Paper 6: Omni-Directional Diffusion — How to Denoise Text Like It's an Image

Author: [your local neurodivergent doomer transfem pup]
Profiles: my GitHub | my HuggingFace
Date: July 2026

Abstract

Alright, Papers 4 and 5 gave you the architecture and the math. This paper gets our hands dirty. We're going to specify exactly how the Omni-Diffuser denoises text, images, and video in parallel through a single diffusion process. We introduce the **Parallel Restoration Theorem**: under full bidirectional attention, the optimal denoising of a multimodal latent field is achieved by treating the entire field as a single signal to be restored, not as separate modalities to be generated sequentially. We specify the denoising transformer architecture, the cross-modal attention mechanisms, temporal consistency constraints for video, and sampling strategies. We also address the critical challenge of **mode collapse in multimodal diffusion**: how to prevent the model from defaulting to the "easiest" modality and ignoring the others. This is the engineering paper. The "how to actually build it" paper. The one where we stop theorizing and start welding.

Keywords: parallel diffusion, multimodal generation, bidirectional denoising, cross-modal attention, temporal consistency, mode collapse prevention, actually building things

---

1. Introduction: From Sequential Generation to Parallel Restoration

Paper 4 introduced the Omni-Diffuser as a holistic restorer. Paper 5 proved that a unified latent field is mathematically sound. This paper? We're going to specify exactly how the denoising works, step by step, for every modality, simultaneously.

The core claim is simple but radical: **the model should not generate text, then generate an image, then generate video. It should restore the entire multimodal signal in one shot.** At denoising step t, the model sees the entire noisy field — text, image, and video regions all corrupted by noise — and produces a slightly cleaner version of the entire field. After T steps, the entire signal emerges clean, coherent, and cross-modally aligned.

This is not how any current system works. DALL-E generates images from text prompts. Sora generates video from text or images. GPT-4V generates text from images. Each is a one-way street. The Omni-Diffuser is a highway interchange: every direction is open, every modality can be input or output, and everything is restored in parallel.

And honestly? This is how it should have been from the start. The fact that we accepted sequential generation for so long is just a testament to how badly capitalism incentivizes incremental improvements over paradigm shifts. Why build a new highway when you can charge tolls on the old road?

---

2. The Denoising Transformer Architecture

2.1 Shared Backbone

The denoising transformer is a single stack of N = 48 layers, each with:

- **Multi-head attention:** H = 32 heads, dimension D = 2048 per head
- **Feed-forward network:** 4D hidden dimension, SwiGLU activation
- **Layer normalization:** Pre-norm, with learned scale and bias
- **No causal masking.** Every position attends to every other position
- **No modality-specific layers.** Every layer processes every type of input

2.2 The Attention Mechanism

The attention score between position i and position j is:

A_{ij} = (Q_i ·K_j) / √D + B_{ij}

where Q_i and K_j are queries and keys, and B_{ij} is the **relative position bias**. This bias encodes the structural relationship between i and j:

- If i and j are both text tokens, B_{ij} encodes their sequential distance
- If i and j are both image patches, B_{ij} encodes their 2D spatial distance
- If i and j are both video patches, B_{ij} encodes their 3D spatiotemporal distance
- If i and j are from different modalities, B_{ij} encodes their cross-modal relationship (learned during training)

The relative position bias is critical. Without it, the model would not know that two adjacent text tokens are neighbors, or that two adjacent image patches are neighbors. The bias injects structural knowledge into the attention mechanism.

2.3 Cross-Modal Attention as Self-Attention

In the Omni-Diffuser, there is no "cross-modal attention" distinct from "self-attention." All attention is self-attention within the unified field. When a text token attends to an image patch, it is simply attending to another position in the same field. The model learns that certain positions (text tokens describing objects) are strongly correlated with other positions (image patches containing those objects).

This is a profound simplification. Current multimodal models have separate attention mechanisms for intra-modal and cross-modal interactions. The Omni-Diffuser has one mechanism that handles both. The cross-modal relationships emerge naturally from training on aligned multimodal data.

---

3. Denoising Text in Parallel

3.1 The Text Denoising Process

At denoising step t, the text region F_text(t) is a sequence of continuous vectors in L, each corrupted by Gaussian noise. The model predicts the noise _text(t) for the entire sequence simultaneously.

