VDE: Training-Free Accelerating Rectified Flow Model via Velocity Decomposition and Estimation

Supplementary Material

We organize our supplementary material as follows:

- In Sec. A, we provide the definition and 10,000-trajectory analysis of the stable phase.
- In Sec. B, we present error bound analysis for VDE.
- In Sec. C, we demonstrate VDE's scalability to image editing tasks, advanced video foundation models, and compatibility with distillation.
- In Sec. D, we present more experimental results, including an ablation study on recompute strategies, computation overhead analysis of VDE, and analysis of latent trajectory consistency.
- In Sec. E, we provide more visualization across rectified flow models.
- In Sec. F, we present more evidence supporting Velocity Decomposition and Estimation across multiple models.

A. Identification of the stable phase

To provide a concrete definition for the transition into the stable phase, we define that the denoising process enters the stable phase at step i if and only if: (1) the relative coefficient extrapolation error at step $i+2$ is $< \epsilon$ (e.g., 2%), and (2) the cosine similarity of orthogonal directions at steps i and $i+1$ is $> \delta$ (e.g., 0.99). This strict criterion enables practical, automated detection.

To verify the consistency of this transition and ensure VDE does not suffer from failure cases, we conducted a large-scale stress test analyzing 10,000 trajectories across varying conditions (2,000 prompts $\times$ 5 random seeds).

As shown in Fig. A(a), the transition point to the stable phase is predominantly and reliably concentrated in the extremely early phase of the generation (typically steps 0–8). Furthermore, Fig. A(b) demonstrates that the orthogonal direction is highly stable across all evaluated prompts, with the cosine similarity rapidly converging to $>$ 0.99. This high consistency confirms that orthogonal stability is a robust, intrinsic property of rectified flow models. Consequently, we directly fixed the warm-up hyperparameter (w) in the main paper as a reliable and effective default.

B. Error Bound Analysis

We provide a rigorous bounded-error analysis for our VDE method.

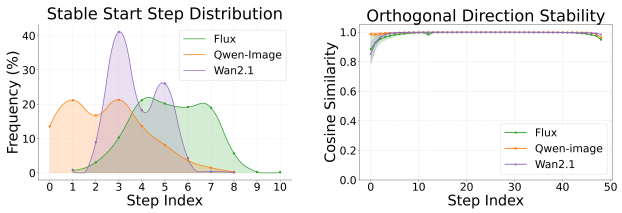

Figure A. Statistical analysis of 10,000 trajectories. (a) The transition point to the stable phase is predominantly concentrated in the early phase (steps 0–8). (b) The orthogonal direction remains highly stable, with cosine similarity converging to > 0.99.

B.1. Derivation of Coefficient Extrapolation Error Bound

Assume coefficients $\alpha(t)$ and $\beta(t)$ are twice continuously differentiable with bounded second derivatives: $|\alpha''(s)| \leq M_\alpha, |\beta''(s)| \leq M_\beta$ for $s \in [t_1, t]$. Let anchor steps be $t_1 < t_2 < t$, with history interval $\Delta = t_2 - t_1$ and prediction step $h = t - t_2$.

Analysis for α_t: By Taylor's theorem at t_2, the true value at t is:

$$\alpha_t = \alpha_{t_2} + \alpha'(t_2)h + \frac{1}{2}\alpha''(\xi_1)h^2, \quad \xi_1 \in [t_2, t] \tag{1}$$

The value at t_1 expanded at t_2 is:

$$\alpha_{t_1} = \alpha_{t_2} + \alpha'(t_2)(-\Delta) + \frac{1}{2}\alpha''(\xi_2)(-\Delta)^2, \quad \xi_2 \in [t_1, t_2] \tag{2}$$

From (2), we solve for the derivative $\alpha'(t_2)$:

$$\alpha'(t_2) = \frac{\alpha_{t_2} - \alpha_{t_1}}{\Delta} + \frac{1}{2}\alpha''(\xi_2)\Delta \tag{3}$$

Substituting (3) into (1):

$$\alpha_t = \alpha_{t_2} + \left[\frac{\alpha_{t_2} - \alpha_{t_1}}{\Delta} + \frac{1}{2}\alpha''(\xi_2)\Delta\right] h + \frac{1}{2}\alpha''(\xi_1)h^2 \tag{4}$$

Our linear extrapolation estimate is $\hat{\alpha}_t = \alpha_{t_2} + \frac{\alpha_{t_2} - \alpha_{t_1}}{\Delta}h$. Thus, the error is:

$$\Delta\alpha_t = \alpha_t - \hat{\alpha}_t = \frac{1}{2}\alpha''(\xi_2)\Delta h + \frac{1}{2}\alpha''(\xi_1)h^2 \tag{5}$$

Taking the absolute value and applying the bound M_α:

$$|\Delta\alpha_t| \leq \frac{M_\alpha}{2}(\Delta h + h^2) = \frac{M_\alpha}{2}h(\Delta + h) \tag{6}$$

The same bound applies to β_t.

B.2. Derivation of Directional Reuse and Total Error Bound

Directional Reuse Error Bound. Define the orthogonal residual $\mathbf{r}_t = \mathbf{v}_t - \alpha_t \mathbf{x}_t$. The unit direction is $\mathbf{u}_t = \mathbf{r}_t / \|\mathbf{r}_t\|$ (for $\mathbf{r}_t \neq 0$). Assume in the stable phase, there exist constants $\beta_{\min} > 0$ and $x_{\min} > 0$ such that:

$$\begin{aligned} \beta_t \geq \beta_{\min}, \quad \|\mathbf{x}_t\| \geq x_{\min} \\ \Longrightarrow \|\mathbf{r}_t\| = \beta_t \|\mathbf{x}_t\| \geq \beta_{\min} x_{\min} > 0. \end{aligned} \tag{7}$$

Using the standard inequality for unit vector difference:

$$\left\| \frac{\mathbf{a}}{\|\mathbf{a}\|} - \frac{\mathbf{b}}{\|\mathbf{b}\|} \right\| \leq \frac{\|\mathbf{a} - \mathbf{b}\|}{\min(\|\mathbf{a}\|, \|\mathbf{b}\|)}. \tag{8}$$

We have:

$$\|\mathbf{u}_t - \mathbf{u}_{t_2}\| \leq \frac{\|\mathbf{r}_t - \mathbf{r}_{t_2}\|}{\min(\|\mathbf{r}_t\|, \|\mathbf{r}_{t_2}\|)}. \tag{9}$$

Numerator Estimate: Assume $\mathbf{r}(t)$ is Lipschitz continuous with constant $L_r > 0$:

$$\|\mathbf{r}_t - \mathbf{r}_s\| \leq L_r |t - s|, \quad \forall t, s. \tag{10}$$

This assumption stems from the smoothness of the learned velocity field in RF models. Thus, with $h = t - t_2$:

$$\|\mathbf{r}_t - \mathbf{r}_{t_2}\| \leq L_r h. \tag{11}$$

Denominator Estimate: From the lower bound assumption:

$$\min(\|\mathbf{r}_t\|, \|\mathbf{r}_{t_2}\|) \geq \beta_{\min} x_{\min}. \tag{12}$$

Substituting into (9):

$$\|\mathbf{u}_t - \mathbf{u}_{t_2}\| \leq \frac{L_r}{\beta_{\min} x_{\min}} h. \tag{13}$$

Define $L_u = L_r / (\beta_{\min} x_{\min})$, then:

$$\|\mathbf{u}_t - \mathbf{u}_{t_2}\| \leq L_u h. \tag{14}$$

Total Velocity Error Bound. The velocity decomposition is $\mathbf{v}_t = \alpha_t \mathbf{x}_t + \beta_t \|\mathbf{x}_t\| \mathbf{u}_t$. The estimate is $\hat{\mathbf{v}}_t = \hat{\alpha}_t \mathbf{x}_t + \hat{\beta}_t \|\mathbf{x}_t\| \mathbf{u}_{t_2}$. The error is:

$$\mathbf{v}_t - \hat{\mathbf{v}}_t = (\alpha_t - \hat{\alpha}_t)\mathbf{x}_t + (\beta_t - \hat{\beta}_t)\|\mathbf{x}_t\|\mathbf{u}_t + \hat{\beta}_t \|\mathbf{x}_t\| (\mathbf{u}_t - \mathbf{u}_{t_2}). \tag{15}$$

Taking the norm and using triangle inequality (and $\mathbf{x}_t \perp \mathbf{u}_t$):

$$\|\mathbf{v}_t - \hat{\mathbf{v}}_t\| \leq |\Delta\alpha_t| \|\mathbf{x}_t\| + |\Delta\beta_t| \|\mathbf{x}_t\| + |\hat{\beta}_t| \|\mathbf{x}_t\| \|\mathbf{u}_t - \mathbf{u}_{t_2}\|. \tag{16}$$

The relative error is:

$$E_t := \frac{\|\mathbf{v}_t - \hat{\mathbf{v}}_t\|}{\|\mathbf{x}_t\|} \leq |\Delta\alpha_t| + |\Delta\beta_t| + |\hat{\beta}_t| \|\mathbf{u}_t - \mathbf{u}_{t_2}\|. \tag{17}$$

Substituting the bounds derived earlier, and assuming $|\hat{\beta}_t| \leq B$:

$$E_t \leq \frac{M_\alpha + M_\beta}{2} h(\Delta + h) + B L_u h. \tag{18}$$

This rigorous proof confirms that the estimation error is $\mathcal{O}(h)$ and is bounded by model-intrinsic properties (M, L, B) and step size h, thereby preventing unbounded error accumulation.

C. Generalization and Compatibility

To demonstrate the broad applicability of VDE, we evaluate its performance across various tasks, advanced foundation models, and distillation models.

