CAMWAM: A WORLD-ACTION MODEL FOR AUTONOMOUS PHOTOGRAPHY

Anonymous authors
Paper under double-blind review



Figure 1: **Overview of CamWAM for autonomous photography.** CamWAM enables autonomous photography by predicting camera actions that transform an unfavorable initial view into a more desirable photographic outcome. The surrounding examples show camera trajectories and corresponding initial and final views across diverse 3D scenes.

ABSTRACT

Autonomous photography aims to enable embodied agents to capture visually appealing photographs without human camera operation, yet remains challenging due to the subjective nature of photographic objectives. Existing approaches typically determine camera actions without explicitly modeling their resulting visual outcomes, limiting their ability to reason about how camera motion affects photographic quality. In this work, we propose **CamWAM**, a world-action model for autonomous photography that jointly models future visual evolution and continuous camera motions. CamWAM cascades a video expert with an action expert through a lightweight adapter, allowing predicted visual evolution to provide informative representations for camera motion prediction. To effectively couple the two experts, we develop a three-stage video-action learning strategy that progressively aligns visual evolution with camera motion. We further introduce a scalable data pipeline that reverses videos synthesized from curated high-quality photographs to construct camera trajectories toward desirable views, while using independently constructed 3D Gaussian Splatting environments for cross-domain evaluation. Extensive experiments demonstrate the effectiveness of CamWAM and its generalization across scenarios. Our models, training code, and dataset will be publicly available.

1 INTRODUCTION

The rapid growth of social media has made photography an important medium for recording and sharing everyday experiences. Yet capturing an appealing photograph remains challenging, as it requires careful camera operation to achieve appropriate viewpoint selection, subject framing, and

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

composition (Hong et al., 2021; Lan & Sekiyama, 2019). Moreover, this conventional photography paradigm becomes impractical when photographers wish to include themselves in the scene.

Inspired by embodied intelligence, autonomous photography has emerged as a promising solution, with existing approaches mainly following two directions: active camera control and post-processing (Byers et al., 2003; AlZayer et al., 2021; Limoyo et al., 2024; Yao et al., 2026). The former controls camera actions based on current observations, while the latter refines captured images through techniques such as automatic cropping, composition adjustment, and generative photographic editing (Yang et al., 2023; Zhang et al., 2026; Zeng et al., 2026). Despite their progress, neither direction explicitly couples visual outcomes with camera motion, limiting the ability to reason about how a scene will evolve before an action is taken.

An intuitive idea is to introduce the recently popular world-action models (WAMs) to autonomous photography. Although WAMs have shown promising results in robotic grasping (Ye et al., 2026; Wang et al., 2026b), navigation (Chen et al., 2026), and driving (Zhao et al., 2025), their direct application to autonomous photography remains challenging due to its highly subjective objectives and lack of explicit target states. On one hand, training requires video clips capturing camera trajectories toward desirable viewpoints, which are difficult to collect at large scale since the quality of the final view cannot be guaranteed during real-world acquisition. On the other hand, the subjective nature of photographic objectives provides limited explicit supervision, making it challenging to model visual evolution and camera motion toward desirable views jointly.

In view of this, we propose **CamWAM**, a world-action model equipped with a novel data construction pipeline for autonomous photography. Specifically, CamWAM cascades a video expert for predicting future visual evolution with an action expert for camera motion prediction, bridged by a lightweight adapter that transforms intermediate visual representations into motion-relevant features. To effectively train CamWAM, we develop a three-stage video-action learning strategy that progressively aligns photographic visual evolution with camera motion by learning visual dynamics, establishing visual–motion correspondence, and distilling motion-relevant information through a lightweight adapter.

For data construction, we build the training and evaluation sets using video generation and 3D Gaussian Splatting (3DGS), respectively. For training, we take curated high-quality photographs as target frames and employ an image-to-video model to generate plausible preceding camera trajectories. By reversing the generated videos, we are able to obtain trajectories that progressively approach photographically desirable views, thereby providing scalable supervision for autonomous photography. For evaluation, we construct interactive environments using 3DGS, enabling cross-domain testing of CamWAM without relying on the video-generation pipeline used for training.

Last, extensive experiments demonstrate the effectiveness of CamWAM, showing that our proposed data pipeline and model generalize well across domains.

Our contributions are summarized as follows.

- We propose CamWAM, a world-action model for autonomous photography that jointly models future visual evolution and camera motions, enabling camera motion prediction to leverage anticipated visual evolution, together with a three-stage learning strategy that progressively aligns photographic visual evolution with camera motion.
- We develop a scalable data pipeline that reverses videos generated from curated high-quality photographs to construct camera trajectories toward desirable views, together with an independently constructed 3DGS photography benchmark for cross-domain evaluation.
- Extensive experiments across diverse scenarios validate the effectiveness of CamWAM and its generalization across domains.

2 Related Work

Autonomous Photography Existing autonomous photography approaches mainly follow two directions, including post-processing of captured images and active camera control. Post-processing methods improve photographic quality through automatic cropping, composition adjustment, and generative editing (Yang et al., 2023; Zhang et al., 2026; Zeng et al., 2026; Yao et al., 2026). While effective for refining captured images, they cannot change the physical camera viewpoint or recover

unseen content outside the current field of view. Active photography methods instead optimize camera viewpoints during capture. Early robotic systems rely on predefined routes and candidate selection (Byers et al., 2003), while AutoPhoto learns camera control with aesthetic rewards and reinforcement learning (AlZayer et al., 2021). Camera View Adjustment Prediction learns camera adjustment from supervision constructed using cropping annotations and pseudo-labels produced by a composition scoring model (Su et al., 2021). Other methods use reference-image matching or large vision-language models for viewpoint selection (Limoyo et al., 2024). Although these methods enable physical viewpoint adjustment, their camera actions are still largely guided by predefined photographic criteria, either directly through handcrafted objectives or indirectly through supervision constructed from such criteria. Both lines of work lack explicit modeling of how the observed scene is expected to evolve under camera motion and therefore cannot leverage future visual prediction to optimize photographic outcomes. Recently, Photography Perspective Composition (PPC) demonstrates that image-to-video models can predict visual evolution toward more aesthetically favorable compositions (Yao et al., 2025). However, PPC generates visual transitions without recovering the corresponding continuous camera actions. In contrast, CamWAM couples future visual prediction with camera action modeling for autonomous photography.

World Action Models World action models couple future visual prediction with action generation, leveraging anticipated visual evolution to provide additional context for physical control (Wang et al., 2026b). Recent studies have demonstrated the effectiveness of this paradigm in robotic manipulation (Ye et al., 2026; Liao et al., 2025), navigation (Chen et al., 2026), and self-driving (Zhao et al., 2025). However, existing WAMs are primarily developed for conventional robotic tasks, whereas autonomous photography poses a substantially different learning objective. In autonomous photography, camera actions are judged by the resulting framing, composition, and consistency with the instruction. Existing WAMs are therefore difficult to directly transfer to autonomous photography, as their training data and learning strategies are not designed to model the relationship between camera motion and photographic visual evolution. CamWAM addresses this gap with a photography-oriented data pipeline and a three-stage learning strategy for aligning visual evolution with camera motion.

3 Data and Benchmark Construction

This section describes how we construct the training data and evaluation benchmark for CamWAM. For training data, we transform curated high-quality photographs into photography-oriented video-action supervision. For evaluation, we construct an interactive 3DGS benchmark with realistic scenes and human subjects.

3.1 Camera Motion Training Data

Training a world action model for autonomous photography requires paired supervision of visual evolution and camera motion. To obtain scalable supervision, we construct training data by converting curated high-quality photographs into paired video-motion sequences, as illustrated in Fig. 2. We first collect portrait photographs from Internet sources and manually retain images with clear human subjects, favorable compositions, and diverse framing and viewpoints, while removing low-quality or unsuitable samples. Each selected image is treated as a desirable photographic endpoint, and Qwen3.6-27B (Qwen Team, 2026a) is used to generate a corresponding photographic instruction. Starting from this endpoint, we use Wan 2.7 (Wan et al., 2025) to synthesize a short video sequence in which the viewpoint gradually departs from the favorable composition. We then reverse the generated video in time, obtaining a new video that starts from a relatively poor viewpoint and progressively approaches the curated high-quality photograph. Based on these reversed videos, we further recover the corresponding camera poses using VGGT-Ω (Wang et al., 2026a) and derive frame-wise relative camera actions from the estimated trajectories. The resulting video, instruction, and camera actions provide aligned supervision for CamWAM. Finally, we manually remove samples with inconsistent visual content or unreliable pose estimates to obtain the final training set.

3.2 3DGS Photography Benchmark

To enable systematic evaluation of autonomous photography, we construct a closed-loop benchmark based on 3D Gaussian Splatting (3DGS), as illustrated in Fig. 3. To obtain suitable photography environments, we first construct two large-scale 3DGS scenes from real-world tourist attractions

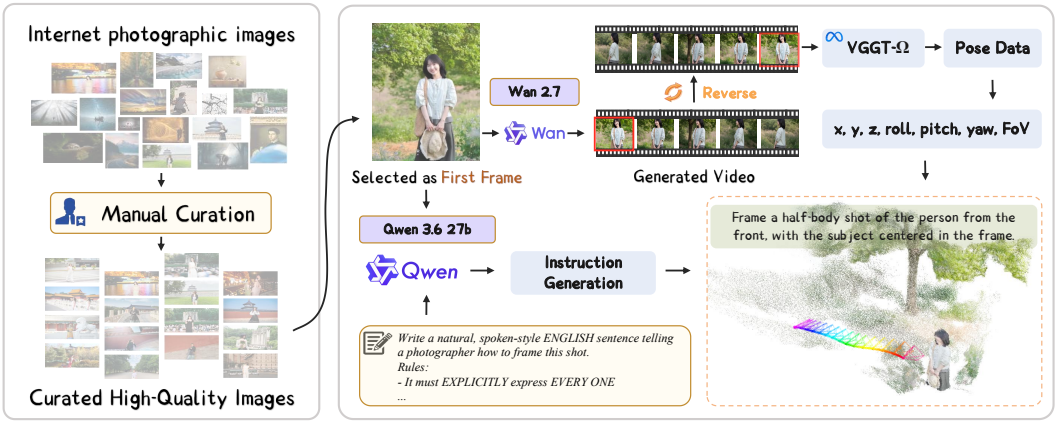


Figure 2: **Overview of the camera motion data construction pipeline.** We manually curate high-quality portrait photographs from Internet sources and use each selected image to generate a short video with Wan 2.7, which is then reversed to form a trajectory that progressively approaches the desired photographic viewpoint. VGGT-Ω recovers the corresponding camera poses and relative motion, while Qwen3.6-27B generates the photographic instruction associated with the target image.

and manually select 6 locations suitable for portrait photography. We then collect 28 human 3DGS assets, including subjects from THuman3.0 (Su et al., 2023) and assets generated using Hyper3D Rodin (Zhang et al., 2024), followed by manual screening for reliable rendering and portrait suitability. Based on these assets, we construct the evaluation cases by combining human subjects with the 3DGS environments and manually selecting challenging initial camera poses.

