This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

NAMI: Efficient Image Generation via Bridged Progressive Rectified Flow Transformers

Yuhang Ma[1*] Bo Cheng[1*] Shanyuan Liu[1*] Hongyi Zhou[2*] Liebucha Wu[1]
Dawei Leng[1†] Yuhui Yin[1]
[1]360 AI Research [2]Tsinghua University



Figure 1. High-quality image synthesis results from NAMI-2B demonstrate its capabilities in precise prompt following, spatial reasoning, and aesthetic quality.

Abstract

Flow-based Transformer models have achieved state-of-the-art image generation performance, but often suffer from high inference latency and computational cost due to their large parameter sizes. To improve inference efficiency without compromising quality, we propose Bridged Progressive Rectified Flow Transformers (NAMI), which decompose the generation process across temporal, spatial, and architectural demensions. We divide the rectified flow into different stages according to resolution, and use a BridgeFlow module to connect them. Fewer Transformer layers are used at low-resolution stages to generate image layouts and concept contours, and more layers are progressively added as the resolution increases. Experiments demonstrate that our approach achieves fast convergence and reduces inference time while ensuring generation quality. The main contributions of this paper are summarized as follows: (1) We introduce Bridged Progressive Rectified Flow Transformers that enable multi-resolution training, accelerating model convergence; (2) NAMI leverages piecewise flow and spatial cascading of Diffusion Transformer (DiT) to rapidly generate images, reducing inference time by 64% for generating 1024×1024 resolution images; (3) We propose a BridgeFlow module to align flows between different stages; (4) We propose the NAMI-1K benchmark to evaluate human pref-

*Equal contribution.
†Corresponding author. lengdawei@360.cn

erence performance, aiming to mitigate distributional bias and comprehensively assess model effectiveness. The results show that our model is competitive with state-of-the-art models.

1. Introduction

Over the past year, text-to-image (T2I) models based on both diffusion [8, 16, 32, 38] and autoregressive approaches [6, 31] have achieved significant advancements in image generation quality and inference efficiency. The diffusion model approach, represented by SD3 [9] and FLUX [20], leverages rectified flow [1, 22, 24] and MM-DiT [9] architectures to achieve excellent performance. However, the increased model parameters have significantly raised training and inference costs, making commercialization more challenging. Concurrently, autoregressive (AR) models generate images through next-token prediction [26]. These methods use vector quantization variational autoencoder (VQVAE) technology [40] to discretize image tokens, and then predict the next component of the image based on the previously generated tokens. The size of the codebook plays a critical role in determining both the efficiency of image generation and the image quality. Currently, AR models such as LlamaGen [39] and Infinity [12] exhibit significant advantages in inference speed, but their image generation quality still falls short.

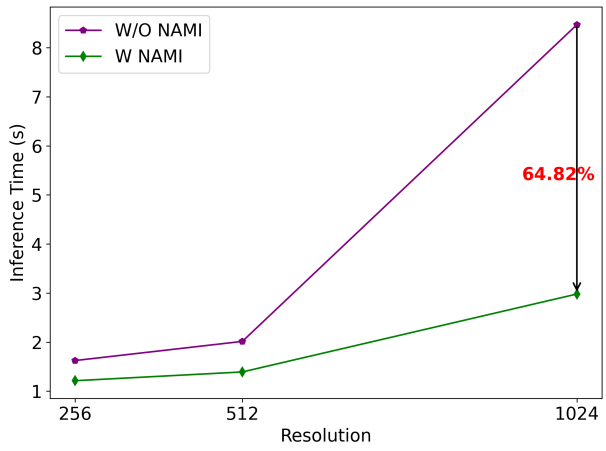


Figure 2. An overview of inference latency between the proposed NAMI-2B and the corresponding FLUX-2B base model of the same size without NAMI. With NAMI, inference performance improvement becomes more significant as image resolution increases. The measurements are conducted with a batch size of 1 on an A100 GPU.

Current research on optimizing the training and inference speed of diffusion models is primarily focused on areas such as latent space downsampling, reducing the number of tokens input into the DiT [27] block, and efficient attention mechanisms [42]. Several approaches have been proposed to reduce the number of input image tokens, one of which involves decomposing the image generation process from low resolution to high resolution. The matryoshka diffusion model (MatryoshkaDM) [11] innovatively integrates the low-resolution diffusion process as part of the high-resolution generation, utilizing a nested UNet [33] architecture to accelerate model training. Another approach involves simultaneous upsampling of the image and denoising within the diffusion model, where the pyramid flow matching [19] method breaks down the video generation process into multiple resolution stages for independent denoising. While these methods improve the efficiency of image training or generation, each still has its own limitations.

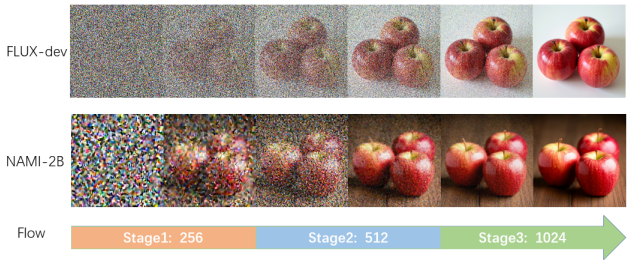


Figure 3. Overview of the image generation process for FLUX-dev [20] and our NAMI-2B, with upscaling alignment applied during the low-resolution stages of NAMI-2B.