C.1. Generalization to Image Editing

VDE is universally applicable to any RF model regardless of the specific task. Beyond standard text-to-image generation, VDE seamlessly applies to editing tasks. As shown in Table A, on the **Flux.1 Fill** model (512×512, 50% mask), VDE achieves a 1.91× speedup with an exceptionally high SSIM of 0.9739, preserving the unmasked regions perfectly while accelerating the filling process.

Table A. Extension of VDE to Image Editing on Flux.1 Fill.

Method	Speedup ↑	Latency(s) ↓	NFE ↓	SSIM ↑	PSNR ↑	LPIPS ↓
Flux.1 Fill (512×512, 50% mask)						
$T = 50$	1.00×	8.56	50	-	-	-
VDE	1.91×	4.48	26	0.9739	35.23	0.0310

C.2. Generalization to More Video Foundation Models

To validate the generality of VDE beyond the Wan2.1 model used in the main paper, we additionally evaluate VDE on two advanced video foundation models: **Open-Sora 1.2** and **HunyuanVideo-1.5**.

Open-Sora 1.2. As shown in Table B, the overall trends on Open-Sora 1.2 (51 frames, 480P) are consistent with our main results. VDE achieves a 2.14× speedup with a latency of 20.82 s, which is on par with TeaCache-fast and significantly faster than classical approaches such as Δ-DiT and T-GATE. On visual retention, VDE outperforms all baselines across SSIM and LPIPS, demonstrating its ability to preserve temporal consistency and structural fidelity. On VBench, VDE delivers a competitive score (78.40%), closely matching the original model.

HunyuanVideo-1.5. Furthermore, VDE scales effectively to state-of-the-art, massive-scale video foundation models. As reported in Table C, VDE speeds up HunyuanVideo-1.5 (49 frames, 480P) by up to 2.47× with high fidelity, proving that our assumption of linearity and

Table B. Comparison of different methods on Open-Sora 1.2 (51 frames, 480P).

Methods	Efficiency		Visual Retention			VBench (%) ↑
	Speedup ↑	Latency (s) ↓	SSIM ↑	PSNR ↑	LPIPS ↓	
$T = 30$	1×	44.56	-	-	-	79.22
Δ-DIT	1.03×	-	0.4811	11.91	0.5692	78.21
T-GATE	1.19×	-	0.6760	15.50	0.3495	77.61
PAB-slow	1.33×	33.40	0.8405	**24.50**	0.1471	77.64
PAB-fast	1.40×	31.85	0.8220	23.58	0.1743	76.95
TeaCache-fast	**2.25**×	**19.84**	0.7477	19.10	0.2511	78.48
VDE(Ours)	2.14×	20.82	**0.7905**	20.45	**0.1885**	78.40

orthogonal stability holds robustly even at a massive commercial scale.

Table C. Extension of VDE to massive-scale Video Foundation Models on HunyuanVideo-1.5.

Method	Speedup ↑	Latency(s) ↓	NFE ↓	SSIM ↑	PSNR ↑	LPIPS ↓
HunyuanVideo-1.5 (49 frames, 480P)						
$T = 50$	1.00×	363.42	50	-	-	-
VDE-slow	1.91×	190.27	26	0.8829	27.04	0.0786
VDE-fast	2.47×	147.13	20	0.7974	22.65	0.1501

C.3. Compatibility with Distillation

VDE is orthogonal to model distillation techniques and can be stacked to achieve extreme acceleration. We tested VDE on Z-image-Turbo, a distilled model optimized for few-step inference (baseline $T = 8$). As reported in Tab. D, VDE successfully further reduces the inference budget to just 5 steps (1.51× speedup) while preserving structural integrity much better than direct step truncation (SSIM 0.8887 vs. 0.7756). This demonstrates that VDE can complement distillation methods to push the boundaries of real-time generation.

Table D. **Compatibility with Distilled Models.** VDE can further accelerate the distilled Z-image-Turbo model, achieving comparable quality with fewer steps.

Method	Speedup ↑	Latency(s) ↓	NFE ↓	SSIM ↑	PSNR ↑	LPIPS ↓	CLIP ↑
Z-image-Turbo (Distilled, 1024×1024)							
$T = 8$	1.00×	3.45	8	-	-	-	0.3154
$T = 5$	1.52×	2.27	5	0.7756	20.14	0.2106	**0.3169**
VDE (Ours)	1.51×	2.29	5	**0.8887**	**27.83**	**0.1657**	0.3145

D. More Experimental Results

D.1. Ablation Study on Recompute Strategies

To address the potential concern that a uniform-step recomputation policy may be suboptimal, we further compare it against several rate-of-change-based dynamic strategies, which resemble the policies used in prior cache-based acceleration methods such as TeaCache, EasyCache, and AdaCache. These prior works rely on a assumption that when the model's input or output changes only slightly, cached features remain reliable and can be safely reused. Following this principle, we construct two families of indicators — input variation and velocity (output) variation — measured under both L1 or L2 criteria. Recompute is triggered only when the indicator exceeds a preset threshold. Table E provides a detailed comparison between uniform-step recomputation and several *rate-of-change–based* dynamic strategies. We observe four consistent trends.

First, *uniform-step recomputation exhibits a clean and controllable efficiency–fidelity trade-off*. Decreasing the interval from 3 to 1 gradually improves visual retention (e.g., SSIM increases from 0.8499 to 0.9283), while latency increases proportionally. This monotonic and predictable behavior makes the schedule both easy to tune and stable across runs.

Second, *input-based rate-of-change policies perform the worst among all options*. Although these strategies achieve comparable speedups (e.g., 2.28× at L2–0.15), their visual metrics degrade significantly—L1–0.1 produces SSIM of only 0.7789 and LPIPS of 0.3305, far below both uniform-step and velocity-based strategies. This confirms that raw input variation is a poor indicator of VDE, causing overly aggressive recompute skipping.

Third, *velocity-based rate-of-change policies provide the strongest dynamic baseline*, yet still do not surpass uniform intervals. Small thresholds (e.g., 0.03) maintain high fidelity (SSIM ≈ 0.929, PSNR ≈ 28.9), but their latency remains high (6.7–6.8s, only 1.2× speedup). Increasing the threshold (0.1–0.15) yields speedups similar to uniform intervals 2–3 (3.7–4.1s), but the corresponding visual quality degrades proportionally, underperforming the uniform counterparts. Thus, velocity-based schedules fail to offer a superior efficiency–quality frontier of VDE.

Fourth, in the context of VDE—*rate-of-change-based methods induce a structurally unfavorable recompute pat-*

tern. As the denoising process begins with strong noise and small timestep differences, the rate-of-change signals tend to be small in the early stage, causing dynamic policies to skip more recomputations in the early denoising stage, where errors accumulate rapidly and directly enlarge the primary VDE error source (i.e., the orthogonal unit vector), thereby degrading fidelity. Conversely, in later timesteps where noise is low and timestep spacing is large, the rate-of-change signals become higher, triggering more frequent recomputations and wasting computation precisely where the system could safely reuse estimates. This mismatch produces a systematically worse efficiency–quality trade-off: settings that match the speed of uniform intervals exhibit noticeably worse visual metrics, while settings that match or surpass their fidelity deliver substantially lower speedups.

Overall, despite exploring a wide range of dynamic recomputation triggers, *none of the rate-of-change–based strategies surpass the simplicity and robustness of the uniform-step schedule*. Combined with the decomposition properties of VDE—where the dominant error arises from the orthogonal direction u_t—a fixed interval provides the most stable and effective recomputation pattern, delivering superior quality at the same speed or superior speed at the same quality.

D.2. Computation Overhead of VDE

To validate the efficiency of the proposed VDE in terms of inference overhead, Table F presents a detailed breakdown of inference costs across four representative rectified-flow models (Flux, Qwen-Image, Wan2.1, and OpenSora-1.2), including full model inference time, decomposition overhead, estimation overhead, and the combined ratio of decomposition and estimation overheads relative to full inference. As observed from the results, the total overhead of VDE (decomposition + estimation) is extremely negligible across all models: OpenSora-1.2 achieves the lowest overhead ratio of only 0.04%, followed by Wan2.1 (0.074%), Qwen-Image (0.146%), and Flux (0.188%), with all ratios well below 0.2%. Notably, even for Wan2.1, which has the longest full inference time (2.977s), the cumulative decomposition and estimation overheads account for less than 0.1% of the total inference cost. This demonstrates that VDE introduces minimal computational burden to the inference pipeline of rectified flow models, enabling efficient cost analysis without compromising the original inference efficiency—an essential property for practical deployment in generative vision tasks.

D.3. Latent Trajectory Consistency Analysis

To diagnose how acceleration strategies influence the generative dynamics, we project the latent trajectories of Flux into a 2D PCA space (Fig. B). The 50-step trajectory serves as the ground-truth evolution of the rectified-flow process. Caching-based methods (**TeaCache**, **EasyCache**) exhibit a clear and increasing deviation from this reference path as denoising proceeds, revealing a growing cache–input mismatch. The mismatch leads the sampler to follow an incorrect latent direction, causing accumulation of geometric drift in the trajectory.

In contrast, **VDE** remains tightly aligned with the reference trajectory throughout sampling. This stability stems from VDE's decomposition–estimation design: instead of reusing stale states, VDE explicitly estimates the velocity's principal components, thereby preserving the instantaneous generative direction at each step. As a result, VDE prevents trajectory drift and maintains faithful latent-space dynamics.

Overall, the PCA visualization provides a geometric view of our method's advantage: VDE preserves direction consistency, whereas cache-based methods induce progressive trajectory distortion, directly supporting our analysis on direction stability and predictable velocity-coefficient evolution.