Specifically, for each of the photography locations, we select four unfavorable initial camera poses spanning diverse viewpoints, distances, elevations, and framings, and pair each pose with six photographic instructions. Each case is then instantiated with two different human subjects, resulting in 288 evaluation cases in total. Since valid photographic outcomes are not unique, the benchmark does not define a target camera pose or reference trajectory. All methods start from the same initial view and are evaluated in a closed loop, where each predicted action updates the camera state and the newly rendered view serves as the next observation. The final rendered view is used as the photographic result for evaluation. Further details on the training data and benchmark construction are provided in Appendix A.1 and Appendix A.2, respectively.

4 Method

4.1 Problem Formulation

Given an initial observation I_0 and a photographic instruction c, autonomous photography aims to predict a sequence of camera actions

$$\mathbf{A} = f_\theta(I_0, c), \qquad \mathbf{A} = \{\mathbf{a}_1, \ldots, \mathbf{a}_T\}, \tag{1}$$

where each $\mathbf{a}_t$ denotes the relative camera motion between consecutive time steps, including translation, rotation, and field-of-view change. Following the world action modeling paradigm, CamWAM also models the future visual evolution $\mathbf{V} = \{I_1, \ldots, I_T\}$ and exploits it to assist camera action prediction. Accordingly, we factorize the prediction process as

$$p(\mathbf{A}, \mathbf{V} \mid I_0, c) = p(\mathbf{V} \mid I_0, c)\, p(\mathbf{A} \mid I_0, c, \mathbf{V}), \tag{2}$$

where $p(\mathbf{V} \mid I_0, c)$ models the future visual sequence conditioned on the observation and instruction, and $p(\mathbf{A} \mid I_0, c, \mathbf{V})$ models the corresponding actions with the future visual evolution as additional context. In this formulation, future visual prediction provides useful context for action prediction, and the process is repeated with updated observations during closed-loop execution.

4.2 CamWAM Architecture

CamWAM adopts a cascaded video-action architecture consisting of a video expert, an action expert, and a lightweight camera motion adapter, as illustrated in Fig. 4. Both experts are conditioned



Figure 3: **Construction of the 3DGS photography benchmark.** We combine 28 human 3DGS assets with two large-scale 3DGS environments reconstructed from real-world tourist attractions. Six suitable photography locations are manually selected across the environments, as indicated by the red markers. Human assets are then combined with these selected locations, and corresponding initial camera views are manually chosen to construct diverse closed-loop portrait photography evaluation cases.

on the initial observation I_0 and photographic instruction c. The video expert is built on a pretrained video generation model (Wan et al., 2025) to predict future visual evolution, while the action expert uses a similar DiT architecture to infer camera motion from its intermediate representations. The two experts have separate parameters and interact through asymmetric attention: the action expert can attend to video features, while no action features are fed back to the video expert. This one-way interaction prevents the randomly initialized action expert from interfering with the pretrained video expert and supports their separate training. Since the intermediate video representations may contain motion-irrelevant information and can deviate from clean training representations during inference due to imperfect video denoising, the camera motion adapter transforms them before action prediction to emphasize motion-relevant cues and improve the robustness of camera-action prediction to representation variations. Detailed configurations are provided in the Appendix B.

4.3 Training Strategy

As illustrated in Fig. 4, CamWAM is trained in three stages dedicated to the video expert, action expert, and camera motion adapter, respectively. The first two stages establish visual prediction and camera action prediction, while the third stage further aligns the intermediate video representations with camera motion prediction.

Stage 1 Video Expert Training. We first train the video expert to predict future visual evolution from the initial observation I_0 and photographic instruction c using a standard flow matching objective. Let $\mathbf{x}_0^{\text{vid}}$ denote the clean video latent and $\boldsymbol{\epsilon}^{\text{vid}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. For a sampled timestep τ_{vid},

$$\mathbf{x}_{\tau_{\text{vid}}}^{\text{vid}} = \left(1 - \sigma_{\tau_{\text{vid}}}^{\text{vid}}\right) \mathbf{x}_0^{\text{vid}} + \sigma_{\tau_{\text{vid}}}^{\text{vid}} \boldsymbol{\epsilon}^{\text{vid}}, \qquad \mathbf{v}^{\text{vid}} = \boldsymbol{\epsilon}^{\text{vid}} - \mathbf{x}_0^{\text{vid}}. \tag{3}$$

The video expert predicts $\hat{\mathbf{v}}^{\text{vid}}$ and is optimized with

$$\mathcal{L}_{\text{vid}} = \mathbb{E}\left[w_{\text{vid}}(\tau_{\text{vid}}) \left\| \hat{\mathbf{v}}^{\text{vid}} - \mathbf{v}^{\text{vid}} \right\|_2^2 \right]. \tag{4}$$

This stage equips the video expert with the predictive capability required for subsequent camera action learning.

Stage 2 Action Expert Training. We then freeze the video expert and train the action expert to predict camera motion from its intermediate representations. Let $\mathbf{x}_0^{\text{act}} \in \mathbb{R}^{H \times 7}$ denote the clean camera action sequence and $\boldsymbol{\epsilon}^{\text{act}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Similarly,

$$\mathbf{x}_{\tau_{\text{act}}}^{\text{act}} = \left(1 - \sigma_{\tau_{\text{act}}}^{\text{act}}\right) \mathbf{x}_0^{\text{act}} + \sigma_{\tau_{\text{act}}}^{\text{act}} \boldsymbol{\epsilon}^{\text{act}}, \qquad \mathbf{v}^{\text{act}} = \boldsymbol{\epsilon}^{\text{act}} - \mathbf{x}_0^{\text{act}}. \tag{5}$$

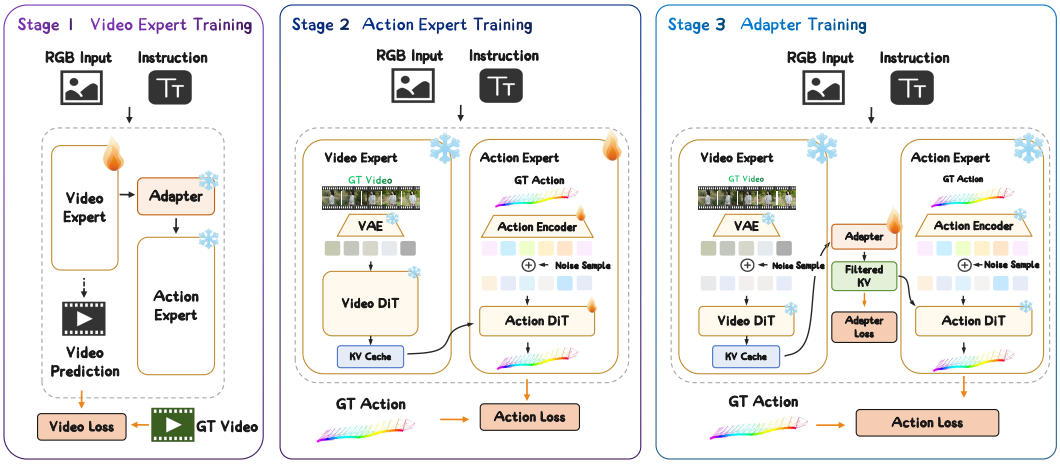


Figure 4: **Overview of the three-stage training strategy of CamWAM.** Stage 1 trains the video expert to predict future visual evolution from the current observation and instruction. Stage 2 freezes the video expert and trains the action expert to infer camera motion from video representations. Stage 3 trains the camera motion adapter with noise-augmented video conditioning and joint adapter and action supervision to improve alignment and robustness.

The action expert is optimized by

$$\mathcal{L}_{\text{act}} = \mathbb{E}\left[w_{\text{act}}(\tau_{\text{act}}) \sum_{h=1}^{H} \sum_{d=1}^{7} \lambda_d \left\| \hat{\mathbf{v}}_{h,d}^{\text{act}} - \mathbf{v}_{h,d}^{\text{act}} \right\|_2^2 \right], \tag{6}$$

where H is the action horizon and λ_d balances different action dimensions. This stage establishes the correspondence between visual evolution and camera motion.

Stage 3 Camera Motion Adapter Training. Stage 2 uses representations derived from ground-truth videos, whereas inference relies on representations produced by predicted videos. To improve robustness to this discrepancy, we freeze both experts and train the camera motion adapter with conditioning noise augmentation. Specifically, we perturb the clean video latent as

$$\tilde{\mathbf{x}}_0^{\text{vid}} = (1 - \sigma_{\text{cond}})\mathbf{x}_0^{\text{vid}} + \sigma_{\text{cond}}\boldsymbol{\epsilon}^{\text{cond}}, \qquad \sigma_{\text{cond}} \sim \mathcal{U}(0, \sigma_{\max}), \tag{7}$$

where $\boldsymbol{\epsilon}^{\text{cond}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. This introduces controlled representation variations while preserving the underlying visual evolution.

To encourage the adapter to suppress nuisance variations while preserving motion-relevant differences, let $g(\cdot)$ denote the adapter and define

$$\rho(\mathbf{h}_i, \mathbf{h}_j) = \frac{\|g(\mathbf{h}_i) - g(\mathbf{h}_j)\|_2}{\|\mathbf{h}_i - \mathbf{h}_j\|_2}, \tag{8}$$

where h_i and h_j denote intermediate video representations obtained from the frozen video expert. We construct nuisance pairs from the same video under different conditioning noise and content pairs from different videos, with corresponding ratios ρ_{nui} and ρ_{con}. The adapter loss is

$$\mathcal{L}_{\text{adapter}} = \rho_{\text{nui}}^2 + \beta\,(\rho_{\text{con}} - 1)^2\,. \tag{9}$$

The overall Stage 3 objective further retains the action loss,

$$\mathcal{L}_{\text{S3}} = \lambda_{\text{act}}\mathcal{L}_{\text{act}} + \lambda_{\text{adapter}}\mathcal{L}_{\text{adapter}}, \tag{10}$$

so that the adapted representations are both robust to nuisance variations and informative for camera motion prediction. More details of the training and implementation are provided in the Appendix C.

5 Experiments

5.1 Baselines

We evaluate CamWAM at three video-expert scales (1.3B, 5B, and 14B) and compare them with four families of baselines. All methods are evaluated from the same initial poses and 3DGS environments as illustrated in Section 3.2, while text-conditioned methods receive the same instructions.

Composition methods. We evaluate VAPNet (Su et al., 2021), UNIC (Liu et al., 2023), and PhotoFramer (You et al., 2026), which predict view adjustments for improving image composition.

Camera-motion methods. We compare with AutoPhoto (AlZayer et al., 2021), which performs sequential camera control, and GenDoP (Zhang et al., 2025), which predicts cinematographic camera trajectories from visual and language conditions. For GenDoP, we evaluate both a text-conditioned variant and an RGB-D variant, both trained on our data.

Direct VLM agents. We evaluate four recent general-purpose multimodal models, Qwen-3.8-Max (Qwen Team, 2026b), Gemini-2.5-Pro (Gemini Team, Google, 2025), GPT-5.6-Sol (OpenAI, 2026), and GLM-5V-Turbo (V Team et al., 2026), as zero-shot camera-control agents. At each step, the agent receives the current RGB observation, photographic instruction, and action history, and predicts either a camera action $\mathbf{a}_t$ or `STOP`.

