Controllable Image and Video Generation with Foundation Models

by

Cong Wei

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2026

© Cong Wei 2026

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner: **Vicente Ordóñez-Román**
Professor
Department of Computer Science
Rice University

Supervisor(s): **Wenhu Chen**
Assistant Professor
David R. Cheriton School of Computer Science
University of Waterloo
Jimmy Lin
Professor
David R. Cheriton School of Computer Science
University of Waterloo

Internal Member: **Yuntian Deng**
Assistant Professor
David R. Cheriton School of Computer Science
University of Waterloo

Internal Member: **Victor Zhong**
Assistant Professor
David R. Cheriton School of Computer Science
University of Waterloo

Internal-External Member: **David Clausi**
Professor
Department of Systems Design Engineering
University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

The research presented in this thesis is based on eight published or accepted papers and one publicly available manuscript. Cong Wei is the first author of the works presented in Sections 3.4 and 4.2. He is a co-first author (* denotes equal contribution) of the works presented in Sections 3.2, 3.3, 4.3, and 5.3. For these works, he contributed to the core ideas and methodology, implementation and model training, experimental design and execution, analysis of results, manuscript writing, and project guidance. Cong Wei is a corresponding author of the work presented in Section 6.2 and is designated as a main contributor (†) to the work presented in Section 6.3. For the works presented in Sections 5.2, 6.3, and 6.2, he contributed to the core ideas and methodology, experimental design and execution, manuscript writing, and project guidance.

The publications and co-authors associated with each study are listed below, grouped by research chapter:

Chapter 3 (Image and Video Generation and Editing with Multimodal Instructions) is based on:

- Cong Wei*, Zheyang Xiong*, Weiming Ren, Xinrun Du, Ge Zhang, and Wenhu Chen. *OmniEdit: Building Image Editing Generalist Models Through Specialist Supervision.* In Proceedings of the *International Conference on Learning Representations* (ICLR), 2025.

- Max Ku*, Cong Wei*, Weiming Ren*, Harry Yang, and Wenhu Chen. *AnyV2V: A Tuning-Free Framework For Any Video-to-Video Editing Tasks.* Published in *Transactions on Machine Learning Research* (TMLR), 2024.

- Cong Wei, Quande Liu, Zixuan Ye, Qiulin Wang, Xintao Wang, Pengfei Wan, Kun Gai, and Wenhu Chen. *UniVideo: Unified Understanding, Generation, and Editing for Videos.* In Proceedings of the *International Conference on Learning Representations* (ICLR), 2026.

Chapter 4 (Temporal Control through Speech and Historical Context) is based on:

- Cong Wei, Bo Sun, Haoyu Ma, Ji Hou, Felix Juefei-Xu, Zecheng He, Xiaoliang Dai, Luxin Zhang, Kunpeng Li, Tingbo Hou, Animesh Sinha, Peter Vajda, and Wenhu Chen. *MoCha: Towards Movie-Grade Talking Character Generation.* In *Advances in Neural Information Processing Systems 38* (NeurIPS), 2025. Selected for a Spotlight presentation.

- Shuo Chen*, Cong Wei*, Sun Sun, Tiancheng Shen, Ping Nie, Kai Zou, Ge Zhang, Ming-Hsuan Yang, and Wenhu Chen. *Context Forcing: Consistent Autoregressive Video Generation with Long Context*. In Proceedings of the *International Conference on Machine Learning* (ICML), 2026.

Chapter 5 (Improving Instruction Adherence through Reward Feedback) is based on:

- Haozhe Wang, Cong Wei, Weiming Ren, Jiaming Liu, Fangzhen Lin, and Wenhu Chen. *RationalRewards: Reasoning Rewards Scale Visual Generation Both Training and Test Time*. In Proceedings of the *Conference on Language Modeling* (COLM), 2026.

- Yuxuan Zhang*, Cong Wei*, Penghui Du*, Bo Li*, Huaisong Zhang, Songcheng Cai, Yubo Wang, Dongfu Jiang, Yuyu Zhang, Changqian Yu, Ping Nie, Wenhu Chen, and Kelsey R. Allen. *RewardHarness: Learning Human Preferences for Image Editing with Only 100 Demonstrations*. In Proceedings of the *Conference on Language Modeling* (COLM), 2026. (The author mentored Yuxuan Zhang.)

Chapter 6 (Benchmarking World Knowledge and Visual Reasoning in Visual Generation) is based on:

- Haozhe Wang, Weijia Feng, Jinpeng Yu, Che Liu, Ping Nie, Fangzhen Lin, Jiaming Liu, Ruihua Huang, Jimmy Lin, Wenhu Chen, and Cong Wei. *Search Beyond What Can Be Taught: Evolving the Knowledge Boundary in Agentic Visual Generation*. (Jiaming Liu and Cong Wei are corresponding authors.) arXiv preprint arXiv:2607.05382, 2026.

- Xuan He†, Cong Wei†, Yuhao Cheng†, Linrui Ma†, Yuxuan Zhang†, Zuojun Li, Yuhao Wen, Jize Jiang, Zeyi Liu, Yuren Hao, Songcheng Cai, Keming Wu, Penghui Du, Kai Zou, Rui Yang, Chenkai Sun, Ke Yang, Ping Nie, Kelsey R. Allen, Chenglong Wang, Michel Galley, Jianfeng Gao, and ChengXiang Zhai. *VGI-Bench: Probing Visual Intelligence in Video Generation Models*. Accepted to the Main Conference of the *Conference on Empirical Methods in Natural Language Processing* (EMNLP), 2026.

Abstract

Diffusion and flow-based generative models have made substantial progress in synthesizing images and videos from text descriptions. However, traditional text-to-image and text-to-video models lack the controllability required for practical applications. In visual creation, users may wish to edit existing images or videos, provide visual references, and specify which details should remain unchanged. For video creation, users may additionally want characters' performances to synchronize with speech or long videos to remain coherent over extended durations. These demands call for generation methods that offer control beyond traditional text conditioning. Yet accepting additional conditions does not guarantee that a model follows them, motivating feedback that can guide improvement. This thesis advances controllable image and video generation with foundation models through four connected challenges: flexible instruction-guided generation and editing, temporal control through speech and history, reward feedback for instruction adherence, and benchmarks that reveal limits beyond visual appearance.

Across image and video generation and editing, users need flexible interfaces to express their intent through multimodal instructions comprising text, images, and videos. Supporting these requests requires models to identify the intended task and interpret the roles of the supplied inputs. We develop generalist systems that support diverse tasks through these flexible interfaces. OmniEdit uses specialist supervision to bring multiple image-editing operations into a generalist model. AnyV2V composes an image editor with an image-to-video generator to extend image-editing capabilities to video without task-specific tuning. To connect instruction understanding with visual synthesis, we introduce UniVideo, which brings multimodal understanding, generation, and editing into a shared model. The shared model can interpret multimodal instructions and distinguish the intended task, supporting nine image and video tasks within a unified framework and demonstrating strong generalization to unseen tasks.

Compared with image generation, video generation introduces an additional temporal dimension of control. Beyond specifying content and edits, we study how speech and historical context can guide the unfolding video. We introduce MoCha for speech-conditioned video generation that aligns performance with audio and coordinates multi-character dialogue, enabling movie-grade storytelling. For long-video generation, maintaining consistency with earlier context becomes increasingly difficult as the video grows longer. We address this with Context Forcing, which combines a long-context teacher that guides the use of historical context with Slow-Fast Memory that balances recent detail with longer-term context. This supports real-time long-video generation while mitigating forgetting and drift.

Enabling models to accept additional control inputs does not ensure that their outputs follow the intended requirements. We study this gap in image generation and editing, using reward feedback to evaluate the requested transformation together with source preservation and visual quality. RationalRewards learns structured critiques from preference data and uses them to guide both reinforcement learning and inference-time refinement. With RewardHarness, we further show that adapting a reward system need not require updating its model weights: evaluation guidelines and tools can instead be learned from a small set of preference demonstrations. Across these approaches, downstream image-generation and editing experiments show that feedback can guide improvement rather than merely judge completed outputs.

When generation requests depend on world knowledge or visual reasoning, visual quality alone cannot establish correctness. We therefore design benchmarks that make these demands explicit and expose model limitations. SearchGen-Bench tests long-tailed and evolving knowledge requirements, revealing failures to depict requested entities and facts accurately. These failures motivate selective retrieval and co-training of a search agent and generator, so that external information can help satisfy requests beyond the generator's existing knowledge. VGI-Bench evaluates visual reasoning in video generation, examining whether models can solve visually grounded problems given an initial image, a goal, and task constraints. The results reveal emerging visual reasoning abilities, but reliable performance under these constraints remains challenging. These benchmarks make knowledge accuracy and visual reasoning explicit criteria for evaluating whether generated content satisfies the request.

Together, these studies advance visual generation from synthesizing visually plausible outputs toward understanding and fulfilling users' creative intent by broadening how that intent can be expressed, improving how models respond to it, and evaluating success beyond visual appearance.

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my supervisor, Professor Wenhu Chen, and my co-supervisor, Professor Jimmy Lin, for their guidance and support throughout my Ph.D. Wenhu encouraged me to pursue ambitious questions in visual generation and gave me the freedom to explore new ideas and collaborations. He taught me how to write papers and make thoughtful research decisions, and I am fortunate to have had him as a mentor throughout my Ph.D. Jimmy offered thoughtful advice and steady support as I navigated the program and brought this thesis to completion. Their complementary perspectives shaped this work and the way I approach research.

I would also like to thank the members of my examining committee, Professors Vicente Ordóñez-Román, David Clausi, Victor Zhong, and Yuntian Deng, for taking the time to read this thesis and participate in my defence. I am grateful for their expertise and constructive feedback, which strengthened this thesis and will inform my future research. I am also grateful for the support of the David R. Cheriton Scholarship, which allowed me to focus more fully on my research during my Ph.D.

The work in this thesis is the result of many collaborations. I am sincerely grateful to all of my co-authors and collaborators for sharing ideas, building systems, running experiments, and revising manuscripts together. This work would not have been possible without their creativity and persistence.

I would especially like to thank my labmates Weiming, Dongfu, Ge, Yubo, Xueguang, Max, Haozhe, Shuo, Yuxuan, and Xuan, as well as the other students in our lab, for their discussions, collaborations, and friendship. Their willingness to exchange ideas, help with experiments, and share both the difficulties and excitement of research made my Ph.D. experience far more rewarding.

I was also fortunate to intern with GenAI at Meta, where the work on MoCha was carried out. I am grateful to Bo, Haoyu, Ji, Felix, Zecheng, Xiaoliang, Luxin, Kunpeng, Tingbo, Animesh, and Peter for their collaboration and support. Working with this team taught me how to develop an ambitious research idea within a fast-moving environment and how sustained collaboration can turn that idea into a complete system.

I am thankful to the faculty, students, and staff of the David R. Cheriton School of Computer Science and to the broader research community at the University of Waterloo. Their conversations, administrative support, and collegial environment made this journey possible.