We observe that in the image generation process of diffusion models, rough conceptual placements and outline layouts are performed in the early stages, while detail enhancement occurs in the later stages. Currently, most methods perform unified denoising across all sampling stages without considering the underlying mechanics of the image generation process. The early layout generation process can not only be quickly executed at low resolution but can also be modeled using a subset of the model parameters. Therefore, jointly considering temporal segmentation in the sampling strategy and the rational allocation of the model space is crucial to improving the efficiency of image generation.

In this paper, we propose a spatiotemporal separation progressive framework, NAMI. We divide the rectified flow into different stages based on resolution, connecting stages with BridgeFlow. Fewer transformer layers are used at the low-resolution stages to generate image layouts and concept contours, while more layers are progressively added as the resolution increases. As shown in the Fig. 3, we visualize the image generation process of FLUX-dev [20] and NAMI-2B under the same inference steps. Despite using a lower resolution and fewer model parameters in the early stages, NAMI-2B still effectively generates image layouts and concepts. Furthermore, considering the overall process,

NAMI-2B achieves a faster generation speed.

Current benchmarks such as GenEval [10], DPG-Benchmark [18], and ImageReward [44], are primarily used to evaluate a model's performance. However, these benchmarks suffer from issues such as limited caption quantity, distribution biases. Therefore, a more comprehensive benchmark is essential for accurately evaluating a model's capabilities. We combine open benchmarks, AI-generated, and human-created test prompts to construct a multidimensional evaluation dataset named NAMI-1K.

In summary, Our primary contributions are as follows:

1. Our method enables direct training on multi-resolution images, facilitating rapid learning of semantics at low resolutions and obtain details and quality at high resolutions, thereby accelerating model convergence.
2. The NAMI-2B employs temporal separation and spatial cascading within the model to efficiently generate images. As shown in Fig. 2, using the same model size, our approach reduces the time required to generate 1024 resolution images by 64%, while still maintaining a high level of image quality.
3. Furthermore, we propose a BridgeFlow module to align flows between different stages.
4. We propose the NAMI-1K benchmark and evaluate our model on both NAMI-1K and open-source benchmarks. The results across multiple metrics consistently highlight the effectiveness of the proposed method.

2. Related Work

Diffusion models [8, 16, 32, 38] have become a powerful framework in image generation, with latent diffusion models further enhancing both quality and efficiency. DiT [27] has introduced a transformative change by replacing the conventional UNet [33] architecture used in models like SDXL [29] with a transformer [41]-based framework. This advancement has led to the development of models such as PixArt-α [4], Hunyuan-DiT [21], LUMINA-Next [48], SD3 [9], and FLUX [20], which efficiently integrate multimodal information to improve text-to-image generation, thereby demonstrating the effectiveness of transformer-based approaches. Notably, SD3 and FLUX employ rectified flow [1, 22, 24], which connects data and noise on a straight line, improving both training and inference efficiency. Additionally, the MM-DiT [9] block has been proposed, achieving state-of-the-art generation performance. Our work builds upon MM-DiT as the fundamental module while adopting rectified flow as a specific forward path selection strategy.

Some studies have focused on optimizing the performance of diffusion models. SANA [43] utilizes a higher downsampling rate for VAE [5] and replaces vanilla quadratic attention modules with linear attention to enhance efficiency. MicroDiT [36] reduces training costs by using masks to decrease the number of input tokens. CLEAR [23] applies convolution-like local attention strategies, constraining interactions to a local window around the query token to achieve linear complexity. Although these methods provide efficiency improvements, they inevitably result in some degradation of image generation quality due to the high compression ratio of VAE [5] or the reduction in token interactions. Our approach differs from these methods by decomposing the generation process, reducing model redundancy while preserving generation quality. Furthermore, our optimization strategy is orthogonal to the aforementioned methods, allowing for parallel integration.

The denoising process [16] in diffusion models can be viewed as an image generation process that progresses from coarse-grained layouts to fine-grained details. Some studies have modeled this process in conjunction with multi-scale image generation. For example, LAPGAN [7] generates images by first producing low-resolution images and then feeding them into a high-resolution model. Pyramidal denoising diffusion probabilistic models [34] have also adopted a similar approach. Pyramid flow matching [19] accelerates inference performance by implementing a pyramidal strategy over time steps. MatryoshkaDM [11] uses different-sized models at different resolution scales, enabling the training of diffusion models in pixel space. These methods have reduced training complexity and accelerated inference by leveraging multiscale resolutions. However, they have not fully addressed the issue of parameter redundancy in DiT [27] models. Our work leverages models of varying sizes across different scale stages to maximize performance efficiency while maintaining generation quality.

3. Method

3.1. Preliminary Study

Flow-based [25, 45] generative models are similar to diffusion models, where the objective is to learn a velocity field $v_\theta(x_t, t)$ that maps random noise $x_0 \sim \mathcal{N}(0, I)$ to data samples $x_1 \sim D$, via an ordinary differential equation (ODE):

$$\frac{dx_t}{dt} = v_\theta(x_t, t), \quad x_0 \sim \mathcal{N}(0, I) \tag{1}$$

Subsequently, a simple simulation-free training objective for flow generative models has been proposed. A representative method is rectified flow [1, 22, 24], which adopts linear interpolation between the noise distribution x_0 and the data distribution x_1. It trains a neural network to approximate the velocity field via the conditional flow matching loss. The optimization procedure is termed reflow:

$$\min_\theta \mathbb{E}_{x_0 \sim \mathcal{N}(0,I), x_1 \sim D}\left[\int_0^1 \|(x_0 - x_1) - v_\theta(x_t, t)\|^2 \, dt\right] \tag{2}$$

where $x_t = tx_1 + (1-t)x_0$.

