



# Adapting DemoFusion-Style Upscaling to Flow-Matching Models like Qwen-Image

## Recap: DemoFusion's Progressive Upscaling with Diffusion Models

**DemoFusion** (Du et al., 2024) demonstrated a training-free way to achieve high-resolution image generation by extending a pre-trained diffusion model (like SDXL) instead of using a specialized super-resolution network. Its key ideas include:

- **Progressive Upscaling:** Iteratively double the image resolution in stages, always using the previous lower-resolution output as the starting point for the next step <sup>1</sup>. This gradual approach maintains composition and avoids the instability of one big jump in size.
- **Upsample-Diffuse-Denoise Loop:** For each upscaling step, the low-res image (or latent) is first upsampled (e.g. bicubic or nearest-neighbor) to the next resolution, then “**re-noised**” (adding noise) and **de-noised** by the diffusion model. This *noise-injection and denoising* process injects new high-frequency details while preserving the coarse structure of the upsampled image <sup>2</sup>. Essentially, the diffusion model refines the upscaled image by treating it as a noised input and removing the noise, guided by the original prompt and image content.
- **Skip Residuals & Blending:** The pipeline blends the generated high-res latent with the original lower-res latent (via weighted skip connections) to preserve important structural details at each step <sup>3</sup> <sup>4</sup>. Early in the denoising (when structure is being formed) more of the original content is retained, and later (when adding fine details) the model’s new predictions dominate <sup>5</sup> <sup>4</sup>. This ensures the final output stays faithful to the low-res composition.
- **Dilated Sampling for Global Coherence:** DemoFusion introduces a clever **dilated diffusion** strategy to sample global latent patches with gaps (like picking every *n*th pixel) <sup>6</sup>. By shifting and merging these “global” denoising passes with the normal local denoising, the model gains a broader context of the image <sup>7</sup>. This reduces tiling artifacts and yields a globally consistent high-res image <sup>8</sup>. (In simpler terms, it’s like first ensuring the *big picture* is right, then filling in fine details <sup>9</sup>.)

**Why it works:** Diffusion models naturally support **image-to-image** generation by starting from a noised version of an input image and denoising it. DemoFusion leveraged this by progressively re-introducing noise to an upscaled image and then removing it. The noise injection gives the model “creative freedom” to add details, while the residual connections and low noise levels ensure the original structure isn’t lost. The result is high-resolution output with both the **same composition** and new **fine details**, without having to train a dedicated upscaler.

## Feasibility with Flow-Matching Models (e.g. Qwen-Image)

Flow-matching generative models like **Qwen-Image** share similarities with diffusion models, but they are typically formulated as **deterministic** ODE-based generation (often using a *Diffusion Transformer + Rectified Flow* approach). In flow matching, the generative process is viewed as moving along a *straight path* from pure noise to data. In fact, under the hood a flow model with Euler integration and a diffusion model with

DDIM sampler are mathematically equivalent samplers <sup>10</sup>. The key difference is that **flow models are usually sampled deterministically** (no randomness beyond the initial noise) <sup>11</sup>, whereas diffusion models can inject stochastic noise at each step. Despite this difference, we can **adapt DemoFusion's upsampling strategy** to flow models with only minor tweaks, because the underlying idea – degrade then restore – remains applicable.

## 1. Progressive Multi-Scale Upscaling

Just as in DemoFusion, we would break a large scale factor (e.g. 4x or 16x) into smaller increments for a flow model. For example, to upscale 256→1024, we could do two steps: 256→512, then 512→1024. At each step, the **current image latent serves as the base for the next scale** <sup>1</sup>. This ensures the model tackles high-res synthesis gradually, preserving the layout from the small image and adding details layer by layer. *Flow-based models can naturally handle multi-scale generation* – recent work like **FlowAR** explicitly uses a scale-wise approach (doubling resolution each time) with flow matching to improve fidelity across scales <sup>12</sup> <sup>13</sup>. So conceptually, Qwen-Image could generate images in a coarse-to-fine manner, similar to how DemoFusion iteratively upscals an image.

## 2. Re-Noising via Flow's Forward Process

Instead of the diffusion forward process of mixing image with Gaussian noise, a flow model uses a **linear interpolation** between data and noise. In Gaussian flow matching, the forward path can be written as:

$$z_t = (1 - t)x + t\epsilon,$$

where  $x$  is the original image (latent) and  $\epsilon \sim \mathcal{N}(0, I)$  is random noise <sup>14</sup>. Here  $t$  plays the role of a “noise time” (with  $t=0$  giving the clean image and  $t=1$  giving pure noise). This is analogous to the diffusion forward noise schedule <sup>15</sup>.