Finally, I owe my deepest gratitude to my parents for their unconditional love, patience, and confidence in me. They supported my choices when the path ahead was uncertain and

gave me the strength to continue through every challenge. I am especially grateful to my partner for her unwavering support, patience, and companionship throughout this journey.

Table of Contents

Examining Committee Membership ii

Author's Declaration iii

Statement of Contributions iv

Abstract vi

Acknowledgements viii

List of Figures xv

List of Tables xviii

1 Introduction 1

1.1 Challenges 2

1.2 Thesis Outline 5

1.3 Contributions 5

2 Background 7

2.1 Diffusion and Flow-Based Generative Models 7

2.1.1 Denoising Diffusion and Latent Representations 7

2.1.2 Flow Matching 8

2.1.3 Diffusion Transformers and Multimodal Attention 9
2.2 Image Editing and Multimodal Conditioning 9
2.2.1 Editing through Inversion and Attention Control 9
2.2.2 Instruction-Based Editing and Reference Conditions 10
2.2.3 Unified Understanding and Generation 11
2.3 Video Generation 11
2.3.1 Video Latents and Spatiotemporal Transformers 11
2.3.2 Speech-Conditioned Video Generation 12
2.3.3 Causal Generation and Few-Step Distillation 12
2.4 Reward Modeling and Post-Training 13
2.4.1 Learning from Human Preferences 13
2.4.2 Preference Optimization and Reinforcement Learning 14
2.4.3 Inference-Time Selection and Refinement 15
2.5 Retrieval and Tool-Augmented Generation 15
2.6 Evaluation of Visual Generation 16

3 Image and Video Generation and Editing with Multimodal Instructions 17
3.1 Overview 17
3.2 OmniEdit: Learning Generalist Image Editing from Specialists 18
3.2.1 Motivation 18
3.2.2 Related Work 20
3.2.3 Preliminaries 20
3.2.4 Method 21
3.2.5 Experiments 27
3.2.6 Discussion 34
3.3 AnyV2V: Tuning-Free Video Editing 35
3.3.1 Motivation 35
3.3.2 Related Work 35

3.3.3 Preliminaries . . . 37
3.3.4 Method . . . 38
3.3.5 Experiments . . . 41
3.3.6 Discussion . . . 49
3.4 UniVideo: Unifying Video Understanding, Generation, and Editing . . . 50
3.4.1 Motivation . . . 50
3.4.2 Related Work . . . 51
3.4.3 Architecture and Training . . . 52
3.4.4 Experiments . . . 54
3.4.5 Discussion . . . 65
3.5 Summary . . . 67

4 Temporal Control through Speech and Historical Context 68
4.1 Overview . . . 68
4.2 MoCha: Speech-Conditioned Video Generation . . . 69
4.2.1 Motivation . . . 69
4.2.2 Related Work . . . 70
4.2.3 Task Definition . . . 71
4.2.4 Speech-Conditioned Video Generation . . . 71
4.2.5 Experiments . . . 78
4.2.6 Discussion . . . 83
4.3 Context Forcing: Real-Time Long-Video Generation . . . 84
4.3.1 Motivation . . . 85
4.3.2 Related Work . . . 85
4.3.3 Method . . . 86
4.3.4 Experiments . . . 92
4.3.5 Discussion . . . 99
4.4 Summary . . . 99

5 Improving Instruction Adherence through Reward Feedback 101
5.1 Overview . . . 101
5.2 RationalRewards: Reasoning-Based Rewards for Visual Generation . . . 102
5.2.1 Motivation . . . 102
5.2.2 Related Work . . . 103
5.2.3 Method . . . 104
5.2.4 Experiments . . . 109
5.2.5 Discussion . . . 117
5.3 RewardHarness: Learning Evaluation Skills . . . 118
5.3.1 Motivation . . . 118
5.3.2 Related Work . . . 119
5.3.3 Method . . . 119
5.3.4 Experiments . . . 124
5.3.5 Discussion . . . 130
5.4 Summary . . . 131

6 Benchmarking World Knowledge and Visual Reasoning in Visual Generation 132
6.1 Overview . . . 132
6.2 SearchGen: Benchmarking World Knowledge in Visual Generation . . . 133
6.2.1 Motivation . . . 133
6.2.2 Related Work . . . 134
6.2.3 SearchGen-Bench: Construction and Evaluation . . . 135
6.2.4 Benchmark Findings . . . 141
6.2.5 Method . . . 145
6.2.6 Co-Training Experiments . . . 147
6.2.7 Discussion . . . 151
6.3 VGI-Bench: Probing Visual Intelligence in Video Generation Models . . . 151

6.3.1 Motivation . . . 151
6.3.2 Related Work . . . 152
6.3.3 Benchmark Design and Evaluation . . . 154
6.3.4 Experiments . . . 156
6.3.5 Diagnostic Findings . . . 159
6.3.6 Discussion . . . 164
6.4 Summary . . . 166

7 Conclusion and Future Work **167**
7.1 Practical Considerations and Limitations . . . 168
7.2 Unified Multimodal Models for Interactive Visual Creation . . . 168
7.3 Persistent Interactive World Simulation . . . 169
7.4 Reliable Evaluation and Dense Feedback for Visual Simulation . . . 170
7.5 Closing Remarks . . . 170

References **171**

List of Figures

1.1 Four challenges in controllable image and video generation 3

3.1 High-resolution, multi-aspect image editing with OmniEdit 19
3.2 Overview of the training pipeline 22
3.3 InternVL2 as a scoring function before (top right) and after (bottom right) fine-tuning on GPT-4o's response 23
3.4 Architecture comparison for OmniEdit's EditNet 26
3.5 Qualitative comparison of OmniEdit with image editing baselines 29
3.6 EditNet versus channel-wise concatenation 33
3.7 Content removal with EditNet and ControlNet variants 33
3.8 AnyV2V supports diverse video-editing tasks without parameter tuning . . 36
3.9 AnyV2V takes a source video as input 39
3.10 AnyV2V is robust in a wide range of prompt-based editing tasks while preserving the background 42
3.11 With different image editing models, AnyV2V can achieve a wide range of editing tasks, including reference-based style transfer 43
3.12 AnyV2V can edit video length beyond the training frame while maintaining motion consistency 47
3.13 Ablation of feature injection and DDIM inversion in AnyV2V 48
3.14 UniVideo is a unified system that can understand multi-modal instructions and generate multi-modal content 51
3.15 Model architecture 53

3.16 Qualitative comparison of UniVideo with SoTA Task Specific Experts on In Context Generation and In Context Editing tasks . . . 62
3.17 Zero-Shot Generalization . . . 63
3.18 Generation with thinking . . . 64
3.19 Qualitative results of generation with thinking . . . 64

4.1 Speech- and text-conditioned video generation with MoCha . . . 70
4.2 MoCha Architecture . . . 73
4.3 Localized audio attention in MoCha . . . 74
4.4 Multi-character Conversation and Character Tagging . . . 76
4.5 Character tagging provides more compact and structured prompts compared to naïve captioning . . . 76
4.6 Progressive multimodal training curriculum for MoCha . . . 78
4.7 Qualitative comparison of lip synchronization on MoCha-Bench . . . 80
4.8 Context Forcing mitigates the forgetting–drifting dilemma . . . 84
4.9 Training paradigms for AR video diffusion models . . . 86
4.10 Context Forcing and Context Management System . . . 87
4.11 Comparison on 1-min Video Generation . . . 89
4.12 Robust context teacher training with error recycling . . . 92
4.13 Visual artifacts in LongLive . . . 97
4.14 Human preference for Context Forcing against long-video baselines . . . 97
4.15 Ablation on Error-Recycling Fine-Tuning (ERFT) . . . 98

5.1 Generation quality and reward curves under different reward models . . . 103
5.2 Three-phase Preference-Anchored Rationalization pipeline . . . 105
5.3 Training-time RL and test-time prompt tuning with RationalRewards . . . 107
5.4 Test-Time Prompt Refinement via "Generate–Critique–Refine" loop with RationalRewards . . . 108
5.5 Qualitative comparison of base, RL-tuned, and prompt-tuned generators . . . 115

5.6 Training curves comparison between RationalRewards and scalar reward model, EditReward and MultiReward used in DiffusionNFT 116
5.7 Selected reward-hacking trajectories 117
5.8 Paradigm comparison 118
5.9 Overview of the RewardHarness self-evolution pipeline 120
5.10 Examples of a Skill and a Tool sampled from the Library at evolution iteration 69 122
5.11 Preference-scoring comparison on EditReward-Bench 128
5.12 Qualitative comparison on ImgEdit-Bench 129
5.13 Self-evolution dynamics over 77 iterations 130

6.1 Dataset composition: entity long-tail and multimodal knowledge gaps . . . 137
6.2 Knowledge-sensitive performance across nine generators 141
6.3 Failure examples from external search 145
6.4 Co-training progression and knowledge boundary shift 148
6.5 VGI-Bench task domains and skill tags 153
6.6 Per-model performance across the skill tags. 158
6.7 Representative cases in the failure modes. 160
6.8 Performance comparison in oracle prompting and in varying input visual styles. 161
6.9 Decoded maze frames at different denoising stages. 164

List of Tables

3.1 Comparison of OmniEdit with existing end-to-end image editing models . . 20
3.2 Task Definitions and Examples 24
3.3 OmniEdit training dataset statistics before and after quality filtering . . . 28
3.4 Main evaluation results on OmniEdit-Bench 30
3.5 Ablation on importance sampling 31
3.6 Ablation of OmniEdit architecture design 31
3.7 Comparison between OmniEdit and specialist models 34
3.8 Video editing methods and supported task types 36
3.9 Quantitative comparisons for our AnyV2V with baselines on prompt-based video editing 44
3.10 Comparisons for three I2V models under AnyV2V framework on novel video editing tasks 47
3.11 Ablation study results for AnyV2V(I2VGen-XL) 49
3.12 Visual understanding and video generation results 55
3.13 Text-to-image generation results on the GenEval benchmark 56
3.14 Image editing results on ImgEdit-Bench and GEdit-Bench-EN 56
3.15 Training hyperparameters across different stages 57
3.16 Quantitative comparison on In-Context Generation 60
3.17 Quantitative comparison with task-specific expert models on In-Context Video Editing 61
3.18 Ablation of joint training and visual conditioning on in-context tasks . . . 65

4.1 Talking Characters task: inputs, output, and evaluation criteria 72
4.2 Human evaluation scores on MoCha-Bench 79
4.3 Comparison with state-of-the-art methods on MoCha-Bench 81
4.4 Ablation study on MoCha-Bench 81
4.5 Comparison of audio cross-attention variants on MoCha-Bench 83
4.6 Ablation of character tagging on MoCha-Bench turn-based dialogue 83
4.7 Single-prompt 60-second video consistency evaluation 94
4.8 Video generation quality and throughput across architecture families . . . 95
4.9 Ablation study on Slow Memory Sampling Strategy, Context DMD, and Bounded Positional Encoding (evaluated on 60s) 98