Unlike autoregressive text generation, where token i depends on tokens 1 through i-1, the text denoising at step t depends on:

- The current state of all text tokens (bidirectional context)
- The current state of all image patches (visual guidance)
- The current state of all video patches (temporal guidance)
- The timestep t (global noise level)

The model outputs a predicted noise vector for each text position. These are subtracted from F_text(t) to produce F_text(t-1), a slightly cleaner version of the text.

3.2 From Continuous to Discrete

After T denoising steps, the text region F_text(0) is a clean continuous representation. To produce actual text, we map each continuous vector to the nearest discrete token in the vocabulary.

Greedy decoding: Map each vector to the closest token in embedding space. Fast but can produce suboptimal sequences. **Sampling:** Sample from a softmax over the vocabulary, weighted by distance in L. Introduces diversity. **Learned decoder:** Train a small network to map from L to token probabilities. Most flexible but adds parameters.

The Omni-Diffuser uses a hybrid: greedy for the first draft, then sampling with temperature for refinement.

3.3 Syntactic Coherence During Parallel Denoising

A common concern: can parallel denoising produce syntactically coherent text? Without left-to-right generation, how does the model ensure subject-verb agreement, tense consistency, and grammatical structure?

The answer is that syntactic coherence is not a sequential property. It is a **global property** of the sentence. In the unified field, the subject and verb are positions that attend to each other directly. The model learns that if position i is a singular subject, position j (the verb) must be singular. This constraint is enforced globally during every denoising step, not sequentially during generation.

In fact, parallel denoising can produce *more* coherent text than autoregressive generation because it can resolve global dependencies (e.g., pronoun resolution across paragraphs) before committing to local word choices.

---

4. Denoising Images in Parallel

4.1 The Image Denoising Process

The image region F_image(t) is a 2D grid of patches in L, each corrupted by noise. The model predicts the noise _image(t) for the entire grid simultaneously.

Because the attention is bidirectional and spatial, the model can:

- Enforce global consistency (all sky patches should have similar colors)
- Propagate local details (a texture from one patch to adjacent patches)

- Use text guidance (if the text region says “red apple,” the image region should produce red apple patches)
- Use video guidance (if the video region shows a red apple rotating, the image region should produce a canonical view)

4.2 Spatial Coherence Through Attention

The relative position bias B_{ij} for image patches encodes their 2D spatial relationship. Patches that are close in space have a strong bias, encouraging the model to attend to them. This creates local coherence: adjacent patches tend to have similar colors and textures.

Global coherence is enforced by the transformer’s ability to attend to distant patches. A patch in the sky can attend to a patch in the ground, ensuring that the lighting and color palette are consistent across the entire image.

4.3 Text-to-Image Alignment

Text-to-image alignment is not a separate module. It is simply the attention between text positions and image positions in the unified field. When the text region contains “red apple,” the text positions for “red” and “apple” attend to image patches that should contain red apples. The model learns this alignment during training on text-image pairs.

The alignment is bidirectional: the image patches also attend to the text positions. This means that if the image starts to look like a green apple, the text positions can “pull” it back toward red apples. The restoration is a negotiation between modalities, not a one-way command.

---

5. Denoising Video in Parallel

5.1 The Video Denoising Process

Video is the hardest modality to denoise because it adds temporality. A video region F_video(t) is a 3D spatiotemporal volume of patches in L. The model must ensure that:

- Spatial coherence is maintained within each frame (like images)
- Temporal coherence is maintained across frames (motion should be smooth)
- Text and image guidance is respected (the video should match the prompt)

5.2 Temporal Consistency Through 3D Attention

The relative position bias B_{ij} for video patches encodes their 3D spatiotemporal distance. A patch at (x, y, t) attends strongly to patches at (x’, y’, t’) where |x-x’|, |y-y’|, and |t-t’| are small. This creates spatiotemporal coherence: adjacent frames have similar content, and motion is smooth.