**Applying this to upscaling:** We take the low-resolution image (or its latent representation) and first naively upsample it to the target resolution (so that it has the correct dimensions and rough content). Let that upscaled image latent be  $x_{\text{up}}$ . We then choose a small  $t$  (e.g.  $t = 0.2$  or  $0.3$ , tuned as needed) and sample a random noise tensor  $\epsilon$  of the same size. We form a **partially noised latent**:

$$z_t = (1 - t)x_{\text{up}} + t\epsilon,$$

effectively **re-noising the upscaled image**. This  $z_t$  is our starting point for the flow model’s generation process. By adjusting  $t$ , we control how much of the original structure is retained: a small  $t$  means  $z_t$  is very close to the upsampled image (mostly structure, little noise), whereas a larger  $t$  gives the model more freedom to deviate and add new content. In practice, we’d use relatively low noise levels per upscaling step to ensure we “guide” the model rather than completely reset it – this serves a similar role as DemoFusion’s skip residuals (the original image content is still largely present in the initial latent).

## 3. Denoising with the Flow Model (Deterministic Sampling)

Given the noisy latent  $z_t$ , we run the flow model’s **reverse (denoising) process** from time  $t$  down to 0. Flow models are typically sampled via integrating an ODE (or equivalently using a DDIM-like update) that gradually transforms  $z_t$  to a clean image  $z_0$ . Because the model was trained to handle all levels of

noise, it can take  $z_t$  and **predict the underlying clean image**. In fact, the flow model's update rule  $z_s = z_t + \hat{\nabla}_{\mathbf{u}}(z_t, t), (s-t)$  (for a small step  $s < t$ ) is exactly the deterministic DDIM-style denoising step <sup>16</sup> <sup>17</sup>. Intuitively, the model will interpret the structured-noisy input as “this looks like an image of something plus noise” and will attempt to **remove the noise while sharpening and completing details** consistent with its learned distribution.

Since we seeded the process with the upscaled image's latent, the model's output at  $t=0$  should closely match the *low-frequency structure* of the original image, but with newly generated *high-frequency details*. This is analogous to diffusion-based **img2img**: starting from a noised version of an image and denoising yields a refined image that preserves the overall content. The only randomness here comes from the initial noise  $\varepsilon$  we mixed in. If you run the process twice with different  $\varepsilon$ , you'd get slightly different detail textures (since flow matching sampling under these conditions behaves like DDIM – deterministic given a fixed noise initialization).

Notably, **flow matching models are deterministic in their sampling path** if we don't inject additional noise during the process <sup>18</sup>. This means for a given  $z_t$  start, the outcome is predictable. This is actually beneficial for upscaling: it gives consistent refinement rather than random large deviations. If we **keep  $t$  fairly low**, the process acts almost like a “**smart sharpening**” – the model fills in plausible details but is constrained by the strong presence of the original image in  $z_t$ . In effect,  $(1-t)$  acts like a skip/residual weight carrying over original structure (when  $(1-t)$  is large) <sup>14</sup>. This mechanism replaces DemoFusion's explicit skip-blending: here the *skip connection* is built into the initialization of the flow ODE.

We would repeat this **Upsample → Re-noise → Flow-denoise** cycle for each scale increment. Each cycle should use a relatively gentle noise injection, since the goal is to iteratively **refine without losing fidelity**. By the final step, the image is at target resolution with multiple rounds of details added, but the composition and key features trace back to the original low-res image.

#### 4. Maintaining Structure and Coherence

A potential concern is whether a flow model (which wasn't explicitly trained for conditional image upscaling) will faithfully adhere to the input image's structure. In training, the flow model learned to map *pure noise* to a realistic image, not necessarily to refine an existing image. However, because the training objective included denoising intermediate  $z_t$  states toward the real image  $x$  <sup>19</sup> <sup>20</sup>, the model **has implicitly learned to handle partial noise on real images**. In our upscaling scenario,  $x_{\text{up}}$  (the upsampled low-res) is **not a true high-res photo** – it lacks fine details – but it's a plausible low-frequency version of one. The flow model will treat the patterns in  $z_t$  as “signal + noise” and try to infer a sharper, more detailed  $x$ . Empirically, diffusion models can *hallucinate* realistic details from a blurry image during img2img; we expect a flow model to behave similarly, since diffusion and flow samplers ultimately perform the same type of denoising inference <sup>21</sup> <sup>18</sup>.