Vision-language-action methods. We first compare with $\pi_{0.5}$ (Physical Intelligence et al., 2025), a representative general-purpose VLA model. To further examine the effect of VLM capacity, we additionally adopt the StarVLA (Community, 2026) framework with VLM backbones of 0.8B, 2B, and 4B parameters. All VLA baselines are trained on our photography data.

Table 1: Evaluator repeatability and human alignment on 300 rollout samples. VLM Average denotes evaluation based on the averaged scores of Qwen3.8-Max and Gemini-2.5-Pro.

	Repeatability Q / I			Human Alignment Q / I
Evaluator	Repeat $\rho \uparrow$	ICC(A,1) $\uparrow$	Run-mean Range $\downarrow$	Human $\rho \uparrow$
Qwen3.8-Max	0.913 / 0.931	0.938 / 0.849	1.20 / 0.90	0.773±0.034 / 0.783±0.036
Gemini-2.5-Pro	0.955 / 0.934	0.970 / 0.929	0.73 / 0.85	0.815±0.025 / 0.741±0.037
VLM Average	0.929 / 0.934	0.954 / 0.936	**0.72 / 0.57**	**0.820±0.029 / 0.810±0.034**
Human–Human		*N/A*		0.850 / 0.827

5.2 Evaluation Metrics

Since autonomous photography admits multiple valid trajectories, we evaluate the final captured image rather than its distance to a ground-truth trajectory. We use two VLM-based evaluators (Qwen Team, 2026b; Gemini Team, Google, 2025), each taking the initial view, final captured image, and photographic instruction as input. Each evaluator outputs an instruction-following score $I \in [0, 100]$ and a photographic-quality score $Q \in [0, 100]$. We average the scores from the two evaluators and use I and Q to characterize the two key aspects of photography performance.

Photographic improvement. To account for different initial-view quality, we measure the photographic improvement as

$$\Delta Q_i = Q(I_T^i) - \bar{Q}_0^i, \tag{11}$$

where $\bar{Q}_0^i$ is the photographic-quality score of the initial view. We report the mean ΔQ over all episodes.

Photography success score. To jointly measure instruction following and photographic quality, we define

$$S_i = I_i \cdot Q_i/100, \qquad \text{PSS} = \frac{1}{N}\sum_{i=1}^{N} S_i, \tag{12}$$

where a high PSS requires both strong instruction following and high photographic quality.

Action–video consistency. For CamWAM, we additionally evaluate the consistency between the predicted future video and the visual sequence obtained by executing the predicted actions:

$$\text{AVC} = \frac{1}{T}\sum_{t=1}^{T} \text{LPIPS}\left(\hat{I}_t, I_t^{\text{exec}}\right), \tag{13}$$

where lower AVC indicates better action–video consistency.

5.3 Evaluator Validation

Since our benchmark relies on VLM-based assessment, we first validate the repeatability of the evaluators and their alignment with human judgments using 300 episodes sampled from the rollout

Table 2: Performance comparison on our 3DGS autonomous photography benchmark. We report the mean results of the two evaluators. For instruction following (I) and photographic quality (Q), we report the raw score together with the percentage of samples scoring at least 65 and below 30.

Method	Instruction Following			Photographic Quality				PSS ↑
	I ↑	$I \geq 65$ (%) ↑	$I < 30$ (%) ↓	Q ↑	ΔQ ↑	$Q \geq 65$ (%) ↑	$Q < 30$ (%) ↓	
Human Reference								
Human	75.75	74.7	1.0	69.77	+29.74	62.8	0	54.72
Composition methods								
VAPNet	–	–	–	40.04	+0.01	0.0	7.6	*N/A*
UNIC	–	–	–	40.90	+0.87	0.7	5.9	*N/A*
PhotoFramer	–	–	–	38.83	-1.20	0.7	14.9	*N/A*
Camera-motion methods								
AutoPhoto	–	–	–	24.42	-15.61	0.0	50.3	*N/A*
GenDoP-Text	25.41	3.8	66.7	35.89	-4.14	3.8	20.1	10.47
GenDoP-RGBD	19.22	1.0	78.1	28.91	-11.12	0.0	44.4	6.79
Direct VLM agents								
Qwen-3.8-Max	38.56	16.4	35.2	44.03	+3.96	21.3	31.4	22.94
Gemini-2.5-Pro	31.59	9.1	48.8	40.19	+0.13	14.6	32.4	17.14
GLM-5V-Turbo	32.88	13.2	46.3	36.14	-3.95	11.8	42.9	18.69
GPT-5.6-Sol	45.90	27.6	25.8	49.62	+9.41	29.8	21.8	28.70
Vision-language-action methods								
$\pi_{0.5}$	37.85	12.5	31.9	47.57	+7.54	25.7	22.2	23.09
StarVLA-0.8B	38.68	16.7	34.4	49.05	+9.02	31.6	21.9	24.07
StarVLA-2B	40.02	13.2	29.5	52.41	+12.39	36.1	16.0	24.78
StarVLA-4B	42.36	17.0	25.7	55.37	+15.34	39.9	14.6	27.37
CamWAM-1.3B	45.59	19.8	20.8	55.34	+15.31	45.1	17.7	29.20
CamWAM-5B	47.10	20.8	18.1	57.70	+17.67	43.4	10.1	30.31
CamWAM-14B	**50.77**	**29.9**	**17.7**	**59.13**	**+19.10**	**50.7**	**6.6**	**33.09**

results of all evaluated methods. Each VLM evaluator scores every episode five times under the same protocol as the main evaluation. We measure repeatability using the average pairwise Spearman correlation across runs for ranking consistency, ICC(A,1) (McGraw & Wong, 1996) for absolute score agreement, and the range of dataset-level mean scores for aggregate drift.

For human alignment, five annotators independently score Q and I using the same criteria. For each evaluator, we compute the Spearman correlation with each annotator over the five runs and report the average over all VLM–human pairs, together with the mean correlation among human annotators as a reference. As shown in Table 1, the evaluators show high ranking and score consistency with little aggregate variation, while their correlations with human judgments are close to human–human agreement, supporting their use for large-scale benchmark evaluation. Additional evaluator analyses and the prompts used for evaluation are provided in Appendix D.2.

5.4 Performance Comparison

Table 2 compares CamWAM with four families of baselines. For reference, we additionally evaluate a human operator who manually adjusts the camera from the same initial views according to the given instructions, providing a human-selected viewpoint reference for all benchmark cases. Methods without text input are not evaluated for instruction following or PSS.

The baseline families exhibit different strengths, with GPT-5.6-Sol performing particularly well in instruction following, and StarVLA-4B achieving stronger photographic quality among baselines. Notably, CamWAM-1.3B already achieves competitive overall performance, reaching photographic quality comparable to StarVLA-4B and a PSS of 29.20, on par with strong general-purpose VLM baselines despite using only a 1.3B video expert. Larger CamWAM variants further improve the overall results. CamWAM-5B reaches a PSS of 30.31, while CamWAM-14B achieves 33.09 together with the highest I, Q, and ΔQ among the evaluated methods. The proportion of low-Q outcomes also decreases from 17.7% at 1.3B to 10.1% at 5B and 6.6% at 14B. These results show that CamWAM remains competitive even at smaller model scales, while increasing model capacity further improves instruction following and photographic quality. A paired bootstrap analysis further confirms the robustness of these comparisons, as detailed in Appendix D.3. Figure 5 further provides qualitative comparisons of the final views produced by representative methods under the same initial observations and photographic instructions. Additional qualitative visualizations and representative failure cases are provided in Appendix D.5.



Figure 5: Qualitative comparison on the 3DGS photography benchmark. Given the same initial view and photographic instruction, we compare the final views produced by different methods. CamWAM achieves more instruction-aligned and favorable photographic results across diverse scenarios.

Table 3: Ablation study of model components and training strategy. Results are averaged over the two VLM evaluators.

Scale	Training	Instruction Following			Photographic Quality				PSS ↑	AVC ↓
		$I\uparrow$	$I \geq 65\,(\%)\uparrow$	$I < 30\,(\%)\downarrow$	$Q\uparrow$	$\Delta Q\uparrow$	$Q \geq 65\,(\%)\uparrow$	$Q < 30\,(\%)\downarrow$		
	Action-only	33.57	11.5	42.7	43.44	+3.42	33.6	33.2	21.08	*N/A*
1.3B	*Joint*	40.44	15.3	28.8	49.76	+9.73	29.2	18.8	25.09	0.51
	S1+S2	43.65	19.1	26.4	53.36	+13.33	42.7	20.1	28.29	0.42
	S1+S2+S3	**45.59**	**19.8**	**20.8**	**55.34**	**+15.31**	**45.1**	**17.7**	**29.20**	**0.41**
5B	*Joint*	36.74	12.5	35.4	46.44	+6.41	29.2	22.6	21.93	0.44
	S1+S2	45.98	22.0	21.6	56.63	+16.55	41.1	11.1	29.42	0.39
	S1+S2+S3	**47.10**	20.8	**18.1**	**57.70**	**+17.67**	**43.4**	**10.1**	**30.31**	**0.39**
14B	*Joint*	40.96	13.5	29.5	51.67	+11.51	17.7	8.3	23.07	0.55
	S1+S2	47.15	21.5	22.6	57.07	+17.04	46.9	9.4	30.12	0.46
	S1+S2+S3	**50.77**	**29.9**	**17.7**	**59.13**	**+19.10**	**50.7**	**6.6**	**33.09**	**0.45**

5.5 Ablation Studies

Table 3 evaluates the effects of future visual prediction and the proposed staged training strategy. Specifically, we compare four training variants. *Action-only* uses the same 1.3B Wan visual backbone to encode only the current observation, without future visual prediction. *Joint* optimizes the video and action experts simultaneously, while *S1+S2* trains the two experts sequentially. *S1+S2+S3* further introduces the camera motion adapter in Stage 3. At the 1.3B scale, *Joint* improves over *Action-only* in both instruction following and photographic quality, showing the benefit of future visual prediction. Across all three model scales, *S1+S2* further improves over *Joint* while consistently reducing AVC, suggesting better alignment between predicted visual evolution and camera actions. Stage 3 maintains or slightly improves AVC while further increasing I, Q, and PSS across all scales. The improvement is most pronounced for CamWAM-14B, where I and Q increase by 3.62 and 2.06 points, respectively, and PSS rises from 30.12 to 33.09. Overall, these results validate the effectiveness of the proposed CamWAM design. Inference efficiency is further analyzed in Appendix E.

6 Conclusion

We present CamWAM, a world-action model for autonomous photography that predicts camera motion by leveraging explicitly modeled future visual evolution. To support this task, we develop a scalable video-action data construction pipeline and an independent 3DGS benchmark for closed-loop photography evaluation. Experiments on our closed-loop benchmark show that CamWAM achieves stronger overall performance than the evaluated baselines. Current limitations include system latency and a remaining gap to human performance. Future work will improve deployment efficiency and further scale the data and benchmark.

AI USE STATEMENT

Large Language Models (LLMs) were used to assist manuscript preparation, limited to language refinement and grammar checking. Specifically, ChatGPT and Qwen3.8-Max were used for grammar checking and language polishing. All AI-assisted modifications were reviewed and verified by the authors, who take full responsibility for the final content of this work, including all text, claims, and artifacts.