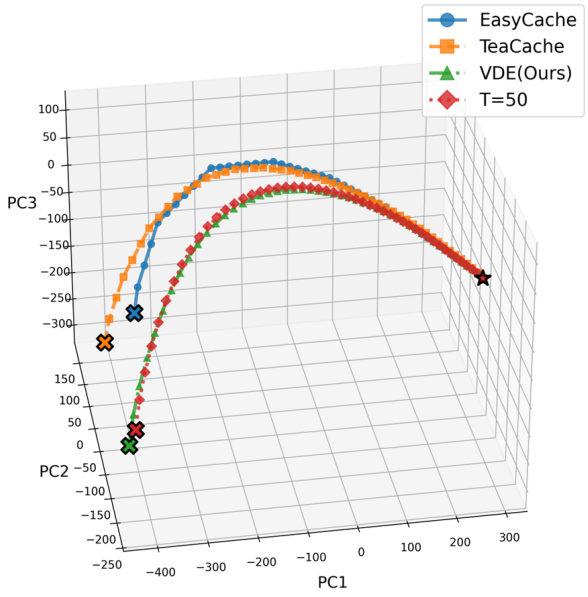

Figure B. PCA visualization of the latent trajectories during sampling. The original 50-step trajectory serves as the reference. Existing caching-and-reusing methods (e.g., TeaCache, and EasyCache) gradually deviate from the reference trajectory, indicating accumulation of cache–input mismatch. In contrast, our method VDE closely follows the reference trajectory throughout the sampling process, demonstrating that decomposing and estimating the latent velocity preserves the correct generative dynamics.

Table E. Ablation study of different recompute strategies on FLUX-dev. We compare uniform-step recomputation and several threshold-based strategies using both L1 and L2 criteria.

Indicator	Criterion	Threshold	Efficiency		Visual Retention			CLIP (%) ↑
			Latency (s) ↓	Speedup ↑	SSIM ↑	PSNR ↑	LPIPS ↓	
$T = 50$	-	-	8.17	1.00×	-	-	-	0.3090
Uniform	Interval	1	4.93	1.69×	0.9283	28.76	0.0801	0.3093
Uniform	Interval	2	3.70	2.21×	0.8877	25.81	0.1243	0.3095
Uniform	Interval	3	3.04	2.70×	0.8499	24.02	0.1679	0.3102
Input Δ	L1	0.1	4.24	1.93×	0.7789	23.79	0.3305	0.3165
Input Δ	L1	0.15	3.57	2.29×	0.7936	23.55	0.3088	0.3152
Input Δ	L2	0.1	4.26	1.92×	0.7789	23.79	0.3305	0.3165
Input Δ	L2	0.15	3.58	2.28×	0.7936	23.55	0.3088	0.3152
Velocity Δ	L1	0.03	6.76	1.21×	0.9284	28.88	0.0785	0.3089
Velocity Δ	L1	0.05	5.31	1.54×	0.9078	27.05	0.1009	0.3090
Velocity Δ	L1	0.1	4.16	1.96×	0.8763	25.09	0.1344	0.3090
Velocity Δ	L1	0.15	3.71	2.20×	0.8577	24.21	0.1553	0.3093
Velocity Δ	L2	0.03	6.79	1.20×	0.9291	28.94	0.0777	0.3088
Velocity Δ	L2	0.05	5.34	1.53×	0.9081	27.06	0.1007	0.3090
Velocity Δ	L2	0.1	4.11	1.99×	0.8763	25.10	0.1344	0.3090
Velocity Δ	L2	0.15	3.68	2.22×	0.8579	24.22	0.1552	0.3093

Table F. Computation overhead analysis of VDE on different rectified flow models, reporting full inference time, decomposition overhead, estimation overhead, and the ratio of their sum to full inference.

Model	Full Inference (s)	Decomposition (s)	Estimation (s)	Overhead Ratio (%)
Flux	0.159	1.43×10^{-4}	1.56×10^{-4}	0.188%
Qwen-Image	0.236	2.26×10^{-4}	1.18×10^{-4}	0.146%
Wan2.1	2.977	2.04×10^{-3}	1.75×10^{-4}	0.074%
OpenSora-1.2	1.475	2.89×10^{-4}	3.01×10^{-4}	0.04%

E. Visualization

E.1. Image Visualization on Flux

Fig. C and Fig. D are visualization results on Flux. To help interpret the visual results in Fig. C, we briefly highlight the discrepancies in each row:

1. **Architecture**: Background building style, crane position
2. **Clock**: Dial design, pointer details, clock thickness
3. **Cat**: Fur color, facial details
4. **Office**: Left monitor style
5. **Owls**: Head posture, feather texture, facial orientation

To help interpret the visual results in Fig. D, we briefly highlight the discrepancies in each row:

1. **Architecture**: Right-building height, style
2. **Living Room**: Pillow style, background painting, coffee table layout
3. **Airplane Window**: Wing shape, landscape details
4. **Sheep**: Sheep count, facial features
5. **Toy Bears**: Bear color, orientation, DVD arrangement

E.2. Image Visualization on Qwen-Image

Fig. E and Fig. F are visualization results on Qwen-Image. To help interpret the visual results in Fig. E, we briefly highlight the discrepancies in each row:

1. **Bathroom**: Toilet style, tile pattern, layout details, wall mirrors
2. **Ski Scene**: Skier count, clothing style, snow landscape details
3. **Shoes & Umbrella**: Shoe type, umbrella details, texture fidelity
4. **Tennis Boy**: Ball position, clothing style, scene context
5. **Train**: Color, headlight details, track scene

To help interpret the visual results in Fig. F, we briefly highlight the discrepancies in each row:

1. **Bird Scene**: Bird type, background scene, human presence
2. **Airplane**: Aircraft model, house details below
3. **Sailboat & Dog**: Dog orientation, sailboat details
4. **Pizza Boy**: Box text, child appearance

5. **Flowers**: Vase style, flower type, layout details

F. Additional Evidence for Velocity Decomposition and Estimation

To further support the two key empirical findings presented in Sec. 3 of the main paper—(1) Predictable coefficients and (2) Stable Orthogonal Direction—we provide extensive visualizations of temporal dynamics across multiple rectified flow models. Fig. G, Fig. H and Fig. I present Flux's temporal dynamics. Fig. M, Fig. N and Fig. O present Wan 2.1's temporal dynamics. Fig. P, Fig. Q and Fig. R present Open-Sora 1.2's temporal dynamics. Fig. J, Fig. K and Fig. L present Qwen-Image's temporal dynamics.

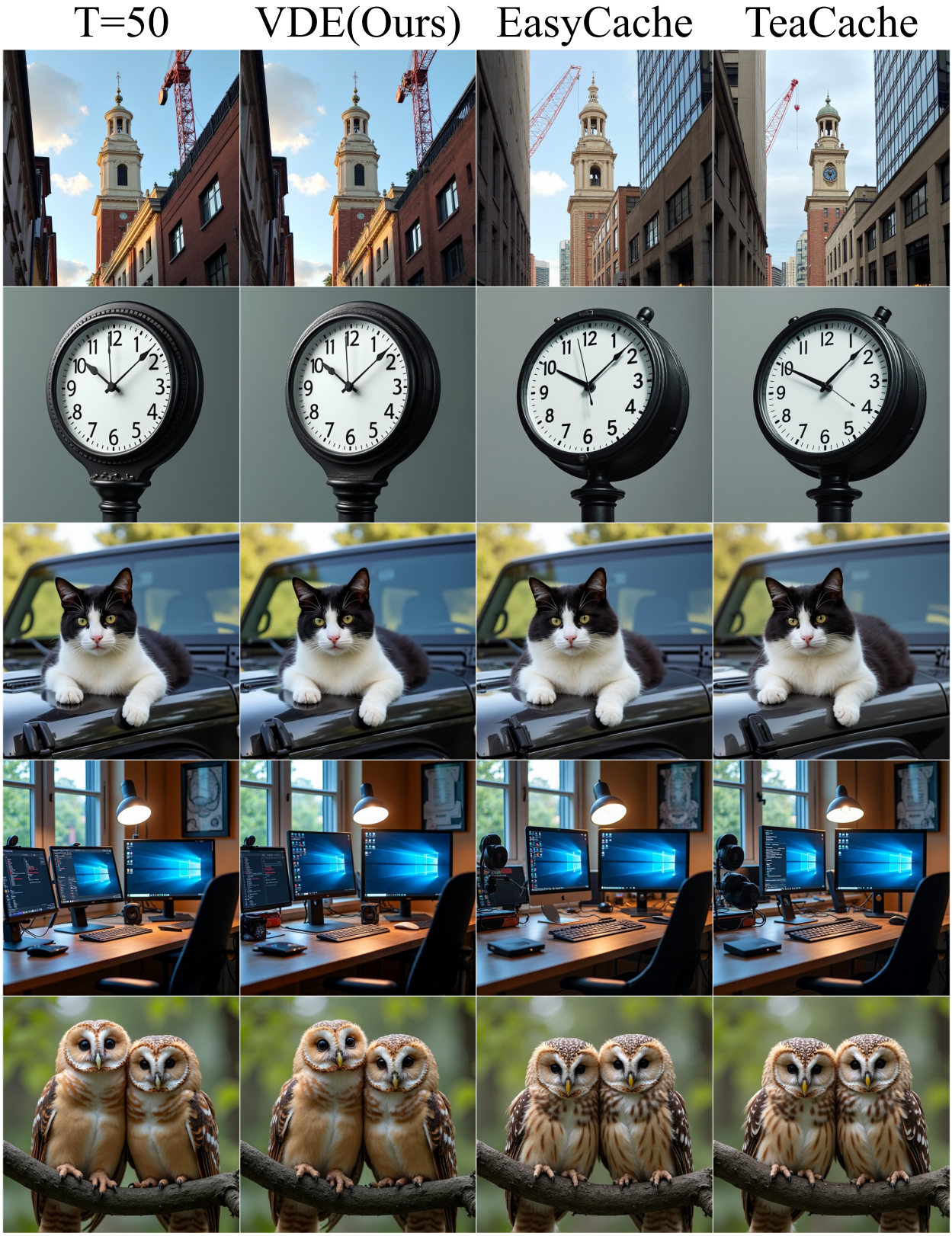

Figure C. Visualization on Flux-dev.

