Distributed and Controllable Mobile Text-to-Image Generation With User Preference Guarantee

Yuxin Kong, Peng Yang, *Member, IEEE*, Xue Qin, Jizhe Zhou, and Xuemin Shen, *Fellow, IEEE*

***Abstract*—In this paper, we investigate controllable mobile text-to-image generation at scale, considering diverse user preferences. In particular, we observe that, by incorporating visual conditions (e.g., Canny maps and depth maps) as supplementary inputs alongside text prompts, fine-grained and controllable image generation could be achieved. To this end, we propose a system design for distributed and controllable mobile text-to-image generation by leveraging edge computing. This system can satisfy diverse user-specified quality preferences at reduced transmission cost through effective cooperation of mobile and edge computing. In particular, the proposed system consists of a *Visual Condition Engineering* module and a *Distributed Denoising Control* module. Since extensive profiling reveals that different visual conditions affect both generation quality and sensitivity to image encoding parameters, the first module selects the optimal configuration of user-specific visual condition on mobile devices. Key to this module is a Pareto Frontier-based model which subtly balances user-preferred generation quality and transmission efficiency. The second module enables collaborative generation by adaptively distributing denoising tasks between mobile devices and the edge server, according to their available computing resources. At the core of this module is an efficient deep reinforcement learning algorithm designed to optimize the dynamic distribution of denoising tasks. By integrating the deep diffusion model, this algorithm achieves superior action space exploration capabilities while maintaining fast convergence and reliable execution, thereby facilitating enhanced adaptability under variable computing resource scenarios. Extensive experimental results reveal that, the designed system can achieve a reduction in transmission cost by over 90% and enhance user satisfaction by up to 18%, with consistent performance across various diffusion models under diverse resource constraints.**

***Index Terms*—Text-to-image generation, edge computing, user preference, distributed systems.**

I. Introduction

RECENT years have witnessed Artificial Intelligence-Generated Content (AIGC) gaining prominence as a novel paradigm for content automation, offering significant advantages in enhancing creativity and enabling user customization on an unprecedented scale. Notable applications are ChatGPT [1], which engages users in interactive prompt-based dialogues, and Stable Diffusion [2], which creates stylistic images from textual descriptions. Those applications have demonstrated the immense potential of AIGC technology. As generative contents continue to evolve, they are poised to play an increasingly integral role in shaping personalized experiences and offering solutions that were once impossible in traditional content creation processes.

However, the superior capabilities of AIGC technologies are primarily driven by the ever-expanding parameter sizes of foundation models [3]. For example, the GPT-3 model includes 175 billion parameters, while that of GPT-4 has surpassed one trillion [4]. Such escalation in model size presents substantial challenges for efficient operation on resource-constrained mobile devices. With the increasing availability of edge servers featuring considerable computational resources near mobile users [5], [6], [7], it has recently become feasible to deliver AIGC services via mobile edge computing paradigm, i.e., *mobile AIGC services* [8], [9]. Through the deep integration of AIGC technologies with mobile edge networks, mobile users can offload their requests to nearby edge servers, thus facilitating distributed yet efficient generation.

Existing studies have made notable strides in advancing mobile AIGC tasks, particularly in text-to-image (T2I) generation [10], [11], [12]. However, they often disregard user transmission costs, given that this task involves merely uploading text prompts with negligible bandwidth usage. As users increasingly seek finer control over the spatial structure of generated images, they tend to provide source images together with text prompts to achieve a precise image generation, which significantly raises bandwidth consumption. A representative approach is utilizing ControlNet [13] for controllable T2I generation, as shown in Fig. 1. It allows images to be generated based on not only the text prompt but also the visual conditions, such as Semantic Segmentation (Seg) maps [14] and Holistically-nested Edge Detection (HED) maps [15]. Despite offering better controllability, this method involves uploading source images to the server for extracting visual conditions, which significantly increases the transmission overhead. For example, uploading an uncompressed image of $512 * 512$ resolution requires up to 768 KB of data volume. In scenarios where many concurrent users accessing AIGC services, the available bandwidth resources allocated to each user

Received 3 May 2025; revised 24 August 2025; accepted 2 October 2025. Date of publication 13 October 2025; date of current version 4 February 2026. This work was supported in part by the Young Elite Scientists Sponsorship Program by CAST under Grant 2022QNRC001 and Grant 2023QNRC001, and in part by Beijing Natural Science Foundation under Grant L253004. Recommended for acceptance by F. Wang. *(Corresponding author: Peng Yang.)*

Yuxin Kong and Peng Yang are with the School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: yxkong@hust.edu.cn; yangpeng@hust.edu.cn).

Xue Qin and Xuemin Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: x7qin@uwaterloo.ca; sshen@uwaterloo.ca).

Jizhe Zhou is with China Academy of Information and Communications Technology, Beijing 100191, China (e-mail: zhoujizhe@caict.ac.cn).

Digital Object Identifier 10.1109/TMC.2025.3620352

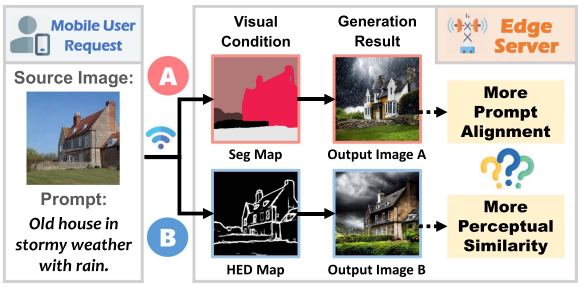


Fig. 1. Examples of controllable T2I generation by mobile edge computing.

may be further restricted [16], causing significant transmission costs.

Moreover, considering the diversity of subjective user preferences, it is non-trivial to improve the Quality of Experience (QoE) for users with controllable T2I generation demands. As depicted in Fig. 1, using the source image of a house and the text prompt of *Old house in stormy weather with rain*, visual conditions are extracted in the form of Seg map for example Ⓐ and HED map for example Ⓑ, resulting in the corresponding generated images. While both outputs exhibit high quality, they prioritize distinct characteristics. Result Ⓐ demonstrates exceptional alignment with the text prompt, accurately representing all specified elements (e.g., stormy weather). In contrast, result Ⓑ achieves superior perceptual similarity, preserving the structural integrity of the source image (e.g., the house layout). This reveals the quality trade-offs inherent in controllable T2I generation, where user preferences can diverge between emphasizing prompt alignment and perceptual similarity. As a result, it is imperative to determine the appropriate visual condition catering to varying user preferences. Moreover, considering that visual conditions serve primarily to offer structural or semantic guidance for image generation, without the requirement to retain pixel-level fidelity, compressing them appropriately presents further opportunities to reduce the transmission overhead.

However, even with well-optimized visual conditions, guaranteeing optimal performance for all users in mobile AIGC scenarios remains challenging. The key issue stems from the overwhelming computational demands of AIGC tasks from multiple users, which often outstrip the constrained computational capacity of the edge server. Moreover, to achieve higher-quality generation results, increased computational resources are required. For instance, in the latent diffusion model-based AIGC services studied in our work, a greater number of denoising steps generally consumes more computing resources but also enhances generation quality [17]. Nonetheless, beyond a certain quality level, further increases in denoising steps contribute only marginal improvements [18]. This highlights the need for careful management of limited edge computing resources to optimize the overall generation quality. Moreover, considering that user devices are increasingly equipped with enhanced computing resources capable of performing AIGC tasks, these resources can be effectively leveraged to support extra denoising steps for performance improvement.

To address the above challenges of *high transmission overheads*, *diverse user preferences* and *limited computational resources*, in this paper, we introduce an efficient distributed service system for mobile controllable T2I generation. By enabling distributed image generation between mobile devices and edge server, our system effectively reduces transmission costs and improves user satisfaction. Specifically, our system starts with *Visual Condition Engineering* executed locally on user devices, determining the most suitable visual condition type and its compression parameters for each user to minimize transmission overhead through a Pareto Frontier-based algorithm. Subsequently, the engineered visual condition images, along with the text prompts, are transmitted to the edge server for generation. In order to enhance both generation quality and resource efficiency, we introduce a distributed denoising strategy. This approach allows the edge server to handle part of the denoising computation while leaving the rest to the mobile device, with only the intermediate latent being transmitted. Therefore, upon receiving all generation requests, the edge server performs *Distributed Denoising Control* to optimally allocate computing resources and make task offloading decisions for each user through a diffusion-enhanced deep reinforcement learning algorithm, thereby improving the overall generation quality. Our main contributions are summarized as follows.

- We introduce a distributed service framework for mobile controllable T2I generation. Our design features uplink transmission of compressed visual conditions and downlink transmission of latent information, significantly reducing communication overhead while maintaining user-expected generation quality.
- We perform comprehensive profiling of diverse visual conditions across varying compression settings, analyzing their quality and communication overhead performances. Building on these insights, we develop a user-centric VCE algorithm for *Visual Condition Engineering* to perform local optimization of visual condition configurations.
- We present a collaborative denoising strategy that orchestrates the T2I generation process between mobile users and the edge server. Central to this strategy is the *Distributed Denoising Control* (DDC) algorithm, which integrates deep diffusion models with deep reinforcement learning (DRL) to dynamically optimize denoising steps and task offloading ratios for each mobile user.
- We evaluate the proposed system through extensive experimentation. The results demonstrate that our solution achieves a reduction in transmission overhead by over 90% and satisfies up to 18% more user requests.

The remainder of the paper is organized as follows. We first present related works in Section II. Section III presents experiments that motivate our work. Section IV elaborates on our system model and problem formulation. Section V describes the details of our algorithm design. Section VI demonstrates our extensive evaluation results. Finally, we conclude our work in Section VII.

II. Related Work

In this section, related work on mobile AIGC services and controllable text-to-image generation is presented.

A. Mobile AIGC Services

Recent studies have delved into the potential power of mobile AIGC services, with the goal of swift service delivery and improved user experiences. Liu et al. [19] introduced an efficient context-loading module that compresses key and value tensors at various levels, aimed at optimizing both overall latency and generation quality in large language model serving systems. Du et al. [20] proposed a distributed AIGC framework based on generative diffusion models. This approach allows semantically similar prompts from different users to share the same diffusion steps, thereby enhancing QoE performance and reducing resource consumption. Furthermore, Liu et al. [12] introduced semantic communications in mobile AIGC to circumvent downlink bandwidth constraints through attention-aware semantic extraction, encoding and prompt engineering. While prior research has established a strong foundation for mobile AIGC services, the exploration of end-edge collaborative solutions for both efficient communication and computation remains limited.

B. Controllable T2I Generation

In order to offer users more refined control over the spatial composition of generated images, Zhang et al. [13] introduced ControlNet, an efficient controllable T2I generation framework. This approach allows generative models to produce images that are guided not only by textual prompts but also by user-specified conditioning images. Then ControlNet-XS [21] is introduced to deliver superior performance with significantly smaller model parameters, making it a promising solution for resource-limited end devices. Some recent works have integrated such techniques into mobile AIGC services. Liu et al. [22] employed the controlled T2I generation as the generative encoder and decoder for semantic communications, further revealing how variations in input information extraction can impact the generation results. Although controlled T2I generation allows for more precise and interactive content creation, efficiently extracting and transmitting the conditioning image for generating user-preferred content still presents a challenge in mobile computing scenarios.

III. Motivation

In this section, we present the experimental results and analysis that motivate our work.

A. Impact of Visual Conditions

We begin with the four most commonly used visual conditions: Canny map [23], HED map [15], Depth map [24], and Seg map [14]. In order to explore how different visual conditions influence generation quality, we first extract the above four visual conditions for each image in the *DreamBooth* dataset [25]. Next, we employ Stable Diffusion v1.5[1] with ControlNet v1.1 [13] to conduct controllable T2I generations for each prompt-condition pair from the dataset.