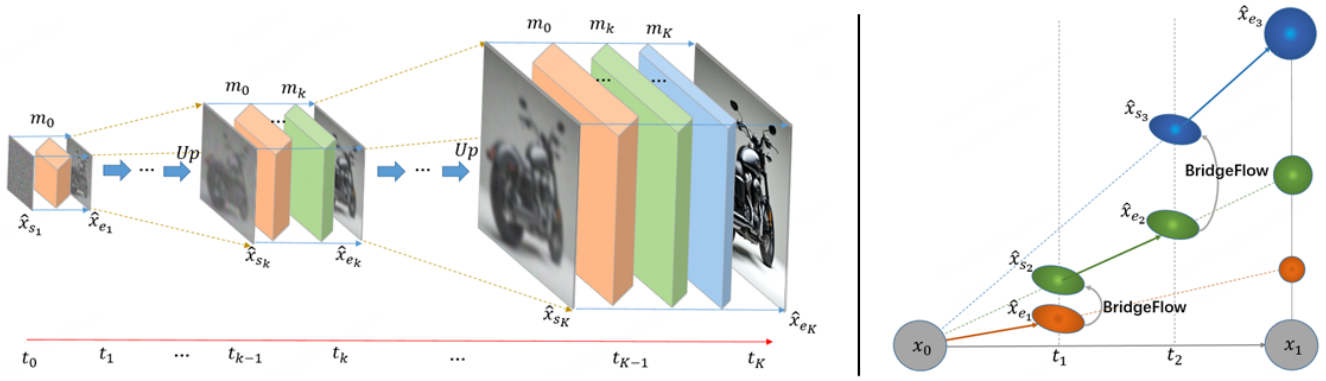


Figure 4. Overview of NAMI: The left figure shows the progressive flow transformers of NAMI, where the same color represents the same module. The right figure depicts the integration of the BridgeFlow module, which establishes connections across adjacent time windows. Specifically, we divide the image generation process into K resolution stages and the entire flow is divided into K time windows, where adjacent stages are connected through upsampling and the BridgeFlow module. We use fewer transformer layers at the low-resolution stages to generate image layouts and concept contours, progressively adding more layers as the resolution increases.

3.2. Progressive Rectified Flow Transformers

Our method treats the image generation process as a transformation from a low-resolution image containing coarse-grained concepts to a high-resolution image with rich details. At low resolution, we employ piecewise flow [47] and partial modules to reduce computational costs. As shown in Fig. 4 , we divide the image generation into K resolution stages, where $Up(.,.)$ denotes the upsampling function and $Down(.,.)$ denotes the downsampling function. For flow processing, we refer to pyramid flow matching [19] and divide the entire flow into K time windows $\{[t_{k-1}, t_k]\}_{k=1}^{K}$, where $1 = t_K > \cdots > t_k > t_{k-1} > \cdots > t_0 = 0$. Linear interpolation is also used between noise and data distribution, that is, $x_{t_k} = t_k x_1 + (1 - t_k)x_0$. The starting point for each time window is given by $\hat{x}_{s_k} = BridgeFlow(Up(Down(x_{t_{k-1}}, 2^{k+1})))$, which is obtained by upsampling the endpoint of the previous stage. The end point is given by $\hat{x}_{e_k} = Down(x_{t_k}, 2^k)$, which is obtained by downsampling x_1 and adding noise. The flow within the window is given by:

$$\hat{x}_t = t'\hat{x}_{e_k} + (1 - t')\hat{x}_{s_k}, \quad t' = (t - t_{k-1})/(t_k - t_{k-1}). \quad (3)$$

Additionally, we divide the corresponding model into K modules, where each module m_k ($k \in [1, K]$) consists of MM-DiT [9] blocks. For the k-th time window, the model is denoted as $\theta_k = \{m_1 \oplus \cdots \oplus m_k\}_{k=1}^{K}$, where $\oplus$ represents the concatenation of the blocks. Finally, our overall optimization process can be expressed as:

$$\min_{\theta_k} \sum_{k=1}^{K} E_{(k,t,(\hat{x}_{s_k},\hat{x}_{e_k}))} \left[\int_{t_{k-1}}^{t_k} \|(\hat{x}_{s_k} - \hat{x}_{e_k}) - v_{\theta_k}(\hat{x}_t, t)\|^2 \, dt \right] \quad (4)$$

3.3. BridgeFlow Module

To ensure the continuity of probabilistic paths across different stages, Pyramid Flow [19] treats the jump points as Gaussian distribution matches and derives the corresponding transitions of the mean and covariance matrices. During inference, it performs rescaling and renoising to handle these transitions. However, this process is non-parametric and lacks learnable adaptation, which limits its robustness. Moreover, the renoising step has a time complexity proportional to the image token length, resulting in unsatisfactory efficiency and performance.

To overcome these limitations, we propose the BridgeFlow module, which introduces a learnable linear transformation to align the probability distributions at stage boundaries in a data-driven manner. Specifically, for the endpoint of each stage, we first apply upsampling to match the resolution, followed by a linear transformation $\hat{x}_{s_k} = W \cdot \mathrm{Up}(\hat{x}_{e_{k-1}}) + B$ to match the distribution of the subsequent stage's starting point. Further analysis is provided in the Appendix C. In practice, every BridgeFlow module is pretrained with a mean squared error (MSE) loss according to the defined flow partitions. The detailed ablation results are presented in Tab. 7.