In addition, pretrained generative models were used as tools in the data construction and evaluation pipelines described in the paper. Wan 2.7 was used to synthesize videos from curated photographs, and Qwen3.6-27B was used with predefined prompts to generate photographic instructions for the training data. Qwen3.8-Max and Gemini-2.5-Pro were used as fixed VLM evaluators with predefined evaluation prompts to assess instruction following and photographic quality. These outputs serve as training annotations, synthesized supervision, or evaluation signals; all scientific claims, analyses, and conclusions are made and verified by the authors.

ETHICS STATEMENT

This work focuses on autonomous photography. Internet photographs are used solely as source images for constructing photography-oriented training supervision. They are not used for identity recognition, biometric analysis, profiling, or any other analysis of personal attributes. The human evaluation is limited to scoring benchmark outputs for photographic quality and instruction following. No sensitive personal information is collected or used in this process. Human 3DGS assets are obtained from existing research datasets or generated assets and are used only to construct synthetic portrait photography scenarios. The use of these data is limited to the research purposes described above. Accordingly, we do not identify any material ethical risks associated with the experiments conducted in this work.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the data construction and 3DGS photography benchmark in Section 3 and Appendix A, including video synthesis, pose recovery, trajectory processing and benchmark case construction. The CamWAM architecture and three-stage training strategy are described in Section 4, with model configurations and parameter counts provided in Appendix B and training and closed-loop implementation details in Appendix C. We will release the trained models, code, and constructed dataset to facilitate reproduction and future research.

REFERENCES

Hadi AlZayer, Hubert Lin, and Kavita Bala. Autophoto: Aesthetic photo capture using reinforcement learning. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 944–951. IEEE, 2021. 2, 3, 7, 18

Zachary Byers, Michael Dixon, Kevin Goodier, C.M. Kavitha, and William D. Smart. An autonomous robot photographer. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2003. 2, 3

Hong Chen, Daqi Liu, Zehan Zhang, Haiguang Wang, Tianhao Lu, Longfei Yan, Haiyang Sun, Fangzhen Li, Hongwei Xie, Bing Wang, et al. Pondering the way: Spatial-perceiving world action model for embodied navigation. *arXiv preprint arXiv:2606.29908*, 2026. 2, 3

StarVLA Community. Starvla: A lego-like codebase for vision-language-action model developing. *arXiv preprint arXiv:2604.05014*, 2026. 7, 18, 21

Gemini Team, Google. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. Technical report, Google DeepMind, 2025. 7, 20

Chaoyi Hong, Shuaiyuan Du, Ke Xian, Hao Lu, Zhiguo Cao, and Weicai Zhong. Composing photos like a photographer. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7057–7066. IEEE, 2021. 2

Kai Lan and Kosuke Sekiyama. Autonomous robot photographer with kl divergence optimization of image composition and human facial direction. *Robotics and Autonomous Systems*, 111:132–144, 2019. 2

Yue Liao, Pengfei Zhou, Siyuan Huang, Donglin Yang, Shengcong Chen, Yuxin Jiang, Yue Hu, Jingbin Cai, Si Liu, Jianlan Luo, Liliang Chen, Shuicheng Yan, Maoqing Yao, and Guanghui Ren. Genie envisioner: A unified world foundation platform for robotic manipulation. *arXiv preprint arXiv:2508.05635*, 2025. 3

Oliver Limoyo, Jimmy Li, Dmitriy Rivkin, Jonathan Kelly, and Gregory Dudek. Photobot: Reference-guided interactive photography via natural language. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2479–2486. IEEE, 2024. 2, 3

Xiaoyu Liu, Ming Liu, Junyi Li, Shuai Liu, Xiaotao Wang, Lei Lei, and Wangmeng Zuo. Beyond image borders: Learning feature extrapolation for unbounded image composition. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 13023–13032. IEEE, 2023. 7, 18

Kenneth O McGraw and Seok P Wong. Forming inferences about some intraclass correlation coefficients. *Psychological methods*, 1(1):30, 1996. 8

OpenAI. Gpt-5.6 system card, July 2026. URL https://deploymentsafety.openai.com/gpt-5-6. 7, 21

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025. 7, 18

Qwen Team. Qwen3.6-27B: Flagship-level coding in a 27B dense model, April 2026a. URL https://qwen.ai/blog?id=qwen3.6-27b. 3

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork, August 2026b. URL https://qwen.ai/blog?id=qwen3.8. 7, 20

Yu-Chuan Su, Raviteja Vemulapalli, Ben Weiss, Chun-Te Chu, Philip Andrew Mansfield, Lior Shapira, and Colvin Pitts. Camera view adjustment prediction for improving image composition. *arXiv preprint arXiv:2104.07608*, 2021. 3, 7, 18

Zhaoqi Su, Tao Yu, Yangang Wang, and Yebin Liu. Deepcloth: Neural garment representation for shape and style editing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2): 1581–1593, 2023. doi: 10.1109/TPAMI.2022.3168569. 4, 16

V Team, Wenyi Hong, Xiaotao Gu, Ziyang Pan, Zhen Yang, Yuting Wang, Yue Wang, Yuanchang Yue, Yu Wang, Yanling Wang, Yan Wang, Xijun Liu, Wenmeng Yu, Weihan Wang, Wei Li, Shuaiqi Duan, Sheng Yang, Ruiliang Lv, Mingdao Liu, Lihang Pan, Ke Ning, Junhui Ji, Jinjiang Wang, Jing Chen, Jiazheng Xu, Jiale Zhu, Jiale Cheng, Ji Qi, Guobing Gan, Guo Wang, Cong Yao, Zijun Dou, Zihao Zhou, Zihan Wang, Zhiqi Ge, Zhijie Li, Zhenyu Hou, Zhao Xue, Zehui Wang, Zehan Qi, Zehai He, Yutao Zhang, Yusen Liu, Yukuo Cen, Yuchen Li, Yuan Wang, Yu Yang, Yongbin Liu, Yijian Lu, Yifan Xu, Yanzi Wang, Yanxiao Zhao, Yanfeng Wang, Yadong Xue, Yabo Xu, Xinyu Zhang, Xinyu Liu, Xiao Liu, Wenyi Zhao, Wenkai Li, Tianyu Tong, Tianshu Zhang, Shudan Zhang, Shengdong Yan, Qinkai Zheng, Mingde Xu, Licheng Bao, lat Long long, Jiaxing Xu, Jiaxin Fan, Jiawen Qian, Jiali Chen, Jiahui Lin, Jiadai Sun, Haozhi Zheng, Haoran Wang, Haochen Li, Hanyu Lai, Han Xu, Fan Yang, Dan Zhang, Da Yin, Chuangxin Zhao, Chengcheng Wu, Boyan Shi, Bowen Lv, Bowei Jia, Bo Li, Bin Chen, Baoxu Wang, Peng Zhang, Debing Liu, Bin Xu, Juanzi Li, Minlie Huang, Yuxiao Dong, and Jie Tang. Glm-5v-turbo: Toward a native foundation model for multimodal agents, 2026. URL https://arxiv.org/abs/2604.26752. 7

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. 3, 5, 14, 17

Jianyuan Wang, Minghao Chen, Shangzhan Zhang, Nikita Karaev, Johannes Schönberger, Patrick Labatut, Piotr Bojanowski, David Novotny, Andrea Vedaldi, and Christian Rupprecht. VGGT-Ω. *arXiv preprint arXiv:2605.15195*, 2026a. 3, 14

Siyin Wang, Junhao Shi, Zhaoyang Fu, Xinzhe He, Feihong Liu, Chenchen Yang, Yikang Zhou, Zhaoye Fei, Jingjing Gong, Jinlan Fu, et al. World action models: The next frontier in embodied ai. *arXiv preprint arXiv:2605.12090*, 2026b. 2, 3

Guo-Ye Yang, Wen-Yang Zhou, Yun Cai, Song-Hai Zhang, and Fang-Lue Zhang. Focusing on your subject: Deep subject-aware image composition recommendation networks. *Computational Visual Media*, 9(1):87–107, 2023. 2

Lujian Yao, Siming Zheng, Xinbin Yuan, Zhuoxuan Cai, Pu Wu, Jinwei Chen, Bo Li, and Peng-Tao Jiang. Photography perspective composition: Towards aesthetic perspective recommendation. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 90306–90327, 2025. 3

Mingde Yao, Zhiyuan You, King-Man Tam, Menglu Wang, and Tianfan Xue. Photoagent: Agentic photo editing with exploratory visual aesthetic planning. *arXiv e-prints*, pp. arXiv–2602, 2026. 2

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026. 2, 3

Zhiyuan You, Ke Wang, He Zhang, Xin Cai, Jinjin Gu, Tianfan Xue, Chao Dong, and Zhoutong Zhang. Photoframer: Multi-modal image composition instruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10197–10207, 2026. 7, 18

Ying Zeng, Miaosen Luo, Guangyuan Li, Yang Yang, Ruiyang Fan, Linxiao Shi, Qirui Yang, Jian Zhang, Chengcheng Liu, Siming Zheng, et al. Smartphotocrafter: Unified reasoning, generation and optimization for automatic photographic image editing. *arXiv preprint arXiv:2604.19587*, 2026. 2

Ke Zhang, Tianyu Ding, Jiachen Jiang, Tianyi Chen, Ilya Zharkov, Vishal M Patel, and Luming Liang. Procrop: Learning aesthetic image cropping from professional compositions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 12600–12608, 2026. 2

Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. Clay: A controllable large-scale generative model for creating high-quality 3d assets. *ACM Transactions On Graphics (TOG)*, 43(4):1–20, 2024. 4, 16

Mengchen Zhang, Tong Wu, Jing Tan, Ziwei Liu, Gordon Wetzstein, and Dahua Lin. Gendop: Auto-regressive camera trajectory generation as a director of photography. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 18229–18239. IEEE, 2025. 7, 18

Zhida Zhao, Talas Fu, Yifan Wang, Lijun Wang, and Huchuan Lu. From forecasting to planning: Policy world model for collaborative state-action prediction. In *Advances in Neural Information Processing Systems*, volume 38, pp. 134585–134611, 2025. 2, 3

A Additional Data and Benchmark Details

This section provides additional details of the camera-motion training data and the 3DGS photography benchmark, including trajectory recovery and filtering for training supervision, as well as the construction of evaluation environments, human assets, and benchmark cases.

A.1 Camera Motion Training Data

Source photographs and video synthesis. Our data construction process turns a static photographic endpoint into paired visual and camera-motion supervision. We first manually curate Internet portrait photographs, retaining images with a clear human subject and a useful photographic composition. For every retained photograph, we generate several five-second, 150-frame videos at 30 fps with Wan 2.7 (Wan et al., 2025). The generation prompts move the camera away from the reference viewpoint through changes in distance, azimuth, elevation, and subject placement. We reverse each generated video so that the training sequence instead begins from an unfavorable viewpoint and converges to the curated photograph. The associated instruction describes the desired endpoint rather than the synthetically degraded initial view. Figure 6 shows four representative generated sequences, each uniformly sampled into six frames.