In the controllable T2I generations, the perceptual similarity and prompt alignment of the generated images are often complementary. Perceptual similarity ensures that the generated image closely resembles the reference in terms of textures, shapes, colors, and other visual details, which is critical for maintaining the visual consistency of the output. On the other hand, prompt alignment evaluates whether the generated image accurately reflects the meaning of the input context, ensuring the generated content conveys the correct message. Relying solely on perceptual similarity may result in semantic deviations, while focusing only on prompt alignment could lead to a decline in image similarity. Therefore, a comprehensive evaluation from both aspects is necessary.

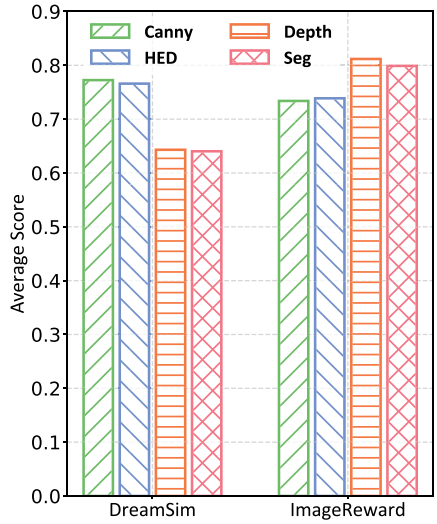


Fig. 2. Average generation scores under different visual conditions.

For perceptual similarity evaluation, the DreamSim (DS) [26] metric is employed. This approach effectively connects low-level metrics with high-level measures, resulting in improved alignment with human similarity perception. Additionally, we utilize ImageReward (IR) [27] to evaluate prompt alignment. As the first general-purpose human preference reward model for text-to-image synthesis, IR is particularly effective in understanding human preferences. Compared to other alternative metrics, both the adopted two not only exhibit stronger alignment with human judgments in recent studies, but also provide more robust and consistent measurements across diverse visual conditions and inference settings. In our work, both metrics are normalized to ensure a fair evaluation. After normalization, higher values for both DS Score and IR Score indicate better quality in the generated images.

The average quality score results for both metrics under different visual conditions are illustrated in Fig. 2. It can be observed that, the Canny and HED maps attain greater perceptual similarity, although with reduced prompt alignment. Conversely, the Depth and Seg map show enhanced prompt alignment but suffer from lower perceptual similarity. The rationale is that both the Canny and HED algorithms are designed to focus on the edge detection and gradient information, thus enhancing the visual structure and similarity of generated images. In contrast, Depth and Seg maps prioritize the semantic aspects of an image by providing information about object boundaries and spatial

[1] https://huggingface.co/runwayml/stable-diffusion-v1-5

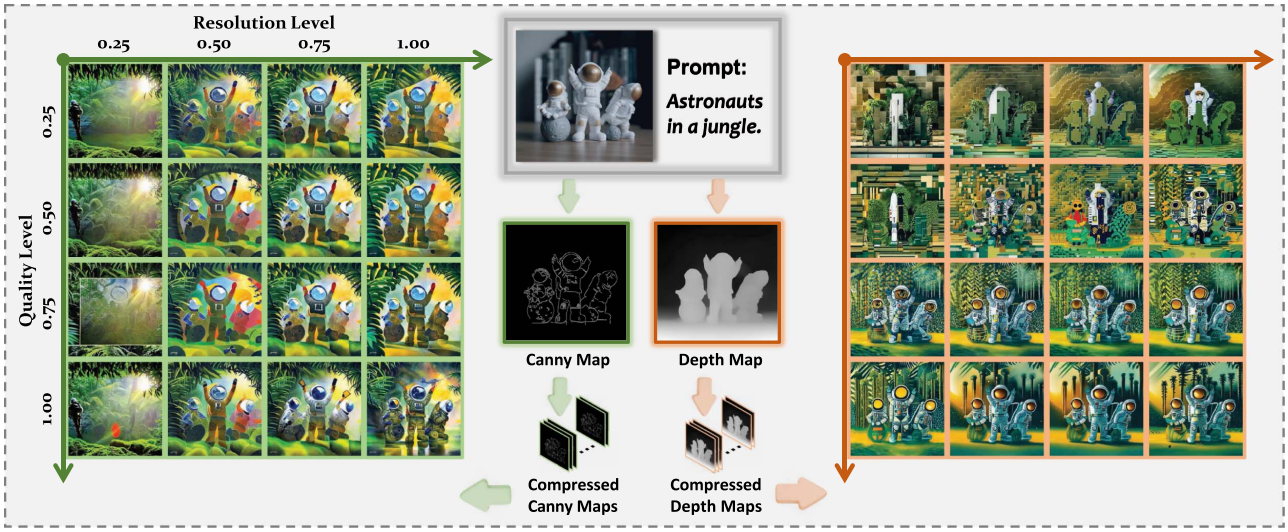


Fig. 3. An example of generation results with different compressed visual conditions.

relationships. This facilitates better prompt alignment, providing models with greater flexibility to create images that align more closely with the intended contexts. Given that different users display distinct preferences in generation, it is crucial to identify the suitable visual conditions to enhance the generation quality.

Furthermore, although the data size of visual condition is smaller than that of the corresponding raw image, it is still much larger compared to text prompts. Therefore, we continue to investigate the potential of compressing visual condition images. Specifically, we focus on changing the resolution level and quality level (i.e., quantization level) of the condition image. As shown in Fig. 3, we explore a generation example with a source image of toy astronauts and the prompt of *Astronauts in a jungle*. We first extract the Canny map and Depth map from the source image as the visual conditions. These visual conditions are then compressed at various resolution and quality levels, resulting in 16 different compressed visual conditions. Based on these compressed conditions, we then generate images respectively and compare their quality.

For images generated with Canny maps, it can be observed that those compressed at the 0.25 resolution level show a marked diminishment of the astronauts, irrespective of the quality level applied. This indicates that compressing the Canny map to a low resolution level results in an unacceptable qualtiy. In contrast, images generated with Depth maps at 0.25 and 0.5 quality levels exhibit noticeable blockiness in the objects, regardless of the resolution level applied. This suggests that when the Depth map is compressed at a low quality level, performance suffers greatly. Therefore, to minimize the transmission cost of visual conditions without compromising final generation quality, it is essential to implement suitable compression strategies tailored to various visual conditions.

B. Impact of Denoising Steps

In this subsection, we further explore how visual conditions under various compression settings, can take advantage of additional denoising steps. Without loss of generality, we take the visual condition of Canny map as an example to conduct the following experiment. The Canny maps for all images in the *DreamBooth* dataset are extracted and compressed at ratios of 20%, 40%, 60%, and 80%. Then, images are generated under these compressed visual conditions using two denoising step configurations: one with 5 steps and the other with 20 steps. This setting allows us to evaluate the potential enhancement brought by the additional 15 denoising steps across different levels of compression.

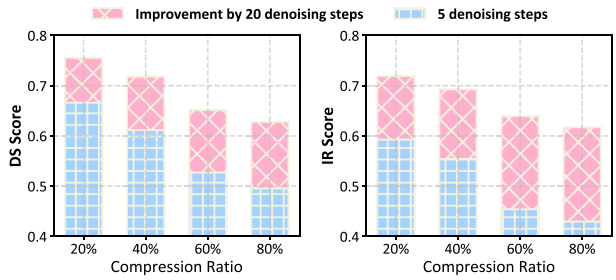


Fig. 4. Generation quality improvement from 5 to 20 denoising steps under different compression configurations for Canny map as visual condition.

As illustrated in Fig. 4, there are two key observations. *First*, increasing the compression ratio of visual conditions leads to more pronounced quality degradation in generated images. For example, a rise in the Canny map compression ratio from 20% to 80% at 20 denoising steps results in a significant 12% drop in the DS Score and a 10% drop in the IR Score. Consequently, determining the optimal compression settings is critical for striking a balance between the transmission cost of visual conditions and the quality of the generated output. *Second*, a greater number of denoising steps is critical to improve the overall generation quality, particularly for visual conditions with higher compression ratios, which exhibit amplified gains in both metrics. To illustrate, a 20% compression of the Canny map leads to an 12% IR Score improvement, while an 80% compression enhances the improvement to 19%. With each additional step, the model is

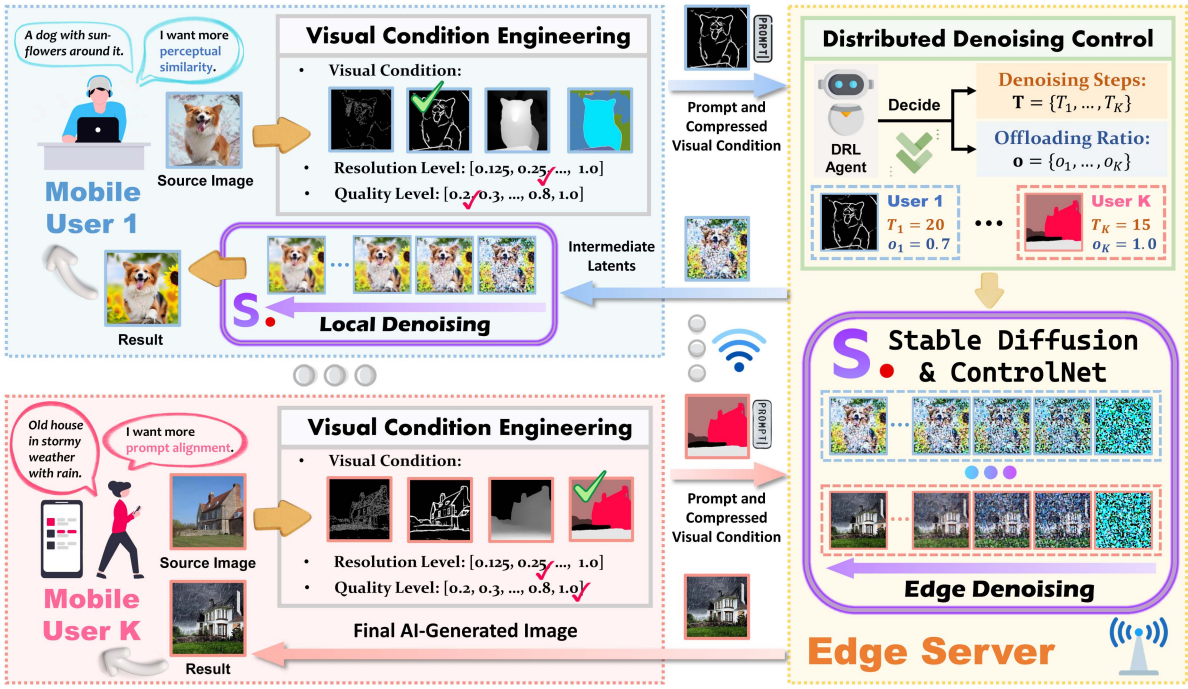


Fig. 5. System framework.

better able to recreate the missing details, enhancing the overall image quality. Consequently, under highly compressed visual conditions, more denoising steps are crucial for achieving better results.

In summary, the interplay between different visual condition configurations and denoising steps both significantly affect the generation quality. Therefore, carefully configuring them is key to optimizing generation experiences for all users.

IV. System Model and Problem Formulation

In this section, we illustrate our system for distributed controllable T2I generation services.

A. System Overview

As shown in Fig. 5, we consider a mobile controllable T2I generation service system consisting of one edge server and K users, where each user expects high-quality image output that satisfies their distinct requirements. For example, User 1 requests an image generation using both a source image of a dog and the prompt of *A dog surrounded by sunflowers*. In addition, this user wants the generated image to preserve more perceptual similarity. Meanwhile, User K requests generating an image from a source image of a house with the prompt of *Old house in stormy weather with rain*. In contrast, this user prioritizes a stronger alignment between the final result and the prompt. To address the unique demands of different users, we introduce a *Visual Condition Engineering* module that operates on the local device of each user. This module is responsible for selecting the most appropriate visual condition type and its compression configurations tailored to the preferences of users. As a result, transmission cost can be significantly reduced without compromising the generated-image quality.

