile device testing was conducted on commercially available hardware without accessing private user data.

I.3 Future Work

Combining *MobilePicasso* with low-bit quantisation (e.g., INT4 via SVDQuant (Li et al., 2025)), pruning (Kwon et al., 2026), and further hardware-aware optimisation could reduce latency and memory beyond the W8A16 configuration reported here. We also expect the core techniques (hallucination-aware loss, ACPT, and model-system co-design) to transfer to other on-device generative tasks, including sketch-to-image generation, style transfer, and image inpainting and outpainting.

J Reproducibility Statement

We have taken several steps to ensure the reproducibility of our results. All datasets employed in this study are publicly available and described in the main text and appendix. Detailed descriptions of model architectures, hyperparameters, and training protocols are provided in Sections 3.2 and 4.1 and Appendix B. Furthermore, the quantisation scheme, export toolchain, target hardware and measurement tools are given in Section 3.5; and the ACPT algorithms are specified in Appendix C. The user-study protocol is documented in Appendix F.

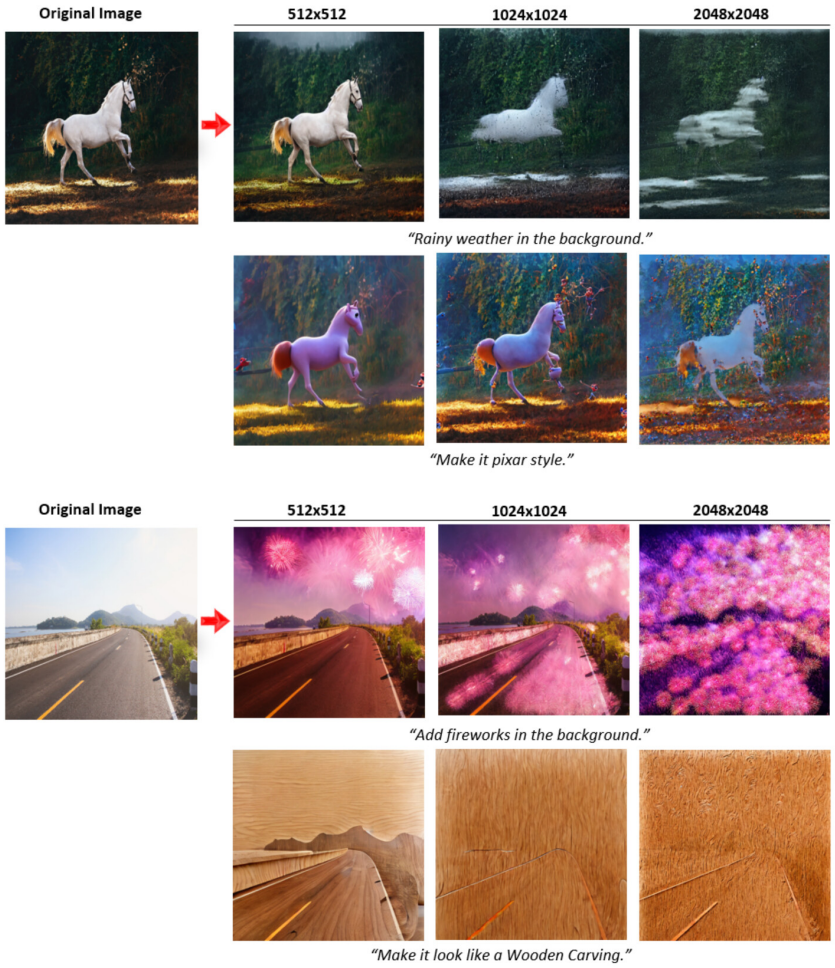


Figure 9: Typical examples of the effect of image resolution on image-to-image (I2I) generation. As image resolutions get larger starting from 512×512 to 2048×2048, I2I image edit models such as IP2P are often unable to produce realistic images that align well with the edit prompt.

K Additional Image Editing Results

In this section, we present additional results that are not included in the main content of the paper due to the page limit.