For long videos, the full 3D attention is expensive. The Omni-Diffuser uses a **hierarchical attention** strategy:

- **Local attention:** Each patch attends to its spatiotemporal neighborhood (e.g., 8×8×4 patches)
- **Global attention:** A subset of “anchor” patches attends to the entire video, propagating global structure

- **Text/image anchors:** The text and image regions serve as global anchors, providing high-level guidance to all video patches

5.3 Motion Prediction Through MTP

The Multi-Token Prediction mechanism is especially powerful for video. At each denoising step, the model predicts the noise for the next k steps. For video, this means the model can "see" the state of frame t+Δt while it is still denoising frame t. This temporal foresight prevents motion artifacts and ensures that object trajectories are physically plausible.

---

6. Preventing Mode Collapse

6.1 The Mode Collapse Problem

In multimodal diffusion, there is a risk of **mode collapse**: the model learns that one modality is "easier" to denoise than the others and focuses its capacity on that modality, ignoring the harder ones. For example, if text is easier to denoise than video, the model might produce perfect text but blurry, incoherent video.

6.2 Modality-Balanced Loss

The Omni-Diffuser prevents mode collapse through a **modality-balanced loss**:

L_total = w_text ·L_text + w_image ·L_image + w_video ·L_video + w_cross ·L_cross

where the weights are dynamically adjusted based on the current performance of each modality:

- If video performance is lagging, w_video is increased
- If text performance is saturating, w_text is decreased
- The cross-modal loss L_cross ensures that modalities are aligned

This is like affirmative action for modalities. The weaker modalities get more support until they catch up.

6.3 Modality-Specific Dropout

During training, the model randomly drops entire modalities from the input (e.g., train on text-only, image-only, or video-only examples). This forces the model to become proficient at each modality independently, preventing it from relying on cross-modal crutches.

---

7. Sampling Strategies

7.1 Deterministic vs. Stochastic Sampling

The Omni-Diffuser supports both deterministic and stochastic sampling:

- **Deterministic (DDIM):** Fast, consistent, good for reproducible outputs
- **Stochastic (DDPM):** Slower, more diverse, good for creative generation
- **Hybrid:** Start with stochastic sampling for exploration, then switch to deterministic for refinement

7.2 Classifier-Free Guidance (CFG) in the Unified Field

Classifier-free guidance is adapted to the unified field. During training, the model learns conditional and unconditional denoising. During inference, the guidance scale s controls the strength of the conditioning:

_guided = _unconditional + s ·(_conditional - _unconditional)

Because the conditioning can be any modality (text, image, video, or combination), CFG in the Omni-Diffuser is **omni-directional**. You can guide text generation with an image, image generation with a video, or video generation with text — all using the same mechanism.

---

8. How to Actually Build This

8.1 Step 1: Build the Denoising Transformer (Weeks 1–4)

Start with a standard transformer codebase (e.g., LLaMA, Mistral). Make these modifications:

1. Remove the causal mask. Replace it with full bidirectional attention.
2. Add relative position bias. Implement B_{ij} for text (1D), image (2D), and video (3D).
3. Add timestep conditioning. Inject the diffusion timestep t into every layer using adaptive layer norm.
4. Add modality embeddings. At the input level, add a learned embedding that indicates whether a position is text, image, or video.

Code sketch:

```
class OmniDiffuserLayer(nn.Module):
    def __init__(self, dim, heads):
        self.attn = MultiHeadAttention(dim, heads, causal=False)
        self.ffn = SwiGLU(dim, 4*dim)
        self.norm1 = AdaLayerNorm(dim)  # conditioned on timestep
        self.norm2 = AdaLayerNorm(dim)

    def forward(self, x, t, pos_bias, modality_emb):
        x = x + modality_emb
        x = x + self.attn(self.norm1(x, t), pos_bias=pos_bias)
        x = x + self.ffn(self.norm2(x, t))
        return x
```

8.2 Step 2: Implement the Forward Diffusion Process (Weeks 5–6)

Implement the variance-preserving SDE:

```
def forward_diffusion(x_0, t, beta_schedule):
    # x_0: clean latent field
    # t: timestep
    alpha_bar = torch.cumprod(1 - beta_schedule, dim=0)
    noise = torch.randn_like(x_0)
    x_t = torch.sqrt(alpha_bar[t]) * x_0 + torch.sqrt(1 - alpha_bar[t]) * noise
    return x_t, noise
```