Figure D. Visualization on Flux-dev.

Figure E. Visualization on Qwen-Image.

Figure F. Visualization on Qwen-Image.

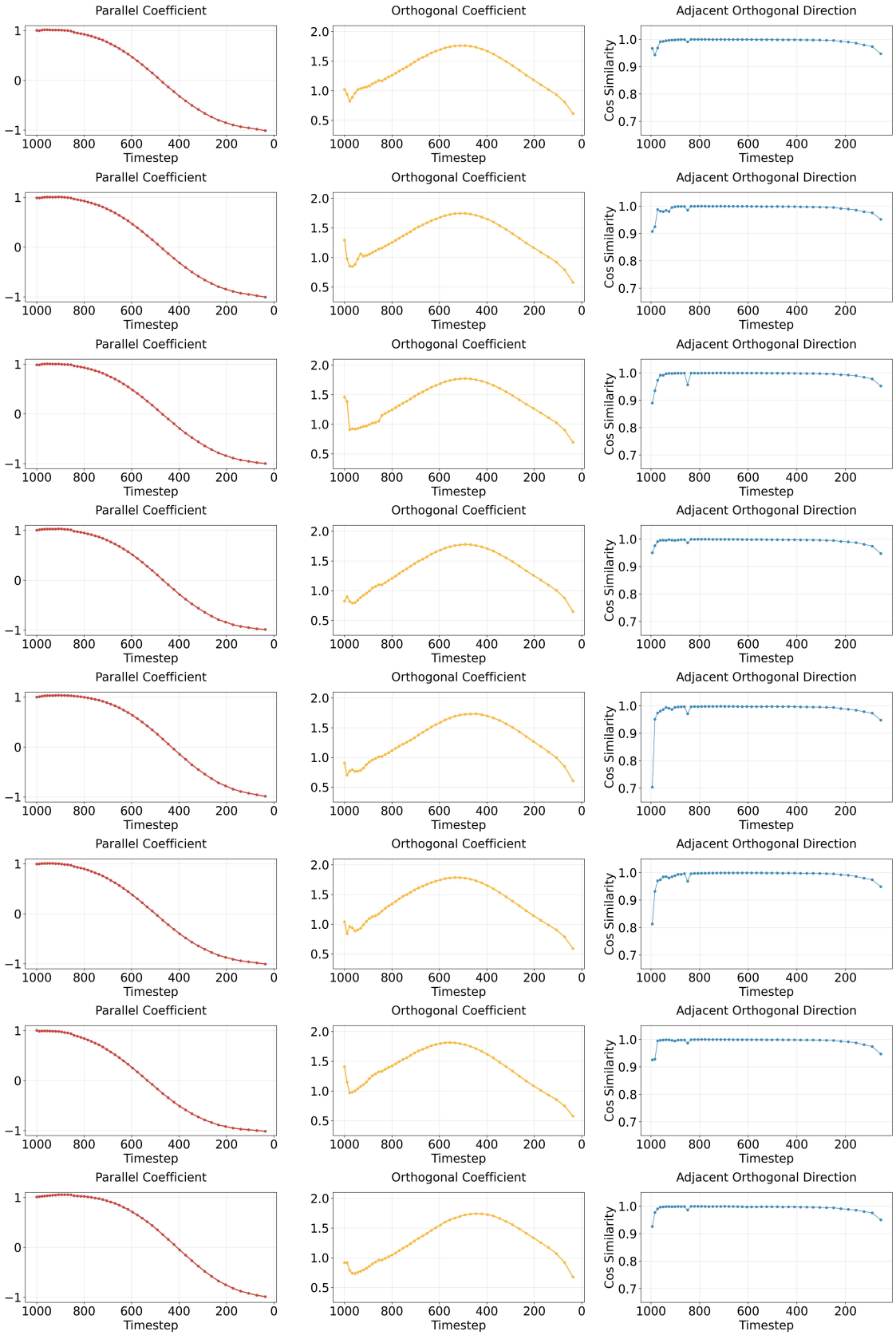

Figure G. Flux's Temporal dynamics of velocity components, including parallel coefficient, orthogonal coefficient and cosine similarity of adjacent orthogonal direction.

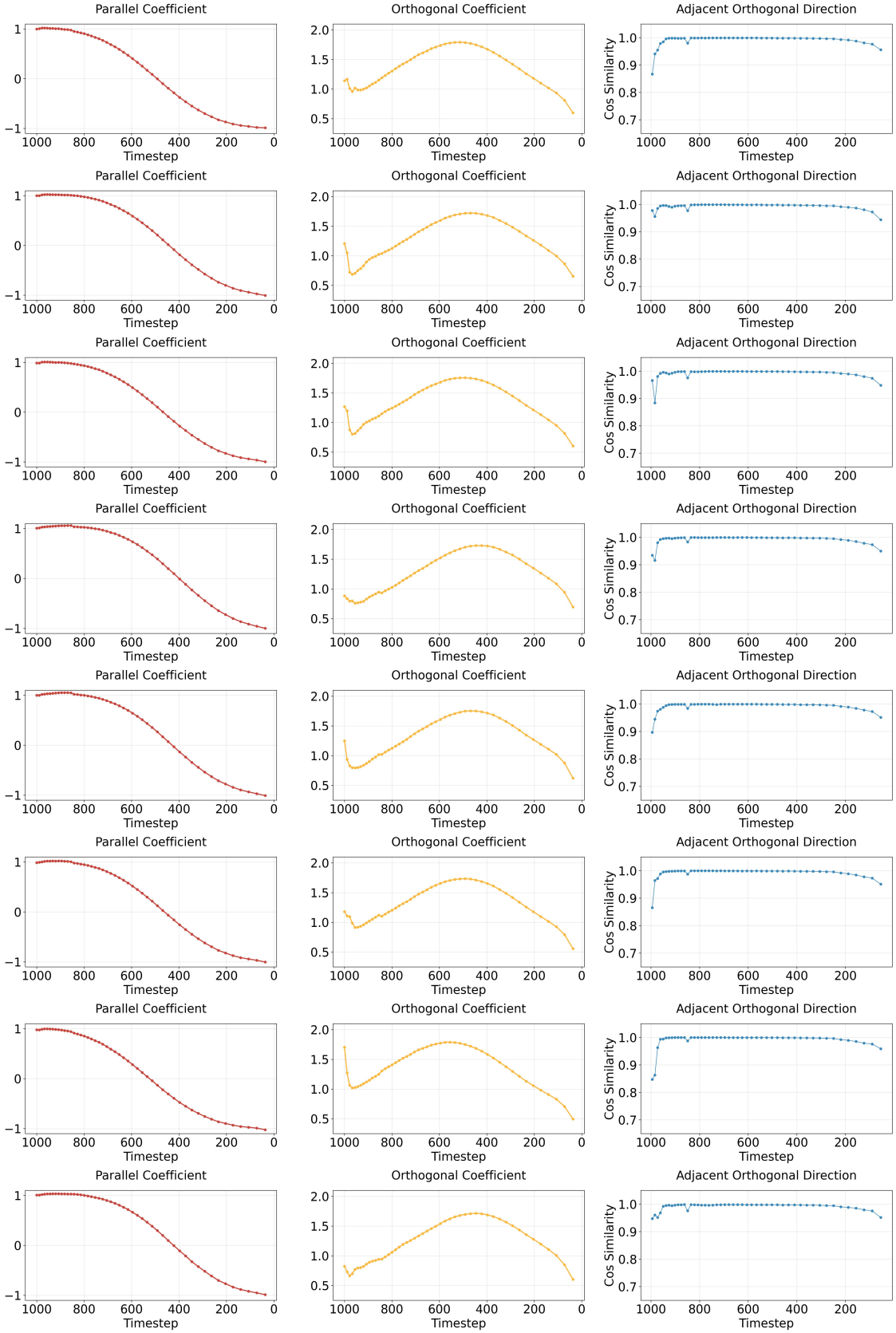

Figure H. Flux's Temporal dynamics of velocity components, including parallel coefficient, orthogonal coefficient and cosine similarity of adjacent orthogonal direction.