To proceed, the prompts and compressed visual conditions are sent to the edge server. Once the edge server receives all generation requests, our proposed *Distributed Denoising Control* module utilizes a well-trained DRL agent enhanced by a diffusion model to optimally decide the denoising steps and offloading ratios for each user. Users with higher computing capacity, for instance, User 1, have part of the denoising process performed at the edge server, and the intermediate latents are returned to the device to perform remaining steps at local. For users with limited computing resources, for instance, User K, the entire denoising process takes place on the edge server, and only the final generated image is sent back. This strategy can strike a good balance between maximizing overall image quality and reducing the overall latency of the generation process for mobile users with diverse level of computing capacity.

B. Visual Condition Engineering

Traditional mobile AIGC services necessitate the upload of source image $\mathcal{I}_k$ to the edge server for visual condition extraction, which consumes considerable bandwidth resources. To overcome this challenge, we propose extracting visual conditions locally at the user device for three main reasons. *First*, recent advances have introduced many lightweight yet accurate methods for extracting visual conditions [28], [29], making it feasible to perform fast and reliable extraction on resource-constrained devices. *Second*, the extracted visual conditions are generally much smaller in size than original images, as they only emphasize critical information rather than retaining

all pixel-level details. *Third*, as shown in Fig. 3, compressing the visual conditions appropriately presents an opportunity to further mitigate the transmission costs while maintaining the quality of the generated images. Therefore, we conduct *Visual Condition Engineering* locally on user devices.

As discussed in Section III-A, both the perceptual similarity and prompt alignment are key criterions for evaluating the generation quality in controllable T2I generation tasks. Let q_k^{I} denote the perceptual similarity score and q_k^{T} denote the prompt alignment score of the generated image for user k. Accordingly, the generation quality vector for user k is defined as $\boldsymbol{q_k} = (q_k^{\mathrm{I}}, q_k^{\mathrm{T}})^{\mathsf{T}}$. Given that different users prioritize different aspects of generation results, integrating their preferences into the quality evaluation is crucial. To this end, we define $\boldsymbol{\omega_k} = (\omega_k^{\mathrm{I}}, \omega_k^{\mathrm{T}})$ as the preference vector for user k, where ω_k^{I} and ω_k^{T} correspond to the user preference for perceptual similarity and prompt alignment, respectively, subject to the constraint that $\omega_k^{\mathrm{I}} + \omega_k^{\mathrm{T}} = 1$. Therefore, we define the generated-image quality $\mathcal{Q}_k$ for user k as

$$\mathcal{Q}_k = \boldsymbol{\omega_k}\boldsymbol{q_k} = \omega_k^{\mathrm{I}} q_k^{\mathrm{I}} + \omega_k^{\mathrm{T}} q_k^{\mathrm{T}}, \tag{1}$$

which is a weighted sum of both scores.

While certain visual conditions emphasize perceptual similarity and others are better aligned with text prompts, choosing the right one is vital for enhancing the generation quality $\mathcal{Q}_k$. Besides, since visual conditions primarily provide semantic and structural guidance, moderate compression can significantly reduce transmission costs while preserving essential details and maintaining generation quality. To this end, for user k, we conduct the *Visual Condition Engineering* to dynamically select the most suitable visual condition type, λ_k, along with the appropriate resolution level, α_k, and quality level, β_k, for compression. We denote all the selected visual condition configurations for user k as a vector $\boldsymbol{\nu}_k = (\lambda_k, \alpha_k, \beta_k)$ and the engineered visual condition as $\mathcal{I}_k(\boldsymbol{\nu}_k)$.

The process of *Visual Condition Engineering* involves four key sequential steps on the user device. *First*, we apply our proposed VCE algorithm to select the optimal visual condition configurations $\boldsymbol{\nu}_k$. *Second*, we proceed to extract visual conditions based on the derived type result λ_k. For the latency of extracting visual condition, it depends only on the computing capacities of users as each visual condition can be extracted through multiple lightweight methods with comparable latencies. *Third*, once extraction is completed, we compress the extracted visual condition image under the compression configuration (α_k, β_k) to further reduce the transmitted data size. The compression latency is also excluded due to its considerably faster operation compared to other stages. *Finally*, upon finishing the compression process, each user sends the prompt along with the engineered visual condition $\mathcal{I}_k(\boldsymbol{\nu}_k)$ to the edge server for image generation under current available uplink bandwidth resource $B^{\uparrow}$. As the data size of text prompt is much smaller than that of the engineered visual condition, its transmission latency is oftentimes considered to be negligible [30]. In summary, the total latency for *Visual Condition Engineering* process $\mathcal{L}_k^1$ can be formulated as

$$\mathcal{L}_k^1 = \mathcal{L}_k^0(f_k) + \frac{\mathcal{D}(\mathcal{I}_k(\boldsymbol{\nu}_k))}{B^{\uparrow}}, \tag{2}$$

where f_k quantifies the device computing capacities of user k in the floating point operation per second (FLOPS), $\mathcal{L}_k^0(f_k)$ represents the extraction latency determined by f_k, and $\mathcal{D}(\cdot)$ refers to the function for calculating the data size.

C. Distributed Denoising Control

When the edge server receives generation requests from multiple users, it proceeds to the generation stage. Without loss of generality, the most widely used T2I model, latent diffusion model [17], is employed for generation. As images are generated primarily through a gradual reversal of the diffusion process, the quality of the resulting images is highly dependent on the number of denoising steps, denoted as T. Specifically, the denoising stage is initiated with a randomly generated Gaussian noise $\boldsymbol{z}_T$ in the latent space. It is then iteratively refined through T denoising steps, guided by the user-specified conditions $\boldsymbol{z_c}$, which include the encoded latent information of both the text prompt and the engineered visual condition. Thus, according to [17], the denoising process can be mathematically expressed as

$$p_\theta(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t) = \mathcal{N}(\boldsymbol{z}_{t-1}; \mu_\theta(\boldsymbol{z}_t, t, \boldsymbol{z_c}), \Sigma_\theta(\boldsymbol{z}_t, t, \boldsymbol{z_c})), \tag{3}$$

which describes the conditional probability of obtaining the next denoised latent variable $\boldsymbol{z}_{t-1}$ given the noisy latent variable $\boldsymbol{z}_t$ at current step t. Here, θ represents the well-trained parameters of UNet-based noise predictor. $\mathcal{N}(\cdot)$ indicates that $\boldsymbol{z}_{t-1}$ is drawn from a Gaussian distribution, characterized by its mean $\mu_\theta(\cdot)$ and variance $\Sigma_\theta(\cdot)$. By iteratively applying the denoising process outlined in (3), the model transforms a fully noisy latent variable $\boldsymbol{z}_T$ into a clean latent representation $\boldsymbol{z}_0$ of the required image. Then, a high-quality image $\boldsymbol{x}_0$ in the human-perceivable pixel space can be generated through a variational autoencoder (VAE) decoder.

Although more denoising steps can enhance output quality, they also lead to higher demands for computing resources, as each step requires the execution of the heavy U-Net to remove Gaussian noise [31]. Edge servers possess significantly greater computing resources than typical end devices, yet they remain resource-constrained compared to cloud servers [32], [33], [34]. Consequently, the denoising steps assigned to each user T_k are limited, making it challenging to guarantee the optimal generation quality and latency for all the users. Given the diverse quality expectations $\widetilde{\mathcal{Q}}_k$ and latency requirements $\widetilde{\mathcal{L}}$ of users, careful selection of denoising steps is necessary to optimize overall performances. Furthermore, as demonstrated in Fig. 4, different visual condition configurations demonstrate distinct potentials in quality improvement, providing additional opportunities for optimizing step selection strategies.

Considering that certain off-the-shelf mobile devices can handle compute-intensive denoising locally, edge resources can be reserved for less capable users, thereby boosting overall performance. Consequently, we propose a *Distributed Denoising* approach to accomplish image generation through the

cooperation of edge server and end devices. Specifically, for user k with T_k denoising step requirements, we introduce an offloading ratio $o_k \in (0, 1]$ to represent the proportion of denoising steps that should be offloaded to the edge server for processing. When the local computing resources are abundant, users can manage part of the denoising tasks on their own devices, leading to a low offloading ratio. On the other hand, users with limited computing resources or those unable to deploy the AIGC models, rely heavily on the edge server to complete generations, thus resulting in a higher o_k.

During the *Distributed Denoising* process, the first $\lfloor o_k T_k \rceil$ denoising steps are conducted on the edge server for each user, where $\lfloor \cdot \rceil$indicates the rounding operation. Therefore, the edge denoising process can be represented as

$$(3), \quad 0 < t \leq \lfloor o_k T_k \rceil. \tag{4}$$

If the task is not fully offloaded to the edge server (i.e., $o_k \neq 1$), the intermediate output latent $\boldsymbol{z}^k(\lfloor o_k T_k \rceil)$ generated at the $\lfloor o_k T_k \rceil$ step will be transmitted to the corresponding user device for subsequent denoising process. Due to the lower dimensionality of the latent space, the transmission cost is much lower than transmitting the full image pixel data. After the user receives the intermediate latent $\boldsymbol{z}^k(\lfloor o_k T_k \rceil)$, it continues the local denoising process, represented as

$$(3), \quad \lfloor o_k T_k \rceil < t \leq T_k. \tag{5}$$

The above cooperative denoising process can yield the clean latent representation $\boldsymbol{z}_0^k$ of the resulting image, after which the VAE decoder on the user device transforms this latent into pixel space as the final image $\boldsymbol{x}_0^k$. In this case, the total distributed denoising latency consists of edge denoising latency, latent transmission latency and local denoising latency. Given that each step of the denoising process performs identical architectural and computational operations, the required computational resources grow almost linearly with the number of denoising steps [35]. Therefore, the total distributed denoising latency $\mathcal{L}_k^2$ can be formulated as

$$\mathcal{L}_k^2 = \frac{\varepsilon_0 \lfloor o_k T_k \rceil}{f_{\mathrm{E}}} + \frac{\mathcal{D}(\boldsymbol{z}^k(\lfloor o_k T_k \rceil))}{B^{\downarrow}} + \frac{\varepsilon_0 \lfloor (1 - o_k) T_k \rceil}{f_k}, \quad \text{if } o_k \neq 1, \tag{6}$$

where ε_0 denotes the required floating point operations (FLOPs) for computing a single step of denoising process. Additionally, f_{E} denotes the computing capacities of the edge server in FLOPS and $B^{\downarrow}$ is the current available downlink bandwidth resources. We ignore the latency introduced by our algorithm and the execution of other model components (e.g., the VAE decoder), as they contribute insignificantly to the overall latency compared to other processes [36].

In cases the generation task is fully offloaded to the edge server (i.e., $o_k = 1$), it will handle all the denoising steps, with only (4) being executed. The final generated image at the pixel space $\boldsymbol{x}_0^k$ is then transmitted back to user k. In this scenario, the distributed denoising latency $\mathcal{L}_k^2$ consists of only the edge denoising latency and the generated-image transmission latency. Thus, $\mathcal{L}_k^2$ can be simplified as

$$\mathcal{L}_k^2 = \frac{\varepsilon_0 T_k}{f_{\mathrm{E}}} + \frac{\mathcal{D}(\boldsymbol{x}_0^k)}{B^{\downarrow}}, \quad \text{if } o_k = 1. \tag{7}$$

In summary, the total AIGC service latency for user k in our system is denoted as $\mathcal{L}_k = \mathcal{L}_k^1 + \mathcal{L}_k^2$.