3.4. Multi-Resolution Training

Considering knowledge sharing between modules and efficient training, we propose a multi-resolution progressive approach during the training process. Different resolutions of data can be used at different stages, and the optimization follows the approach in Eq. (4). The training process is shown in Algorithm 1. Unlike the conventional method of training at low resolutions first and then fine-tuning at higher resolutions separately, our approach enables the joint training of data with different resolutions simultaneously. This facilitates knowledge sharing within the model and helps prevent catastrophic forgetting during fine-tuning at high resolutions. At each stage, we train using images with

Algorithm 1 Multi-resolution Training with Progressive Rectified Flow Transformers

Input: Number of windows K, Multi-resolution Datasets $D = \{D_k\}_{k=1}^{K}$
Create K time windows $[t_{k-1}, t_k]$ for $k = 1$ to K with $t_K = 1, t_0 = 0$
Initialize $\theta_k = \{m_1 \oplus \cdots \oplus m_k\}_{k=1}^{K}$
repeat
  for each time window $[t_{k-1}, t_k]$ **do**
    Sample $x'_1 \sim D'\{D_k, D_{k+1}, ..., D_K\}$
    do $Down(D', .)$ until $x'_1 \sim \{D_k, D_k, ..., D_k\}$
    Compute the start point
    if $k = 1$ **then**
      $\hat{x}_{s_k} \leftarrow x_0$
    else
      $\hat{x}_{s_k} \leftarrow \text{BridgeFlow}(\text{Up}(\text{Down}(x'_{t_{k-1}}, 2)))$
    end if
    Compute the end point $\hat{x}_{e_k} = x'_{t_k}$
    Sample $t \in [t_{k-1}, t_k]$, compute $\hat{x}_t = t'\hat{x}_{e_k} + (1 - t')\hat{x}_{s_k}$ with $t' = (t - t_{k-1})/(t_k - t_{k-1})$
    Compute the loss:

$$\text{loss} = \|(\hat{x}_{s_k} - \hat{x}_{e_k}) - v_{\theta_k}(\hat{x}_t, t)\|^2$$

  end for
  Weighting the losses for different time windows and perform backpropagation
until convergence

resolutions greater than or equal to the current resolution. The $Down(.,.)$ function is applied to downsample the images to the corresponding resolution, from which the starting and ending points are computed to obtain the target and loss for that stage. The loss of different stages is jointly optimized by dynamically adjusting their weights according to the training process. Additionally, during training, we also follow SD3 [9] and use logit-normal sampling for each stage. To enable CFG [15], we apply a 0.1 probability for random dropping of the prompt.

3.5. Inference

During inference, we sample a noise at the minimum resolution, start from stage $k = 1$ and proceed until $k = K$, where at each stage, we use the Flow-Euler-discretization sampler. For the jump points between stages, We use upsampling and BridgeFlow to perform the transformation. Our approach can reduce 64% time during inference.

4. Experiments

4.1. Model Details

In this paper, we use the MM-DiT block from FLUX [20] to build our NAMI due to its state-of-the-art performance in text-to-image generation. As shown in Tab. 1, our NAMI consists of 22 layers, each with 2048 channels, and uses 16 attention heads, totaling 2B parameters. Additionally, we perform ablation experiments by scaling the model down to 0.6B parameters. The layer distribution for different stages of the two models is visible in the table under layers ratio. For the text encoder, we use mT5 [46] and mCLIP [3] to enable our model with multilingual capabilities.

Table 1. Architecture details of the proposed NAMI.

Model	Width	Depth	Heads	Layers Ratio
NAMI-0.6B	1536	12	12	5:4:3
NAMI-2B	2048	22	16	9:7:6

4.2. Inference Time Details

As shown in Tab. 2, we provide the inference time of the NAMI-2B model at each resolution, using 10 inference steps per stage. The "Overall" time additionally includes the upsampling and BridgeFlow modules at the connection points between stages. Compared to the Flux-based 2B model with the same total of 30 sampling steps, our inference time is significantly reduced 64%. The design of the flow piece according to resolution reduces the computation time by 53%, while the model partition further decreases the time by an additional 11%. Refer to Fig. 9 for details.

Table 2. The inference time details of the NAMI architecture (measured in seconds).

Method	256	512	1024	Overall	Reduction
Baseline	-	-	8.47	8.47	-
NAMI-2B	0.27	0.45	2.21	2.98	**64.82%**

4.3. Experiments Details

We trained our NAMI-2B model on LAION [35], GRIT-20M [28], further fine-tuned it with 100K high-quality internal data, applying semantic and aesthetic filtering to all datasets. The total size of the training set is approximately 100 million. We set the generated image resolution to 1024 and the number of stages K to 3. The resolution for each stage is set to 256, 512, and 1024, respectively, with the corresponding time window division ratio being 1:1:1.

We first pretrain the BridgeFlow modules between stages according to the time window division, using a learning rate of 1e-3. Convergence is typically reached within 10k training steps. Subsequently, we trained the NAMI-2B model using the multi-resolution simultaneous training strategy described in Method. At the early stage of training, we set the learning rate to 1e-4 and sampled three resolutions

within each batch at a ratio of 4:2:1 (from low to high resolution) for 120k steps. Subsequently, we adjusted the sampling ratio to 2:4:1 and continued training for a further 80k steps. Finally, we modified the ratio to 1:2:4, reduced the learning rate to 5e-5, and trained for 50k steps to obtain the final NAMI-2B model.