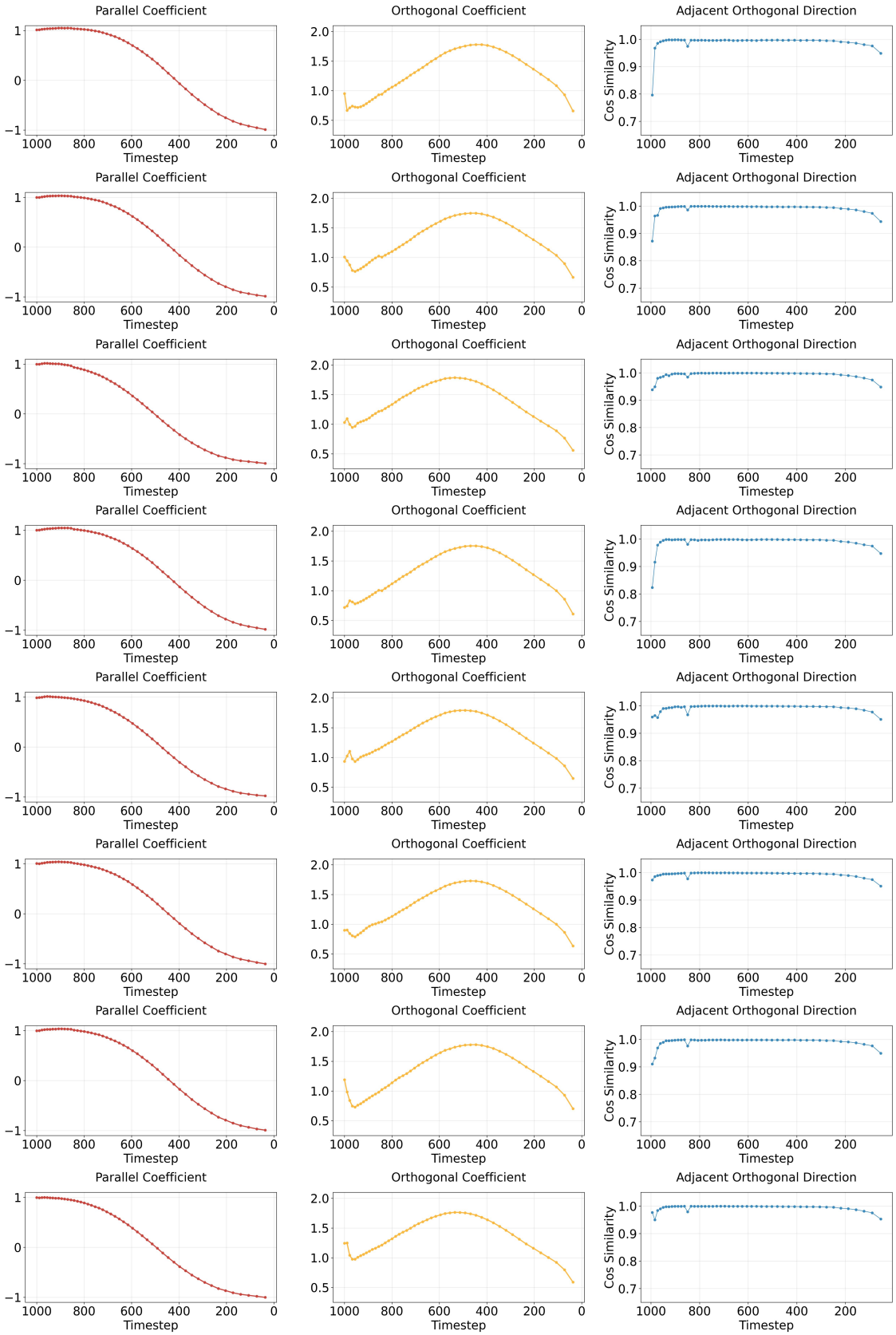

Figure I. Flux's Temporal dynamics of velocity components, including parallel coefficient, orthogonal coefficient and cosine similarity of adjacent orthogonal direction.

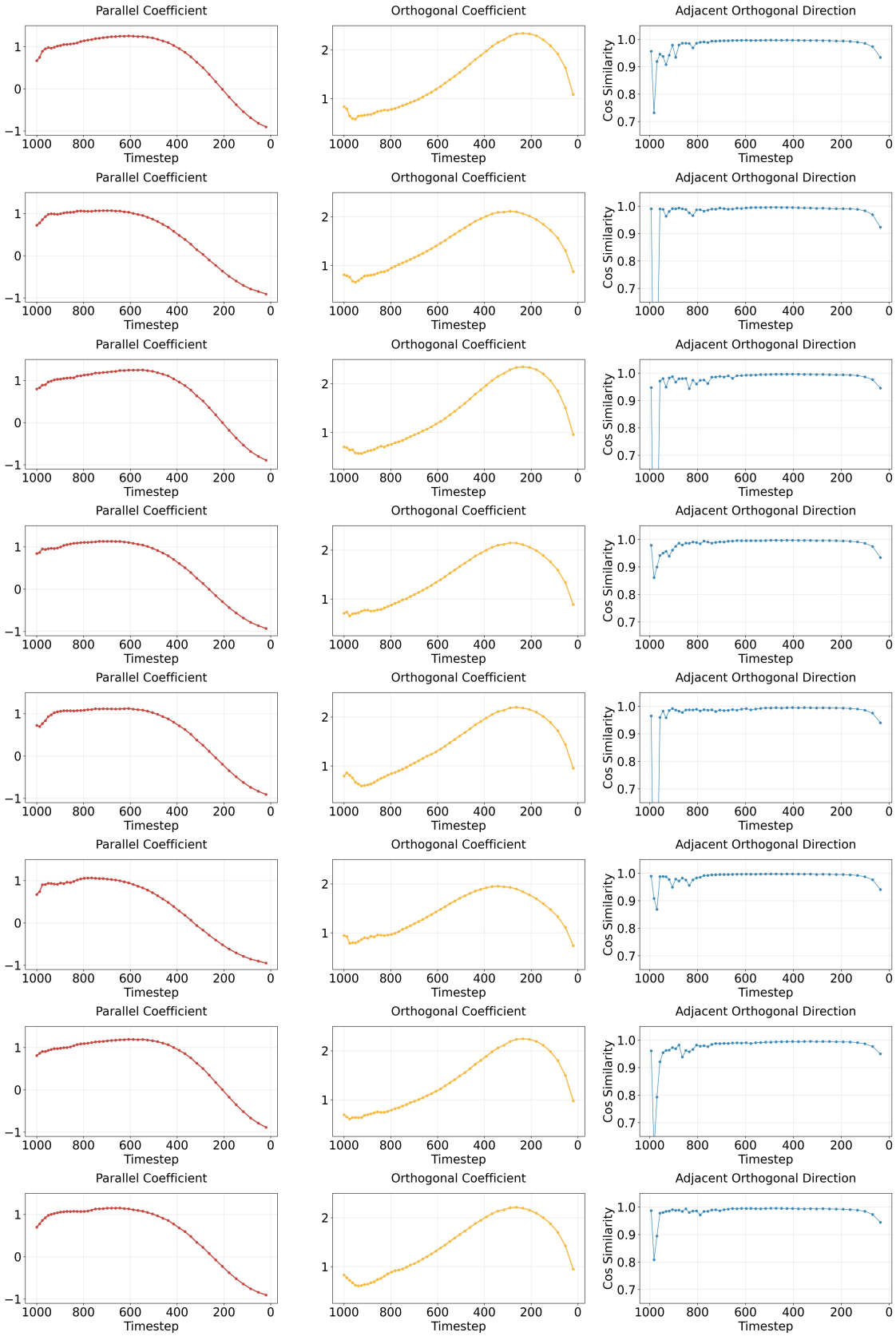

Figure J. Qwen-Image's Temporal dynamics of velocity components, including parallel coefficient, orthogonal coefficient and cosine similarity of adjacent orthogonal direction.

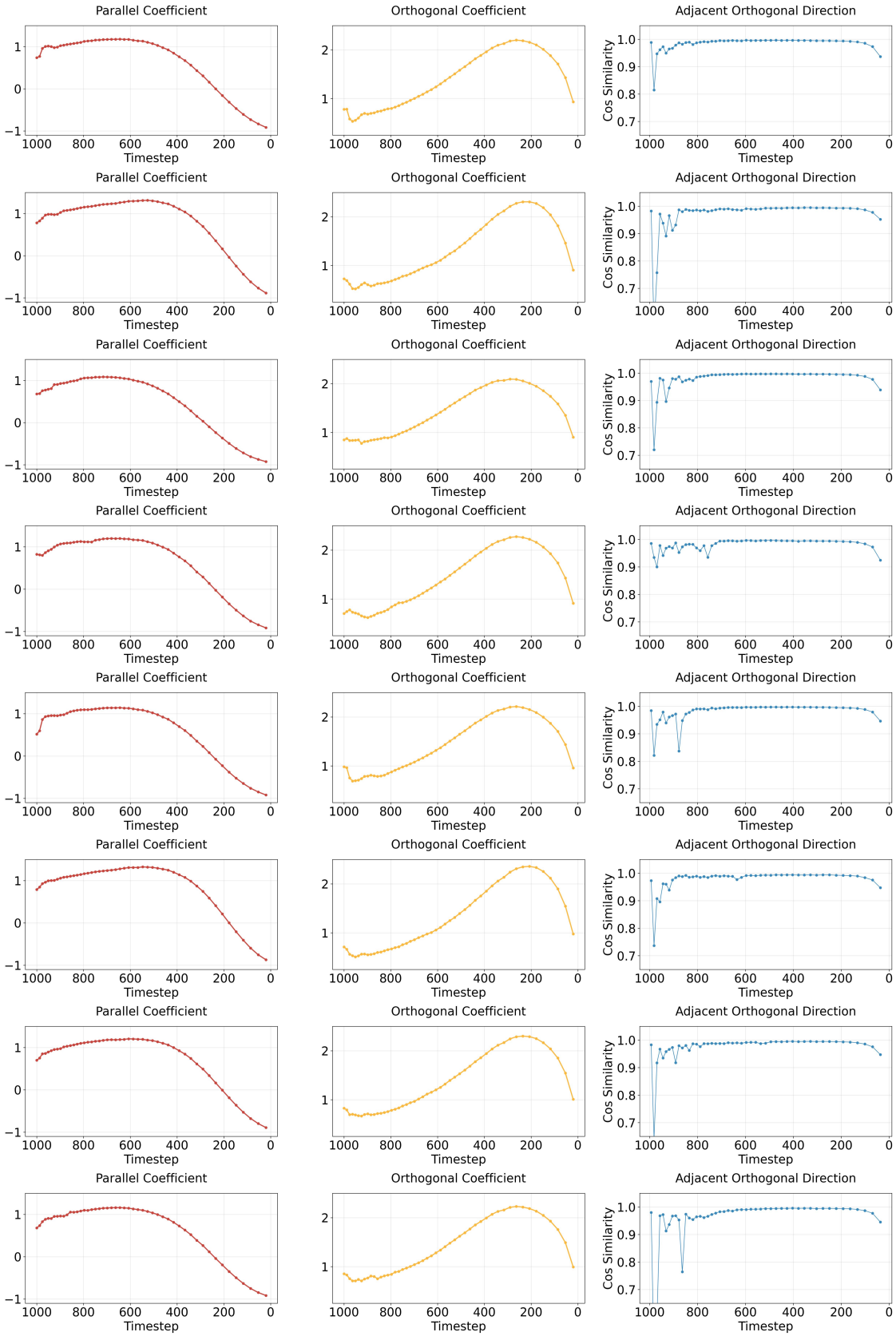

Figure K. Qwen-Image's Temporal dynamics of velocity components, including parallel coefficient, orthogonal coefficient and cosine similarity of adjacent orthogonal direction.

