CVPR 2026 Paper Talk

MPDiT: Multi-Patch Global-to-Local Transformer

Efficient flow matching and diffusion generation

- Coarse-to-fine transformer: **large patches early, small patches late**
- Conditioning upgrades: **FNO time embedding** and **multi-token class embedding**
- Target: reduce DiT compute without sacrificing ImageNet sample quality

Generated samples from MPDiT-XL, CFG scale $w = 3$, epoch 160.

Problem: DiT uses the same token budget everywhere

Why standard DiT is expensive

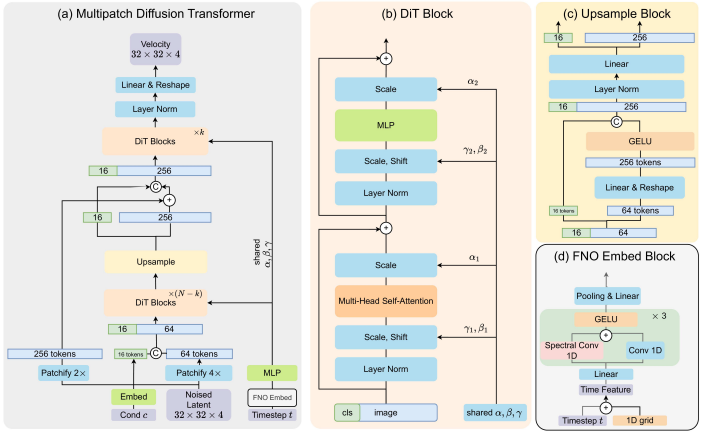
- Latent ImageNet-256 becomes a 32×32 grid after VAE compression.
- DiT-B/2 uses $p = 2$: $16 \times 16 = 256$ **image tokens** in every block.
- Self-attention cost scales roughly with L^2, where L is token length.
- Early blocks may not need all local detail - they mainly build global structure.

Key question: Can we spend most transformer depth on cheap global tokens, then use a few fine-resolution blocks for detail?

$$\frac{64}{256} = 25\%$$

tokens in early global blocks

Architecture overview: global-to-local Multi-Patch DiT



(a) MPDiT: coarse tokens first, fine tokens last.

(b) Block: shared conditioning instead of per-block heavy modulation.

(c,d) Modules: upsample and FNO time embedding.

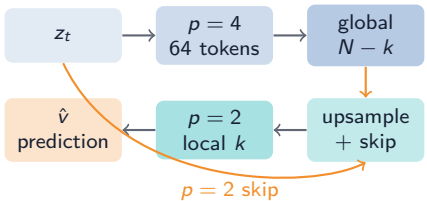
Core design: cheap global blocks, selective local refinement

MPDiT token schedule

1. $\texttt{PatchEmbed}_{p=4}(z_t)$ for the first $N-k$ blocks.
2. Process only 64 **tokens** to model global layout.
3. Upsample $64 \to 256$ tokens and add a $\texttt{PatchEmbed}_{p=2}$ skip.
4. Final k **blocks** operate at $p=2$ for local details.

Flow matching objective:

$$L_{FM} = \mathbb{E}_{z,t,n}\|f_\theta(z_t,t,c)-(n-z)\|_2^2, \quad z_t=(1-t)z+tn.$$



Design intuition

The coarse path captures scene-level structure; the late fine path restores spatial detail with a small number of expensive blocks.

Conditioning modules: small changes, large training gains

Shared AdaIN

Replaces repeated per-block modulation with a shared conditioning path. It cuts parameters while preserving performance.

Multi-token class embedding

Uses m class tokens, not one. More condition tokens let image tokens interact with richer label semantics.

FNO time embedding

Uses MixedFNO blocks over grid-based time features, giving smoother temporal representations for flow matching.

Method	Params (M)	GFLOPs ↓	FID ↓
DiT-B/2	130.0	23.0	34.84
+ Shared AdaIN	90.3	22.9	35.31
+ Multi-token class ($m = 16$)	101.9	24.3	28.56
+ FNO time embedding	101.2	24.3	24.52
+ MPDiT ($k = 6$)	104.8	**16.6**	**24.74**

Table 1: Component ablation from the paper, trained for 80 epochs.