4.3.1. Quantitative Comparison

We conducted a quantitative evaluation of NAMI across multiple open-source benchmarks, comparing our approach with other open-source methods. Table 3 demonstrates the text-image alignment capability for short prompts on GenEval [10]. It can be observed that our approach achieves a leading overall ranking when compared under comparable model parameter scales. Table 4 shows the complex semantic alignment and instruction-following capability for long texts on DPG-Benchmarks [18]. Our model still demonstrates exceptional performance in certain dimensions.

Table 3. Comparison of different methods on GenEval. With highlight the **best**, second best entries. Ovr & Sgl & Two & Cnt & Col & Pos & CA mean: Overall & Single & Two & Counting & Colors & Position & Color Attribution.

Models(Params)	Ovr	Sgl	Two	Cnt	Col	Pos	CA
FLUX-dev(12B)	0.67	**0.99**	0.81	**0.79**	0.74	0.20	0.47
FLUX-schnell(12B)	**0.71**	**0.99**	**0.92**	0.73	0.78	0.28	**0.54**
LUMINA-Next(2B)	0.46	0.92	0.46	0.48	0.70	0.09	0.13
SDXL(2.6B)	0.55	0.98	0.74	0.39	0.85	0.15	0.23
Hunyuan-DiT(1.5B)	0.63	0.97	0.77	0.71	**0.88**	0.13	0.30
SD3-medium(2B)	0.62	0.98	0.74	0.63	0.67	**0.34**	0.36
Sana(1.6B)	0.66	**0.99**	0.77	0.62	**0.88**	0.21	0.47
NAMI-2B(2B)	0.65	**0.99**	0.78	0.64	0.82	0.20	0.45

We found that GenEval [10] and DPG-Benchmark [18] primarily focus on text-image alignment, with limited prompt diversity.

Table 4. Comparison of different methods on DPG-Benchmark. With highlight the **best**, second best entries. Ovr & Gbl & Ent & Attr & Rel & Oth mean: Overall & Global & Entity & Attribute & Relation & Other

Models(Params)	Ovr	Gbl	Ent	Attr	Rel	Oth
FLUX-dev(12B)	84.0	82.1	89.5	88.7	91.1	89.4
FLUX-schnell(12B)	**84.8**	**91.2**	91.3	**89.7**	86.5	87.0
LUMINA-Next(2B)	74.6	82.8	88.7	86.4	80.5	81.8
SDXL(2.6B)	74.7	83.3	82.4	80.9	86.8	80.4
Hunyuan-DiT(1.5B)	78.9	84.6	80.6	88.0	74.4	86.4
SD3-medium(2B)	84.1	87.9	91.0	88.8	80.7	88.7
Sana(1.6B)	**84.8**	86.0	**91.5**	88.9	**91.9**	**90.7**
NAMI-2B(2B)	83.8	90.3	89.4	88.3	88.0	90.6

4.3.2. Human Evaluation

To address the limited prompt diversity of open-source benchmarks and to accurately evaluate the model's performance, We propose the NAMI-1K evaluation dataset from the perspective of human preference performance.

Table 5. Human evaluation results on NAMI-1K dataset. Rele & Cohe & Aes & Real mean: Relevance & Coherence & Aesthetic & Realism. With highlight the **best**, second best entries.

Models(Params)	Rele	Cohe	Aes	Real	Overall
Flux-dev(12B)	**83.93**	**83.28**	**84.26**	**90.16**	**85.05**
SD3-medium(2B)	75.74	65.90	61.64	75.74	69.97
Infinity(2B)	76.39	65.25	61.97	74.43	69.77
Hunyuan-DiT(1.5B)	74.43	69.51	63.93	64.92	68.95
SANA(1.6B)	75.41	62.30	60.00	72.46	67.80
NAMI-2B(2B)	76.07	66.89	62.30	76.72	70.69

Dataset Construction We developed a hierarchical evaluation dataset, NAMI-1K, comprising 1,000 prompts with diverse topic categories and varying length distributions. This dataset was constructed by integrating open benchmarks, AI-generated prompts, and human-authored prompts. Specifically, 360 short prompts were selected from the open benchmarks GenEval [10] and Lumiere-Set [2] to characterize the alignment capability of text to image. While 320 human-created prompts were collected from community contributions and user interactions to reflect the model's performance in real-world user scenarios. Additionally, 320 long prompts generated by Cogvlm2 [17], were used to assess performance in complex semantic alignment and instruction-following capabilities.

As shown in Fig. 5, the prompt lengths in GenEval are primarily concentrated within 10 words, while those in the DPG-benchmark are mainly distributed between 50 and 80 words. In contrast, NAMI-1K exhibits a diverse distribution across different lengths within 120 words. As described in Appendix A, compared to GenEval and the DPG-benchmark, the topic distribution of NAMI-1K is more comprehensive and balanced.

Evaluation Process The evaluation was conducted by five professional annotators through cross-assessment, considering four dimensions of relevance, coherence, aesthetic quality, and realism. The numerical range of each dimension is from 0 to 100, with a higher score indicating better performance. The final overall score was calculated through a weighted aggregation of the scores across the different dimensions, with the following weights assigned: relevance (0.3), coherence (0.3), aesthetic quality (0.2) and realism (0.2).