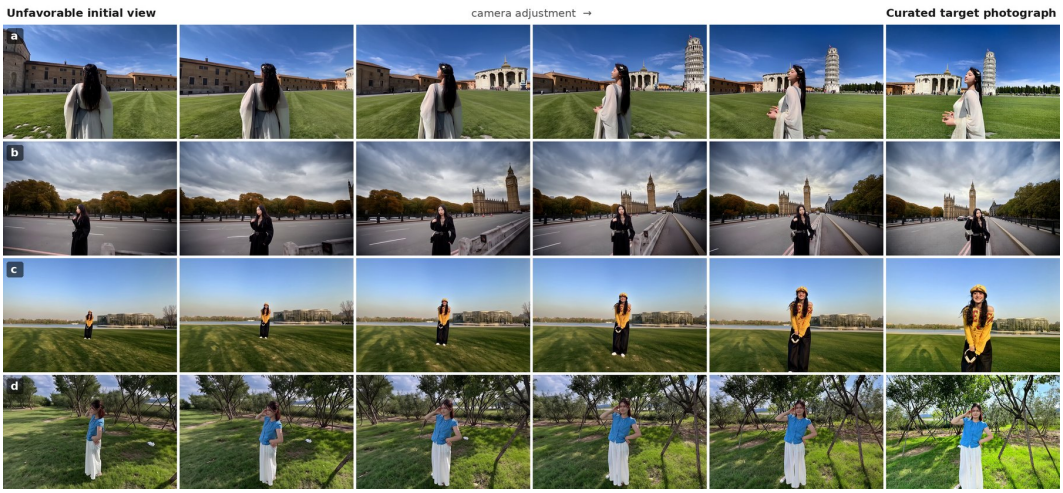


Figure 6: **Examples of video sequences.** Each row shows six uniformly sampled frames from one reversed five-second video. The sequence begins from a synthetically displaced camera view and progressively approaches the curated target photograph in the last column. The examples cover changes in camera distance, subject scale, lateral placement, and elevation.

Pose recovery and trajectory processing. Since the generated videos contain no camera metadata, we recover the camera trajectory and intrinsics using VGGT-Ω (Wang et al., 2026a). Figure 7 shows representative reconstructed scenes and recovered camera trajectories. We further normalize the recovered trajectories to obtain a consistent translation scale across videos and smooth the pose sequences to suppress high-frequency reconstruction noise.

Let $\mathbf{T}_t \in SE(3)$ denote the camera-to-world pose at frame t. We define the camera motion between two consecutive frames in the coordinate system of the current camera as

$$\Delta\mathbf{T}_t = \mathbf{T}_t^{-1}\mathbf{T}_{t+1}. \tag{14}$$

The translation component of $\Delta\mathbf{T}_t$ gives the local camera displacement, while its rotation component is converted to relative roll, pitch, and yaw. Together with the change in horizontal field of view derived from the recovered camera intrinsics, each transition is represented by a seven-dimensional action

$$\mathbf{a}_t = [\Delta x_t, \Delta y_t, \Delta z_t, \Delta\phi_t, \Delta\theta_t, \Delta\psi_t, \Delta f_t]. \tag{15}$$

Expressing the motion relative to the current camera frame makes the action representation independent of the arbitrary world coordinate system produced by reconstruction.

For training, each 150-frame source video is sampled into 65-frame windows and resized from 1176×784 to 384×256 while preserving its 3:2 aspect ratio. The video VAE maps each window to 17 temporal latent frames. Accordingly, every four frame-level camera transitions are aggregated into one action, resulting in 16 action transitions aligned with the 16 latent-frame transitions.

Filtering and final corpus. We apply several stages of filtering to remove unreliable camera trajectories and unsuitable training samples. First, we discard sequences with failed or unstable pose reconstruction and physically implausible camera motion. We further remove trajectories containing isolated motion spikes, excessive rotations above $360°$, or nearly static motion below $3°$. Finally, since camera actions should explain the dominant visual change, we remove clips in which the subject itself exhibits substantial motion, including large lateral displacement or consistent person-camera following motion.

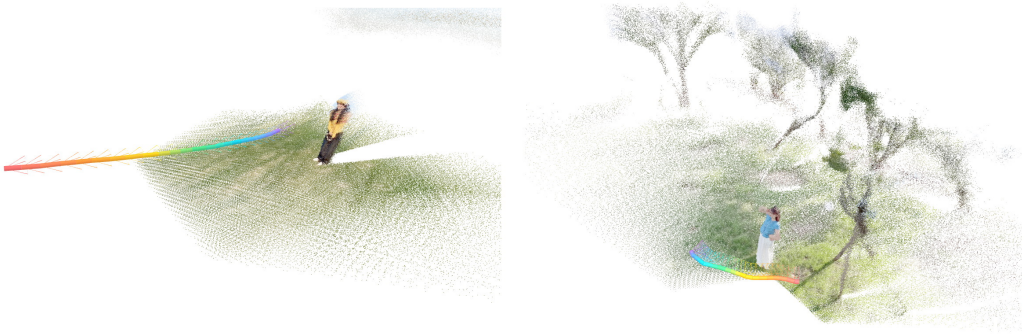


Figure 7: **Recovered camera trajectories.** Two representative examples are shown with camera poses and trajectories estimated by VGGT-Ω.

Table 4 summarizes the resulting data attrition. Starting from 21,122 reversed videos, pose-quality checks retain 12,368 trajectories, trajectory filtering retains 12,245 episodes, and subject-motion filtering leaves 10,176 effective training episodes.

Table 4: Attrition of the camera-motion data pipeline. Retention is measured relative to all generated reversed sequences.

Stage	Episodes	Retention
Generated reversed sequences	21,122	100.0 %
Pose quality filtering	12,368	58.6 %
Action trajectory filtering	12,245	58.0 %
Static-subject filtering	10,176	48.2 %

The final dataset contains 10,176 episodes and 1,526,400 source frames. Using the fixed seed-42 split yields 9,661 training episodes and 515 disjoint validation episodes. All videos synthesized from the same endpoint photograph are assigned to the same split. With the window-start stride used in our main configuration, the training set contains 106,271 candidate 65-frame windows. Each sample contains the RGB video, seven-dimensional camera state, body-frame camera actions, and the corresponding photographic instruction.

A.2 3DGS Photography Benchmark

This section expands the 3DGS photography benchmark in Section 3.2. The evaluation environments are constructed independently of the video-generation training pipeline, so that CamWAM is tested in interactive 3DGS scenes rather than on the synthesized videos used for supervision.

Reconstructed environments. We reconstruct two large-scale 3DGS scenes from real-world tourist attractions, denoted Scene 1 and Scene 2, and manually select 6 locations suitable for portrait photography. Table 5 reports the scale of the two reconstructions. Scene 1 contains 13.5 million Gaussians and covers about $7.3 \times 10^3\,\mathrm{m}^2$. Scene 2 contains 51.3 million Gaussians and covers

about 3.2×10^5 m^2. These extents and densities indicate that the evaluation worlds are metric reconstructions of outdoor public spaces rather than small indoor assets or purely synthetic sets. At each selected location, we keep the reconstructed background geometry, vegetation, and architectural detail, and only then place a human subject for portrait evaluation.

Table 5: Scale of the two reconstructed 3DGS environments. Covered area is the convex hull of Gaussian centers on the ground plane after discarding the most extreme 1% of coordinates.

Environment	Gaussians	Covered area	Vertical extent
Scene 1	13.5M	7.3×10^3 m^2	16 m
Scene 2	51.3M	3.2×10^5 m^2	141 m

Human assets and screening. We collect 28 human 3DGS assets, including subjects from THuman3.0 (Su et al., 2023) and assets generated using Hyper3D Rodin (Zhang et al., 2024). The retained assets are a screened subset of a larger candidate pool. Automatic checks discard Gaussians that are too fine to remain visible at portrait distances, nearly transparent subjects, and assets whose standing extent falls outside an adult height range of $[1.2, 2.2]$ m. We then manually review the remaining candidates for reliable rendering and portrait suitability, removing assets that fail to appear as a standing adult in the composed scene or that are unsuitable as a photographic subject. The 28 kept assets have standing heights between 1.41 m and 1.79 m and contain 2.0–17.0 million Gaussians each. The pool covers a wide range of photographic poses, so the shorter end of this range is not a child-sized body. It comes from subjects that are bent or stooped, which reduces the measured vertical extent. Figure 8 shows representative examples of both families after they have been placed in a reconstructed environment. Human assets that originate as meshes are converted into 3DGS by sampling anisotropic Gaussians on the mesh surface, so that they can be composed with the reconstructed scenes under the same renderer.



Figure 8: **Screened human 3DGS assets used in the photography benchmark.** (a) THuman3.0 subjects. (b) Hyper3D Rodin-generated subjects. All examples are shown after composition into a reconstructed environment.

Case construction. As described in Section 3.2, we assign 4 unfavorable initial camera poses and 6 photographic instructions to each of the 6 photography locations, yielding 144 location-pose-instruction cases. Each case is instantiated with 2 human subjects, for 288 evaluation cases in total. The benchmark does not define a target camera pose or reference trajectory. All methods start from the same initial view and act in a closed loop. Each predicted action updates the camera, and the newly rendered 1176×784 observation is returned as the next input. The initial horizontal field of view is $54.4°$, matching the 3:2 training aspect ratio. The final rendered view is the photographic result used for evaluation.

B Additional Model Details

This section provides additional implementation details of the CamWAM architecture introduced in Section 4, including the video and action backbones, camera-motion adapter, parameter counts, and the action-only variant.

Video expert. CamWAM is instantiated with three video-expert capacities, denoted CamWAM-1.3B, CamWAM-5B, and CamWAM-14B. The video experts are initialized from pretrained Wan video-generation backbones (Wan et al., 2025). The 1.3B and 14B variants use Wan 2.1 backbones with 30 and 40 DiT blocks, respectively, while the 5B variant uses the 30-block Wan 2.2 backbone. All variants take a 384×256 initial observation and generate 65-frame visual sequences conditioned on the photographic instruction. The corresponding Wan VAE encodes each sequence into 17 temporal latent frames. The text encoder and VAE remain frozen throughout training. In Stage 1, the 1.3B and 5B video experts are fully optimized, while the 14B video expert is adapted using LoRA with rank 32.

Action expert. The action expert is a compact DiT that predicts the $H{=}16$ camera-action sequence in $\mathbb{R}^7$ associated with the visual evolution, matching the latent-frame action representation in Section A.1. It uses hidden width 1024 and FFN width 4096 at every scale. We use 30 action blocks with the 1.3B video expert (0.64B parameters) and keep the action expert near 1.0B parameters at the two larger scales (30 blocks with the 5B video expert, 20 blocks with the 14B video expert). The two experts have separate weights and interact only through the asymmetric attention of Section 4.2. The video expert attends to itself, while the action expert may additionally attend to video keys and values.

Camera motion adapter. The adapter is a residual bottleneck of rank 256 with a GELU nonlinearity, applied independently to the video keys and values before they are read by the action expert. It is identity-initialized, so Stage 3 starts from the Stage 2 mapping. We set $\beta = 4$ in the adapter objective of Section 4.3, and draw the conditioning-noise level from $\mathcal{U}(0, \sigma_{\max})$ with $\sigma_{\max} = 0.45$.

Table 6: Parameter counts of the reported variants. For CamWAM, visual parameters are the Wan video DiT. Action-only uses the same 1.3B Wan video DiT as a current-frame encoder and does not generate a future clip. The frozen Wan text encoder is shared and not included.