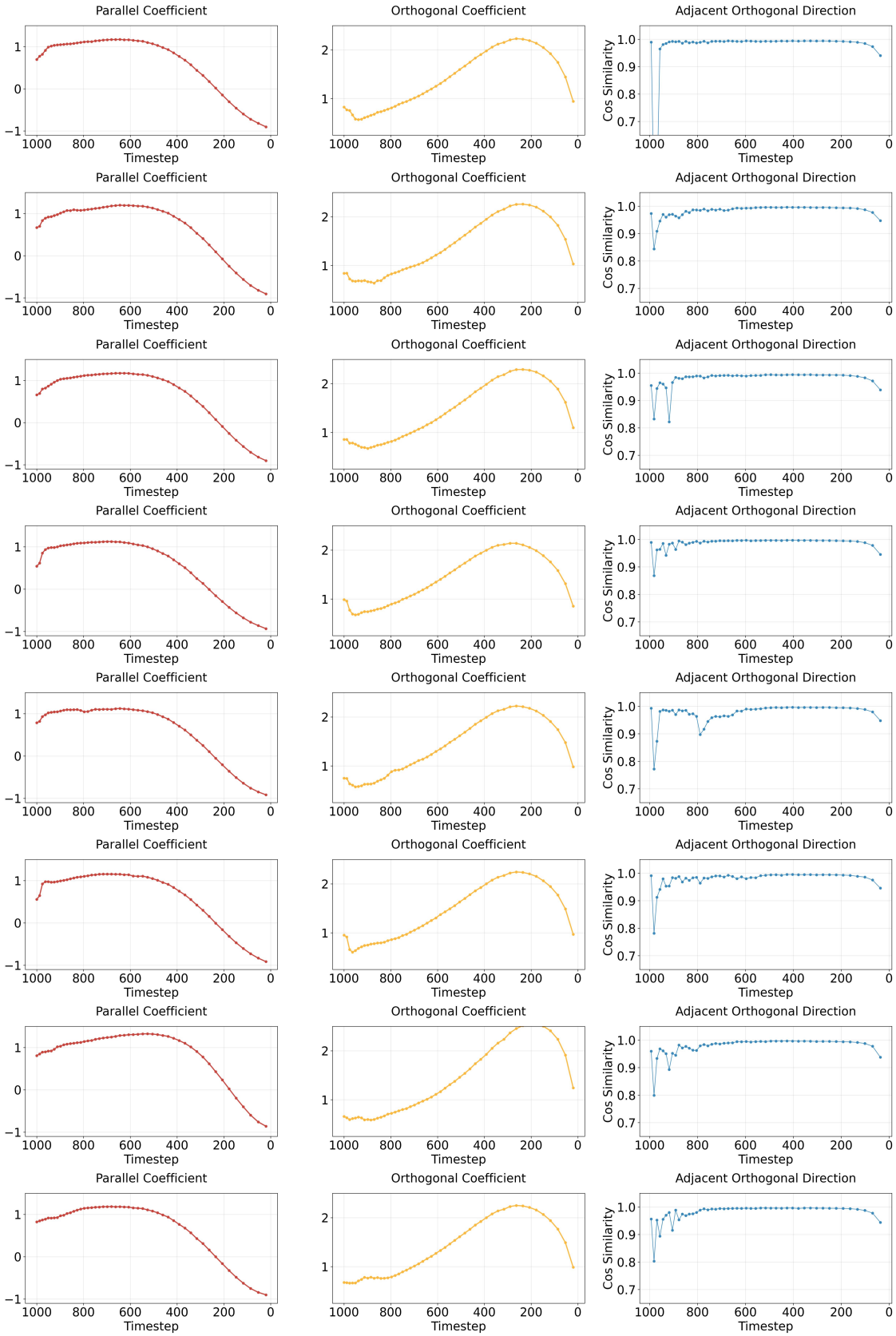

Figure L. Qwen-Image's Temporal dynamics of velocity components, including parallel coefficient, orthogonal coefficient and cosine similarity of adjacent orthogonal direction.

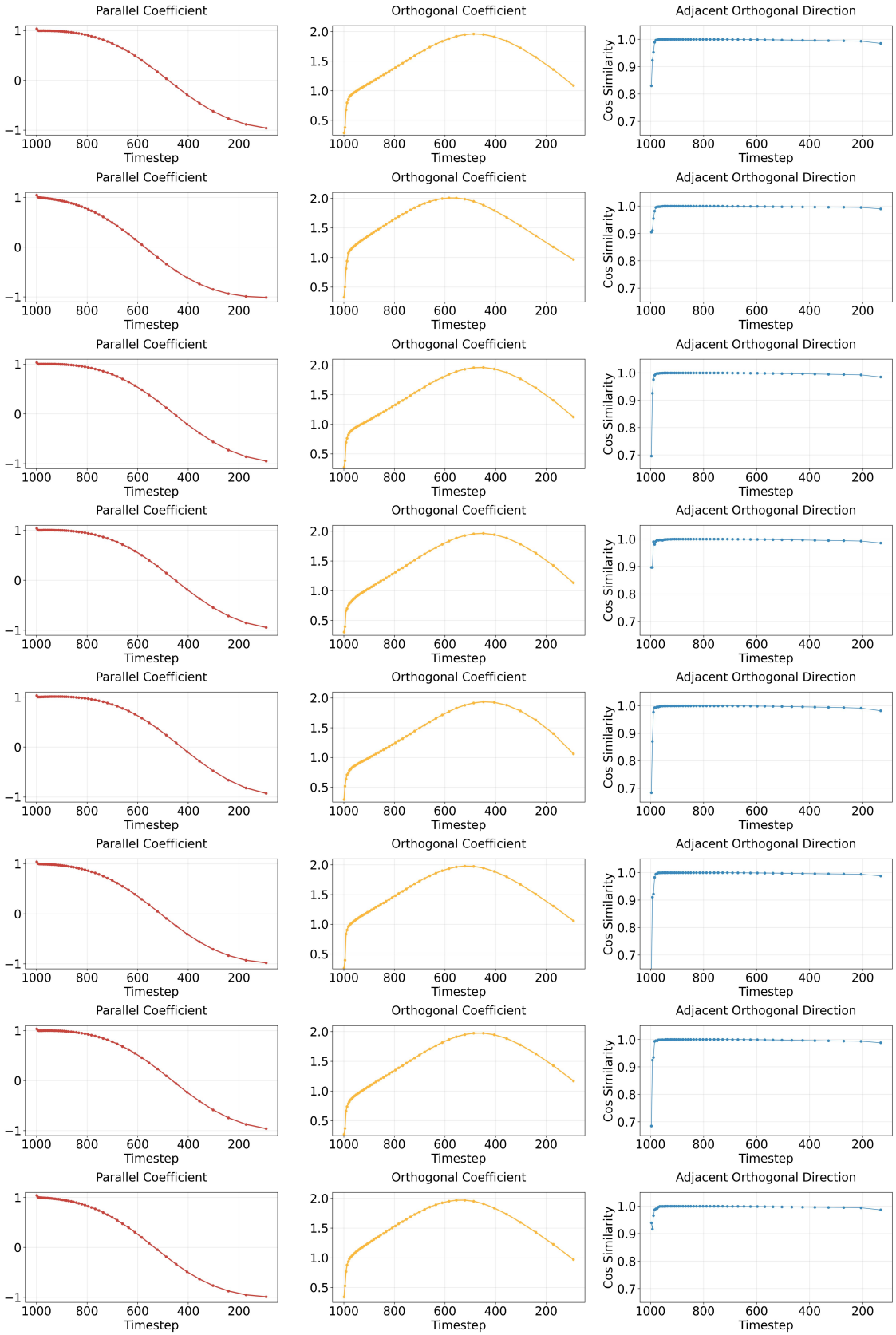

Figure M. Wan 2.1's Temporal dynamics of velocity components, including parallel coefficient, orthogonal coefficient and cosine similarity of adjacent orthogonal direction.