D. Problem Formulation

Our objective is to maximize the total generation quality across all users, while adhering to the limitations imposed by available computing resources, quality requirements, and latency constraints. To this end, we first define a utility function $\mathcal{U}_k$ to indicate the utility score for each user as

$$\mathcal{U}_k = \mathcal{Q}_k(\boldsymbol{\omega_k}, \boldsymbol{\nu}_k, T_k) - \eta \mathcal{L}_k(f_k, \boldsymbol{\nu}_k, T_k, o_k), \tag{8}$$

where η represents the weight constant for normalization. To maximize the overall utility scores, our system necessitates a meticulous joint selection of the visual condition configurations $\boldsymbol{\nu}_k$, denoising steps T_k, and offloading ratios o_k for all users. This problem can be mathematically formulated as

$$\mathbf{P}_0 : \max_{\{\boldsymbol{\nu}, \mathbf{T}, \mathbf{o}\}} \sum_{k \in K} \mathcal{U}_k \tag{9}$$

$$\text{s.t.} \quad \lambda_k \in \boldsymbol{\lambda}, \alpha_k \in \boldsymbol{\alpha}, \beta_k \in \boldsymbol{\beta}, \quad \forall k \in K, \tag{10}$$

$$o_k \in (0, 1], \quad \forall k \in K, \tag{11}$$

$$\varepsilon_0 \sum_{k \in K} \lfloor o_k T_k \rceil \leq C_{\mathrm{E}}, \quad \forall k \in K, \tag{12}$$

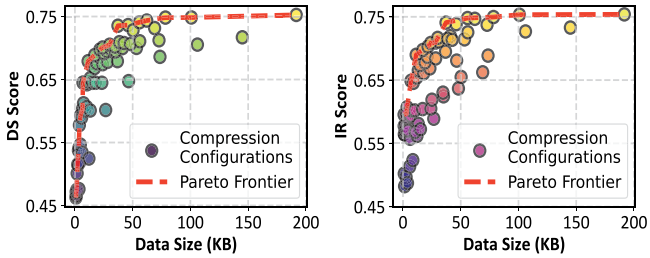
$$\varepsilon_0 \lfloor (1 - o_k) T_k \rceil \leq C_k, \quad \forall k \in K, \tag{13}$$

$$\widetilde{\mathcal{Q}}_k \leq \mathcal{Q}_k(\boldsymbol{\omega_k}, \boldsymbol{\nu}_k, T_k), \quad \forall k \in K, \tag{14}$$

$$\mathcal{L}_k(f_k, \boldsymbol{\nu}_k, T_k, o_k) \leq \widetilde{\mathcal{L}}, \quad \forall k \in K, \tag{15}$$

where (10) and (11) ensure the visual condition configuration parameters and offloading ratios remain within the selectable ranges. (12) and (13) limit the computing resource consumption on the edge server and user device within the time slot, bounded by C_{E} and C_k, respectively. Moreover, (14) ensures the QoE threshold of each user is satisfied, and (15) guarantees the total service latency is within the permissible threshold.

The problem $\mathbf{P}_0$ includes discrete variables $\boldsymbol{\nu}_k$, integer variables T_k and continuous variables o_k with nonlinear constraints. A brute-force way to solve this problem is computationally intensive, as it entails an extensive search across both large discrete and continuous domains. To tackle this challenge, we propose a problem decomposition strategy, which divide the original problem $\mathbf{P}_0$ into two sub-problems. This strategy is driven by the necessity to handle the heterogeneous decision-making processes: visual condition configurations $\boldsymbol{\nu}_k$ need to be decided locally on each user device, whereas denoising steps T_k and offloading ratios o_k must be optimized collaboratively for multiple users at the edge.



(a) $\mathcal{P}_{\lambda}^{\mathrm{I}}$: DS Score-Latency (b) $\mathcal{P}_{\lambda}^{\mathrm{T}}$: IR Score-Latency

Fig. 6. Examples of DS Score-Latency and IR Score-Latency Pareto Frontiers for the visual condition of Canny map. As the latency $\mathcal{L}_k^1$ is linear to the data size of compressed visual condition for a given mobile user, the Pareto Frontier for Score-Latency and Score-Data Size is equivalent.

Algorithm 1: The VCE Algorithm.

Input: $\boldsymbol{\omega_k}, \widetilde{\mathcal{Q}}_k, \boldsymbol{\mathcal{P}}$.
Output: $\boldsymbol{\nu}_k^* = [\lambda_k^*, \alpha_k^*, \beta_k^*]$.
if $\omega_k^{\mathrm{I}} > \omega_k^{\mathrm{T}}$ **then**
  $\boldsymbol{\lambda} \leftarrow [\textsc{Canny}, \textsc{Hed}], \quad \mathcal{P}_{\lambda}^* \leftarrow \mathcal{P}_{\lambda}^{\mathrm{I}}$
else
  $\boldsymbol{\lambda} \leftarrow [\textsc{Depth}, \textsc{Seg}], \quad \mathcal{P}_{\lambda}^* \leftarrow \mathcal{P}_{\lambda}^{\mathrm{T}}$
for $\lambda_k \in \boldsymbol{\lambda}$ **do**
  $\boldsymbol{\nu}_{k,\lambda}^*, U_{k,\lambda}^* \leftarrow$ **getBestConfig**$(\lambda_k, \widetilde{\mathcal{Q}}_k, \mathcal{P}_{\lambda}^*)$
$\boldsymbol{\nu}_k^* \leftarrow \arg\max_{\boldsymbol{\nu}_{k,\lambda}^*} U_{k,\lambda}^*$
return $\boldsymbol{\nu}_k^*$

V. Algorithm Design

In this section, we elaborate on the two proposed algorithms designed to address the above sub-problems respectively.

A. Visual Condition Engineering Algorithm

The first sub-problem aims at determining the optimal visual condition configurations $\boldsymbol{\nu}_k$ for each user. By treating the denoising steps T_k and offloading ratio o_k as fixed parameters, we only focus on the impact of visual condition configurations $\boldsymbol{\nu}_k$ on the generation quality and latency. Hence, the sub-problem can be formulated as

$$\mathbf{P}_1 : \max_{\{\boldsymbol{\nu}\}} \quad \sum_{k \in K} \mathcal{U}_k(T_k = T, o_k = O) \tag{16}$$

$$\text{s.t.} \quad (10), (14), (15). \tag{17}$$

Although with a fully discrete feasible region and relaxed constraints, the search space of $\mathbf{P}_1$ remains large due to the numerous possible combinations of visual condition configurations across multiple users. Considering the utility of each user is based only on their own configuration with no interdependencies with others, we aim to develop a configuration selection algorithm that maximizes the utility for each user.

For each visual condition, we first establish profiles that relate both the DS Score and IR Score to latency $\mathcal{L}^1$ under all compression configurations. Then we identify the Pareto Frontier for each profile. The Pareto Frontier means the set of configurations $\boldsymbol{\nu}$ in the Pareto-optimal set $\mathcal{P}^*$, where no alternative configuration $\tilde{\boldsymbol{\nu}}$ can achieve lower latency while providing higher quality. Formally, $\mathcal{P}^*$ is defined as

$$\mathcal{P}^* = \{\boldsymbol{\nu} \in \mathcal{V} : \{\tilde{\boldsymbol{\nu}} \in \mathcal{V} : \mathcal{Q}(\tilde{\boldsymbol{\nu}}) > \mathcal{Q}(\boldsymbol{\nu}), \mathcal{L}(\tilde{\boldsymbol{\nu}}) < \mathcal{L}(\boldsymbol{\nu})\} = \varnothing\}. \tag{18}$$

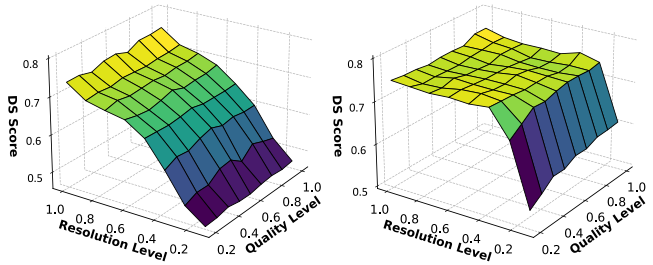
Thus, the configurations in the Pareto Frontier reliably guarantee a balance between quality and latency, where higher quality can only be achieved at the expense of increased latency. Examples of DS Score-Latency Pareto Frontier $\mathcal{P}_{\lambda}^{\mathrm{I}}$ and IR Score-Latency Pareto Frontier $\mathcal{P}_{\lambda}^{\mathrm{T}}$ for the Canny map visual condition (i.e., $\lambda =$ Canny) are presented in Fig. 6.

Based on the profiled Pareto Frontiers, we develop the *Visual Condition Engineering* (VCE) algorithm, whose overall pipeline is presented in Algorithm 1. We first determine the set of potential visual conditions based on user preferences $\boldsymbol{\omega_k}$. As indicated in Section III-A, for users prioritizing perceptual similarity, Canny and HED maps are selected, with $\mathcal{P}_{\lambda}^{\mathrm{I}}$ serves as the candidate set for compression configuration exploration. Otherwise, Depth and Seg maps are preferred, leveraging $\mathcal{P}_{\lambda}^{\mathrm{T}}$ as the candidate set. Then we employ Algorithm 2 (i.e., **getBestConfig** function) to identify the optimal compression configurations for each candidate visual condition and evaluate their utility scores, selecting the configuration with the highest score as the final choice.

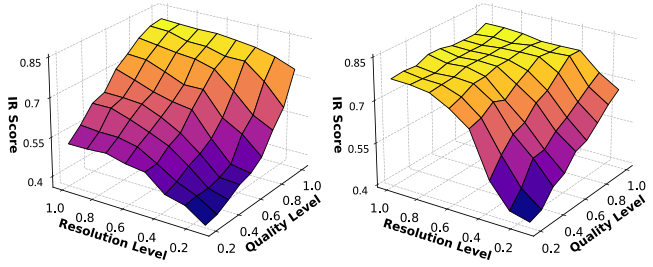
Although the configurations within the Pareto Frontier provide a favorable trade-off between quality and latency, identifying the optimal configuration remains non-trivial due to the extensive search space. Therefore, there is a need for an algorithm capable of rapidly and precisely selecting the ideal compression configuration from the Pareto-optimal set, i.e., realizing the **getBestConfig** function.

In particular, we delve deeper into the interplay between resolution level, quality level, and generation quality across a range of visual conditions through profiling. As shown in Fig. 7, we observe that generation quality reacts more significantly to resolution level changes when the Canny map is compressed, but with Depth map compression, the performance is more affected by quality level adjustments. These numerical results align precisely with the visible outcomes illustrated in Fig. 3. In terms of the HED map and Seg map, similar trends are observed, where a moderate compression configuration allows each to reach a satisfactory performance. Based on the profiling results, we propose Algorithm 2 for efficient compression configuration selection according to the visual condition λ_k and corresponding Pareto Frontier $\mathcal{P}_{\lambda}^*$.

Initially, we sort the Pareto-optimal set $\mathcal{P}_{\lambda}^*$ in ascending order first by resolution level and then by quality level. Subsequently, we iterate over $\mathcal{P}_{\lambda}^*$, updating the compression configurations based on the given type of visual condition. When using Canny map as the visual condition, we first focus on enhancing the resolution level to achieve substantial quality gains. Once the highest resolution is reached, we then shift to enhancing the quality level (Lines 6-8). Conversely, for the Depth map,



(a) DS Score for Canny map (b) DS Score for HED map



(c) IR Score for Depth map (d) IR Score for Seg map

Fig. 7. The relationship among resolution level, quality level and generation quality across different types of visual conditions. Since DS Score and IR Score demonstrate comparable patterns under different configurations, we present only one type of score result per visual condition as an example.

we prioritize enhancing the quality level first and only adjust the resolution once quality has reached its maximum (Lines 9-11). For the remaining two types of visual conditions, we adjust both parameters concurrently to achieve the satisfactory quality level as quickly as possible (Lines 12-13). The search process concludes only upon meeting both the specified quality requirements and latency thresholds. Finally, the resulting compression configurations along with the corresponding utility score are returned to Algorithm 1, with the configuration yielding the highest score selected as the final output. Following this searching paradigm, we can promptly obtain the accurate configuration results.