Evaluation Results The evaluation metrics are presented in Tab. 5. Although our model still trails behind

FLUX-dev [20], which have considerably larger parameter sizes, it shows a clear advantage over SD3-medium [9], Infinity [12], Hunyuan-DiT [21], and SANA [43], which have comparable parameter sizes. As illustrated in Appendix F, a comparison of the generation results is provided between different methods.

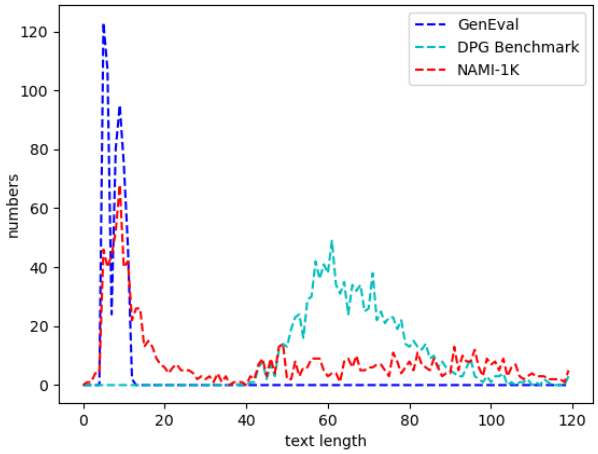


Figure 5. The distribution of text lengths across GenEval, DPG-Benchmark and NAMI-1K.

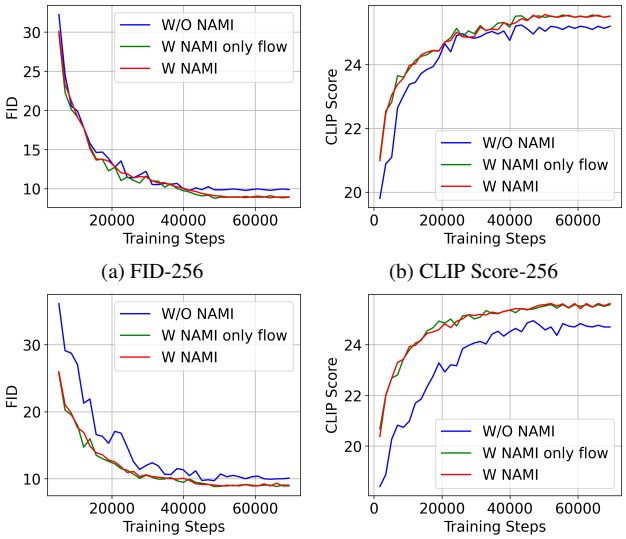
4.4. Ablation Study

We sampled 2 million training data for the ablation experiment, and 1,000 for the test data. We used the NAMI-0.6B model for the experiments.

Training Efficiency We compare the training efficiency of NAMI with multi-resolution training against FLUX-based training of the same size at a final resolution of 256. The experiments are conducted using 8 A100 GPUs (80 GB memory each). As shown in Tab. 6 NAMI demonstrates higher training efficiency compared to the FLUX-based baseline.

Components Effectiveness We conducted ablation experiments to separately verify the effectiveness of flow piecewise and model partitioning. We conducted two sets of experiments at 256 and 512 resolutions, with the following configurations: (1) the conventional flow model of FLUX-base, (2) the semi-NAMI structure with only flow piecewise (W NAMI only flow), and (3) the full NAMI structure incorporating both flow piecewise and model spatial decomposition.

As shown in Fig. 6, experiment (1) and (3) show that NAMI converges faster, and achieves better FID [14] and CLIP [30] scores at both resolutions. This advantage becomes more pronounced at the 512 resolution, further highlighting the effectiveness of the NAMI structure in accelerating high-resolution image training and improving generation quality. Experiment (1) and (2) demonstrate the effectiveness of piecewise flow and multi-resolution progressive generation. Experiment (2) and (3) compare the impact of model partitioning. We observe that although the full model has a slight convergence speed advantage in the early stages, the performance of both approaches is generally comparable. However, using partitioning provides a clear advantage in inference speed.



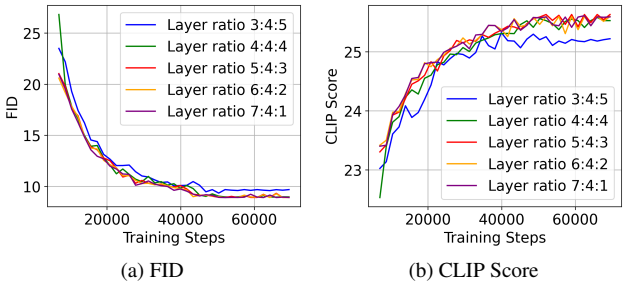
(a) FID-256 (b) CLIP Score-256

(c) FID-512 (d) CLIP Score-512

Figure 6. The effectiveness of the NAMI components at resolutions of 256 and 512.

Model Layers Partitioning We conducted experiments on the model's layers ratio at a resolution of 512 to evaluate the impact of layers partitioning on performance. We adjusted the layer distribution ratio across three stages, with the total number of layers fixed at 12.

As shown in Fig. 7, it can be seen that too few layers in the low-resolution stage lead to a decline in performance. However, as the number of layers increases to a certain point, the model's performance tends to saturate, and adding more layers results in redundancy in the model's capacity. In practical applications, this can be adjusted as a hyperparameter based on an equal distribution as a baseline.



(a) FID (b) CLIP Score

Figure 7. Ablation study on the distribution ratio of NAMI layers at different stages.