5.1 Four-dimensional scoring anchors for RationalRewards 107
5.2 Training data composition before and after consistency filtering 109
5.3 Comparison of reward models as evaluators 110
5.4 Hyperparameters for RL fine-tuning via DiffusionNFT 112
5.5 Ablation of RationalRewards for Text-to-image RL on UniGenBench++ . 112
5.6 Ablation of RationalRewards as dual-space optimizer on editing tasks . . . 113
5.7 Comparison of image editing evaluators on editing reward benchmarks . . 125
5.8 Image editing results after RL fine-tuning with RewardHarness 127

6.1 Taxonomy of search-intensive visual generation 136
6.2 Released data resources for agentic visual generation 138
6.3 Evaluation components of the SearchGen-Bench judge 140
6.4 Knowledge bottleneck on the Search-Intensive subset of SearchGen-Bench . 142
6.5 SearchGen-Bench: overall and knowledge-sensitive scores 142
6.6 SearchGen-Bench: rendering and appearance scores 143
6.7 Comparison of search policies across three knowledge settings 145
6.8 Compute budget overview 147
6.9 Main results: the co-training progression 150

6.10 Comparison with related video reasoning benchmarks 152
6.11 Evaluation results of video generation models on VGI-Bench 156
6.12 Generation configurations of evaluated video models 157
6.13 VGI-Bench component means and strict success 158
6.14 VLM evaluator reliability and component ablations 159
6.15 Synthetic fine-tuning performance by training-distribution overlap 162
6.16 Domain and skill performance of VBVR-Wan2.2 and its base model 163
6.17 Solution-state transitions between decoded denoising steps 164

Chapter 1

Introduction

Diffusion and flow-based generative models have made substantial progress in synthesizing images and videos from textual descriptions. Advances in denoising-based generation, latent representations, and transformer architectures have made these models a reusable foundation for visual content creation [39, 56, 118, 129]. Building on this progress, users increasingly seek systems that can modify existing content, incorporate visual references and speech, combine different instructions, and support both long-video and real-time generation.

Traditional text-to-image and text-to-video models lack the controllability required for these practical applications. Across image and video tasks, users need flexible interfaces to express their intent through text, images, and videos. A system must identify the intended task and interpret how the supplied visual inputs should guide generation or editing. Video generation adds the question of how content unfolds: speech can constrain the timing of visible performance, while historical context can constrain a coherent continuation. Addressing these requirements involves decisions about training data, conditioning representations, model composition, temporal context, and inference efficiency.

Accepting these control inputs does not ensure that a model follows them, motivating reward feedback that identifies deviations and guides improvement. This thesis investigates such feedback in image generation and editing. Assessing control also requires looking beyond whether an output appears to satisfy a request. A text-to-image request may require accurate knowledge of an unfamiliar entity, while a video request may call for visual reasoning about a spatial or physical problem. In these cases, a model can produce a plausible image or video without satisfying the request. Progress therefore depends on both the range of tasks supported by a system and how successfully it meets their requirements.

This thesis studies these questions through nine works organized around four challenges: limited task flexibility, limited temporal control, missing robust and informative feedback, and evaluation blind spots beyond visual quality.

This thesis investigates how to give users greater control over image and video generation with foundation models, improve how faithfully the outputs follow their instructions, and identify the limits of this control.

In this thesis, *controllability* refers to how well a user can specify a requirement through the inputs to a generator, and how reliably the output meets that requirement. The inputs may be a text instruction, a source image or video, a reference image, a speech track, or frames generated earlier in the same video. Control therefore has two parts. The first is what a user can specify: which tasks a model supports and which inputs it can take. The second is whether the output meets what was specified, judged against the requirement of each task. An edit should change what the instruction asks for and leave the rest of the source unchanged. A talking character's lip motion should match the speech. A long video should stay consistent with its earlier frames. An image of a named entity should show that entity correctly, and a video generated for a reasoning task should actually solve it. Chapters 3 and 4 extend what users can specify. Chapter 5 improves how closely outputs meet the requirements, and Chapter 6 measures whether they are met when the requirements involve world knowledge or visual reasoning.

The phrase *foundation models* describes the reusable models on which the studied systems are built: visual generators, multimodal understanding models, and language-based reasoners. A generator produces visual content; an understanding model interprets conditions; a reward system judges candidate outputs; and a search agent selects external information. Some proposed methods update model parameters, while others change the inference procedure or the information supplied to a frozen model. The studies develop complementary methods under their own experimental settings, with a common concern for adapting pretrained capabilities to task requirements, improving outputs through feedback, and evaluating the limits of those capabilities.

1.1 Challenges

In practical image and video creation, users need to express different tasks through combinations of text, images, and videos. Compared with images, videos add a temporal dimension of control: speech must guide the unfolding performance, and earlier context must remain relevant as generation continues. Even when a system accepts these conditions, its outputs may not follow them, while useful feedback about such failures can be

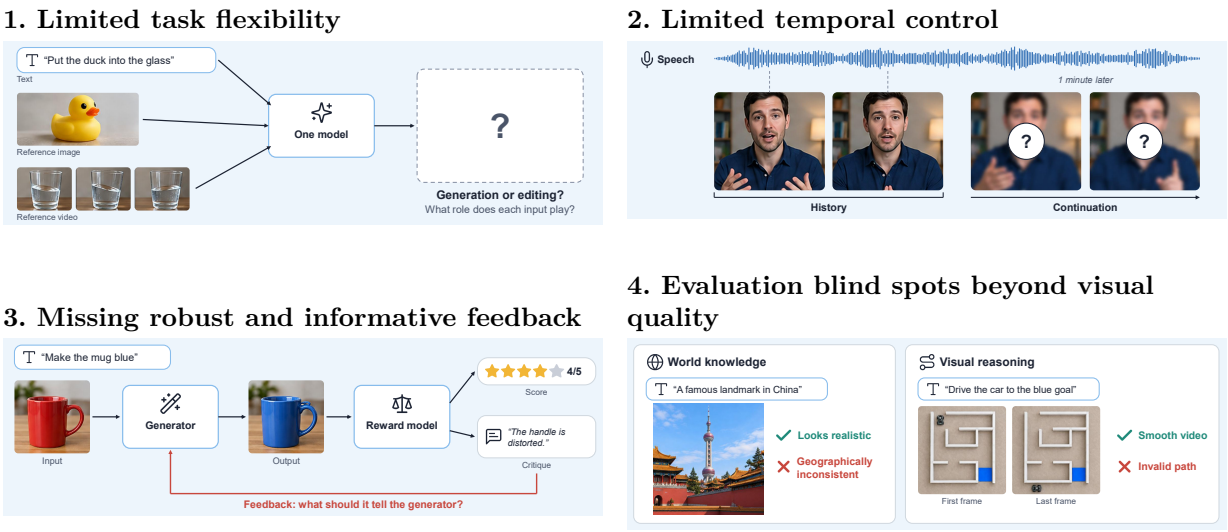


Figure 1.1: Four challenges in controllable image and video generation: limited task flexibility, limited temporal control, missing robust and informative feedback, and evaluation blind spots beyond visual quality. The examples are conceptual illustrations, not experimental results. Question marks indicate gaps not resolved by the available inputs or judgments alone.

difficult to obtain. Requests that depend on world knowledge or visual reasoning pose a further difficulty: success cannot be established from visual quality alone. These demands expose four challenges in making generation responsive to user intent, illustrated in Figure 1.1.

Limited task flexibility. Traditional text-to-image and text-to-video models synthesize visual content from text descriptions, but this capability alone does not meet the range of requests that arise in practical visual creation. Users may want to modify existing content, incorporate visual references, or combine text, images, and videos to specify a desired result. Supporting these requests requires more than accepting additional input modalities: a model must identify the intended task and interpret how each input should guide the output. The same image, for example, may provide content to edit, an appearance reference, or context for generating a video. Models and interfaces tailored to one operation or a fixed combination of inputs can limit this flexibility, requiring separate training or adaptation to support other requests. The challenge is to support diverse generation and editing tasks while interpreting multimodal instructions and the roles of their inputs appropriately.

Limited temporal control. Compared with image generation, video generation introduces a temporal dimension: users need to control not only what appears, but also how events unfold over time. Traditional text-to-video models guide this process through text descriptions, which offer limited control over precise timing and continuity. In movie generation, for example, users may want performances synchronized with a speech track and dialogue turns assigned to the intended speakers. Long-video creation also requires earlier context to constrain how subjects and scenes evolve as generation continues. These scenarios call for temporal conditions beyond text and for models that maintain their influence across frames, even over extended durations. Longer history further increases computation and memory demands, while real-time generation requires sustained throughput.

Missing robust and informative feedback. Accepting an instruction does not ensure that a model follows it, but recognizing a poor result does not automatically provide useful guidance for improvement. In image generation and editing, a judgment may need to account jointly for the instruction, source content, and visual quality. An attractive result can still omit a requested change, while an edit can satisfy one requirement at the expense of another. Overall preference scores may hide these distinctions, and detailed human feedback is costly to collect. Feedback must also remain useful when it is used to improve outputs. An evaluator that predicts held-out preferences accurately may reward superficial cues or provide an exploitable signal during optimization. The challenge is to obtain feedback that recognizes task-relevant errors and supports improvement, rather than merely assigning plausible scores to completed outputs.

Evaluation blind spots beyond visual quality. Some generation requests require knowledge of unfamiliar entities or the ability to solve visual reasoning tasks. A visually appealing image may depict the wrong entity or omit a defining cultural detail. Likewise, a coherent-looking video does not by itself show that a model can solve a spatial or physical reasoning problem. These requests call for explicit tests of knowledge and reasoning alongside visual quality and human preference. World knowledge is long-tailed and evolving, making it difficult to specify what a generator should know and to distinguish missing knowledge from failure to use supplied information. Visual reasoning poses a different evaluation challenge: performance can depend on task difficulty, input appearance, and physical or spatial constraints. Assessing these capabilities requires explicit success criteria and controlled comparisons that reveal what models can accomplish and where their limitations lie.

1.2 Thesis Outline

Chapter 2 introduces the common technical background. Chapter 3 studies flexible multimodal interfaces for image and video generation and editing through OmniEdit, AnyV2V, and UniVideo. Chapter 4 studies temporal control through speech and historical context with MoCha and Context Forcing. Chapter 5 develops reward feedback for instruction adherence in image generation and editing through RationalRewards and RewardHarness. Chapter 6 centers on benchmark design and diagnosis of world knowledge and visual reasoning through SearchGen-Bench and VGI-Bench. Chapter 7 draws together the findings, discusses their limitations, and identifies future research directions.

1.3 Contributions

This thesis makes four contributions, each addressing a corresponding challenge in controllable image and video generation.

Expanding task flexibility (Chapter 3). We develop OmniEdit, AnyV2V, and UniVideo to extend traditional generation capabilities to more diverse user requests. OmniEdit combines specialist supervision in one model to support multiple image-editing operations. AnyV2V transfers different first-frame editing operations to video through model composition, broadening the supported editing interfaces without task-specific video tuning. UniVideo addresses the need to interpret the task as well as the inputs: it combines multimodal understanding, generation, and editing in a shared model that interprets multimodal instructions and distinguishes the intended task. It supports nine image and video tasks and generalizes to unseen tasks. These approaches address both the range of requests a system can execute and the flexibility with which users can express them.