Although only four typical visual conditions are presented in this paper, the designed system can be easily extended to new visual conditions by profiling their quality-latency trade-offs. Since the VCE algorithm relies on profiling-based Pareto selection and pattern-aware configuration, newly profiled conditions can be integrated without structural changes.

B. Distributed Denoising Control Algorithm

In this subsection, we introduce the proposed algorithm for *Distributed Denoising Control*. With the optimal visual conditions $\boldsymbol{\nu}_k^*$ derived from Algorithm 1, our next objective is to select the optimal denoising steps T_k and offloading ratio o_k for all the users, which can be formulated as

$$\mathbf{P}_2: \max_{\{\mathbf{T},\mathbf{o}\}} \quad \sum_{k\in K} \mathcal{U}_k(\boldsymbol{\nu}_k = \boldsymbol{\nu}_k^*) \tag{19}$$

$$\text{s.t.} \quad (11) - (15). \tag{20}$$

Algorithm 2: Compression Configuration Selection.

1 **Function** `getBestConfig` $(\lambda_k, \widetilde{\mathcal{Q}}_k, \mathcal{P}_\lambda^*)$**:**
2 Sort $\mathcal{P}_\lambda^*$ first by α then by β in ascending order
3 $\boldsymbol{\nu}_{k,\lambda}^* = [\lambda_k, \alpha_k^0, \beta_k^0] \leftarrow$ `initialization`
4 **for** $\boldsymbol{\nu}_{k,\lambda}^i \in \mathcal{P}_\lambda^*$ **do**
5 **switch** λ **do**
6 **case** *Canny* **do**
7 $\alpha_k^* \leftarrow \alpha_k^i$ **if** $\alpha_k^* < \alpha_k^i$
8 $\beta_k^* \leftarrow \beta_k^i$ **if** $\alpha_k^* = \alpha_k^{\max}$
9 **case** *Depth* **do**
10 $\beta_k^* \leftarrow \beta_k^i$ **if** $\beta_k^* < \beta_k^i$
11 $\alpha_k^* \leftarrow \alpha_k^i$ **if** $\beta_k^* = \beta_k^{\max}$
12 **otherwise do**
13 $\alpha_k^* \leftarrow \alpha_k^i, \beta_k^* \leftarrow \beta_k^i$
14 $U_{k,\lambda}^*, \mathcal{Q}_{k,\lambda}^*, \mathcal{L}_{k,\lambda}^* \leftarrow$ `getUtility`$(\nu_{k,\lambda}^*)$
15 **if** $\widetilde{\mathcal{Q}}_k \le \mathcal{Q}_{k,\lambda}^*$ **and** $\mathcal{L}_{k,\lambda}^* \le \widetilde{\mathcal{L}}$ **then break**
16 **return** $\boldsymbol{\nu}_{k,\lambda}^*, U_{k,\lambda}^*$

Despite the intricate coupling among variables is alleviated by the obatined $\boldsymbol{\nu}_k^*$, the nonlinear characteristics of complex objectives and constrains remains, hindering conventional optimization methods from efficiently achieving optimal results. Therefore, we propose applying the DRL techniques to solve problem $\mathbf{P}_2$, leveraging their advanced ability to effectively manage complex constraints while formulating adaptive policies. With the remarkable analytical abilities, multiple models have been validated as effective components for augmenting the problem-solving capacity of DRL architectures, such as Generative Adversarial Networks [37], VAEs [38], Transformers [39], and deep diffusion models [11]. Therefore, we propose enhancing the policy network $\pi_\phi(\boldsymbol{a}|\boldsymbol{s})$ within our DRL framework by incorporating a deep diffusion model for generating distributed denoising control schemes. Specifically, the policy network $\pi_\phi(\boldsymbol{a}|\boldsymbol{s})$ serves as a denoiser, gradually transforming the initial random Gaussian noise $\boldsymbol{a}_T$ into the optimal actions $\boldsymbol{a}_0$, which can be expressed as

$$\begin{aligned}\pi_\phi(\boldsymbol{a}|\boldsymbol{s}) &= p_\phi(\boldsymbol{a}_{0:T}|\boldsymbol{s}) \\ &= \mathcal{N}(\boldsymbol{a}_T; 0, \mathbf{I}) \prod_{t=1}^{T} p_\phi(\boldsymbol{a}_{t-1}|\boldsymbol{a}_t, \boldsymbol{s}),\end{aligned} \tag{21}$$

where $p_\phi(\boldsymbol{a}_{t-1}|\boldsymbol{a}_t, \boldsymbol{s})$, defined in (3), denotes the progressive denoising process of T steps to refine the action. Through this integration, our algorithm effectively supports the DRL agent to capture the dependencies between state and action spaces, facilitating a more efficient learning process [40].

Beyond its generative capability, this diffusion-based policy structure fundamentally enhances the convergence of the mapping from parameters ϕ to the action distribution $\pi_\phi(\boldsymbol{a}|\boldsymbol{s})$. In standard actor-critic methods, deep neural network policies often exhibit high sensitivity to parameter updates, leading to

large Lipschitz constants L_g for the policy gradient $\nabla_\phi J_\pi(\phi)$, which forces the use of small learning rates and slows convergence [41]. In contrast, our approach reformulates action generation as a T-step denoising process as denoted in (21), where each transition $p_\phi(\boldsymbol{a}_{t-1}|\boldsymbol{a}_t, \boldsymbol{s})$ is a Gaussian distribution modeled by a parameter-shared U-Net. This decomposes the policy into T locally smooth mappings, each exhibiting reduced sensitivity to changes in ϕ. Under mild regularity assumptions, the overall Lipschitz constant of the diffusion-enhanced policy gradient satisfies:

$$L_{g,\text{diff}} \leq \ell^T L_g, \tag{22}$$

where $\ell \ll 1$ is the per-step contraction factor from denoising process. Thus, the maximum stable learning rate scales as

$$\eta_{\max} = \mathcal{O}\left(\frac{1-\gamma}{L_{g,\text{diff}}^2}\right) \propto \mathcal{O}(T^2), \tag{23}$$

implying a theoretical advantage over standard policies. In summary, this improved policy induces an exponential contraction effect that reduces the overall Lipschitz constant by a factor of ℓ^T. This reduction allows for larger stable learning rates that scale as $\mathcal{O}(T^2)$, thereby accelerating convergence.

Next, we demonstrate the key elements of our proposed algorithm, characterized by the following definitions of state space $\boldsymbol{s}$, action space $\boldsymbol{a}$, and reward function $\boldsymbol{r}$.

- *State:* The state space encompasses the key factors influencing decisions during each time slot. For the generation tasks of K users, these factors include input visual condition configurations, QoE and latency requirements, and the available computing resources on both the edge and user sides, expressed as:

$$\boldsymbol{s} = \{\{\boldsymbol{\nu}_k^*, \widetilde{\mathcal{Q}}_k, C_k\}_{k=1}^K, \widetilde{\mathcal{L}}, C_E\}. \tag{24}$$

- *Action:* The action space incorporates decisions on both the denoising step and offloading ratio for each user, leading to a hybrid action space, represented as

$$\boldsymbol{a} = \{\{T_k, o_k\}_{k=1}^K\}. \tag{25}$$

  To handle the discrete nature of T_k, we leverage the Gumbel-Softmax method [42], which generates a probability distribution over its potential discrete values, effectively converting T_k into a continuous representation. This transformation ensures compatibility with gradient-based continuous optimization frameworks.
- *Reward:* The reward is formulated to align with the primary goal of maximizing the overall utility $\sum_{k\in K} \mathcal{U}_k$. Besides, to address constraint violations, negative rewards are used as penalty terms. Therefore, given the state $\boldsymbol{s}$, the reward r of taking action $\boldsymbol{a}$ is defined as

$$\begin{aligned} r = &\sum_{k\in K} \mathcal{U}_k - \delta_1 \max\left(0, \varepsilon_0 \sum_{k\in K} \lfloor o_k T_k \rceil - C_{\text{E}}\right) \\ &- \delta_2 \sum_{k\in K} \max(0, \varepsilon_0 \lfloor (1-o_k) T_k \rceil - C_k) \\ &- \delta_3 \sum_{k\in K} \max(0, \widetilde{\mathcal{Q}}_k - \mathcal{Q}_k) \\ &- \delta_4 \sum_{k\in K} \max(0, \mathcal{L}_k - \widetilde{\mathcal{L}}), \end{aligned} \tag{26}$$

  where δ_i are coefficients for these penalties respectively. The above penalties terms exists only when the constraints are violated, facilitating the policy to satisfy the constraints while striving to maximize the overall utility.

Algorithm 3: The DDC Algorithm.

Initialize: Actor network π_ϕ, critic networks Q_{ψ_1} and Q_{ψ_2}, target networks $Q'_{\psi_1} \leftarrow Q_{\psi_1}$ and $Q'_{\psi_2} \leftarrow Q_{\psi_2}$, and experience replay buffer $\mathcal{E}$.
for *episode = 1 to MAX_EPISODE* **do**
  Observe the initial state $\boldsymbol{s}$ and initialize a random Gaussian noise $\boldsymbol{a}_T \sim \mathcal{N}(0, \mathbf{I})$
  for *time step = 1 to MAX_STEP* **do**
    Generate action $\boldsymbol{a}_0$ via the denoising process defined in Eq. (21), conditioned on state $\boldsymbol{s}$ and exploration noise
    Execute the distributed denoising process with action $\boldsymbol{a}_0$, then calculate the reward r by Eq. (26) and observe next state $\boldsymbol{s}'$
    Store transition $(\boldsymbol{s}, \boldsymbol{a}_0, r, \boldsymbol{s}')$ in replay buffer $\mathcal{E}$
    Sample a random batch of records $\mathcal{E}_b$ from $\mathcal{E}$
    Update critic networks Q_{ψ_1}, Q_{ψ_2} by Eq. (28)
    Update actor network π_ϕ by Eq. (27)
    Update target networks Q'_{ψ_1}, Q'_{ψ_2} by Eq. (30)
return Optimized policy π^*

To solve problem $\mathbf{P}_2$ with continuous action spaces, we opt for the Soft Actor-Critic (SAC) architecture, as it effectively maximizes both the cumulative reward expectation and the entropy of the policy, promoting exploratory behavior and increasing the robustness of the learning process. The architecture of SAC algorithm comprises five key neural networks: an actor network $\pi_\phi(\boldsymbol{a}|\boldsymbol{s})$, which generates an action distribution based on the given state $\boldsymbol{s}$; two critic networks $Q_{\psi_1}(\boldsymbol{s}, \boldsymbol{a})$ and $Q_{\psi_2}(\boldsymbol{s}, \boldsymbol{a})$, responsible for estimating the state-action value; and two target critic networks $Q_{\psi'_1}(\boldsymbol{s}, \boldsymbol{a})$ and $Q_{\psi'_2}(\boldsymbol{s}, \boldsymbol{a})$, designed to stabilize the training process. As the primary goal of the SAC algorithm is to strike a balance between exploration and exploitation by maximizing both the reward and the entropy, the actor network $\pi_\phi(\boldsymbol{a}|\boldsymbol{s})$ is optimized by minimizing the following objective:

$$\mathcal{J}_\pi = \mathbb{E}_{\boldsymbol{s}\sim\mathcal{D}, \boldsymbol{a}\sim\pi}\left[\xi \log \pi_\phi(\boldsymbol{a}|\boldsymbol{s}) - \min_{j=1,2} Q_{\psi_j}(\boldsymbol{s}, \boldsymbol{a})\right], \tag{27}$$

where the parameter ξ regulates the balance between entropy-driven exploration and reward-focused exploitation. While the actor network focuses on improving the policy, the critic networks play a pivotal role in grounding these improvements in precise value predictions. As such, the critic networks $Q_{\psi_1}(\boldsymbol{s}, \boldsymbol{a})$ and $Q_{\psi_2}(\boldsymbol{s}, \boldsymbol{a})$ are trained to minimize the temporal difference

error, with the loss function as

$$\mathcal{J}_Q = \mathbb{E}_{(\boldsymbol{s},\boldsymbol{a},r,\boldsymbol{s}')\sim\mathcal{D}}\left[\left(Q_\psi(\boldsymbol{s},\boldsymbol{a}) - y'\right)^2\right], \tag{28}$$

where $(\boldsymbol{s}, \boldsymbol{a}, r, \boldsymbol{s}')$ represents samples drawn from the experience replay buffer $\mathcal{D}$, and y' is the target value, defined as:

$$y' = r + \gamma(\min_{j=1,2} Q_{\psi_j}(\boldsymbol{s}', \boldsymbol{a}') - \xi \log \pi_\phi(\boldsymbol{a}'|\boldsymbol{s}')). \tag{29}$$

Here, the minimum output between two critic networks is chosen in order to alleviate the issue of overestimation during training process and γ is the discount factor.