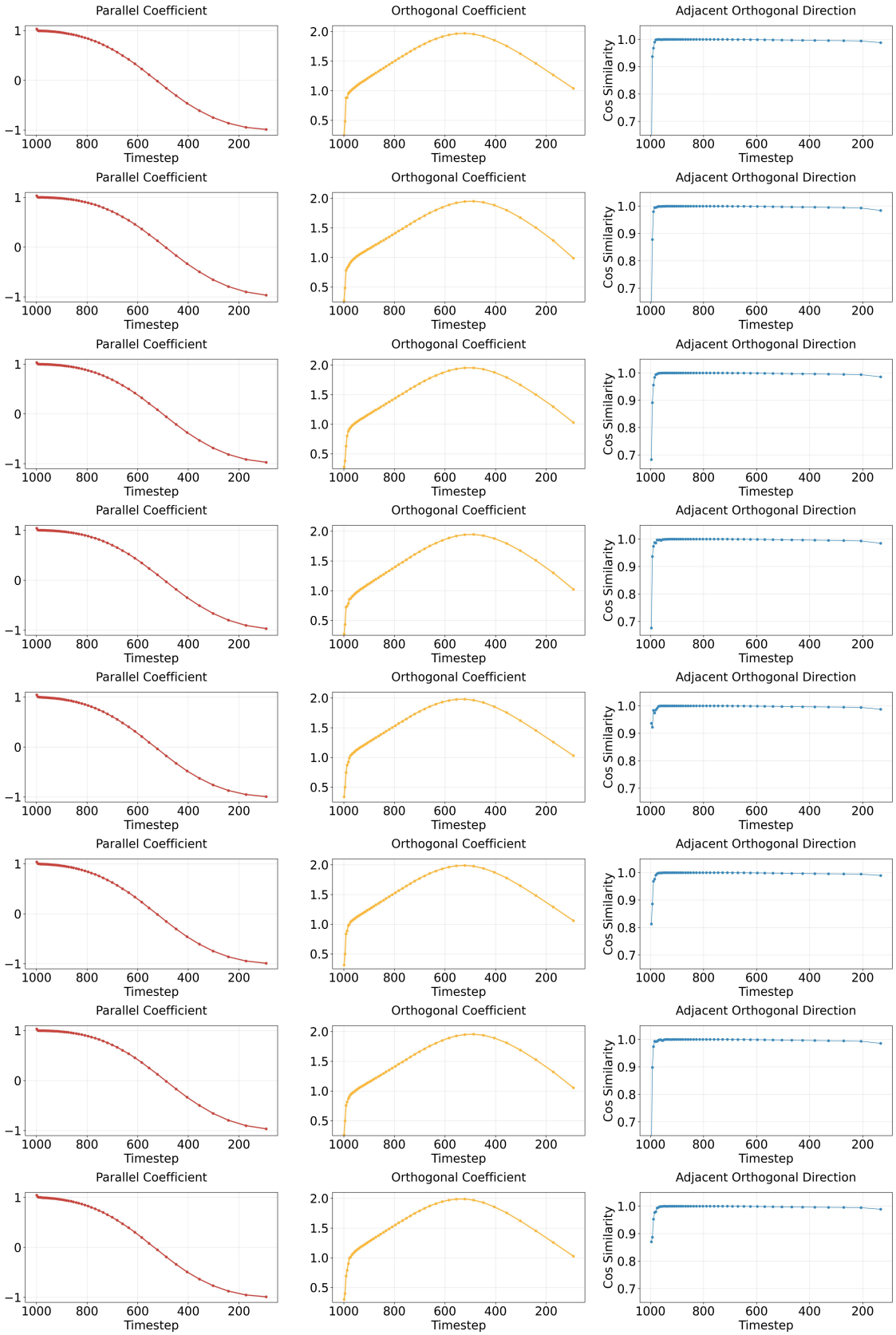

Figure N. Wan 2.1's Temporal dynamics of velocity components, including parallel coefficient, orthogonal coefficient and cosine similarity of adjacent orthogonal direction.

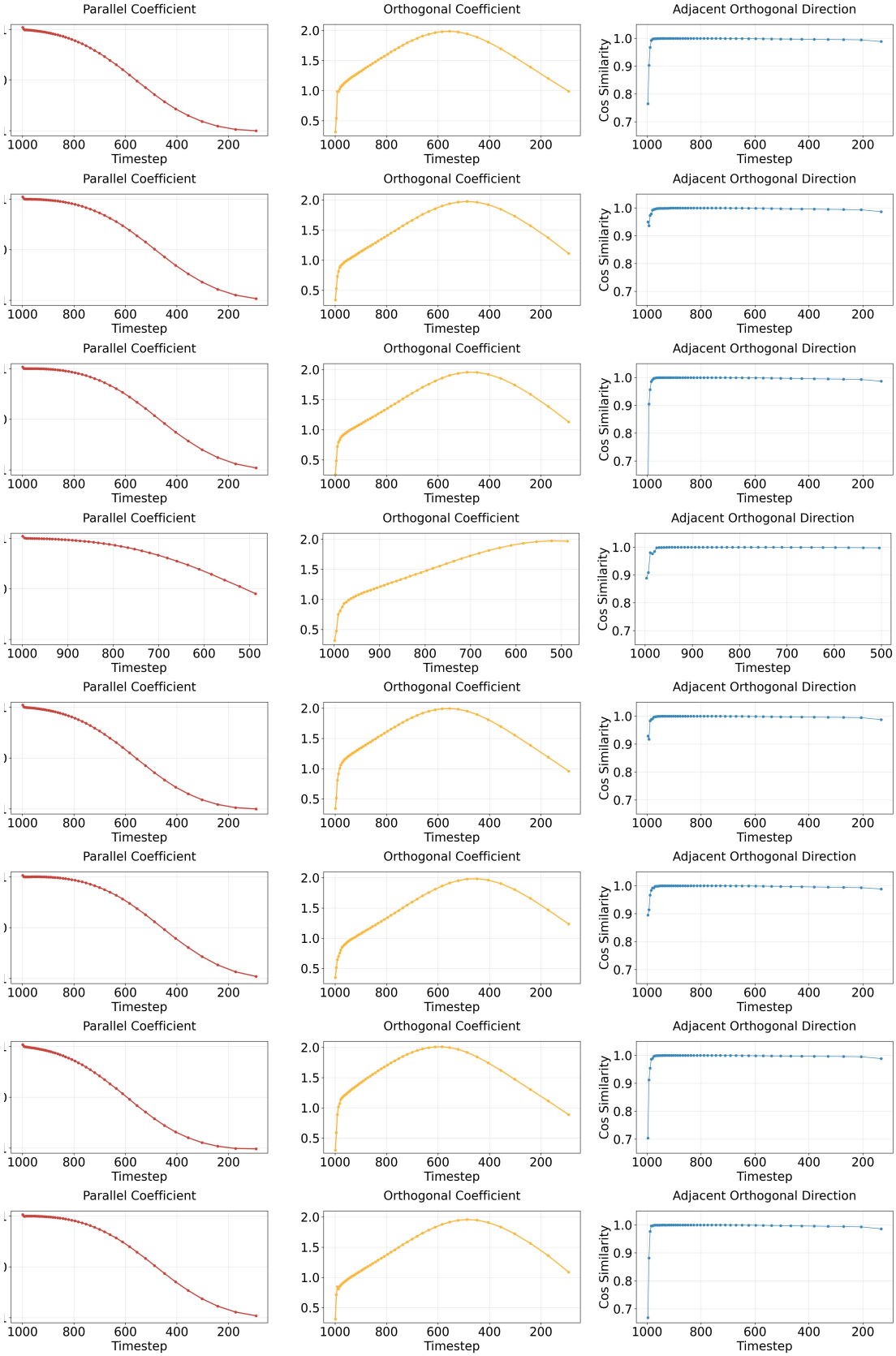

Figure O. Wan 2.1's Temporal dynamics of velocity components, including parallel coefficient, orthogonal coefficient and cosine similarity of adjacent orthogonal direction.