Model	Visual backbone params	Action expert
Action-only	1.3B	0.64B
CamWAM-1.3B	1.3B	0.64B
CamWAM-5B	5.0B	1.02B
CamWAM-14B	14B	1.02B

Action-only variant. As in Table 3, Action-only uses the action expert without future visual prediction. The visual encoder is the same Wan 2.1 video DiT used by CamWAM-1.3B, applied to the current observation only. Those tokens, together with the same instruction embedding, condition the 30-block action DiT. There is no predicted visual sequence and no camera-motion adapter. Training uses the same 16-step, 7-dimensional action parameterization and the same action flow-matching objective as Stage 2. This isolates future visual prediction from the action head, while keeping the visual backbone matched.

C Additional Training and Implementation Details

C.1 Training

All CamWAM variants use the same optimization settings. For the full three-stage training, Stage 1, Stage 2, and Stage 3 are each trained for 3,000 steps, resulting in a total budget of 9,000 optimization steps. For a fair comparison, ablation variants with fewer stages, including Action-only and Joint, are trained for additional steps in their active stage(s) so that every variant receives the same total optimization budget of 9,000 steps.

All stages are trained on 8 NVIDIA H20 GPUs with a batch size of 8 per GPU. Optimization uses AdamW in bfloat16, with learning rate 1×10^{-4}, weight decay 0.01, and gradient clipping at 1.0. The action and adapter stages follow a cosine schedule with a short 3% warmup. Camera actions are z-score normalized from the training set. Both experts are trained with flow matching over 1000 discrete timesteps and a time shift of 5.0. The timestep weights $w_{\text{vid}}(\tau)$ and $w_{\text{act}}(\tau)$ follow the Wan flow-matching scheduler and are normalized to unit mean.

The losses follow Section 4.3. Stage 1 uses $\mathcal{L}_{\text{vid}}$ with unit weight. Stage 2 uses $\mathcal{L}_{\text{act}}$ with $\lambda_{\text{act}} = 1$. The seven action-dimension weights are $\lambda_d = (1, 1, 1, 5, 5, 5, 1)$ for translation, rotation, and field-of-view change, so rotation is upweighted. These weights are then renormalized to unit mean, which leaves the scale of λ_{act} unchanged. Joint training uses both experts at once with unit weight on $\mathcal{L}_{\text{vid}}$ and $\mathcal{L}_{\text{act}}$, and the same λ_d. Action-only uses the Stage 2 action loss with the same λ_{act} and λ_d. Stage 3 keeps $\lambda_{\text{act}} = 1$ and adds the adapter objective with $\beta = 4$ and $\sigma_{\max} = 0.45$. The adapter coefficient is $\lambda_{\text{adapter}} = 31.0$. Padded steps after a trajectory ends are supervised as zero motion, so the action expert can express a stop by predicting negligible movement.

C.2 Implementation

Closed-loop evaluation is conducted in the 3DGS photography environments of Section 3.2. At each round, the model receives the current rendered view and the photographic instruction, and predicts a 16-step camera-action sequence. Only the first eight steps are executed. The environment then renders a new observation, and the model replans from that view. Executing half of the predicted horizon lets later motion be revised after the camera has actually moved. We also use a simple stop rule. If several consecutive executed steps produce only negligible translation, and at least two rounds have already been completed, the rollout ends. Otherwise, the interaction continues for up to ten rounds. The last rendered frame is the photographic result used for scoring. Within each round, the video expert first predicts the visual evolution with 10 flow matching steps, and the action expert then reads those representations to produce the camera actions.

D Additional Experimental Details

This section records how the baselines in Section 5.1 were run, then expands the evaluator study, the main comparison, and the training-protocol ablation of Sections 5.3 through 5.5. Unless noted otherwise, CamWAM numbers use the same closed-loop protocol as Table 2, with dual-judge means over the 288 photography cases.

D.1 Baseline Protocols

Section 5.1 already names the four baseline families, and here we record the closed-loop settings across families.

Composition methods. VAPNet (Su et al., 2021), UNIC (Liu et al., 2023), and PhotoFramer (You et al., 2026) improve a photograph by adjusting the current view, not by driving a camera through a scene. To place them on the same 3DGS benchmark, we turn each predicted adjustment into a camera motion, apply it to the current viewpoint, and feed back the newly rendered image. The loop stops when the method decides the framing is complete, or when a short round limit is reached.

Camera-motion methods. AutoPhoto (AlZayer et al., 2021) uses a discrete action policy and stops when it selects its native stop action, or when a step budget is exhausted. GenDoP-Text (Zhang et al., 2025) predicts a full trajectory in one pass and therefore has no interactive stop. GenDoP-RGBD replans for up to four rounds and additionally receives rendered depth.

Direct VLM agents. The four general-purpose agents in Table 2 run up to 16 rounds. Each round executes a four-frame camera increment, so the cap is 64 micro-steps. That budget yields about five to six meters of travel, which is close to CamWAM under the stop rule of Section C.2. At each round, the model outputs either a camera increment or `STOP`. A `STOP` after the first round ends the episode without executing that round.

VLA methods. We evaluate $\pi_{0.5}$ (Physical Intelligence et al., 2025) together with three StarVLA (Community, 2026) variants using 0.8B, 2B, and 4B VLM backbones. Unlike the zero-

shot VLM agents, all VLA baselines are trained on our photography data using the same seven-dimensional camera-action representation. They predict action chunks from the current observation and photographic instruction and are executed iteratively with replanning from updated views.

CamWAM uses the stop rule of Section C.2. Table 2 therefore compares each family under a closed-loop budget that matches how it is used.

D.2 VLM Evaluator

Table 1 already shows that repeated scoring preserves sample rankings and that VLM–human agreement is close to human–human agreement. Here we record what that stability means for the reported scores. Both VLM judges receive the photographic instruction, the initial view, and the final photograph, and they return integer scores for q_{initial}, q_{final}, and instruction following. The prompts used in our evaluation are also submitted as supplementary files.

Figure 9 shows pairwise Spearman correlations among the five independent runs of the dual-judge mean. Off-diagonal entries remain above 0.91 for Q and above 0.92 for I. The ranking of episodes is therefore stable under re-scoring.

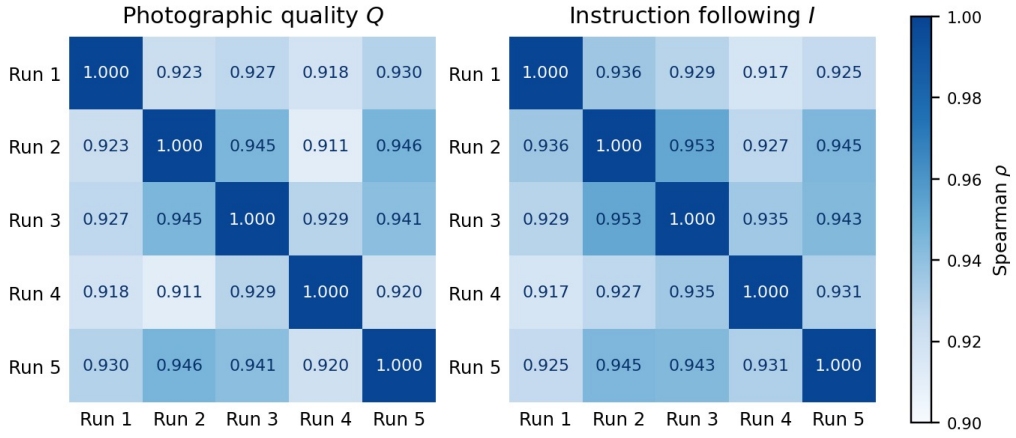


Figure 9: Pairwise Spearman correlation among five independent scoring runs of the dual-judge mean, computed on the same 300 validation images used in Table 1. Rankings of Q and I remain stable across runs.

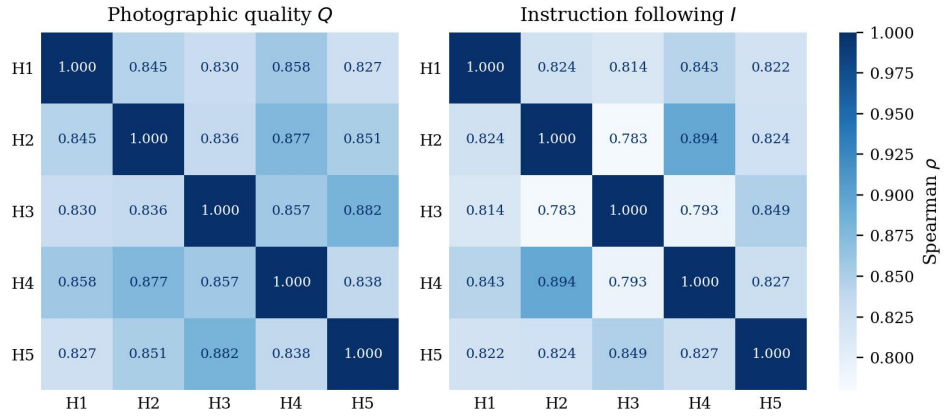


Figure 10: Pairwise Spearman correlation among the five human annotators on the 300 validation photographs. People agree more tightly on Q than on I, but both matrices stay in a narrow high range.

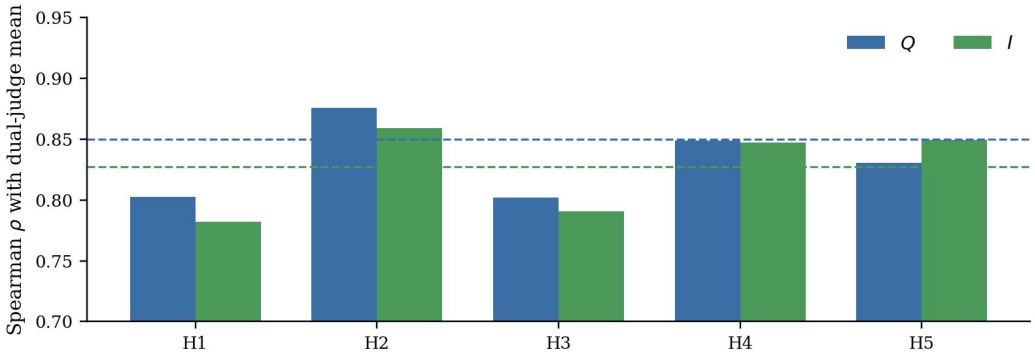


Figure 11: Spearman correlation of the dual-judge mean with each annotator, using the five-run ensemble mean. Dashed lines are the mean human–human ρ from Figure 10.

Human study. The human numbers in Table 1 come from five annotators who scored the same 300 photographs independently. The 300 images are stratified by photographic score so that both strong and weak results appear. The annotators read the same scoring instructions as the VLM judges and used the same definitions of Q and I. Each person saw the photographic instruction and the final photograph. Method names were not shown. They scored only the final photograph. They did not score the initial view, because the study is a check of the judged result rather than of ΔQ. The VLM judges still produce q_{initial} for the main ΔQ metric. Human–VLM and human–human agreement is therefore computed on final Q and on I.