With the defined training objectives for both the actor and critic networks, the network parameters are then optimized by gradient-based methods. Subsequently, to ensure stability during training, the target networks $Q_{\psi'_1}(\boldsymbol{s}, \boldsymbol{a})$ and $Q_{\psi'_2}(\boldsymbol{s}, \boldsymbol{a})$ are slowly updated using a soft update formula:

$$\psi'_j \leftarrow \tau\psi_j + (1-\tau)\psi'_j, \quad j \in \{1, 2\}, \tag{30}$$

where τ denotes the update rate of the target network.

The training details of the proposed DDC algorithm for *Distributed Denoising Control* process is summarized in Algorithm 3. For each episode, the process starts with the observation of the initial state $\boldsymbol{s}$ and a random generated Gaussian noise $\boldsymbol{a}_T$ for exploration. At each step, an action $\boldsymbol{a}_0$ is generated through the denoising process of the deep diffusion model, according to the current state and exploration noise. The action then is executed in the system, producing a reward r and a new state $\boldsymbol{s}'$. This transition tuple is stored in the experience replay buffer $\mathcal{E}$, and a random batch of experiences $\mathcal{E}_b$ is sampled for updates. Subsequently, the critic networks are updated by minimizing the temporal difference error, the actor network is optimized to maximize reward and entropy, and the target networks are updated softly to stabilize learning. This iterative process continues over multiple episodes, resulting in an optimized policy π^* for distributed denoising control decisions. Importantly, by integrating the diffusion model into the SAC framework, the action generation process benefits from enhanced smoothness and flexibility, leading to more stable policy updates and improved exploration-exploitation trade-offs during training. During the inference phase, the well-trained policy π^* is employed to generate actions directly based on the observed state, enabling efficient and adaptive distributed denoising control decisions in our system.

VI. Performance Evaluation

A. Experimental Settings

1) Implementation: We consider an edge server providing AIGC services for five users, i.e., $K = 5$. The edge server is equipped with an NVIDIA GeForce RTX 3080Ti GPU, featuring 12GB of memory and delivering 34.1 TFLOPS of FP16 (half-precision floating point) computational performance. To simulate multiple mobile users with different computing capacities, we use a platform of an Nvidia AGX Orin with 64GB of memory and provides 10.6 TFLOPS of FP16 performance. Besides, Stable Diffusion v1.5 and ControlNet v1.1 models [13] are utilized for controllable text-to-image generation. The important experimental parameters are summarized in Table I.

TABLE I
Summary of Important Experimental Parameters

Symbol	Description	Value
$\boldsymbol{\lambda}$	Visual condition set	{Canny, HED, Depth, Seg}
$\boldsymbol{\alpha}$	Resolution level set	{0.125, 0.25, ..., 0.875, 1.0}
$\boldsymbol{\beta}$	Quality level set	{0.2, 0.3, ..., 0.9, 1.0}
f_{E}	Edge computing capacity	ε_0/0.11 FLOPS
f_k	Computing capacities of five user devices	$\{\frac{1}{10}, \frac{1}{4}, \frac{1}{2}, \frac{1}{3}, \frac{1}{2}\}f_{\mathrm{E}}$
C_{E}	Edge resource budget	$60\varepsilon_0$ FLOPs
C_k	Computing resource budget on five user devices	$\{\frac{1}{50}, \frac{1}{10}, \frac{1}{5}, \frac{1}{10}, \frac{1}{5}\}C_{\mathrm{E}}$
$\widetilde{\mathcal{Q}}_k$	QoE threshold	[0.6, 0.8]
$\widetilde{\mathcal{L}}$	Latency threshold	10 s
δ_i	Reward weighting factors	{0.15, 0.15, 0.6, 0.1}
$B^{\uparrow}, B^{\downarrow}$	Bandwidth condition	[5, 20] Mbps

2) Requests: We leverage the *DreamBooth* dataset [25] to generate our requests. This dataset includes 30 categories of subjects, comprising common objects, live subjects, and pets. Specifically, it contains a total of 158 images, each with a resolution of 512∗512, and every image is associated with 25 different prompts. Consequently, the dataset provides 3950 unique generation requests in total.

3) Benchmarks: We compare our proposed AIGC service system with the following benchmarks.

- *Random:* The proposed distributed AIGC service framework is utilized, but all the configurations are chosen randomly. This method serves as an ablation study to evaluate the effectiveness of all the proposed algorithms.
- *SemGen [22]: SemGen* selects the visual condition according to one desired evaluation metric and leverages a well-trained Deep Q-Network to determine the optimal downscaling factor for each visual condition. The downscaled visual conditions are then sent to the edge server for centralized generation with equal denoising steps.
- *AIGC-as-a-Service (AaaS) [11]:* Users first forward the source images and prompts to the edge server where the AIGC model operates. Then the model extracts the user-specified visual condition, completes the generation process with user-customized denoising steps.
- *OQC+VCE (OQC+) [43]:* The original *OQC* system employs successive convex approximation method to jointly optimize offloading decisions and denoising steps. The offloading decisions are binary, limiting the whole denoising execution to either the edge server or the user device. We integrate our VCE algorithm into the original *OQC* framework, resulting in the *OQC+* benchmark to ensure a fair comparison.

B. Overall Performances

1) Case Studies: We begin by presenting two distinct case studies to visually analyze the generation quality of various benchmarks, as illustrated in Fig. 8. Both cases aim for a target quality score of 0.8. In case Ⓐ, the user expresses a

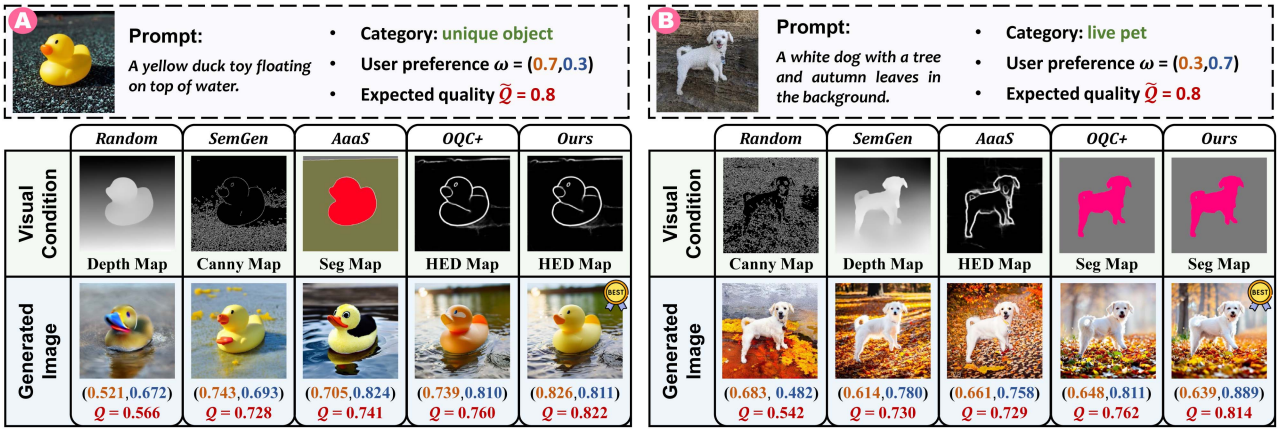


Fig. 8. Two generation case studies of different methods, where $(q^{\mathrm{I}}, q^{\mathrm{T}})$ denotes the DS Score and IR Score of the generated image, respectively.

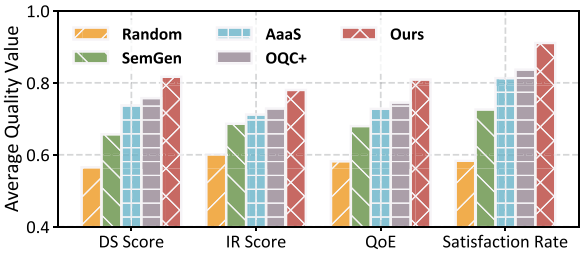


Fig. 9. Average generation quality values under different metrics.

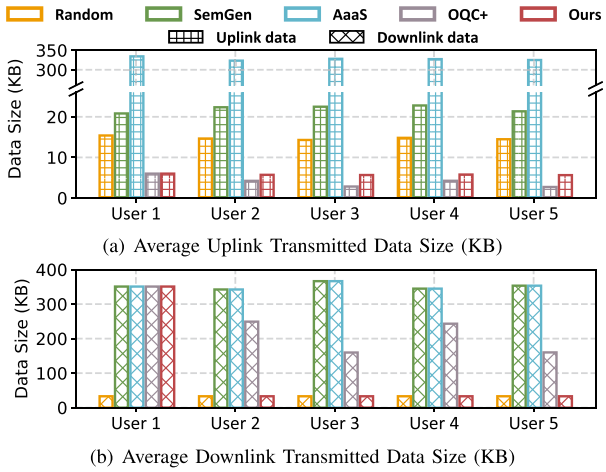


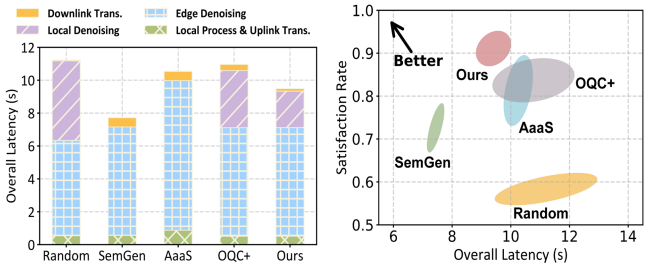
Fig. 10. Comparison of uplink and downlink transmitted data size of multiple users across different methods.

stronger preference for perceptual similarity, with preference vector being $\boldsymbol{\omega} = (0.7, 0.3)$ while in case Ⓑ, the user favors prompt alignment more strongly, with $\boldsymbol{\omega} = (0.3, 0.7)$.

The *Random* method selects generation settings randomly, leading to the poorest quality in both cases. Both *SemGen* and *AaaS* excel at only one evaluation metric due to their disregard of user preferences and inadequate denoising steps, thus resulting in a compromised overall generation quality. In contrast, *OQC+* and our system both incorporate the VCE algorithm for selecting and compressing visual conditions, leading to the satisfaction of user preferences. For instance, in case Ⓐ, we use the HED map as the visual condition, enabling the generation of images that closely resemble the original. By contrast, in case Ⓑ, we adopt the Seg map, which explicitly captures the full text prompt (e.g., both the tree and the autumn leaves), whereas other methods convey only part of the prompt details. Moreover, compared with *OQC+*, our proposed DDC algorithm provides a more effective denoising step decision, generating higher-quality images in both cases and being the only method that meets the user-expected quality of 0.8.