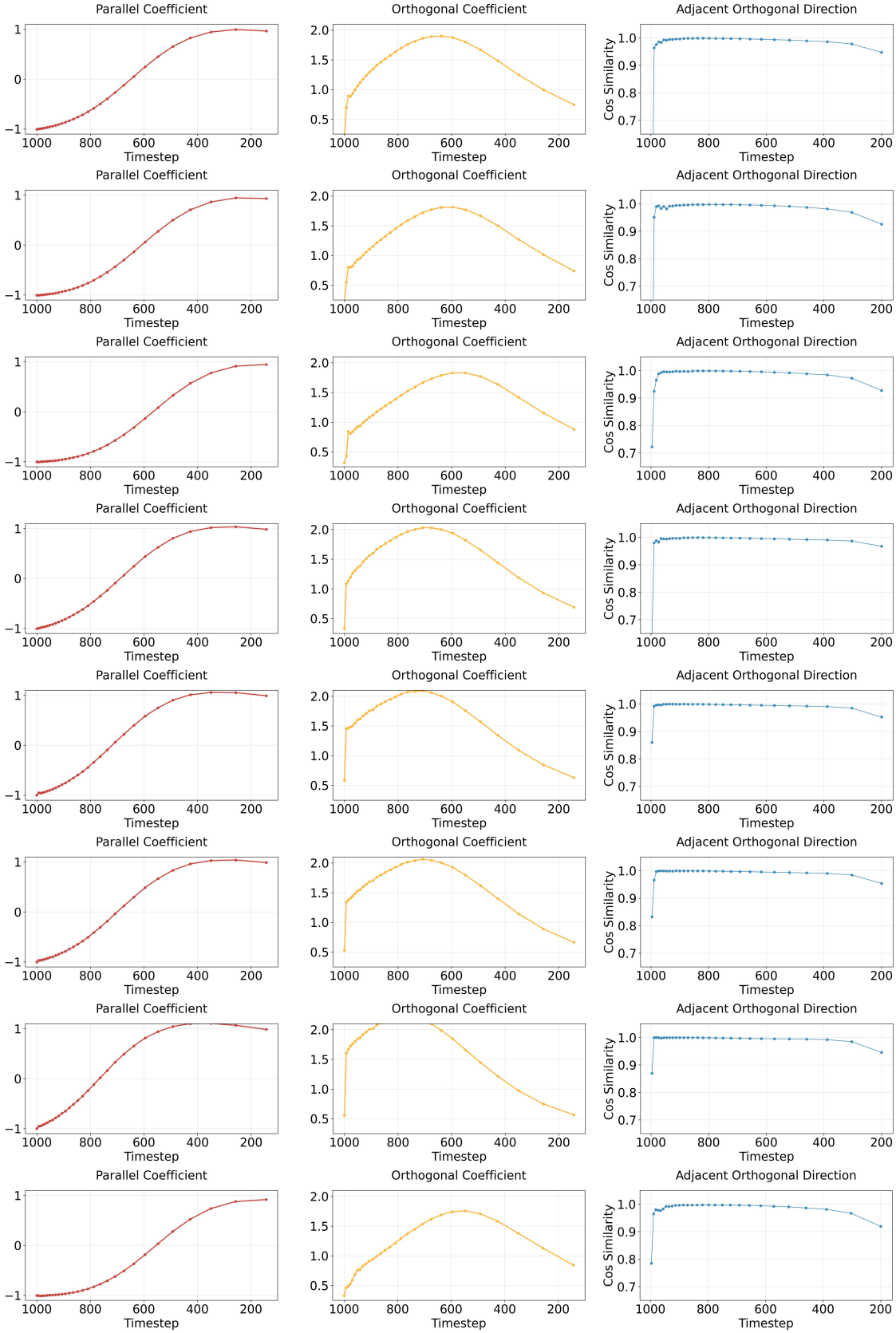

Figure P. Open-Sora 1.2's Temporal dynamics of velocity components, including parallel coefficient, orthogonal coefficient and cosine similarity of adjacent orthogonal direction.

References

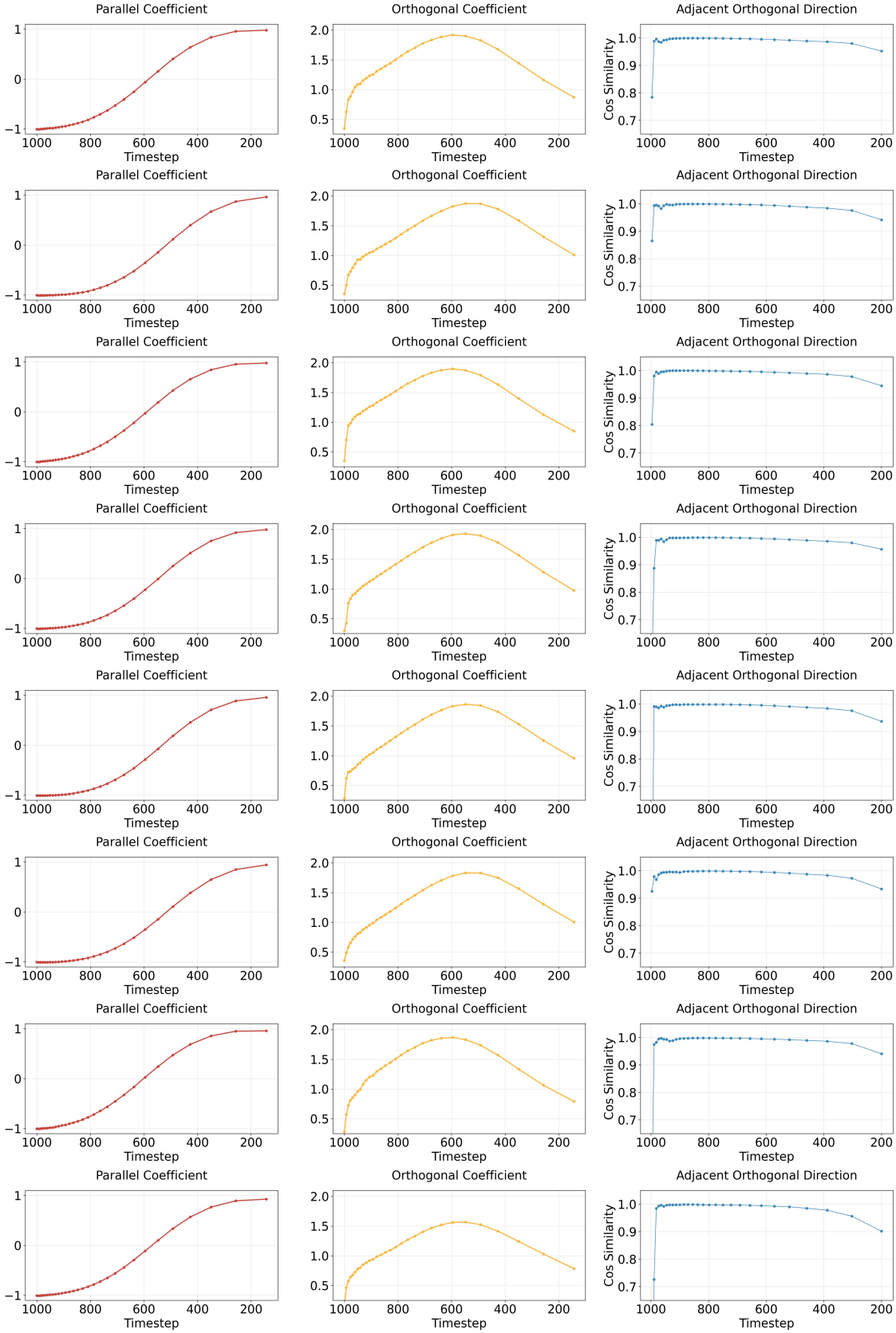

Figure Q. Open-Sora 1.2's Temporal dynamics of velocity components, including parallel coefficient, orthogonal coefficient and cosine similarity of adjacent orthogonal direction.

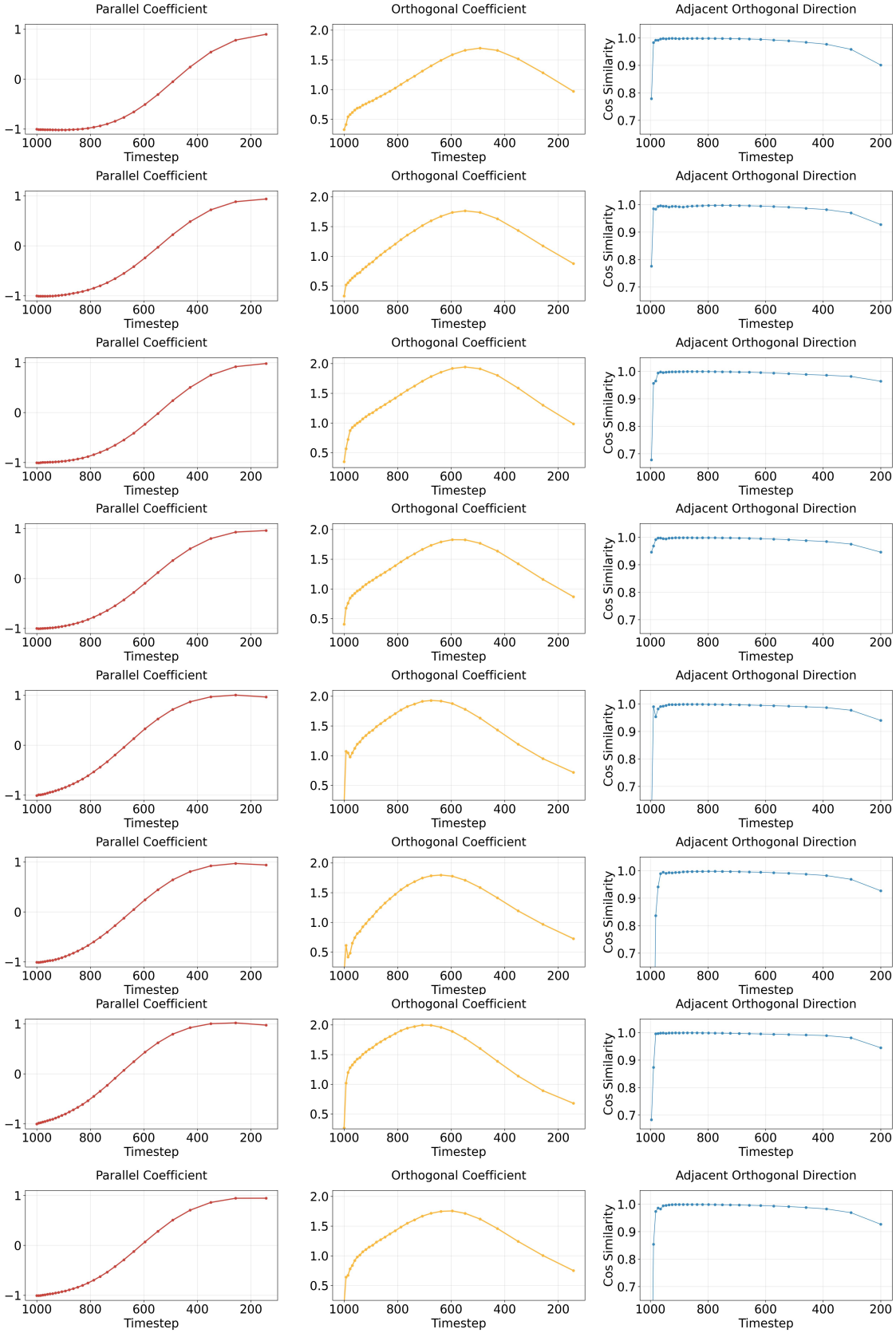

Figure R. Open-Sora 1.2's Temporal dynamics of velocity components, including parallel coefficient, orthogonal coefficient and cosine similarity of adjacent orthogonal direction.