Figure 10 shows pairwise Spearman correlations among the five people. Off-diagonal entries for Q lie between 0.827 and 0.882. For I they lie between 0.783 and 0.894. The mean human–human ρ is 0.850 for Q and 0.827 for I, which is the Human–Human row of Table 1. Figure 11 compares each annotator with the dual-judge mean, using that person's scores against the five-run ensemble mean. The dashed lines mark the mean human–human ρ. The ensemble stays close to this reference for every annotator. For several annotators, it matches human–human agreement, and it never falls far below it.

Averaging Qwen3.8-Max (Qwen Team, 2026b) and Gemini-2.5-Pro (Gemini Team, Google, 2025) further tightens the aggregate. The five-run range of the mean score is 0.72 for Q and 0.57 for I, smaller than either judge alone on I and no larger than the more stable judge on Q.

D.3 Statistical Robustness of the Main Comparison

To further examine the robustness of the performance differences in Table 2, we conduct a paired bootstrap analysis over the 288 benchmark cases. For each pair of methods, we compute the case-wise differences in instruction following I, photographic quality Q, and PSS, and estimate 95% confidence intervals by bootstrap resampling while preserving the pairing between methods. Table 7 reports the mean paired differences and their corresponding confidence intervals. Positive values favor the first method, and intervals that exclude zero are highlighted in bold.

The paired analysis provides additional support for the main performance comparison. Most notably, CamWAM-14B shows positive differences over both GPT-5.6-Sol and StarVLA-4B in all three metrics, with all corresponding 95% confidence intervals excluding zero. This confirms that the advantage of the largest CamWAM variant is not driven by a small subset of benchmark cases. The analysis also supports the overall benefit of increased model capacity. CamWAM-14B improves over CamWAM-1.3B in I, Q, and PSS, with all three intervals above zero, and further improves over CamWAM-5B in both I and PSS. Although the adjacent 1.3B–5B differences are less clearly separated, their point estimates remain positive across all three metrics.

The smaller CamWAM variants also exhibit complementary advantages over the strongest baselines. Compared with GPT-5.6-Sol, both CamWAM-1.3B and CamWAM-5B show clear improvements in photographic quality Q. Compared with StarVLA-4B, both variants achieve higher instruction-following scores, while CamWAM-5B additionally obtains a positive PSS difference whose confidence interval excludes zero. These results indicate that even the smaller CamWAM variants remain

Table 7: Paired performance differences on the 288-case 3DGS photography benchmark. Each entry reports the mean difference between two methods together with its paired-bootstrap 95% confidence interval. Positive values favor the first method, and bold entries indicate intervals that exclude zero.

Comparison	n	ΔI	ΔQ	ΔPSS
Across CamWAM scales				
CamWAM-14B − CamWAM-5B	288	**+3.67 [+0.50, +6.80]**	+1.43 [−1.21, +4.14]	**+2.78 [+0.10, +5.46]**
CamWAM-14B − CamWAM-1.3B	288	**+5.19 [+1.76, +8.73]**	**+3.80 [+0.74, +6.84]**	**+3.89 [+0.97, +6.87]**
CamWAM-5B − CamWAM-1.3B	288	+1.51 [−1.50, +4.52]	+2.36 [−0.56, +5.31]	+1.11 [−1.45, +3.67]
Comparison with GPT-5.6-Sol				
CamWAM-14B − GPT-5.6-Sol	288	**+4.87 [+0.42, +8.75]**	**+9.51 [+5.42, +12.94]**	**+4.39 [+0.84, +7.49]**
CamWAM-5B − GPT-5.6-Sol	288	+1.20 [−2.63, +4.91]	**+8.08 [+4.74, +11.81]**	+1.61 [−1.39, +4.47]
CamWAM-1.3B − GPT-5.6-Sol	288	−0.31 [−4.17, +3.28]	**+5.72 [+1.95, +8.80]**	+0.50 [−2.65, +3.35]
Comparison with StarVLA-4B				
CamWAM-14B − StarVLA-4B	288	**+8.41 [+4.91, +11.90]**	**+3.76 [+0.28, +7.13]**	**+5.72 [+2.62, +8.77]**
CamWAM-5B − StarVLA-4B	288	**+4.74 [+1.49, +8.02]**	+2.33 [−0.90, +5.47]	**+2.94 [+0.16, +5.74]**
CamWAM-1.3B − StarVLA-4B	288	**+3.23 [+0.20, +6.29]**	−0.03 [−3.19, +3.10]	+1.83 [−0.86, +4.45]

competitive with strong general-purpose and VLA baselines, while CamWAM-14B provides the most consistent gains across instruction following, photographic quality, and overall PSS.

D.4 Additional Comparison

Table 2 pools Scene 1 and Scene 2. The slices below keep the same scoring protocol and only change how cases are grouped.

Scene 1 versus Scene 2. Table 8 splits the 192 cases in Scene 1 from the 96 cases in Scene 2. Every method is weaker in Scene 2, where the reconstructed world is much broader and the initial views are often farther from a usable portrait. CamWAM-1.3B drops most, from a PSS of 31.69 to 24.21. CamWAM-5B and CamWAM-14B keep a smaller gap, from 31.10 to 28.72 and from 34.28 to 30.71. GPT-5.6-Sol (OpenAI, 2026) and StarVLA-4B (Community, 2026) also lose several PSS points in Scene 2. CamWAM-14B remains ahead of both in either environment.

Table 8: Scene split of the main closed-loop evaluation. Scene 1 and Scene 2 are the two reconstructions in Section A.2.

	Scene 1 (n=192)			Scene 2 (n=96)		
Method	$I\uparrow$	$\Delta Q\uparrow$	PSS $\uparrow$	$I\uparrow$	$\Delta Q\uparrow$	PSS $\uparrow$
GPT-5.6-Sol	47.46	+12.08	29.67	42.54	+4.06	26.60
StarVLA-4B	43.52	+17.16	29.02	40.03	+11.71	24.06
CamWAM-1.3B	48.46	+18.44	31.69	39.83	+9.04	24.21
CamWAM-5B	47.79	+19.49	31.10	45.71	+14.03	28.72
CamWAM-14B	51.54	+20.60	34.28	49.22	+16.12	30.71

Per-evaluator results. Table 9 reports the results under Qwen3.8-Max and Gemini-2.5-Pro separately, together with the dual-judge evaluation used in Table 2. Despite evaluator-specific preferences, both judges consistently favor the stronger CamWAM variants. Under Qwen3.8-Max, CamWAM-14B achieves the highest I, while CamWAM-5B and CamWAM-14B obtain nearly identical Q and PSS, with PSS comparable to GPT-5.6-Sol and higher than StarVLA-4B. Under Gemini-2.5-Pro, CamWAM-14B achieves the highest I, Q, and PSS with clear margins over both baselines. After combining the two evaluators, CamWAM-14B ranks highest in all three metrics. Moreover, all three CamWAM variants achieve higher PSS than GPT-5.6-Sol and StarVLA-4B, all three achieve higher Q than GPT-5.6-Sol, and all three achieve higher I than StarVLA-4B. These results show that the main performance trend is preserved across evaluators rather than being driven by the preference of a single judge, while the dual-judge evaluation provides a more balanced aggregate assessment.

Table 9: Per-evaluator results on the 288-case benchmark under the main closed-loop protocol.

	Qwen3.8-Max			Gemini-2.5-Pro			Dual-judge		
Method	$I\uparrow$	$Q\uparrow$	PSS $\uparrow$	$I\uparrow$	$Q\uparrow$	PSS $\uparrow$	$I\uparrow$	$Q\uparrow$	PSS $\uparrow$
GPT-5.6-Sol	49.29	51.04	30.66	42.51	48.20	26.29	45.90	49.62	28.70
StarVLA-4B	43.30	53.75	27.34	41.41	56.99	27.94	42.36	55.37	27.37
CamWAM-1.3B	46.17	53.15	28.34	45.00	57.52	30.56	45.59	55.34	29.20
CamWAM-5B	49.02	57.15	31.67	45.18	58.26	29.80	47.10	57.70	30.31
CamWAM-14B	49.74	57.11	31.54	51.81	61.16	35.43	50.77	59.13	33.09

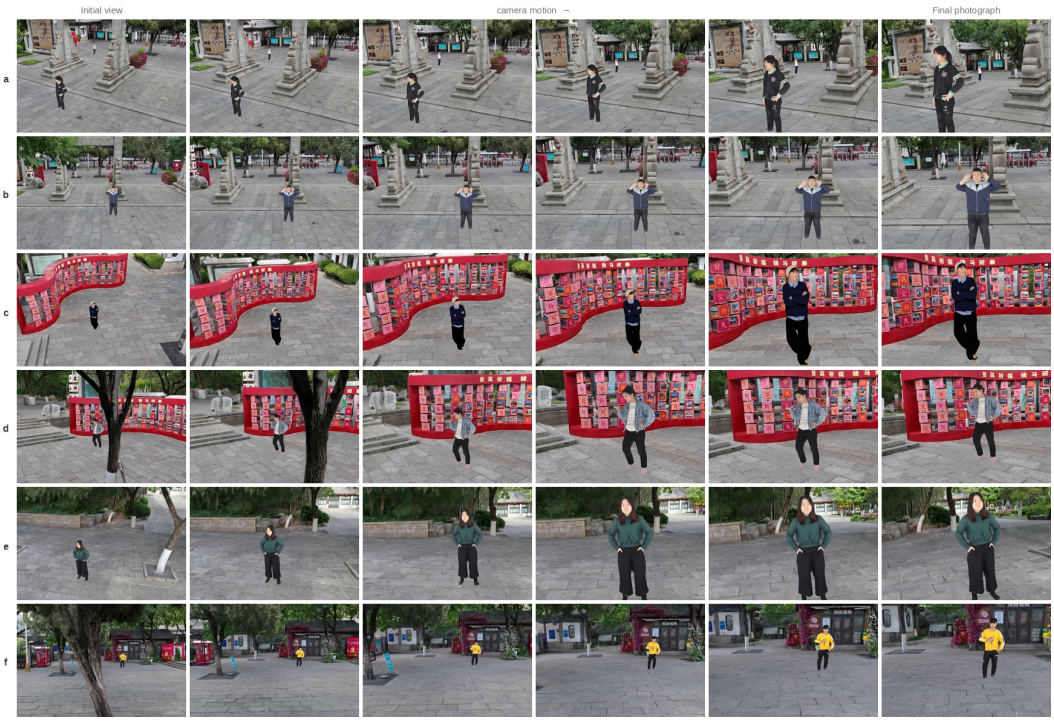


Figure 12: Additional closed-loop rollouts of CamWAM-1.3B. Each row shows six uniformly sampled frames along an executed camera trajectory, from the initial view to the final photograph.

D.5 Visual Results

Qualitative examples. Figure 12 presents additional closed-loop rollouts of CamWAM-1.3B under the stopping protocol described in Section C.2. Each row shows six frames uniformly sampled along the executed camera trajectory, from the initial observation to the final photograph.

Figure 13 visualizes the future visual evolution predicted by the video expert. Each row starts from an Internet photograph I_0 and a corresponding photographic instruction, from which the 14B video expert predicts a future visual sequence at 384×256 resolution. We uniformly sample eight frames from each sequence, from the initial observation to the predicted final frame. The examples cover outdoor portraits with changes in camera distance, elevation, and subject placement.