K.1 Additional High-Resolution Image Editing Results based on IP2P

We present additional results of the image editing model, InstructPix2Pix (IP2P) (Brooks et al., 2022), generating edited images for various image resolutions ranging from a standard resolution of 512×512 to higher resolutions such as 1024×1024 and 2048×2048. As image resolutions get larger starting from 512×512 to 2048×2048, I2I image edit models often cannot produce realistic images, as demonstrated in Figure 9.

K.2 Additional Image Editing Results Comparing *MobilePicasso* and Baselines

In this subsection, we provide additional image editing results to demonstrate the effectiveness of our proposed method, *MobilePicasso*, qualitatively. Figures 10 and 11 show the comparison between our method and all the baselines evaluated in this work.

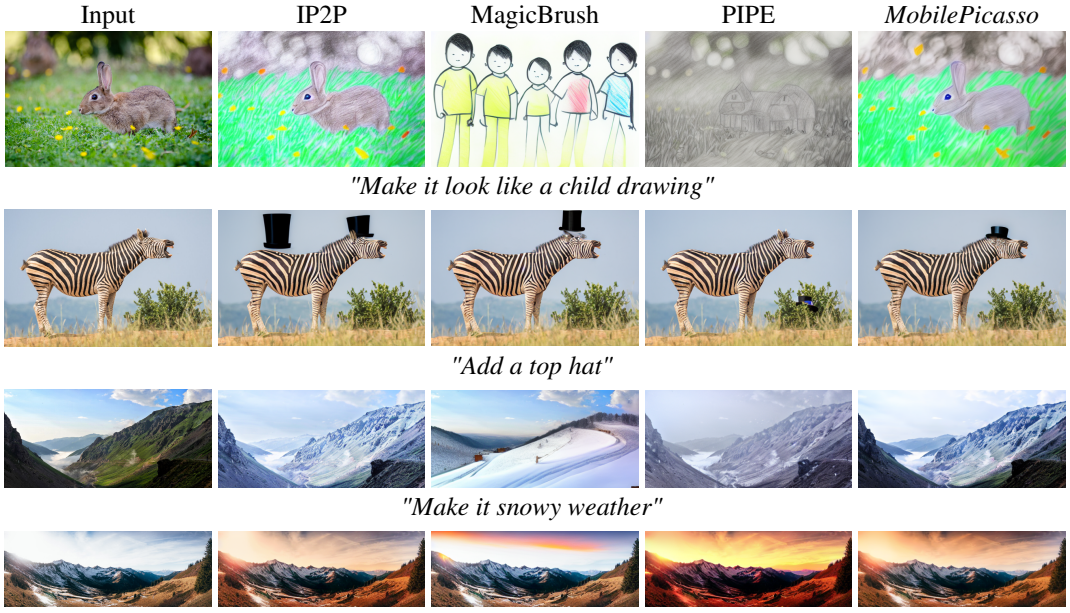


Figure 10: Qualitative comparison among image editing models and *MobilePicasso* given images.



Figure 11: Qualitative comparison among instruction-based image editing models and *MobilePicasso* for high-resolution input images (2K+). (1) Note that the baselines are based on IP2P without the tiling approach to simulate the server-based image editing which serves as the upper bound. (2) The other baselines are IP2P with 0% tiling ratios. OL indicates an overlap between neighbouring tiles.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the contributions and scope of the paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of the paper in Section 6.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: This paper does not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We include experimental settings in Section 4.1, implementation details in Section 3.5, and further details in Appendices A, B and C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: All datasets used (IP2P, PIPE, MagicBrush, LIU4K, MS-COCO) are publicly available and cited; the training code and on-device binaries are not released with this submission.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: We include the details of training and inference settings in Sections 3 and 4, and Appendices A, B, and C.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We report mean and standard error for the user study across 46 participants (Tables 2 and 6), which supports the main claims of the paper.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We include compute resources in Section 3.5 and Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conforms, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss both potential positive and negative societal impact, with mitigations, in Appendix I.2.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We discuss these points in Appendix I.2.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We include citations to the original creators of the code, data and models used in this work.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: This paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: The instructions, rating criteria and interface screenshots are included in Appendix F.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The study involved anonymous rating of images with no foreseeable risk.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.