Extending temporal control (Chapter 4). We develop MoCha and Context Forcing to make speech and historical context effective controls over how videos unfold. MoCha incorporates audio conditioning into video generation to align visible performance with speech and coordinate multi-character dialogue, enabling movie-grade storytelling. Audio supplies timing and speaker cues beyond the text description. Context Forcing addresses the loss of historical constraints during long-video generation: a long-context teacher guides the use of earlier context, while Slow-Fast Memory balances recent detail with longer-term information. Together, these components mitigate forgetting and drift while supporting

real-time generation. The two studies extend control from the content of individual frames to speech-aligned performance and history-conditioned continuation.

Developing informative reward feedback (Chapter 5). We develop RationalRewards and RewardHarness to provide feedback that accounts for instruction adherence, source preservation, and visual quality, rather than reducing these requirements to an undifferentiated preference judgment. RationalRewards learns structured critiques from preference data, making feedback informative about errors and using scores and critiques to guide reinforcement learning and inference-time refinement. RewardHarness learns reusable evaluation guidelines and tools from a small set of preference demonstrations, adapting the evaluation procedure without updating model weights. We also evaluate the feedback through downstream image-generation and editing experiments to measure its effect on generated outputs. These experiments show that the learned feedback can improve outputs, rather than merely judge them.

Addressing evaluation blind spots beyond visual quality (Chapter 6). We introduce SearchGen-Bench and VGI-Bench to assess correctness that visual quality and broad preference judgments can overlook. SearchGen-Bench uses visual checklists and controlled retrieval settings to diagnose missing long-tailed and evolving knowledge and failures to use external evidence. Its findings motivate selective retrieval and co-training of a search agent and generator. VGI-Bench evaluates visual reasoning through task completion and compliance with physical and spatial constraints, distinguishing convincing-looking videos from successful solutions. Together, these benchmarks make knowledge and reasoning requirements explicit, expose persistent limitations, and provide targeted criteria for evaluating progress in controllable generation.

Across these contributions, the thesis connects versatile visual generation systems with the controls needed to perform a task, the feedback needed to improve performance, and the evaluation needed to measure task-specific success.

Chapter 2

Background

This chapter introduces the models, training objectives, and evaluation methods used in visual generation. It begins with diffusion and flow matching, then describes image editing, multimodal conditioning, and video generation. The remaining sections cover reward-based post-training, retrieval and tool use, and evaluation. Throughout this chapter, x denotes visual data, c denotes conditioning information, and θ denotes generator parameters. The variable t indexes diffusion or flow time; video time is indexed separately.

2.1 Diffusion and Flow-Based Generative Models

2.1.1 Denoising Diffusion and Latent Representations

Denoising diffusion probabilistic models (DDPMs) learn to reverse a process that progressively corrupts data with Gaussian noise [56]. Let x_0 be a clean sample. At a selected noise level t, the forward marginal and a noise-prediction training objective can be written as

$$x_t = \alpha_t x_0 + \sigma_t \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I), \tag{2.1}$$

$$\mathcal{L}_{\text{noise}}(\theta) = \mathbb{E}_{x_0,c,t,\epsilon}\left[w(t)\|\epsilon - \epsilon_\theta(x_t, t, c)\|_2^2\right]. \tag{2.2}$$

The schedule determines the signal and noise coefficients α_t and σ_t, and $w(t)$ weights the training contribution of each noise level. Although training uses independently sampled noise levels, generation applies the learned predictor repeatedly, starting from Gaussian

noise and moving toward a clean sample. DDPM sampling uses stochastic reverse transitions; DDIM constructs an alternative sampling family that includes a deterministic trajectory [135].

Latent diffusion reduces the cost of this process by moving denoising from pixels to a compressed representation [129]. An autoencoder maps an image to $z_0 = E(x_0)$, the generator models z_0, and a decoder produces the image $D(z_0)$. Spatial downsampling reduces the number of positions processed by the denoiser while retaining information needed for reconstruction. In Stable Diffusion, a U-Net operates on these latents, with text features supplied through cross-attention. The autoencoder's compression ratio and reconstruction fidelity determine the computational savings and retained image detail.

Classifier-free guidance (CFG) strengthens conditioning by combining conditional and unconditional predictions [55]. In the noise-prediction parameterization,

$$\widehat{\epsilon}_\theta(x_t, t, c) = \epsilon_\theta(x_t, t, \varnothing) + s\Big[\epsilon_\theta(x_t, t, c) - \epsilon_\theta(x_t, t, \varnothing)\Big], \tag{2.3}$$

where $\varnothing$ denotes a dropped condition and s is the guidance scale. Training randomly removes conditions so that the same network supports both predictions. Guidance adjusts the sampling direction while keeping the learned parameters fixed. The guidance scale is selected to balance conditioning strength, diversity, and image quality.

2.1.2 Flow Matching

Flow matching learns a velocity field that transports a simple distribution to a data distribution [91]. To keep clean data at $t = 0$ as above, consider the linear path

$$x_t = (1 - t)x_0 + t\epsilon, \qquad t \in [0, 1]. \tag{2.4}$$

The pathwise velocity is $\epsilon - x_0$. Conditional flow matching therefore minimizes

$$\mathcal{L}_{\text{flow}}(\theta) = \mathbb{E}_{x_0, c, t, \epsilon}\left[\|v_\theta(x_t, t, c) - (\epsilon - x_0)\|_2^2\right]. \tag{2.5}$$

Generation integrates $dx_t/dt = v_\theta(x_t, t, c)$ backward from $t = 1$ to $t = 0$. Other presentations reverse the time convention, in which case the path and velocity signs change together.

Rectified-flow image generators, including Stable Diffusion 3, combine this formulation with latent representations and transformer backbones [39]. The noise-level sampling distribution determines which portions of the path receive more training emphasis, while the solver and number of integration steps determine inference cost.

2.1.3 Diffusion Transformers and Multimodal Attention

A Diffusion Transformer (DiT) replaces the convolutional U-Net denoiser with a transformer operating on latent patches [118]. For a latent grid of size $H_\ell \times W_\ell$, patches of size $p \times p$ yield $N = H_\ell W_\ell / p^2$ tokens. A learned projection embeds each patch, positional information distinguishes locations, and a final projection reconstructs the predicted latent patches. Attention allows spatially distant regions to interact:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}} + M\right)V, \tag{2.6}$$

where Q, K, and V are projected queries, keys, and values, d_k is the key dimension, and M is an optional attention mask. Dense self-attention forms an $N \times N$ interaction matrix, so token count strongly affects computation and memory.

DiT supplies diffusion-time and class information through adaptive layer normalization. For a token representation h and conditioning embedding e, the modulation has the form

$$\text{adaLN}(h, e) = \big(1 + \gamma(e)\big) \odot \text{LN}(h) + \beta(e). \tag{2.7}$$

Its adaLN-Zero blocks also predict residual gates initialized to zero, making the blocks initially behave as identity mappings. This provides a concrete way to inject global conditions throughout the network rather than appending them only at the input.

For text-conditioned generation, cross-attention uses visual queries and text-derived keys and values. The multimodal diffusion transformer (MMDiT) in Stable Diffusion 3 instead uses separate text and image processing weights with joint attention over the two token sequences [39]. Text and visual representations can consequently be updated through repeated interaction. The term *diffusion transformer* describes this family of denoising architectures even when the network is trained with a flow-matching objective.

2.2 Image Editing and Multimodal Conditioning

2.2.1 Editing through Inversion and Attention Control

Image editing conditions generation on a source image x_{src} and an instruction u, with the desired distribution $p_\theta(x_{\text{edit}} \mid x_{\text{src}}, u)$. Unlike text-to-image generation, the model must retain source content outside the requested transformation. Early diffusion editors obtain this correspondence through the sampling process rather than paired editing supervision.

SDEdit adds an intermediate amount of noise to a source image and then denoises under the target condition [107]. The noise level controls a practical trade-off: greater corruption allows larger changes but discards more source structure. Inversion instead attempts to recover a noise state or trajectory from which the source can be reconstructed. Null-Text Inversion optimizes unconditional text embeddings along a DDIM-based reconstruction trajectory, allowing subsequent prompt changes while retaining the recovered image structure [109].

Prompt-to-Prompt controls cross-attention maps associated with words in the prompt, preserving spatial organization while changing selected content [53]. Blended Diffusion uses masks to combine edited and preserved regions [3]. These mechanisms supply different controls: a mask identifies where to edit, attention links prompt tokens to spatial locations, and inversion anchors the computation to an existing image. Extending edits to video also requires correspondence over time; TokenFlow propagates diffusion features using inter-frame correspondences to promote consistent edits [42].

2.2.2 Instruction-Based Editing and Reference Conditions

InstructPix2Pix treats editing as supervised learning from triples $(x_{\text{src}}, u, x_{\text{edit}})$ [13]. It creates training pairs using language-generated editing instructions and diffusion-generated before-and-after images. The source-image latent is concatenated with the noisy target latent, while the instruction enters through text conditioning. A representative training objective is

$$\mathcal{L}_{\text{edit}} = \mathbb{E}\left[\|\epsilon - \epsilon_\theta(z_t, t, E(x_{\text{src}}), u)\|_2^2\right], \tag{2.8}$$

where z_t is a noised target-image latent. This trains an editor that accepts a new instruction without optimizing its parameters for each source image. Separate guidance terms for the image and text conditions control source preservation and instruction strength.

Other conditioning mechanisms target information that is difficult to express in text. ControlNet introduces a trainable conditioning branch connected to a frozen diffusion backbone through zero-initialized convolutions, allowing spatial inputs such as edges, depth maps, or poses to guide generation [198]. IP-Adapter encodes a reference image and uses decoupled cross-attention for image and text features [185]. Spatial conditions specify geometry, image references convey identity, appearance, or style, and editing instructions describe the requested transformation. These inputs provide complementary information for conditioning a visual generator.

2.2.3 Unified Understanding and Generation

Multimodal language models typically encode images into features that support textual interpretation. A visual generator, however, must also recover color, texture, and spatial detail. Unified systems connect these two functions through different representations and objectives. Chameleon represents text and images as discrete tokens and trains an autoregressive model over mixed-modal sequences [140]. Transfusion combines next-token prediction for text with diffusion over continuous image representations within a shared transformer [210]. Janus separates visual encoding for understanding and generation while using a unified transformer to process the resulting representations [165].

These designs organize shared computation and task-specific representations in different ways, from a common discrete token space to separate visual encodings and training objectives. In connector-based designs, a projection or learned query module maps language-model features into the condition space of a separate visual generator. Which components are frozen, which are trained, and whether source-image details reach the generator directly determine what information the system can preserve.

2.3 Video Generation

2.3.1 Video Latents and Spatiotemporal Transformers

A video generator must model both spatial appearance and temporal change. One approach extends a pretrained image diffusion model with temporal layers. Video LDM and AnimateDiff add temporal modeling to image-generation backbones, reusing their spatial representations while learning motion from video [11, 51]. Factorized designs apply spatial attention within a frame and temporal attention across frames, reducing the cost relative to unrestricted attention over all space-time locations.