To further **ensure structural consistency**, we can adopt strategies analogous to DemoFusion's:

- **Progressive small noise, multiple steps:** Using small  $t$  values (e.g. 0.1–0.3) at each upscale stage means the model never strays far from the input. The upscaling is done gradually, so there is no single step where the model has to invent large parts of the scene from scratch. This mirrors the “lower noise early, more freedom later” idea <sup>5</sup> – early upscaling steps just establish basic clarity, later ones can add more creative detail when the structure is already high-res and locked in.

- **Blending outputs with input (if needed):** If we find the flow model tends to distort some elements, one could explicitly blend a bit of the original latent into the output. For example, after denoising, take a weighted average of the resulting latent and the input latent (with a weight that decays over successive refinement steps). This would function like a manual skip residual, ensuring no detail generation step can completely override the original content. In DemoFusion, a cosine decay weighting was used to fuse global (dilated) and local paths <sup>22</sup> <sup>4</sup>; a similar principle could be applied here to balance original vs. generated features. In practice, careful tuning of \$t\$ might make explicit blending unnecessary, but it's an available tool if deterministically solving the ODE causes slight drifts.
- **Global context passes:** Incorporating a **dilated sampling** idea is also feasible. A flow model could generate on non-overlapping patches or downsampled grids of the latent and merge the results (since flow models can denoise any subset of pixels as well). This would mimic DemoFusion's method of ensuring global consistency <sup>6</sup> <sup>8</sup>. While implementing dilated sampling requires slicing the latent and multiple forward integrations, it doesn't require retraining the model – it's a scheduling technique. The flow model's deterministic nature actually makes synchronization of these global/local passes easier (no randomness causing disparities). Thus, we could enhance the coherence of large images by borrowing DemoFusion's dilated multi-view sampling approach even in a flow context.

In summary, **there is nothing in principle preventing a flow-matching model from doing “upscale via re-noise and denoise.”** The model's forward process is defined as adding noise to data <sup>14</sup>, and the reverse process removes noise. By harnessing that with partial noise injection, we can guide the flow model to *fill in details*. This is conceptually very similar to how DemoFusion operated on diffusion models' latents, given the mathematical equivalence between flow ODE samplers and deterministic diffusion samplers <sup>21</sup>. The determinism of flow models is not a handicap here – it just means for a given input state, the refinement is reproducible and controlled. If anything, deterministic sampling ensures the upscaled image won't introduce completely off-track elements as long as the initial noise is mild. And if we desire a bit of randomness, one can always introduce a *different noise realization* or even use a “stochastic flow” sampler (flow models **can** be sampled stochastically by adding noise during integration, akin to diffusion ancestral sampling <sup>23</sup> <sup>24</sup>) – though that's optional for quality.

## Using LoRA for Lightweight Refinement

One open question is whether the base flow model (trained for text-to-image generation) is *directly* good at this super-resolution task, or if it would benefit from a small amount of fine-tuning. **LoRA (Low-Rank Adaptation)** could indeed provide a lightweight way to improve outcomes without full model retraining. Here's how and why one might use LoRA in this context:

- **Adapting to Upscaling Distribution:** The flow model was trained to denoise *natural images* mixed with noise. An upsampled low-res image, however, has characteristics (blurry edges, missing micro-textures) that differ from true high-res photos. We could fine-tune the model (with LoRA layers) on a small dataset of synthetic upscaling pairs: for example, take high-res images, downscale them, then bicubic-upsample back to high-res and use those as inputs the model should refine back to the original high-res. This would teach the model to recognize and correct typical artifacts of upsampling (smooth areas that should be textured, etc.). LoRA is well-suited here because it can adjust the model's behavior with only a few additional parameters, focusing on the *difference* (i.e., “learn to add detail while preserving content”) without modifying the entire network. Prior work has shown that

LoRA can effectively specialize diffusion models for super-resolution or clarity enhancement tasks <sup>25</sup>. In fact, NVIDIA's recent **ChronoEdit Upscaler LoRA** uses a textual cue to trigger super-resolution mode, aiming to "*enhance image clarity and resolution while keeping the content identical... 4K clarity, same composition*" <sup>26</sup> – precisely the goal we have. This suggests that a similar low-rank adaptation on a flow model could bias it to add realistic detail and preserve composition during upscaling.