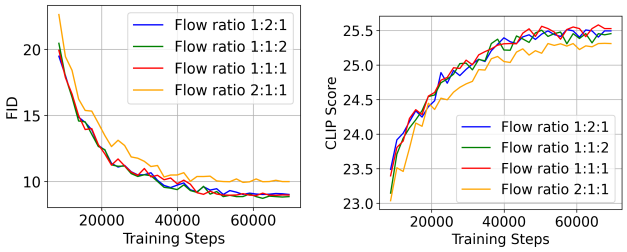
Table 6. Comparison of Training Efficiency between NAMI and FLUX-based Architectures.

Model	Batch Size	Throughput(img/s)	8xA100 GPU Hours	Peak Memory	FID	CLIP Score
NAMI	768	274	156	55G	8.9293	25.57
FLUX-based	768	241	176	67.6G	9.76	25.20

Table 7. Comparison of different implementations at jump point for 256 resolution. Our proposed BridgeFlow module achieves the best overall performance, while more complex modules do not lead to further improvements.

Module	Infer Time (s)	FID	CLIP Score
BridgeFlow	0.05	8.9293	25.57
Upsample + MLP	0.16	9.0842	25.71
Pixel Shuffle + CNN	0.23	8.8975	25.46
Pyramid Flow	0.12	9.8231	24.21

Time Window Division Ratio We adjusted the flow window distribution ratio across the three stages to observe the impact of different stage assignments. As shown in Fig. 8, it is evident that overly assigning the time windows to either the low or high stages does not result in significant improvements and may even lead to a decrease in accuracy due to insufficient length in other stages. Therefore, a uniform distribution of time windows is sufficient.



(a) FID (b) CLIP Score

Figure 8. Ablation study on the flow piecewise ratio of NAMI at different stages.

BridgeFlow module We conduct experiments at a resolution of 256 to compare different methods at the jump point, including: (1) Upsampling with Renoising as in pyramid flow matching [19], (2) Upsampling followed by a 3-layer MLP, and (3) Pixel Shuffle [37] followed by a 3-layer residual CNN [13]. As shown in Tab. 7, PyramidFlow exhibits suboptimal performance, with its renoising process causing further increases in inference time at higher resolutions. Overall, our BridgeFlow module achieves a better trade-off between quality and speed, while introducing more complex structures does not yield further gains.

The inference time of NAMI Components We performed 30 inference steps on an 80GB VRAM A100 GPU, excluding the time spent on the VAE and text encoders. We evaluated the inference time advantages brought by adjusting the flow and transformer components, respectively. As shown in Fig. 9, NAMI method at a resolution of 1024 shows a 53.01% reduction in inference time when only flow piecewise is applied. Additionally, with model partitioning, the inference time further decreased by 11.81%.

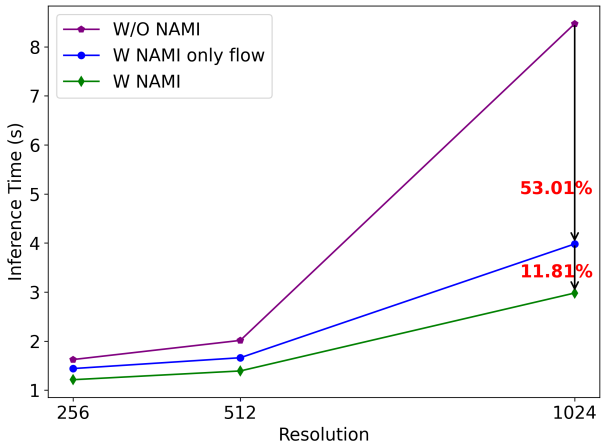


Figure 9. The inference time of NAMI Components.

5. Conclusion

In this paper, we introduce a bridge-connected multi-stage flow and spatially nested model architecture and employs a multi-resolution joint training strategy to accelerate model convergence while ensuring performance. By dividing the rectified flow into multiple stages based on resolution and progressively increasing the number of transformer layers as the resolution grows, our method efficiently balances computational cost and performance. The proposed NAMI architecture reduces inference time by 64% at 1024 resolution. Additionally, we have constructed the NAMI-1K benchmark to address the distribution biases and limited prompt diversity issues present in open-source benchmarks. NAMI is also convenient and promising for applications in other tasks. A simple training-free example of directly applying NAMI to image editing is provided in Appendix D. In future work, we plan to further explore NAMI's potential in other tasks and its integration with other efficient methods.

References

[1] Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *arXiv preprint arXiv:2209.15571*, 2022. 2, 3

[2] Omer Bar-Tal, Hila Chefer, Omer Tov, Charles Herrmann, Roni Paiss, Shiran Zada, Ariel Ephrat, Junhwa Hur, Guanghui Liu, Amit Raj, et al. Lumiere: A space-time diffusion model for video generation. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–11, 2024. 6

[3] Guanhua Chen, Lu Hou, Yun Chen, Wenliang Dai, Lifeng Shang, Xin Jiang, Qun Liu, Jia Pan, and Wenping Wang. mclip: Multilingual clip via cross-lingual transfer. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13028–13043, 2023. 5

[4] Junsong Chen, Jincheng Yu, Chongjian Ge, Lewei Yao, Enze Xie, Yue Wu, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, et al. Pixart-α: Fast training of diffusion transformer for photorealistic text-to-image synthesis. *arXiv preprint arXiv:2310.00426*, 2023. 3

[5] Junyu Chen, Han Cai, Junsong Chen, Enze Xie, Shang Yang, Haotian Tang, Muyang Li, Yao Lu, and Song Han. Deep compression autoencoder for efficient high-resolution diffusion models. *arXiv preprint arXiv:2410.10733*, 2024. 3