Video transformers instead operate on tokens derived from a spatiotemporally compressed video. A video autoencoder maps F frames to a latent tensor with temporal extent F_ℓ, spatial dimensions H_ℓ, W_ℓ, and latent channels. Patches of size $p_t \times p_h \times p_w$ produce

$$N = \frac{F_\ell}{p_t}\frac{H_\ell}{p_h}\frac{W_\ell}{p_w} \tag{2.9}$$

tokens when the dimensions are divisible by their patch sizes. Joint spatiotemporal attention lets any visible token interact with the others; factorized or windowed attention

restricts this interaction to reduce cost. Position encodings must distinguish temporal order as well as spatial location.

Movie Gen illustrates a large-scale video-transformer pipeline: a temporal autoencoder compresses video, a 30B-parameter transformer predicts flow velocities in latent space, and text encoders supply conditioning [120]. CogVideoX similarly combines a video autoencoder with transformer-based text–video modeling [183]. The compressor determines token resolution, the backbone determines interactions among tokens, and the sampling procedure determines how many network evaluations are needed.

2.3.2 Speech-Conditioned Video Generation

Speech-driven animation requires temporal alignment between audio and visible motion. Audio features are extracted at a sequence of time positions and associated with the corresponding video frames or latent intervals. A frame can attend to neighboring audio features to capture phonetic context, while identity and appearance can be supplied by a reference image. Temporal compression must be accounted for when mapping audio timestamps to video tokens.

Talking-head methods differ in whether they explicitly predict an intermediate motion representation. SadTalker predicts 3D motion coefficients from speech and uses them to animate a portrait [203]. EMO conditions a diffusion-based portrait animator on audio without requiring such an explicit intermediate 3D face representation [144]. Speech-conditioned performance involves synchronizing mouth motion with speech and coordinating facial expression, head movement, and body gestures across their different temporal scales.

2.3.3 Causal Generation and Few-Step Distillation

A bidirectional video denoiser processes a clip jointly, allowing tokens to use context from earlier and later video positions. Autoregressive generation instead factorizes the video into ordered chunks:

$$p_\theta(x^{1:K} \mid c) = \prod_{k=1}^{K} p_\theta(x^k \mid x^{<k}, c). \tag{2.10}$$

Each chunk predictor can itself be a diffusion or flow model. There are consequently two time axes: the index k of the generated video chunk and the noise level t within its

sampling process. Block-causal attention permits interactions within the current chunk and with earlier chunks, but masks future chunks.

CausVid distills a bidirectional video diffusion teacher into a causal generator, combining blockwise generation with a reduced number of denoising steps [192]. Cached keys and values allow previous chunks to supply context without recomputing all their representations for every new chunk. Cache size still grows with retained history; sliding windows bound this cost but remove access to older content.

Training on clean ground-truth history creates a mismatch when inference uses imperfect generated history. Self Forcing addresses this by rolling out the autoregressive model during training and conditioning subsequent predictions on its own outputs [63]. Such rollout-based training exposes the model to the states it encounters during generation.

Few-step distillation complements autoregressive streaming by reducing the number of denoiser evaluations per chunk. Distribution Matching Distillation trains a student using differences between score estimates for the teacher distribution and the student's generated distribution [191]. Consistency models learn mappings from points on a probability-flow trajectory to a common clean endpoint, supporting generation with fewer evaluations [136]. Generation speed is assessed through output frame rate and startup latency, alongside memory usage and the duration of the generated video.

2.4 Reward Modeling and Post-Training

2.4.1 Learning from Human Preferences

A preference dataset pairs a condition c with preferred and dispreferred outputs (x^+, x^-). A scalar reward model $r_\phi(c, x)$ can be trained using the Bradley–Terry preference model:

$$P_\phi(x^+ \succ x^- \mid c) = \text{sigmoid}\left(r_\phi(c, x^+) - r_\phi(c, x^-)\right). \tag{2.11}$$

Minimizing negative log-likelihood encourages higher scores for preferred outputs. The learned score differences predict the relative preference between outputs under the same condition.

ImageReward learns text-to-image preferences with a multimodal model, while PickScore adapts a CLIP-based scoring model using human comparisons [76, 175]. Editing rewards must additionally observe the source image: EditReward assesses the instruction

and before-and-after images to account for both the requested change and preservation [168]. Generative evaluators can produce textual judgments or rationales before a decision rather than only a scalar [105].

2.4.2 Preference Optimization and Reinforcement Learning

Direct Preference Optimization (DPO) uses preference pairs to update a policy relative to a reference policy without fitting a separate scalar reward model [127]. At the distribution level, define

$$\Delta_\theta = \log \frac{p_\theta(x^+ \mid c)}{p_{\text{ref}}(x^+ \mid c)} - \log \frac{p_\theta(x^- \mid c)}{p_{\text{ref}}(x^- \mid c)}, \tag{2.12}$$

$$\mathcal{L}_{\text{DPO}} = -\mathbb{E}\left[\log \text{sigmoid}(\beta \Delta_\theta)\right]. \tag{2.13}$$

Here $\beta > 0$ controls the scale of the reference-relative preference margin. Unlike a language model's token likelihood, a diffusion model's clean-image likelihood is generally not available as a simple tractable expression. Diffusion-DPO derives a diffusion-specific objective using a variational formulation and denoising losses [149].

Online reinforcement learning instead samples outputs and optimizes their rewards. DDPO treats successive denoising transitions as policy actions and assigns a reward to the completed image [8]. The policy-gradient estimator uses the log probabilities of those stochastic transitions, so a non-differentiable evaluator can supply the reward without backpropagation through it.

Flow-GRPO applies group-relative policy optimization to flow-matching generators [94]. For several outputs sampled under one prompt, rewards are normalized within the group to form advantages $A_i = (R_i - \bar{R})/(\text{std}(R) + \varepsilon)$. A clipped surrogate uses the transition probability ratio $\rho_{i,t}$ between the current and sampling policies:

$$\mathbb{E}_{i,t}\left[\min\left(\rho_{i,t} A_i,\ \text{clip}(\rho_{i,t}, 1-\delta, 1+\delta) A_i\right)\right]. \tag{2.14}$$

Here δ sets the clipping range; regularization can additionally constrain changes from a reference model. To obtain stochastic transitions with computable densities, Flow-GRPO converts the flow ODE to an SDE designed to preserve its time marginals. It also reduces sampling steps during training.

DiffusionNFT takes a different route: it uses rewarded clean samples and forward-process flow-matching objectives, contrasting positive and negative generations to define an improvement direction [209]. It does not require likelihood estimates for the reverse sampling trajectory. This separates sample generation from the solver-specific transition ratios used by policy-gradient approaches.

2.4.3 Inference-Time Selection and Refinement

Feedback can improve outputs without updating generator parameters. Best-of-n selection samples several candidates and returns the one with the highest evaluator score. Iterative refinement instead uses a critique to revise the prompt or editing instruction, generates a new candidate, and repeats. PromptEnhancer and ReasonEdit exemplify prompt refinement and feedback-driven editing workflows [153, 189].

Selection spends computation on additional candidates; refinement spends it on both evaluation and another generation round. Because both procedures optimize the evaluator's score, their downstream gains are measured with independent judgments of output quality. Evaluation thus considers preference accuracy, independently assessed generation quality, and computational cost.

2.5 Retrieval and Tool-Augmented Generation

A parametric generator stores information implicitly in its weights. Retrieval supplies an additional, explicit context r, yielding a conditional model $p_\theta(x \mid c, r)$ rather than relying only on $p_\theta(x \mid c)$. In language generation, RAG couples retrieved documents to a generator [84]. Visual retrieval can instead return image–text pairs or image features that describe the appearance of an unfamiliar subject.

Re-Imagen conditions image synthesis on retrieved image–text examples, while Retrieval-Augmented Diffusion Models use retrieved representations from an external database [10, 20]. The retrieval index determines which evidence is available; the conditioning pathway determines how it affects synthesis. This differs from concept-specific fine-tuning, which changes the generator's parameters.

Tool-using language models extend this pipeline into an iterative interaction. ReAct interleaves reasoning with actions and observations: the model chooses a tool call, receives its result, and selects the next action using the accumulated context [184]. For generation, possible actions include retrieving a reference, revising a prompt, invoking a generator, or inspecting an output. Adaptive retrieval methods such as Self-RAG make retrieval conditional on the request and intermediate judgments rather than treating it as mandatory [2].

Feedback can also update the agent's context rather than its weights. Reflexion stores verbal reflections on previous attempts to guide subsequent decisions [133]. Prompts, memories, and reusable tools are external artifacts, whereas supervised fine-tuning or policy optimization changes the model itself. This distinction matters when comparing adaptation costs and when using a fixed model through an API.

2.6 Evaluation of Visual Generation

Distributional quality and condition alignment. Fréchet Inception Distance (FID) compares Gaussian approximations to real and generated image-feature distributions [54]:

$$\mathrm{FID} = \|\mu_r - \mu_g\|_2^2 + \mathrm{Tr}\Big(\Sigma_r + \Sigma_g - 2(\Sigma_r^{1/2}\Sigma_g\Sigma_r^{1/2})^{1/2}\Big)\,. \tag{2.15}$$

The means and covariances summarize sets of feature vectors, making FID a measure of distributional similarity between generated and real images. Fréchet Video Distance applies the analogous comparison to video features [148].

At the level of individual image–prompt pairs, CLIP-based similarity measures semantic alignment in a joint embedding space [126]. Fine-grained checks of object counts and relations complement this embedding-level similarity: GenEval tests object presence, count, color, and position [44], while TIFA decomposes a prompt into visual questions and evaluates answers using visual question answering [60]. For image editing, assessment covers both adherence to the instruction and preservation of source content outside the requested change.

Video consistency and human judgments. VBench decomposes video quality into dimensions such as subject consistency, background consistency, motion smoothness, and text alignment [64]. Speech-driven generation also requires audio–visual synchronization measures, as used in lip-synchronization systems [121]. Human pairwise comparisons complement these measurements by assessing gesture naturalness and interaction quality using specified annotation criteria.

Reasoning, knowledge, and evaluator reliability. Video-reasoning benchmarks such as TiViBench and V-ReasonBench evaluate spatial, physical, and logical tasks through generated sequences [18, 100]. Evaluation examines both final outcomes and intermediate states, including compliance with task rules. Knowledge-intensive assessment checks whether the depicted entity, attributes, or relations agree with external evidence.

VLM-based judges scale these assessments by producing automated judgments of generated outputs. Their reliability is assessed through agreement with human preferences, sensitivity to prompts or presentation order, and performance across task categories. Gecko combines fine-grained prompt coverage with human-grounded evaluation [162], providing an example of evaluating automatic metrics against explicit assessment criteria.

Chapter 3

Image and Video Generation and Editing with Multimodal Instructions

This chapter is based on the following works. Author contributions are described in the Statement of Contributions.