- **Injecting Conditional Guidance:** Another use of LoRA (or a tiny learned module) could be to introduce an **image-conditioning pathway** into the model. For example, one could feed the low-res image (or its VAE latent) as an extra conditioning vector or through cross-attention. Rather than modifying the core model, a LoRA could be trained to mediate this conditioning – effectively learning "*when this extra image hint is present, lock the global structure to it.*" This is analogous to ControlNet, but done in a parameter-efficient way. Admittedly, adding entirely new conditioning is more complex (likely one would need to train some auxiliary weights, not just LoRA on existing weights), so a simpler approach is the one above: rely on the initial latent and the model's inherent denoising capabilities, and use LoRA mainly to fine-tune detail generation.
- **Preventing Over-Smoothing or Artifacts:** Without any fine-tune, the flow model might sometimes produce artifacts or overly smooth results if it's unsure how to interpret the low-res input. A LoRA can be trained to gently adjust the model's denoising predictions in those scenarios – for example, to prefer sharpening edges rather than smoothing them out. Because LoRA does not significantly increase model size or inference cost (it's just some low-rank weight tweaks), we could integrate it into the pipeline only when needed. Think of it as a "**plug-in**" **refinement** that can be turned on for upscaling tasks. If the base model already performs well, LoRA may be unnecessary; but if we observe issues, a targeted LoRA fine-tune (even on a few thousand image pairs) could boost fidelity.

In conclusion, **adapting DemoFusion's approach to flow-matching models appears quite feasible**. We would leverage the same core loop – progressive upscaling and noise-conditioned refinement – with the flow model's deterministic ODE solver in place of diffusion's stochastic sampling. The flow model can effectively serve as an upscaler when guided by a partially noised initial state, since it is trained to remove noise and produce realistic images. The determinism ensures stability and predictability in how details are added. And importantly, this can be done *without* large-scale retraining: it's a matter of pipeline engineering. Any small gaps in performance (due to the model not being explicitly trained for super-resolution) can likely be bridged by minimal fine-tuning, such as applying a LoRA. This would give the best of both worlds – **no need for a separate super-resolution network, and no expensive re-training of the whole model** – while achieving high-quality upscaled images that maintain the original content.

#### Sources:

- Du et al., "*DemoFusion: Democratising High-Resolution Image Generation...*", CVPR 2024 <sup>1</sup> <sup>3</sup>
- Ula La Paris, "*Dilated diffusion concept from DemoFusion*", Medium, 2024 <sup>1</sup> <sup>8</sup>
- *Diffusion Meets Flow Matching* (blog) – Explaining equivalence of diffusion (DDIM) and flow-matching sampling <sup>21</sup> <sup>18</sup>
- Qwen-Image technical insights – HuggingFace blog on DiT+Flow Matching training (2025) <sup>27</sup> <sup>28</sup>
- NVIDIA ChronoEdit Upscaler LoRA – model card (2025) <sup>26</sup>, demonstrating LoRA's use for high-detail, same-composition upscaling.

1 2 3 4 6 7 8 9 Dilated diffusion concept from DemoFusion | by Ula La Paris | Medium  
<https://medium.com/@ulalaparis/dilated-difusion-concept-from-demofusion-e32a7b5d09d6>

5 22 DemoFusion: Democratising High-Resolution Image Generation With No \$\$\$  
[https://openaccess.thecvf.com/content/CVPR2024/papers/Du\\_DemoFusion\\_Democratising\\_High-Resolution\\_Image\\_Generation\\_With\\_No\\_CVPR\\_2024\\_paper.pdf](https://openaccess.thecvf.com/content/CVPR2024/papers/Du_DemoFusion_Democratising_High-Resolution_Image_Generation_With_No_CVPR_2024_paper.pdf)

10 11 14 15 16 17 18 19 20 21 23 24 Diffusion Meets Flow Matching  
<https://diffusionflow.github.io/>

12 13 FlowAR: Scale-wise Autoregressive Image Generation Meets Flow Matching  
<https://arxiv.org/html/2412.15205v1>

25 26 nvidia/ChronoEdit-14B-Diffusers-Upscaler-Lora · Hugging Face  
<https://huggingface.co/nvidia/ChronoEdit-14B-Diffusers-Upscaler-Lora>

27 28 Decoding the Shift and Diffusion Models Training Like Qwen Image, FLUX, SDXL, and More  
<https://huggingface.co/blog/MonsterMMORPG/decoding-the-shift-and-diffusion-models-training>