[6] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International conference on machine learning*, pages 1691–1703. PMLR, 2020. 2

[7] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. *Advances in neural information processing systems*, 28, 2015. 3

[8] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 2, 3

[9] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024. 2, 3, 4, 5, 7

[10] Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36:52132–52152, 2023. 3, 6

[11] Jiatao Gu, Shuangfei Zhai, Yizhe Zhang, Joshua M Susskind, and Navdeep Jaitly. Matryoshka diffusion models. In *The Twelfth International Conference on Learning Representations*, 2023. 2, 3

[12] Jian Han, Jinlai Liu, Yi Jiang, Bin Yan, Yuqi Zhang, Zehuan Yuan, Bingyue Peng, and Xiaobing Liu. Infinity: Scaling bitwise autoregressive modeling for high-resolution image synthesis. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 15733–15744, 2025. 2, 7

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 8

[14] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 7

[15] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 5

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2, 3

[17] Wenyi Hong, Weihan Wang, Ming Ding, Wenmeng Yu, Qingsong Lv, Yan Wang, Yean Cheng, Shiyu Huang, Junhui Ji, Zhao Xue, et al. Cogvlm2: Visual language models for image and video understanding. *arXiv preprint arXiv:2408.16500*, 2024. 6

[18] Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. Ella: Equip diffusion models with llm for enhanced semantic alignment. *arXiv preprint arXiv:2403.05135*, 2024. 3, 6

[19] Yang Jin, Zhicheng Sun, Ningyuan Li, Kun Xu, Hao Jiang, Nan Zhuang, Quzhe Huang, Yang Song, Yadong Mu, and Zhouchen Lin. Pyramidal flow matching for efficient video generative modeling. *arXiv preprint arXiv:2410.05954*, 2024. 2, 3, 4, 8

[20] Black Forest Labs. Flux. https://github.com/black-forest-labs/flux, 2024. 2, 3, 5, 7

[21] Zhimin Li, Jianwei Zhang, Qin Lin, Jiangfeng Xiong, Yanxin Long, Xinchi Deng, Yingfang Zhang, Xingchao Liu, Minbin Huang, Zedong Xiao, et al. Hunyuan-dit: A powerful multi-resolution diffusion transformer with fine-grained chinese understanding. *arXiv preprint arXiv:2405.08748*, 2024. 3, 7

[22] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022. 2, 3

[23] Songhua Liu, Zhenxiong Tan, and Xinchao Wang. Clear: Conv-like linearization revs pre-trained diffusion transformers up. *arXiv preprint arXiv:2412.16112*, 2024. 3

[24] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022. 2, 3

[25] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021. 3

[26] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *International conference on machine learning*, pages 4055–4064. PMLR, 2018. 2

[27] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023. 2, 3

[28] Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, and Furu Wei. Kosmos-2: Grounding multimodal large language models to the world. *arXiv preprint arXiv:2306.14824*, 2023. 5

[29] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023. 3

[30] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021. 7

[31] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International conference on machine learning*, pages 8821–8831. Pmlr, 2021. 2

[32] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 2, 3

[33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 2, 3

[34] Dohoon Ryu and Jong Chul Ye. Pyramidal denoising diffusion probabilistic models. *arXiv preprint arXiv:2208.01864*, 2022. 3

[35] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in neural information processing systems*, 35:25278–25294, 2022. 5

[36] Vikash Sehwag, Xianghao Kong, Jingtao Li, Michael Spranger, and Lingjuan Lyu. Stretching each dollar: Diffusion training from scratch on a micro-budget. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 28596–28608, 2025. 3

[37] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016. 8

[38] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. pmlr, 2015. 2, 3

[39] Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024. 2

[40] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 2

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 3

[42] Jing Wang, Ao Ma, Jiasong Feng, Dawei Leng, Yuhui Yin, and Xiaodan Liang. Qihoo-t2x: An efficient proxy-tokenized diffusion transformer for text-to-any-task. *arXiv preprint arXiv:2409.04005*, 2024. 2

[43] Enze Xie, Junsong Chen, Junyu Chen, Han Cai, Haotian Tang, Yujun Lin, Zhekai Zhang, Muyang Li, Ligeng Zhu, Yao Lu, et al. Sana: Efficient high-resolution image synthesis with linear diffusion transformers. *arXiv preprint arXiv:2410.10629*, 2024. 3, 7

[44] Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imagereward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36:15903–15935, 2023. 3

[45] Yilun Xu, Ziming Liu, Max Tegmark, and Tommi Jaakkola. Poisson flow generative models. *Advances in Neural Information Processing Systems*, 35:16782–16795, 2022. 3

[46] Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mt5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 483–498, 2021. 5

[47] Hanshu Yan, Xingchao Liu, Jiachun Pan, Jun Hao Liew, Qiang Liu, and Jiashi Feng. Perflow: Piecewise rectified flow as universal plug-and-play accelerator. *Advances in Neural Information Processing Systems*, 37:78630–78652, 2024. 4

[48] Le Zhuo, Ruoyi Du, Han Xiao, Yangguang Li, Dongyang Liu, Rongjie Huang, Wenze Liu, Xiangyang Zhu, Fu-Yun Wang, Zhanyu Ma, et al. Lumina-next: Making lumina-t2x stronger and faster with next-dit. *Advances in Neural Information Processing Systems*, 37:131278–131315, 2024. 3