- Cong Wei, Zheyang Xiong, Weiming Ren, Xinrun Du, Ge Zhang, and Wenhu Chen. *OmniEdit: Building Image Editing Generalist Models Through Specialist Supervision*. In Proceedings of the *International Conference on Learning Representations* (ICLR), 2025. (Section 3.2)
- Max Ku, Cong Wei, Weiming Ren, Harry Yang, and Wenhu Chen. *AnyV2V: A Tuning-Free Framework For Any Video-to-Video Editing Tasks*. Published in *Transactions on Machine Learning Research* (TMLR), 2024. (Section 3.3)
- Cong Wei, Quande Liu, Zixuan Ye, Qiulin Wang, Xintao Wang, Pengfei Wan, Kun Gai, and Wenhu Chen. *UniVideo: Unified Understanding, Generation, and Editing for Videos*. In Proceedings of the *International Conference on Learning Representations* (ICLR), 2026. (Section 3.4)

3.1 Overview

This chapter studies how to support diverse image and video generation and editing tasks through flexible instruction interfaces. Users may express their intent through text, images,

and videos, with visual inputs serving as content to edit or references to guide generation. Supporting these requests requires identifying the intended task and using the supplied inputs accordingly.

OmniEdit (Section 3.2) learns a broad range of image-editing operations through specialist supervision. AnyV2V (Section 3.3) extends flexible image editing to the video domain by propagating an edited first frame throughout a video. By composing an image editor with an image-to-video generator, it brings diverse prompt- and reference-based editing tasks to video without task-specific fine-tuning. UniVideo (Section 3.4) brings multimodal understanding, generation, and editing into a shared model that interprets multimodal instructions and distinguishes the intended task. It supports nine image and video tasks and demonstrates strong generalization to unseen tasks.

The studies broaden task coverage through specialist supervision, extend interface flexibility through model composition, and unify task understanding and execution through multimodal learning.

3.2 OmniEdit: Learning Generalist Image Editing from Specialists

3.2.1 Motivation

Instruction-based image editing requires changing the requested content while preserving the rest of the source image. As discussed in Chapter 2, a model can learn this behavior from source–instruction–target triples. The difficulty is obtaining supervision that covers both localized operations, such as object replacement, and global transformations, such as style or environment changes.

Synthetic editing pairs make large-scale training feasible, but the data generator determines which edits succeed and which errors the student encounters. Prompt-to-Prompt [53] and inpainting-based pipelines offer different strengths; neither should be assumed to supply equally reliable supervision for every operation. Quality filtering and the distribution of training resolutions introduce further constraints. Table 3.1 presents a preliminary comparison across seven editing skills and motivates a more diverse source of supervision.

OmniEdit combines seven task-specific specialists to train a generalist editor. A multimodal evaluator scores the resulting pairs for importance weighting, and the EditNet architecture connects source-image information with the generative backbone. Training

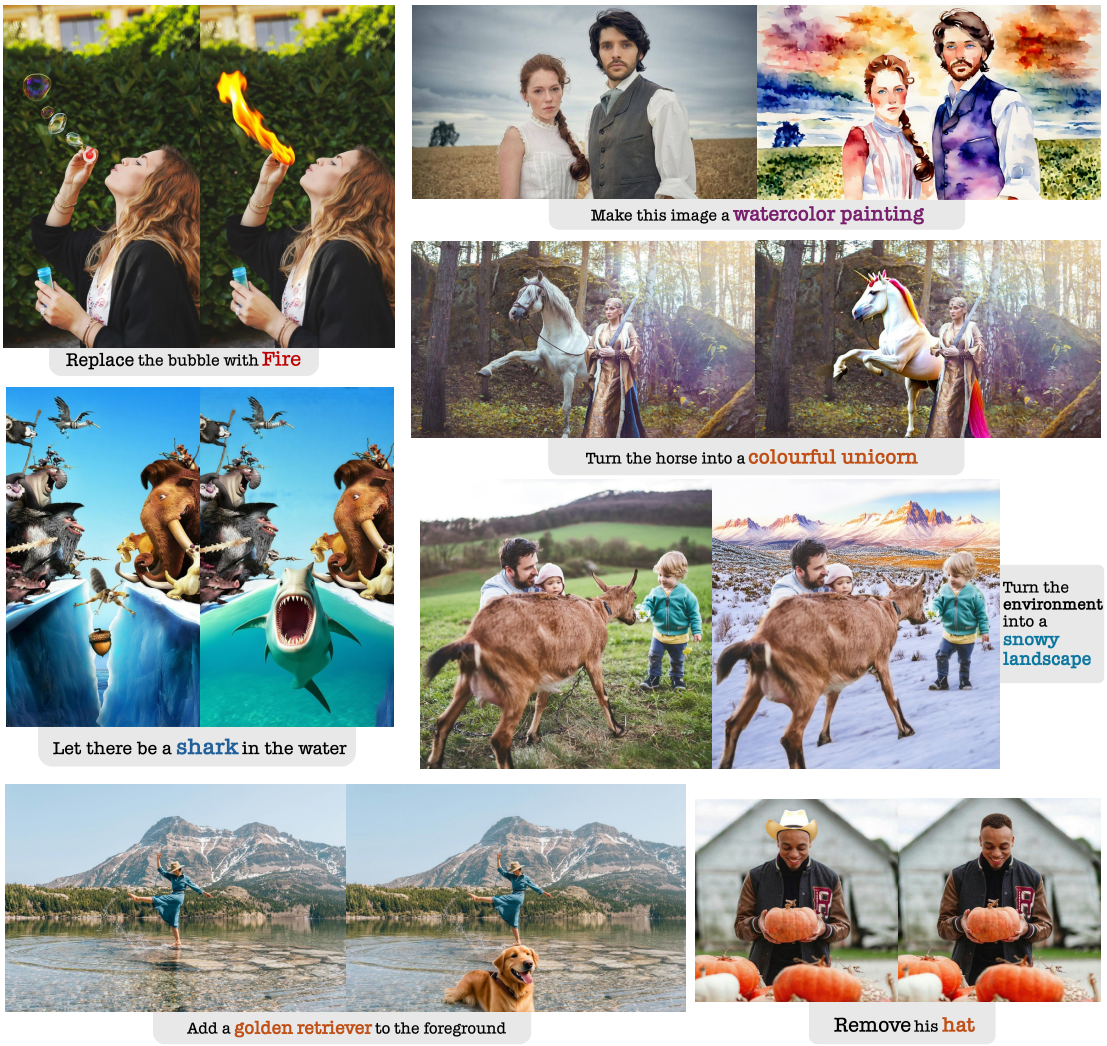


Figure 3.1: **Editing high-resolution multi-aspect images with OmniEdit.** OmniEdit is an instruction-based image editing generalist capable of performing diverse editing tasks across different aspect ratios and resolutions. It accurately follows instructions while preserving the original image's fidelity.

includes different resolutions and aspect ratios. The section develops this learning objective and data pipeline, then evaluates instruction adherence and source preservation using automatic metrics, human judgments, and controlled ablations.

Table 3.1: Comparison of OmniEdit with the evaluated end-to-end image editing models. Scores are based on a preliminary study of approximately 50 prompts.

Property	InstructP2P	MagicBrush	UltraEdit	MGIE	HQEdit	CosXL	OmniEdit
Training Dataset Properties							
Real Image?	✗	✓	✓	✓	✗	✗	✓
Any Res?	✗	✗	✗	✗	✗	✗	✓
High Res?	✗	✗	✗	✗	✓	✗	✓
Fine-grained Image Editing Skills							
Obj-Swap	★★☆	★★☆	★★☆	★½☆	★★☆	★☆☆	★★½
Obj-Add	★☆☆	★★☆	★☆☆	★½☆	★☆☆	★☆☆	★★½
Obj-Remove	★☆☆	★★☆	★☆☆	★½☆	★☆☆	★☆☆	★★½
Attribute	★★☆	★☆☆	★★☆	★½☆	★☆☆	★☆☆	★★½
Back-Swap	★★☆	★★☆	★★☆	★½☆	★★☆	★★☆	★★½
Environment	★☆☆	★☆☆	★☆☆	★½☆	★☆☆	★★☆	★★½
Style	★★☆	★☆☆	★★☆	★½☆	★☆☆	★★½	★★½

3.2.2 Related Work

Chapter 2 distinguishes inference-time editing from supervised adaptation. Representative inference-time methods include SDEdit, Prompt-to-Prompt, and Null-Text Inversion [53, 107, 109]. OmniEdit instead follows the instruction-supervised setting exemplified by InstructPix2Pix [13].

The relevant distinction is the source and coverage of supervision. MagicBrush uses human-guided data construction, HQ-Edit uses generated image pairs, and Emu Edit and UltraEdit develop large-scale editing datasets [65, 131, 197, 207]. OmniEdit combines specialists with different editing strengths and scores their outputs before generalist training.

3.2.3 Preliminaries

3.2.3.1 Text-to-Image Diffusion Models

Diffusion models [56, 135] are a class of latent variable models parameterized by θ, defined as $p_\theta(\mathbf{x}_0) := \int p_\theta(\mathbf{x}_{0:T})\, d\mathbf{x}_{1:T}$, where $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ represents the original data, and $\mathbf{x}_1, \ldots, \mathbf{x}_T$ are progressively noisier latent representations of the input image $\mathbf{x}_0$. Throughout the process, the dimensionality of $\mathbf{x}_0$ and the latent variables $\mathbf{x}_{1:T}$ remains consistent, with $\mathbf{x}_{0:T} \in \mathbb{R}^d$, where d corresponds to the product of the image's height, width, and channels. The forward (diffusion) process, denoted as $q(\mathbf{x}_{1:T}|\mathbf{x}_0)$, is a predefined Markov chain that incrementally adds Gaussian noise to the data according to a pre-defined schedule $\{\beta_t\}_{t=1}^T$.

The process of forward diffusion is defined as:

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}), \quad q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t\mathbf{I}), \tag{3.1}$$

where $\mathcal{N}$ denotes a Gaussian distribution, and β_t controls the amount of noise added at each step. The objective of diffusion models is to reverse this diffusion process by learning the distribution $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, which enables the reconstruction of the original data $\mathbf{x}_0$ from a noisy latent $\mathbf{x}_t$. This reduces to a denoising problem where the model ϵ_θ is trained to denoise the sample $\mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{x}_0)$ back into $\mathbf{x}_0$. The maximum log-likelihood training objective breaks down to minimizing the weighted mean squared error between the model's prediction $\hat{\mathbf{x}}_\theta(\mathbf{x}_t, c)$ and the true data $\mathbf{x}_0$:

$$\arg\max_\theta \log p_\theta(\mathbf{x_0}|c) = \arg\min_\theta \mathbb{E}_{(\mathbf{x}_0,c)\sim\mathcal{D}} \left[\mathbb{E}_{\epsilon,t}\left[w_t \cdot \|\hat{\mathbf{x}}_\theta(\mathbf{x}_t, c) - \mathbf{x}_0\|_2^2\right]\right], \tag{3.2}$$

where $(\mathbf{x}_0, c)$ pairs come from the dataset $\mathcal{D}$, with c representing the text prompt. The term w_t is a weighting factor applied to the loss at each timestep t. For simplicity, prior papers [56, 75, 135] will set w_t to be 1.