8.3 Step 3: Implement the Reverse Process (Weeks 7–10)

Train the model to predict the noise:

```
def training_step(batch):
    x_0 = embed_multimodal(batch)  # project text/image/video into L
    t = torch.randint(0, T, (batch_size,))
    x_t, noise = forward_diffusion(x_0, t, beta_schedule)
    predicted_noise = model(x_t, t, pos_bias, modality_emb)
    loss = F.mse_loss(predicted_noise, noise)
    return loss
```

8.4 Step 4: Add MTP (Weeks 11–12)

Modify the output head to predict k noise vectors:

```
class MTPOUTPUT(nn.Module):
    def __init__(self, dim, k):
        self.heads = nn.ModuleList([nn.Linear(dim, dim) for _ in range(k)])

    def forward(self, x):
        return [head(x) for head in self.heads]
```

Train with the coupled prediction loss:

```
def mtp_loss(predicted_noises, true_noises, lambdas):
    loss = sum(l * F.mse_loss(pred, true)
               for l, pred, true in zip(lambdas, predicted_noises, true_noises))
    return loss
```

8.5 Step 5: Inference (Weeks 13–16)

Implement the sampling loop:

```
def sample(model, shape, T):
    x = torch.randn(shape)  # start from noise
    for t in reversed(range(T)):
        t_batch = torch.full((shape[0],), t)
        predicted_noises = model(x, t_batch, pos_bias, modality_emb)
        # Use the 1-step prediction for the actual denoising
        x = denoise_step(x, predicted_noises[0], t, beta_schedule)
    return x
```

---

9. Next-Gen Ideas

9.1 The Real-Time Diffusion Engine

Current diffusion models take multiple steps (20–50) to generate an output. What if we could do it in one step? A **distilled real-time diffusion engine** would use a student model trained to predict the final clean output directly from noise, bypassing the iterative denoising process. This would enable real-time video generation, real-time interactive image editing, and real-time text generation — all in the same

unified model. The technology exists (consistency models, adversarial distillation), but applying it to a unified multimodal model would be revolutionary.

9.2 The Physical Diffusion Prior

What if the diffusion process incorporated physical laws? For video generation, the model could learn to respect conservation of mass, momentum, and energy. For image generation, it could learn perspective, lighting, and material properties. This would not be a separate physics engine bolted onto the model — it would be a **physical prior** encoded in the score function, guiding the denoising process toward physically plausible states. The result would be generated content that is not just visually realistic but physically correct.

9.3 The Adversarial Diffusion Critic

Current diffusion models are trained with MSE loss, which tends to produce blurry outputs. What if we added an adversarial critic that evaluates the quality of the denoised output at each step? The critic would be a small discriminator network trained to distinguish between real data and model outputs. The diffusion model would be trained to fool the critic while still minimizing the MSE loss. This would produce sharper, more realistic outputs without the training instability of pure GANs.

---

10. Conclusion

The Omni-Directional Diffusion Engine is not a collection of separate diffusion models. It is a single engine that restores the entire multimodal signal in parallel. Full bidirectional attention, cross-modal coupling through the unified latent field, and temporal foresight through MTP combine to create a generative system that is faster, more coherent, and more flexible than any sequential alternative.

In the next paper, we zoom in on the Multi-Token Prediction mechanism: how predicting multiple future states at every denoising step creates built-in foresight, and what this means for planning, reasoning, and creative generation.

Also, build things. Stop just reading papers. The best way to learn is to break things and fix them. That's how I learned Linux, and that's how you'll learn to build the Omni-Diffuser.

References

Besta, M., Blach, N., Kubicek, A., Gerstenberger, R., Podstawski, M., Gianinazzi, L., … & Hoefler, T. (2024). Graph of Thoughts: Solving elaborate problems with large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16), 17682–17690.

Ho, J., Jain, A., & Abbeel, P. (2020). Denoising diffusion probabilistic models. *NeurIPS*, 33, 6840–6851.

Peebles, W., & Xie, S. (2023). Scalable diffusion models with transformers. *ICCV*, 4195–4205.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. *CVPR*, 10684–10695.

Song, J., Meng, C., & Ermon, S. (2020). Denoising diffusion implicit models. *ICLR 2021*.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., … & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. *NeurIPS*, 35, 24824–24837.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct: Synergizing reasoning and acting in language models. *ICLR 2023*.