Main result: better quality at much lower compute

Model	Epochs	GFLOPs ↓	FID ↓	sFID ↓	IS ↑	Rec ↑
SiT-B/2	80	23.02	34.84	6.59	41.53	0.64
DiG-B/2	80	17.07	39.50	8.50	37.21	–
DiCo-B	80	16.88	27.20	–	56.52	0.61
MPDiT-B	80	**16.60**	**24.74**	**6.32**	**57.40**	**0.65**
SiT-XL/2	80	118.66	18.04	5.07	73.90	0.64
DiG-XL/2	80	89.40	18.53	6.06	68.53	0.64
DiCo-XL	80	87.30	11.67	–	100.42	0.61
MPDiT-XL	80	**59.30**	**9.92**	**5.05**	**102.79**	**0.64**
SiT-XL/2	1400	118.66	9.35	6.38	126.06	0.68
DiG-XL/2	240	89.40	8.60	6.46	**130.03**	0.68
MPDiT-XL	240	**59.30**	**7.36**	**5.09**	127.57	**0.69**
MPDiT-XL-G	240	**59.30**	**2.05**	**4.51**	278.73	**0.61**

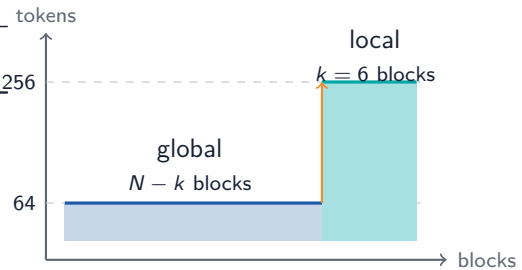
Table 2: ImageNet 256 results. Selected rows from the paper table.

Headline: MPDiT-XL reaches **7.36 FID** at **59.3 GFLOPs**; with guidance it reaches **2.05 FID** after 240 epochs.

Efficiency: same depth, fewer tokens for most blocks

Model	N	k	Dim D	GFLOPs ↓	Ratio ↓
MPDiT-B	12	6	768	**16.6**	72.1%
MPDiT-XL	28	6	1152	**59.3**	**49.9%**

Table 3: Model configuration and computational cost.



Interpretation

MPDiT-XL keeps the same general transformer scaling story, but the majority of its blocks operate on coarse $p = 4$ tokens. This brings the GFLOPs to about half of DiT-XL/2 while preserving or improving FID.

Why this works

Global structure is learned cheaply first. Local refinement is delayed until the end, where high-resolution tokens have the most impact on sample quality.

Ablations: how much local refinement is needed?

Method	Params (M)	GFLOPs ↓	FID ↓
Configuration B			
DiT-B/2 †	101.2	24.3	24.52
MPDiT $k = 4$	104.8	**13.9**	26.94
MPDiT $k = 6$	104.8	16.6	**24.74**
MPDiT $k = 8$	104.8	19.3	**24.62**
Configuration XL			
DiT-XL/2 †	473.1	125.5	9.22
MPDiT $k = 4$	481.2	**53.2**	11.11
MPDiT $k = 6$	481.2	59.3	9.92
MPDiT $k = 8$	481.2	65.4	**9.73**

Table 4: k ablation. † includes shared AdaIN, multi-token class, and FNO time embedding.

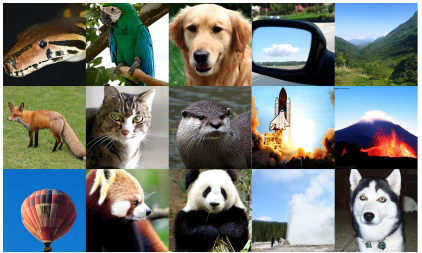
Reading the table

- $k = 4$: maximum compute saving, but less local refinement.
- $k = 6$: good default - close to DiT quality with much lower GFLOPs.
- $k = 8$: marginally better FID, but more compute.

$k = 6$
balanced point for quality and efficiency

Qualitative samples

MPDiT-XL qualitative samples.



ImageNet 512 qualitative samples.

Visual takeaway

The architecture preserves global object coherence while local blocks restore texture and class-specific details.

Takeaways and closing pitch

Three takeaways

1. MPDiT turns DiT into a **coarse-to-fine** transformer.
2. Most blocks use **64 tokens**; only late blocks use **256 tokens**.
3. FNO time and multi-token class embeddings improve convergence.

Limitation: results focus on ImageNet class-conditional generation; large text-to-image and video settings are future work.

Final message

One-line pitch: MPDiT keeps DiT quality by delaying expensive local tokens until they matter most.

Why it matters

- The method reduces compute without changing the generative training objective.
- It is a simple architectural reallocation: global reasoning first, local detail later.
- The same idea is promising for larger-scale generative transformers.