3.2.3.2 Instruction-Based Image Editing in Supervised Learning

Instruction-based image editing can be formulated as a supervised learning problem. Existing methods [13, 197] often adopt a paired training dataset of text editing instructions and images before and after the edit. An image editing diffusion model is then trained on this dataset. The latent diffusion objective is defined as:

$$\arg\max_\theta \log p_\theta(\mathbf{x'_0}|\mathbf{x_0}, c) = \arg\min_\theta \mathbb{E}_{(\mathbf{x}'_0,\mathbf{x}_0,c)\sim\mathcal{D}} \left[\mathbb{E}_{\epsilon,t}\|\hat{\mathbf{x}}_\theta(\mathbf{x}_t, c) - \mathbf{x}'_0\|_2^2\right], \tag{3.3}$$

where $(\mathbf{x}'_0, \mathbf{x}_0, c)$ triples are sampled from the dataset $\mathcal{D}$ with $\mathbf{x}_0$ denoting the source image, c denoting the editing instruction and $\mathbf{x}'_0$ denoting the target image.

3.2.4 Method

In this section, we introduce the entire specialist-to-generalist learning framework to build OmniEdit. We describe the overall learning objective in Section 3.2.4.1. We then describe how we learn the specialists in Section 3.2.4.2 and the importance weighting function in Section 3.2.4.3. In Figure 3.2, we show the overview of the OmniEdit training pipeline.

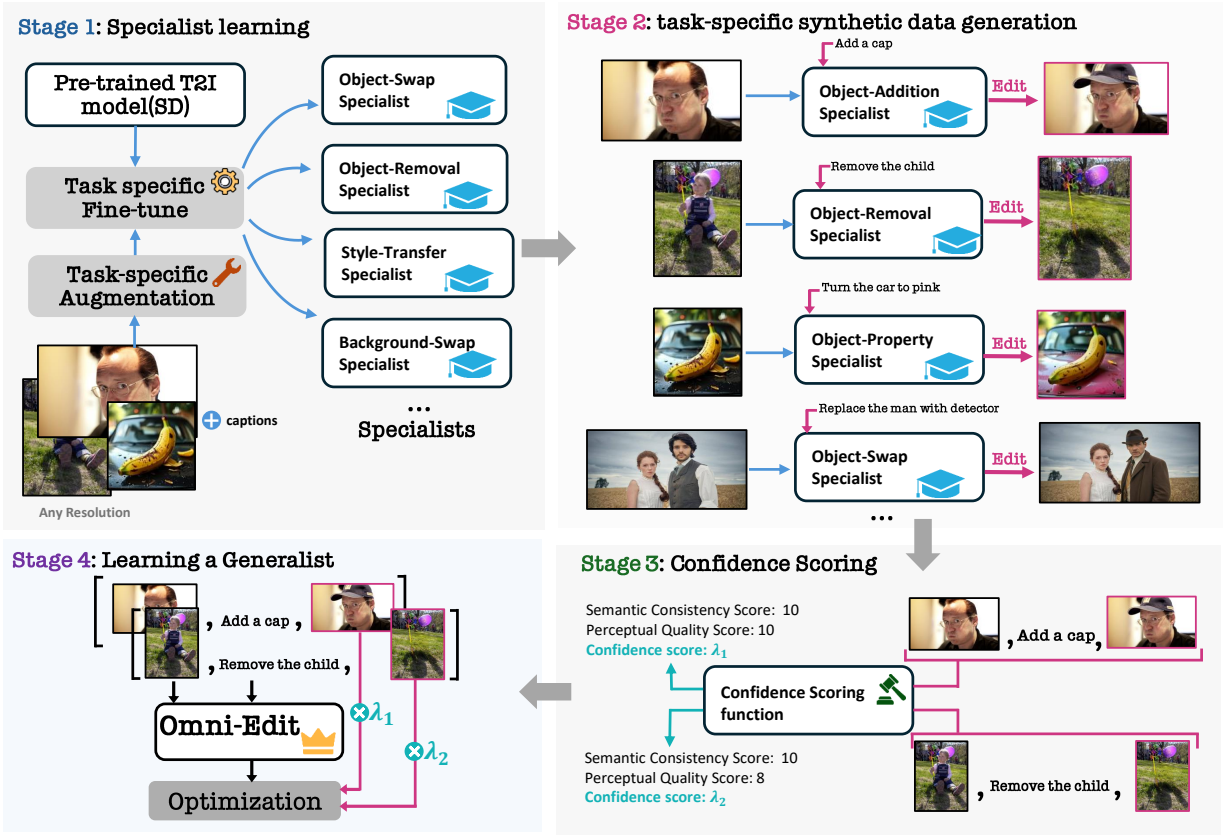
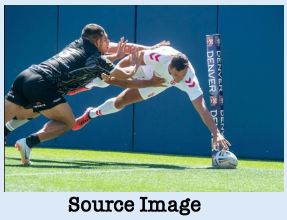


Figure 3.2: Overview of the OmniEdit training pipeline. The pipeline consists of four stages: (1) task-specific specialist models are trained for diverse editing tasks; (2) these specialist models are used to generate a large, high-resolution, multi-aspect-ratio dataset; (3) a cost-efficient distilled large multi-modal model (LMM) assigns importance weights to each pair of image-editing data; and (4) the final generalist model is trained on the weighted dataset.

3.2.4.1 Learning Objective

We assume there is a groundtruth editing model $p(\mathbf{x}'|\mathbf{x}, c)$, which can perform any type of editing tasks perfectly according to the instruction c. Our goal is to minimize the divergence between $p_\theta(\mathbf{x}'|\mathbf{x}, c)$ and $p(\mathbf{x}'|\mathbf{x}, c)$ by updating the parameters θ:

$$L(\theta) := \sum_{\mathbf{x},c} D_{KL}(p(\mathbf{x}'|\mathbf{x}, c)\|p_\theta(\mathbf{x}'|\mathbf{x}, c)) = -\sum_{\mathbf{x},c}\sum_{\mathbf{x}'} p(\mathbf{x}'|\mathbf{x}, c)\log p_\theta(\mathbf{x}'|\mathbf{x}, c) + C \tag{3.4}$$



Source Image



Edited Image

Editing Instruction:
Replace the rugby player with astronaut in space suit

InternVL2:
PQ:
{ **reasoning**: ....**There are no visible distortions**...The lighting and shadows are consistent, **score**: **10** }
SC:
{ **reasoning**: ...**The edited image perfectly follows this instruction**. The rugby player is completely replaced by an astronaut in a space suit... The editing is effective and matches the instruction without any overediting...,
score: **[10, 10]**}

InternVL2-Distilled-from-GPT4o:
PQ:
{**reasoning**: The image has some **notable distortions**, **especially in body proportions**, as the legs and arms of the individuals appear unnaturally elongated...,
score: **4**}
SC:
{**reasoning**: The editing **partially follows the instruction** by adding padded suits that slightly resemble space suits**, but it doesn't convincingly replace the players with astronauts**... The players' bodies, poses, and proportions remain largely intact, **making it feel like an incomplete transformation.**
score: **[5, 6]**}

Figure 3.3: InternVL2 as a scoring function before (top right) and after (bottom right) finetuning on GPT-4o's response. On the top right, the original InternVL2 fails to identify the unusual distortions in the edited image it also does not spot the error when the edited image fails to meet the specified editing instructions. On the bottom right, finetuned-InternVL2 successfully detects such failures and serves as a reliable scoring function.

where C is a constant, which we leave out in the following derivation. However, since we don't have access to $p(\mathbf{x}'|\mathbf{x}, c)$, we adopt importance sampling for approximation:

$$\begin{aligned} L(\theta) &= -\sum_{\mathbf{x},c}\sum_{\mathbf{x}'} q(\mathbf{x}'|\mathbf{x}, c)\frac{p(\mathbf{x}'|\mathbf{x}, c)}{q(\mathbf{x}'|\mathbf{x}, c)}\log p_\theta(\mathbf{x}'|\mathbf{x}, c) \\ &\approx -\mathbb{E}_{(\mathbf{x},c)\sim D}\left[\mathbb{E}_{\mathbf{x}'\sim q(\mathbf{x}'|\mathbf{x},c)}\left[\lambda(\mathbf{x}', \mathbf{x}, c)\log p_\theta(\mathbf{x}'|\mathbf{x}, c)\right]\right] \\ &\approx -\mathbb{E}_{(\mathbf{x},c)\sim D}\left[\mathbb{E}_{\mathbf{x}'\sim q_s(\mathbf{x}'|\mathbf{x},c)}\left[\lambda(\mathbf{x}', \mathbf{x}, c)\log p_\theta(\mathbf{x}'|\mathbf{x}, c)\right]\right] \end{aligned} \tag{3.5}$$

where $q(\mathbf{x}'|\mathbf{x}, c)$ is the proposal distribution and $\lambda(\cdot)$ is the importance function. To better approximate the groundtruth distribution $p(\mathbf{x}'|\mathbf{x}, c)$, we propose to use an ensemble model $q(\mathbf{x}'|\mathbf{x}, c)$. In essence, $q(\mathbf{x}'|\mathbf{x}, c) := q_s(\mathbf{x}'|\mathbf{x}, c)$, where q_s is a specialist distribution decided by the type of the instruction c (e.g. object removal, object addition, stylization, etc). Combining this with Equation 3.3, our objective can be rewritten as:

$$\arg\min_\theta L(\theta) = \arg\min_\theta \mathbb{E}_{(\mathbf{x},c)\sim D}\mathbb{E}_{\mathbf{x}'\sim q_s(\mathbf{x}'|\mathbf{x},c)}\lambda(\mathbf{x}', \mathbf{x}, c)\left[\mathbb{E}_{\epsilon,t}\|\hat{\mathbf{x}}_\theta(\mathbf{x}_t, \mathbf{x}, c) - \mathbf{x}'\|_2^2\right]. \tag{3.6}$$

Table 3.2: Task Definitions and Examples

Editing Tasks	**Definition**	**Instruction c Example**
Object Swap	c describes an object to replace by specifying both the object to remove and the new object to add, along with their properties such as appearance and location.	Replace the black cat with a brown dog in the image.
Object Removal	c describes which object to remove by specifying the object's properties such as appearance, location, and size.	Remove the black cat from the image.
Object Addition	c describes a new object to add by specifying the object's properties such as appearance and location.	Add a red car to the left side of the image.
Attribute Modification	c describes how to modify the properties of an object, such as changing its color and facial expression.	Change the blue car to a red car.
Background Swap	c describes how to replace the background of the image, specifying what the new background should be.	Replace the background with a space-ship interior.
Environment Change	c describes a change to the overall environment, such as the weather, lighting, or season, without altering specific objects.	Change the scene from daytime to nighttime.
Style Transfer	c describes how to apply a specific artistic style or visual effect to the image, altering its overall appearance while keeping the content the same.	Apply a watercolor painting style to the image.