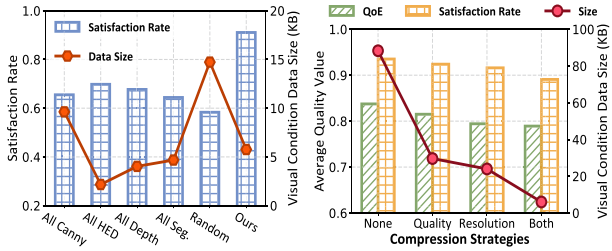
2) Overall Quality Comparison: We employ multiple metrics to evaluate the generation quality. Specifically, QoE is calculated through (1), based on user-specific preferences. The Satisfaction Rate is calculated to indicate the proportion of requests that achieve the QoE threshold set by users. The average generation quality results are shown in Fig. 9. The *Random* method arbitrarily selects configurations, meeting only 58% of user requests. *SemGen* considers either perceptual similarity or prompt alignment but neglects QoE and denoising step requirements, yielding 73% satisfaction. *AaaS* generates images using user-preferred, uncompressed visual conditions, but limited user knowledge and suboptimal resource utilization often lead to poor choices in visual conditions and denoising steps, resulting in 10% lower satisfaction than our system. *OQC+* combines our VCE algorithm with convex optimization for offloading decisions, achieving 83% satisfaction. In contrast, our framework offers finer control over local resource use, and our DDC algorithm enables more precise denoising step adjustment, boosting satisfaction to 91%.

3) Transmission Cost Comparison: We then evaluate the average uplink and downlink data sizes per user across different methods, as shown in Fig. 10. *AaaS* uploads raw images without extracting or compressing visual conditions, resulting in 335 KB per user. *SemGen* downscales visual conditions but still transmits nearly three times more data than ours. Both *AaaS* and *SemGen* generate images entirely on the edge, requiring an additional 350 KB per user for downlink transmission. *Random* method uses our distributed framework, uploading compressed visual



(a) Latency decomposition. (b) QoE-Latency trade-off.

Fig. 11. Overall latency analysis of different methods.



(a) Satisfaction rate and data volume comparison under different visual condition selection methods. (b) QoE and visual condition data size comparison under different compression strategies.

Fig. 12. Effectiveness of the VCE algorithm.

conditions and downloading intermediate latents, both with smaller data sizes. *OQC+* either generates images locally to eliminating uplink traffic or transmits compressed visual conditions for edge generation, slightly reducing uplink size compared to ours. However, when generating at the edge, it still incurs high downlink costs due to full-image transmission. In contrast, our method consistently uploads only compressed visual conditions (6 KB on average). Furthermore, our DDC algorithm provides a more fine-grained image generation scheme for different users. For User 1 with insufficient computing resources for local denoising, the generated image must be transmitted back at a high transmission cost. While for other users who can perform the denoising process locally, the cost is greatly reduced due to the transmission of only intermediate latents, which requires an average of 33 KB of data volume.

In summary, the designed system addresses fluctuating bandwidth in mobile scenarios with two adaptive mechanisms: the VCE algorithm dynamically adjusts compression settings based on uplink conditions to balance transmission efficiency and visual quality, while the DDC algorithm reallocates denoising tasks according to downlink bandwidth, reducing offloaded data at lower transmission overhead.

4) Overall Latency Analysis: We then analyze the overall generation latency for each round, which consists of generation requests from all users, as illustrated in Fig. 11(a). The *Random* method achieves low transmission latency due to small data size, but its arbitrary offloading strategy increases local computational load and thus local processing latency. Both *SemGen* and *AaaS* generate images entirely on the edge server, and therefore incur no local denoising latency. Nevertheless, *AaaS* suffers from higher transmission latency as it involves transmitting raw images, and also experiences longer edge generation latency due to the necessity of extracting visual conditions for all users on the edge server, resulting in 10.53 s overall generation latency. In contrast, both *OQC+* and our method extract visual conditions locally and transmit the compressed data, leading to a latency of only about 0.5 s. Our method further reduces local generation latency by 1.2 s compared to *OQC+* due to the DDC algorithm's optimal denoising decisions. Additionally, by primarily transmitting intermediate latent data, our approach minimizes downlink transmission latency. We further provide a detailed comparison of the QoE-Latency trade-off for different methods in Fig. 11(b). With the satisfaction rate of 91% and the overall latency of 9.5 s, our method achieves the highest satisfaction rate with only a slight increase in latency than *SemGen*, demonstrating our superior balance between generation quality and latency.

C. Effectiveness of the VCE Algorithm

1) Impact of Visual Condition Type: To evaluate the performance of our proposed VCE algorithm, we begin by comparing the satisfaction rate and transmission cost across various visual condition selection methods, as illustrated in Fig. 12(a). While employing just one visual condition type may offer slight reductions in data size consumption with our compression configuration selection algorithm, they tend to excel in just one aspect of image quality, failing to address the diverse QoE requirements of users with different preferences. Consequently, even the best-performing *All HED* solution can achieve only about 70% user satisfaction. Besides, the *Random* method randomly selects visual conditions and compression settings, leading to low QoE and high transmission cost. In comparison, our method adapts visual conditions based on user preferences, leading to a 21% improvement in satisfaction while increasing the data volume by just 3 KB.

2) Impact of Compression Settings: Next, we demonstrate the effectiveness of our compression configuration selection algorithm in terms of generation quality and transmission cost, illustrated in Fig. 12(b). When no compression is applied, the original visual conditions demand an average transmission size of 88 KB, resulting in the best generation quality of 94% satisfaction rate. When either resolution or quality is compressed individually, with the other fixed at its optimal setting, our algorithm identifies the best compression configurations to achieve a significant reduction in data size with minimal quality degradation. Moreover, by jointly compressing resolution and quality, our method enables better data size reduction while preserving considerable generation quality, demonstrating the efficiency of compressing both parameters.

D. Effectiveness of the DDC Algorithm

1) DRL Algorithm Comparison: We first compare the training convergence of our proposed DDC algorithm with two benchmark DRL methods: PPO [44] and SAC [45]. As shown in Fig. 13, PPO struggles to converge within 40 k steps and

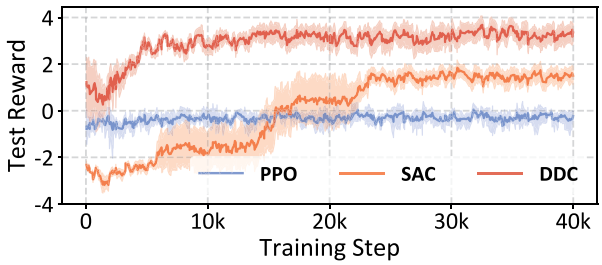


Fig. 13. Comparison of test reward curves across different DRL methods.

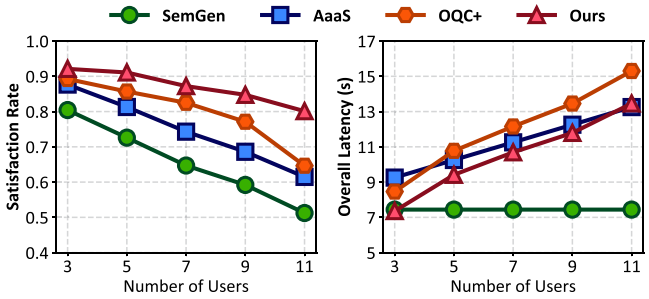
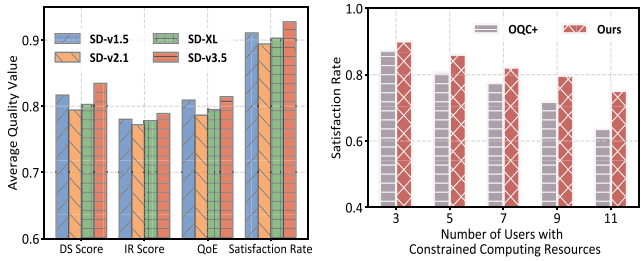


Fig. 14. Impact of user numbers on the satisfaction rate and overall latency.

yields low test rewards, while SAC converges around 25 k steps with moderate performance. In contrast, DDC integrates a deep diffusion model into the SAC framework, enhancing policy learning with stronger modeling capabilities. This leads to faster convergence and higher test rewards, validating the effectiveness of our design. For practical deployment on edge servers, we adopt a lightweight policy network with moderate hidden dimensions and fix the denoising steps to three. As a result, DDC inference takes about 600 ms per decision, which is only slightly higher than original SAC's 400 ms and remains negligible compared to the image generation time.

2) Impact of Number of Users: We then investigate how the number of users affects the generation quality and overall latency, as presented in Fig. 14. To scale the experiment to more users, we systematically assign computing capacities by iterating through the five user types defined in Table I in a round-robin manner. The results show that all methods experience performance degradation as the number of users increases, mainly due to limited edge resources restricting denoising steps per user. Methods like *AaaS* and *SemGen*, which rely solely on edge computation, are particularly impacted, showing notable quality drops. For *SemGen*, latency remains relatively constant since edge denoising steps are fixed, while *AaaS* incurs increasing latency due to extracting visual conditions for more users. In contrast, *OQC+* and our method leverage user-side resources to offload denoising, improving generation quality. As user count rises to 11, our method sees only a 9% drop in satisfaction and a latency increase of about 3.9 s. To adapt to new users, the DRL policy within the DDC algorithm needs to be retrained to accommodate updated computing resource constraints and user demand profiles.



(a) Performance comparison across different T2I models on the average generation quality.

(b) Impact of user numbers on the satisfaction rate where all users are computing resource constrained.

Fig. 15. System generalizability analysis.

E. System Generalizability

1) Performance Under More T2I Models: To further validate generalizability, we conduct experiments using recent diffusion-based T2I models, including SD-v2.1, SD-XL, and SD-v3.5, beyond the baseline SD-v1.5. As shown in Fig. 15(a), our system demonstrates consistently strong performance across all models, with SD-v3.5 achieving the highest satisfaction rate of 92.76%. However, SD-v2.1 and SD-XL exhibit slightly lower metrics, primarily due to our experiments use a fixed 512∗512 resolution, which aligns with SD-v1.5's design. However, SD-v2.1 and SD-XL are optimized for higher resolutions, and generating at 512∗512 underutilizes their capabilities, resulting in degraded outputs. In contrast, SD-v3.5 better accommodates varying resolutions and resource conditions, aligning well with our distributed framework.

2) Performance Under Low-End Devices: To evaluate the robustness of our system when all user devices lack sufficient computational resources for local denoising, we compare our method with *OQC+* under different numbers of users and measure the satisfaction rate, as shown in Fig. 15(b). Results show that although the system performance decreases due to limited edge capacity, our approach still performs well. For example, with five users, the satisfaction rate drops by only 4%. As user numbers grow, our DRL-based step allocation strategy becomes more advantageous. When the number of users reaches eleven, our method achieves a satisfaction rate 10% higher than *OQC+*, demonstrating its effectiveness even in severely resource-constrained environments.

VII. Conclusion

In this paper, we have presented a system design for distributed and controllable T2I generation tasks that accommodates diverse user preferences. Specifically, we have proposed the VCE algorithm for visual condition engineering, optimizing visual condition configurations according to comprehensive offline profiles of generation quality and transmission efficiency. Furthermore, we have developed the DDC algorithm, leveraging a deep diffusion model-enhanced DRL approach to effectively choose the best cooperative denoising strategy between the edge and user for completing the generation task. The outcome of this paper could benefit significantly to large-scale and user-centric

AIGC applications in resource-constrained mobile scenarios. For future work, we aim to enhance the scalability of the proposed system. This includes extending it to a multi-edge collaborative framework to support a larger user base and exploring challenges such as user-edge coordinated scheduling under dynamic network conditions.