Failure cases. As shown in Table 2, 17.7% of the final photographs produced by CamWAM-1.3B receive $Q < 30$. Figure 14 visualizes six representative failure trajectories under the stopping protocol of Section C.2. These failures mainly arise from two factors: where the camera stops and how it is oriented at that position. A successful portrait requires both an appropriate camera distance and a viewing direction that keeps the subject properly framed. The first four rows illustrate failures caused by inaccurate stopping, where the camera continues moving after reaching a usable view and eventually stops too close to the subject or nearby geometry. The last two rows illustrate failures



Figure 13: Predicted visual evolution. Each row shows eight uniformly sampled frames from the predicted sequence, from the initial observation to the predicted final frame.

in camera orientation, where the camera has already reached a reasonable position but its viewing direction remains misaligned with the subject.



Figure 14: Failure cases of CamWAM-1.3B. Each row shows six uniformly sampled frames from a closed-loop trajectory whose final photograph has $Q < 30$.

Prediction–execution alignment. Figure 15 compares predicted and executed visual sequences for four CamWAM-14B cases. Within each group, all rows share the same initial observation and predicted visual sequence. The first row shows the visual evolution predicted by the video expert, while the second and third rows show the executed rollouts using the action predictions after Stage 2 and after Stage 3, respectively. Six frames are uniformly sampled from each sequence. Across these examples, the rollouts after Stage 3 more closely follow the predicted visual evolution than those only after Stage 2.



Figure 15: Prediction–execution alignment of CamWAM-14B. Each group shows the predicted visual sequence and the executed rollouts after Stage 2 and Stage 3, respectively.

E Inference Cost

This section reports the inference cost of CamWAM. We time one closed-loop round on a single NVIDIA H20 in bfloat16. The input is a 384×256 observation and a photographic instruction. The model predicts a 65-frame visual window and a 16-step action chunk with 10 flow-matching steps, matching Section C.2. Each number is the mean of seven rounds after two warmups. Peak memory is the process-level allocated size and does not include the CUDA context of about 0.5 GB.

Wall-clock time and peak memory. Figure 16 and Table 10 summarize the four variants in Table 6. CamWAM-5B is the fastest full model, at 3.88 s per round. CamWAM-1.3B takes 5.42 s despite fewer parameters. CamWAM-14B takes 26.25 s and 38.3 GB, and is the only variant whose

memory is dominated by the video expert. Action-only finishes in 0.37 s and 4.4 GB. It still loads the 1.3B Wan video backbone, but only encodes the current frame and never denoises a future clip.

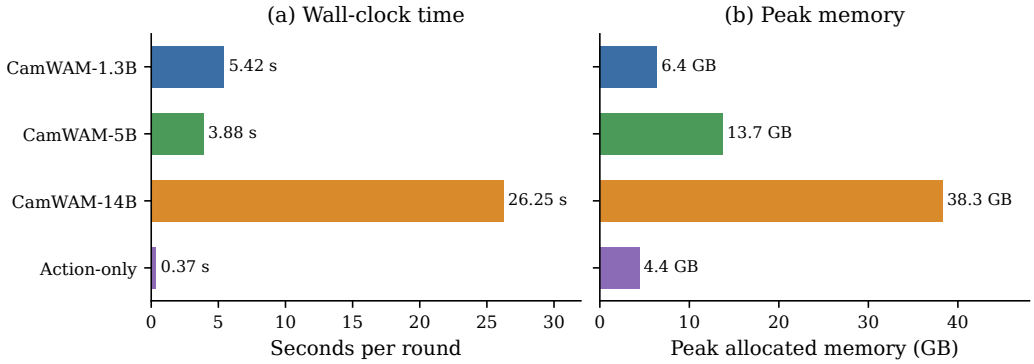


Figure 16: Per-round wall-clock time and peak allocated memory on one NVIDIA H20, excluding 3DGS rendering. Action-only is the current-frame Wan variant in Table 6.

Table 10: Per-round inference cost on one NVIDIA H20 with 10 flow-matching steps, excluding 3DGS rendering. Tokens are video DiT tokens after the Wan spatial patch. Weights is the size of loaded parameters. Action-only is the 1.3B-scale current-frame Wan variant in Table 6.

Model	Time (s) ↓	Peak (GB) ↓	Weights (GB)	Tokens
Action-only	0.37	4.4	4.1	384
CamWAM-1.3B	5.42	6.4	4.1	6528
CamWAM-5B	3.88	13.7	12.5	1632
CamWAM-14B	26.25	38.3	34.2	6528

Why CamWAM-5B is faster than CamWAM-1.3B. Runtime follows parameter count times token count, not parameter count alone. The 5B video expert uses the Wan 2.2 VAE, which downsamples space by 16, so a 384×256 clip becomes a $17 \times 16 \times 24$ latent. The 1.3B and 14B experts use Wan 2.1, which downsamples by 8 and produces a $17 \times 32 \times 48$ latent. After the spatial patch, CamWAM-5B therefore has 1632 video tokens, while the other two have 6528. Table 11 converts this into an approximate FLOP count per denoising step.

CamWAM-14B has about $4\times$ as many tokens as CamWAM-5B and about $3.3\times$ as many video parameters, so it does about $13\times$ more multiply-adds per step. The measured step times (2.21 s versus 0.17 s) match that ratio. Both large models reach about 70% of the measured BF16 peak on this GPU, so the gap is not under-utilization. CamWAM-1.3B has almost the same FLOP per step as CamWAM-5B, because its extra tokens cancel its smaller width, but it occupies only about 37% of peak and is therefore slower. Peak memory rises from 6.4 GB to 13.7 GB, the latent channel count grows from 16 to 48, and VAE decode becomes a much larger share of the round, as shown next.

Table 11: Video-token geometry and an approximate FLOP count per denoising step, $2\times$ (video parameters) $\times$ (DiT tokens). Throughput is that FLOP count divided by the measured time of one joint video-action step.

Model	VAE	Latent $t\times h\times w$	Tokens	FLOP / step (T)	Time / step (s)	TFLOPS
CamWAM-1.3B	$8\times$	$17\times32\times48$	6528	18.5	0.36	51
CamWAM-5B	$16\times$	$17\times16\times24$	1632	16.3	0.17	95
CamWAM-14B	$8\times$	$17\times32\times48$	6528	216	2.21	98

Where the time goes. Figure 17 splits one round into joint video-action denoising, VAE decode of the predicted clip, the later action pass that reads the denoised video, and VAE encode of the current frame. On CamWAM-14B, denoising is 84% of the round. On CamWAM-5B, the same stage is only 44%, and VAE decode takes another 43%, because the $16\times$ spatial compression has to be inverted.

CamWAM-1.3B sits in between, with denoising at 67% and decode at 20%. VAE encoding of the first frame is negligible on every scale. The photography scores in Table 2 are computed on 3DGS rerenders of the executed trajectory, not on the video that CamWAM generates. Skipping VAE decode is therefore a valid deployment choice when only the camera actions are needed. Table 12 measures that setting directly, rather than subtracting the decode time from the full round. Time drops by 20%, 43%, and 6% at 1.3B, 5B, and 14B. Peak memory barely moves, because the weights already dominate the allocation.

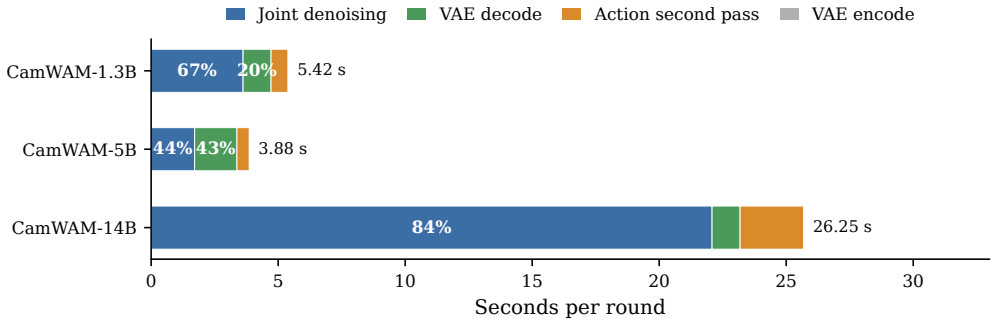


Figure 17: Breakdown of one round at 10 flow-matching steps. Percentages are shown only for slices that occupy at least 18% of the round.

Number of flow-matching steps. The main tables use 10 steps. Figure 18 sweeps 1 through 30. Time grows linearly with the step count. A least-squares fit $t = a + b \times$ steps is reported in Table 13. The intercept a is the step-independent work (VAE, prefill, and scheduling). It is 27% of a 10-step round at 1.3B, 48% at 5B, and 15% at 14B. Cutting the step count therefore helps CamWAM-14B much more than CamWAM-5B. Even one step would still cost about 1.9 s at 1.3B and 2.1 s at 5B. Peak memory remains nearly unchanged over the tested step range

Action-only remains about 1.00 s even at 30 steps. Almost all of its 0.37 s at 10 steps is the action expert reading a single-frame Wan encoding. Dropping future visual prediction is therefore cheap, at the quality cost already shown in Table 3.

Table 12: Cost of skipping VAE decode of the predicted clip, measured by stubbing the decoder in the same benchmark process. The photography evaluator never reads that clip.

Model	Full (s)	No decode (s)	Saved	Full (GB)	No decode (GB)
CamWAM-1.3B	5.42	4.34	20%	6.4	6.1
CamWAM-5B	3.88	2.22	43%	13.7	13.5
CamWAM-14B	26.25	24.79	6%	38.3	38.0

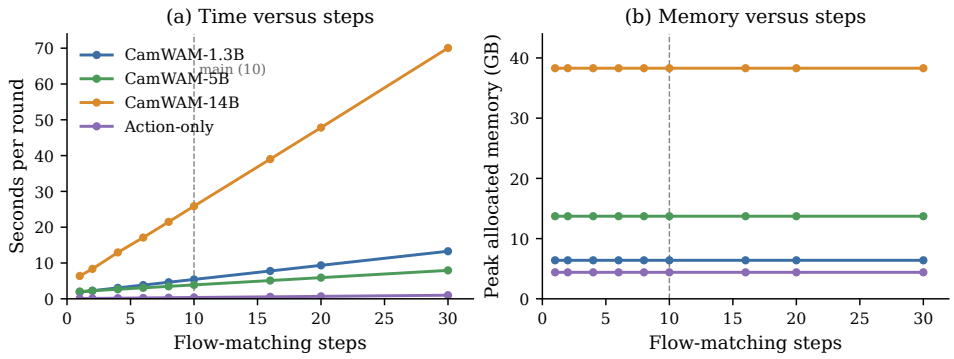


Figure 18: Cost versus the number of flow-matching steps. The dashed line marks the 10 steps used in the main tables. Time is linear in the step count, while peak memory is constant.

Table 13: Linear fit of per-round time against the flow-matching step count. a is the step-independent overhead. b is the cost of one additional step. The last column is a as a fraction of the fitted 10-step time.

Model	a (s)	b (s/step)	Fitted $t(10)$ (s)	a at 10 steps
CamWAM-1.3B	1.48	0.394	5.41	27%
CamWAM-5B	1.85	0.203	3.88	48%
CamWAM-14B	4.01	2.196	25.96	15%