REFERENCES

[1] T. B. Brown, "Language models are few-shot learners," 2020, *arXiv: 2005.14165*.

[2] F.-A. Croitoru, V. Hondru, R. T. Ionescu, and M. Shah, "Diffusion models in vision: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 9, pp. 10850–10869, Sep. 2023.

[3] J. Wang et al., "A unified framework for guiding generative AI with wireless perception in resource constrained mobile edge networks," *IEEE Trans. Mobile Comput.*, vol. 23, no. 11, pp. 10344–10360, Nov. 2024.

[4] X. Zhang et al., "Beyond the cloud: Edge inference for generative large language models in wireless networks," *IEEE Trans. Wireless Commun.*, vol. 24, no. 1, pp. 643–658, Jan. 2025.

[5] J. Hou, P. Yang, X. Dai, T. Qin, and F. Lyu, "Enhancing cooperative LiDAR-based perception accuracy in vehicular edge networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 26, no. 6, pp. 8283–8296, Jun. 2025.

[6] S. D. Okegbile, H. Gao, O. Talabi, J. Cai, D. Niyato, and X. Shen, "Optimizing federated semantic learning in distributed AIGC-enabled human digital twins: A multi-criteria and multi-shard user selection framework," *IEEE Trans. Mobile Comput.*, vol. 24, no. 7, pp. 5916–5933, Jul. 2025.

[7] Y. He, P. Yang, T. Qin, J. Hou, and N. Zhang, "Joint encoding and enhancement for low-light video analytics in mobile edge networks," *IEEE Trans. Mobile Comput.*, vol. 24, no. 4, pp. 3330–3345, Apr. 2025.

[8] M. Xu et al., "Unleashing the power of edge-cloud generative AI in mobile networks: A survey of AIGC services," *IEEE Commun. Surv. Tut.*, vol. 26, no. 2, pp. 1127–1170, Second Quarter 2024.

[9] C. Zhou, S. Hu, J. Gao, X. Huang, W. Zhuang, and X. Shen, "User-centric immersive communications in 6G: A data-oriented framework via digital twin," *IEEE Wireless Commun.*, vol. 32, no. 3, pp. 122–129, Jun. 2025.

[10] H. Du et al., "Exploring collaborative distributed diffusion-based AI-generated content (AIGC) in wireless networks," *IEEE Netw.*, vol. 38, no. 3, pp. 178–186, May 2024.

[11] H. Du et al., "Diffusion-based reinforcement learning for edge-enabled AI-generated content services," *IEEE Trans. Mobile Comput.*, vol. 23, no. 9, pp. 8902–8918, Sep. 2024.

[12] Y. Liu et al., "Cross-modal generative semantic communications for mobile AIGC: Joint semantic encoding and prompt engineering," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 14871–14888, Dec. 2024.

[13] L. Zhang, A. Rao, and M. Agrawala, "Adding conditional control to text-to-image diffusion models," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 3836–3847.

[14] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3431–3440.

[15] S. Xie and Z. Tu, "Holistically-nested edge detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2015, pp. 1395–1403.

[16] Y. Liang, P. Yang, Y. He, and F. Lyu, "Resource-efficient generative AI model deployment in mobile edge networks," in *Proc. IEEE Glob. Commun. Conf.*, 2024, pp. 2647–2652.

[17] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 6840–6851.

[18] S. Gao, P. Yang, Y. Kong, F. Lyu, and N. Zhang, "Characterizing and scheduling of diffusion process for text-to-image generation in edge networks," *IEEE Trans. Mobile Comput.*, vol. 24, no. 10, pp. 11137–11150, Oct. 2025, doi: 10.1109/TMC.2025.3574065.

[19] Y. Liu et al., "CacheGen: KV cache compression and streaming for fast large language model serving," in *Proc. ACM SIGCOMM Conf.*, 2024, pp. 38–56.

[20] H. Du et al., "User-centric interactive AI for distributed diffusion model-based AI-generated content," 2023, *arXiv:2311.11094*.

[21] D. Zavadski, J.-F. Feiden, and C. Rother, "ControlNet-XS: Designing an efficient and effective architecture for controlling text-to-image diffusion models," 2023, *arXiv:2312.06573*.

[22] G. Liu et al., "Semantic communications for artificial intelligence generated content (AIGC) toward effective content creation," *IEEE Netw.*, vol. 38, no. 5, pp. 295–303, Sep. 2024.

[23] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

[24] T. Kanade and M. Okutomi, "A stereo matching algorithm with an adaptive window: Theory and experiment," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 9, pp. 920–932, Sep. 1994.

[25] N. Ruiz, Y. Li, V. Jampani, Y. Pritch, M. Rubinstein, and K. Aberman, "DreamBooth: Fine tuning text-to-image diffusion models for subject-driven generation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 22500–22510.

[26] S. Fu et al., "DreamSim: Learning new dimensions of human visual similarity using synthetic data," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2023, pp. 50742–50768.

[27] J. Xu et al., "ImageReward: Learning and evaluating human preferences for text-to-image generation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2023, pp. 15903–15935.

[28] Z. Su et al., "Pixel difference networks for efficient edge detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 5117–5127.

[29] B. Dong, P. Wang, and F. Wang, "Head-free lightweight semantic segmentation with linear transformer," in *Proc. AAAI Conf. Artif. Intell.*, 2023, pp. 516–524.

[30] R. Cheng, Y. Sun, D. Niyato, L. Zhang, L. Zhang, and M. A. Imran, "A wireless AI-generated content (AIGC) provisioning framework empowered by semantic communication," *IEEE Trans. Mobile Comput.*, vol. 24, no. 3, pp. 2137–2150, Mar. 2025.

[31] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 10684–10695.

[32] P. Yang, Q. Feng, Z. Huang, J. Chen, and N. Zhang, "Hierarchical forwarding resource allocation with proactive frame dropping for VR video streaming," *IEEE Trans. Netw. Sci. Eng.*, early access, Jul. 28, 2025, doi: 10.1109/TNSE.2025.3593299.

[33] C. Wang, P. Yang, J. Hou, Z. Liu, and N. Zhang, "Dependence-aware multitask scheduling for edge video analytics with accuracy guarantee," *IEEE Internet Things J.*, vol. 11, no. 16, pp. 26970–26983, Aug. 2024.

[34] Y. Kong, P. Yang, and Y. Cheng, "Adaptive on-device model update for responsive video analytics in adverse environments," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 35, no. 1, pp. 857–873, Jan. 2025.

[35] X. Xu et al., "Accelerating mobile edge generation (MEG) by constrained learning," 2024, *arXiv:2407.07245*.

[36] C. Yan et al., "Hybrid SD: Edge-cloud collaborative inference for stable diffusion models," 2024, *arXiv:2408.06646*.

[37] Y. Hua, R. Li, Z. Zhao, X. Chen, and H. Zhang, "GAN-powered deep distributional reinforcement learning for resource management in network slicing," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 2, pp. 334–349, Feb. 2020.

[38] Y.-L. Jin, Z.-Y. Ji, D. Zeng, and X.-P. Zhang, "VWP: An efficient DRL-based autonomous driving model," *IEEE Trans. Multimedia*, vol. 26, pp. 2096–2108, 2022.

[39] S. Hu, L. Shen, Y. Zhang, Y. Chen, and D. Tao, "On transforming reinforcement learning with transformers: The development trajectory," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 12, pp. 8580–8599, Dec. 2024.

[40] H. Du et al., "Enhancing deep reinforcement learning: A tutorial on generative diffusion models in network optimization," *IEEE Commun. Surv. Tuts.*, vol. 26, no. 4, pp. 2611–2646, Fourth Quarter 2024.

[41] L. Xiao, "On the convergence rates of policy gradient methods," *J. Mach. Learn. Res.*, vol. 23, no. 1, 2022, Art. no. 282.

[42] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," 2016, *arXiv:1611.01144*.

[43] Y. Wang, C. Liu, and J. Zhao, "Offloading and quality control for AI generated content services in 6G mobile edge computing networks," in *Proc. IEEE Veh. Technol. Conf.*, 2024, pp. 1–7.

[44] J. Schulman et al., "Proximal policy optimization algorithms," 2017, *arXiv: 1707.06347*.

[45] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1861–1870.

Yuxin Kong received the BE degree in electronic information engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2023, where he is currently working towards the ME degree in Information and Communication Engineering. His research interests include mobile edge computing, video analytics, and mobile AIGC.

Peng Yang (Member, IEEE) received the BE degree in communication engineering and the PhD degree in information and communication engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2013 and 2018, respectively. He was with the Department of Electrical and Computer Engineering, University of Waterloo, Canada, as a visiting PhD student from 2015 to 2017 and a postdoctoral fellow from 2018 to 2019. Since 2020, he has been an associate professor with the School of Electronic Information and Communications, HUST. His current research focuses on mobile-edge computing, video analytics, and virtual reality, and mobile generative AI.

Xue Qin received the BEng degree from the North University of China, in 2008, and the MASc degree from the University of Windsor, Windsor, ON, Canada, in 2022. She is currently working toward the PhD degree in electrical and computer engineering with the University of Waterloo, Waterloo, ON, Canada. Her research interests include mobile edge computing, AI, and network security.

Jizhe Zhou received the BS degree in electronic and information engineering from Beihang University, China, in 2015, the MS degree in computer science from Columbia University, in 2017, and the PhD degree from the Beijing University of Posts and Telecommunications (BUPT), China, in 2021. She has been a senior engineer with the China Academy of Information and Communications Technology (CAICT). She is also the vice chair of Agent Communication Network Sub-Working Group (TC5WG12SWG1) of China Communications Standards Association (CCSA). Her research interests include network architecture, edge intelligence, and computing-aware networks.

Xuemin (Sherman) Shen (Fellow, IEEE) received the PhD degree in electrical engineering from Rutgers University, New Brunswick, New Jersey, in 1990. He is a University professor with the Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research focuses on network resource management, wireless network security, Internet of Things, 5G and beyond, and vehicular networks. He is a registered professional engineer of Ontario, Canada, an Engineering Institute of Canada fellow, a Canadian Academy of Engineering fellow, a Royal Society of Canada fellow, a Chinese Academy of Engineering foreign member, and an International fellow of the Engineering Academy of Japan. He received "West Lake Friendship Award" from Zhejiang Province, in 2023, the President's Excellence in Research from the University of Waterloo, in 2022, the Canadian Award for Telecommunications Research from the Canadian Society of Information Theory (CSIT), in 2021, the R.A. Fessenden Award, in 2019 from IEEE, Canada, the Award of Merit from the Federation of Chinese Canadian Professionals (Ontario), in 2019, the James Evans Avant Garde Award, in 2018 from the IEEE Vehicular Technology Society, the Joseph LoCicero Award, in 2015 and Education Award, in 2017 from the IEEE Communications Society (ComSoc), the Technical Recognition Award from Wireless Communications Technical Committee, in 2019, and the AHSN Technical Committee, in 2013. He has also received the Excellent Graduate Supervision Award, in 2006 from the University of Waterloo and the Premier's Research Excellence Award, in 2003 from the Province of Ontario, Canada. He serves/served as the general chair for the 6G Global Conference'23, and ACM Mobihoc'15, Technical Program Committee Chair/Co-Chair for IEEEGlobecom'24, 16 and 07, IEEE Infocom'14, IEEE VTC'10 Fall, and the chair for the IEEE ComSoc Technical Committee on Wireless Communications. He is the past president of the IEEE ComSoc, the vice president for Technical & Educational Activities, vice president for Publications, a Member-at-Large on the Board of Governors, a chair of the Distinguished Lecturer Selection Committee, and a member of IEEE Fellow Selection Committee of the ComSoc. He served as an editor-in-chief for *IEEE Internet of Things Journal*, *IEEE Network*, and *IET Communications*.