The whole process can be described as: we first sample a pair from dataset D, and then choose the corresponding specialist q_s to sample demonstrations $\mathbf{x}'$ for our editing model $\hat{\mathbf{x}}_\theta(\mathbf{x}_t, \mathbf{x}, c)$ to approximate with an importance weight of $\lambda(\mathbf{x}', \mathbf{x}, c)$.

In our specialist-to-generalist framework, we need to have a series of specialist models $\{q_s(\cdot)\}_s$ and an importance function $\lambda(\cdot)$. We describe them separately in Section 3.2.4.2 and Section 3.2.4.3.

3.2.4.2 Constructing Specialist Models

We group the image editing task into 7 categories as summarized in Table 3.2. For each category, we train or build a task specialist $p_s(\mathbf{x}' \mid \mathbf{x}, c)$ to generate millions of examples. Table 3.2 provides detailed information on task groups and example editing instructions c. In this section, we briefly summarize each specialist, with details available in Section 3.2.4.3.

Object Replacement. We train an image-inpainting specialist $q_{\text{obj_replace}}$ for object replacement. Given an image $\mathbf{x}$, an object caption c_{obj}, and a mask M_{obj}, the specialist fills the masked region with the object described by the caption. We construct replacement pairs by masking an existing object and inpainting a new one.

Object Removal. We trained an image inpainting model to serve as the specialist $q_{\text{obj_removal}}$ for object removal. We use a similar procedure as in the object replacement but

use a predicted background content caption to inpaint the masked image.
Object Addition. We treat object addition as the inverse task of object removal.
Attribute Modification. We adopt the Prompt-to-Prompt (P2P) [53] pipeline to generate examples. To enable precise modification, we adapt the method from Sheynin et al. [131] where we provide a mask M_{obj} for the object and force P2P to only make edits inside the mask.
Background Swap. We use a similar procedure as in the object replacement but use an inverse mask of the object to indicate the background and guide the inpainting.
Environment Modification. For environment modification, we use the P2P pipeline to generate original and edited image pairs.
Style Transfer. We use CosXL-Edit [12] as the specialist model as its training data contains a large number of style-transfer examples. We provide CosXL-Edit with $(\mathbf{x}, c)$, and let it generate the edited image $\mathbf{x}'$.

3.2.4.3 Importance weighting

The importance weighting function λ takes as input a tuple of source image, edited image, and editing prompt. Its purpose is to assign higher weights to data points that are more likely to be sampled from the ground truth distribution, and lower weights to the unlikely ones. This is essentially a quality measure to up-weight high-quality samples. Unlike previous work, we do not use CLIP score because prior work [69] has shown its low correlation with human judgments. Instead, we propose to use large multimodal models (LMMs) to approximate the weighting function, as they demonstrate strong image understanding. Following VIEScore [79], we designed a prompting template for GPT-4o [1] to evaluate the image editing pairs and output a score on a scale from 0 to 10. We retain examples with scores greater than or equal to 9, so the LMM essentially serves as a binary weighting function:

$$\lambda(\mathbf{x}', \mathbf{x}, c) = \begin{cases} 1, & \text{if LMM}(\text{prompt}, \mathbf{x}', \mathbf{x}, c) \geq 9 \\ 0, & \text{otherwise} \end{cases}$$

The scoring criteria and filtering procedure are described below.

While the GPT-4o is an effective choice for this task, scoring large-scale datasets with millions of examples is extremely costly and time-consuming. Therefore, we employ knowledge distillation from GPT-4o to a smaller 8B model, InternVL2 [23]. For each task, we sample 50K data points and instruct GPT-4o to output both a score and a score rationale. We fine-tune InternVL2 on these GPT-4o-generated examples. The smaller fine-tuned InternVL2 model reduces the cost of scoring the full dataset. Finally, we apply the fine-tuned

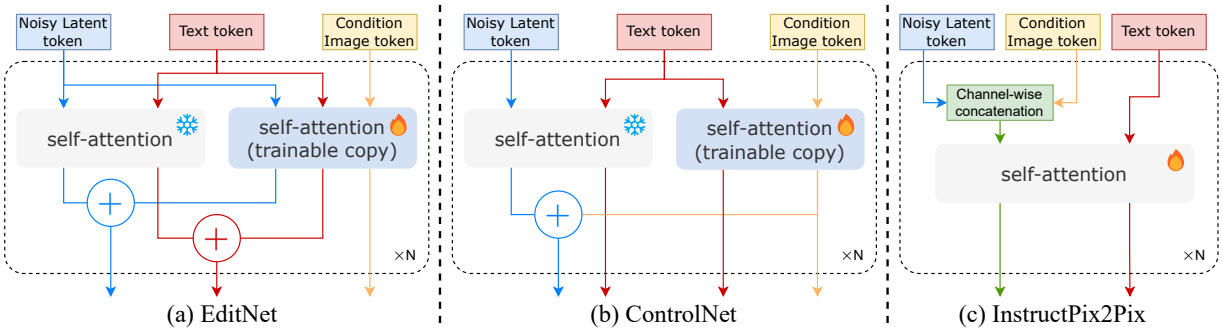


Figure 3.4: Architecture Comparison between **EditNet(ours)**, ControlNet and InstructPix2Pix(Channel-wise concatenation) for DiT models. Unlike ControlNet's parallel execution, EditNet allows adaptive adjustment of control signals by intermediate representations interaction between the control branch and the original branch. EditNet also updates the text representation, enabling better task understanding.

InternVL2 model to filter data across a dataset with millions of samples. Only examples with a score of ≥ 9 are retained, resulting in a curated training dataset of 1.2M examples. We visualize InternVL2's response as a scoring function before and after fine-tuning in Figure 3.3. We observe that fine-tuning InternVL2 on GPT-4o's response effectively turns InternVL2 into a scoring function that identifies distortions and edits that fail to follow the instruction. Table 3.3 provides the task-level dataset statistics.

Architecture. We found that directly fine-tuning a pre-trained high-quality diffusion model like SD3 using channel-wise image concatenation methods [13] compromises the model's original representational capabilities (see Figure 3.6 and Section 3.2.5.2 for a detailed comparison).

To enable a diffusion transformer to perform instruction-based image editing while preserving its original capabilities, we introduce **EditNet** to build OmniEdit. EditNet can effectively transform common DiT models like SD3 into editing models. As illustrated in Figure 3.4, we replicate each layer of the original DiT block as a control branch. The control branch DiT blocks allow interaction between the original DiT tokens, conditional image tokens, and the editing prompts. The output of the control branch tokens is then added to the original DiT tokens and editing prompts. Since the original DiT blocks are trained for generation tasks and are not aware of the editing instructions specifying which contents to modify and how to modify them, this design allows the control branch DiT to adjust the representations of the original DiT tokens and editing prompts according to the editing in-

struction, while still leveraging the strong generation ability of the original DiT. Compared to ControlNet [198], our approach offers two key advantages that make it more suitable for image editing tasks: First, ControlNet does not update text representations, making it challenging to execute editing tasks based on instruction, particularly object removal, as it fails to understand the "removal" intent (see Figure 3.7). Secondly, ControlNet's control branch operates in parallel without access to the original branch's intermediate representations. This fixed precomputation of control signals restricts the overall representation power of the network. We provide an ablation study on the OmniEdit architecture design in Section 3.2.5.2.

Specialist data and quality control. Object replacement and removal use inpainting specialists with GroundingDINO/SAM masks; addition reverses removal pairs. Attribute and environment edits use masked Prompt-to-Prompt, preserving non-target regions; environment editing preserves foreground latents during the first 0.7 of sampling. Background swapping inpaints the inverse object mask, and style transfer uses CosXL-Edit. Instructions are rewritten to describe the resulting source–target transformation. Table 3.3 records task balance before and after filtering.

Semantic consistency assesses both execution of the requested edit and preservation of unrelated content on a 0–10 scale; the edit-success score is zero if the named source object is absent. Perceptual quality separately assesses distortions, anatomy, geometry, and other rendering artifacts, from 0 (substantial artifacts) to 10 (artifact-free). An intentionally unreal situation is not itself an artifact. The evaluator receives the instruction and source/edited images and returns scores with short rationales. Filtering retains overall scores of at least 9. Retained demonstrations and importance weights are stored together and used with Equation 3.6.

3.2.5 Experiments

The evaluation tests whether specialist supervision improves coverage across editing operations and whether the architecture can execute the requested change without losing relevant source content. Comparisons with existing editors assess the resulting task performance, while the ablations examine the supervision and architectural choices behind it. In this section, we first provide statistics of the OmniEdit training set and test set in the dataset description below and Table 3.3. Then we introduce the human evaluation protocol in Section 3.2.5, and comparative baseline system in 3.2.5. We present the main results in Section 3.2.5.1, highlighting the advantages of OmniEdit in handling multi-aspect-ratio,

Table 3.3: OmniEdit training dataset statistics reflecting the number of samples before and after importance scoring and filtering with o-score ≥ 9.

Task	**Before filtering**	**After filtering**
Object Swap	1,500,000	180,000
Object Removal	1,000,000	200,000
Object Addition	1,000,000	200,000
Background Swap	500,000	180,000
Environment Change	500,000	160,000
Style Transfer	250,000	50,000
Object Property Modification	450,000	250,000
Total	**5,200,000**	**1,220,000**

multi-resolution, and multi-task image editing. In Section 3.2.5.2, we study the advantages of importance sampling for synthetic data. In Section 3.2.5.2, we perform an analysis to study the design of OmniEdit.

OmniEdit Training Dataset. We constructed the training dataset $\mathcal{D}$ by sampling high-resolution images with a minimum resolution of 1 megapixel from the LAION-5B [130] and OpenImageV6 [83] databases. The images cover a range of aspect ratios including 1:1, 2:3, 3:2, 3:4, 4:3, 9:16, and 16:9. For the task of object swap, we employed a specialist model to generate 1.5 million entries. We then applied InternVL2 for importance weighting, retaining samples with scores of 9 or higher, resulting in a dataset of 180K entries for this task. For the other tasks, we generate 250K–1M samples and apply the same quality-based filtering to construct the final dataset.

The final training dataset comprises 1.2M entries, with detailed information provided in Table 3.3.

OmniEdit-Bench. To create a high-resolution, multi-aspect ratio, multi-task benchmark for instruction-based image editing, we manually collected 62 images from Pexels [119] and LAION-5B [130]. These images cover a variety of aspect ratios, including 1:1, 2:3, 3:2, 3:4, 4:3, 9:16, and 16:9. We ensured that the images feature a diverse range of scenes and object counts, from single to complex compositions. Additionally, we selected images with a relatively high aesthetic score to better align with the practical use cases of image editing. For each image, we tasked the model with performing 7 tasks as outlined in Table 3.2. This results in a total of 434 edits.



Figure 3.5: Qualitative comparison between baselines and OmniEdit on a subset of the test set.

OmniEdit implementation details. The OmniEdit model is built upon Stable Diffusion 3 Medium [39] with EditNet architecture. Stable Diffusion 3 has 24 DiT layers. Each layer has a corresponding EditNet layer. We train OmniEdit on the 1.2M OmniEdit training dataset for 2 epochs on a single node with 8 H100 GPUs.

Baseline models. We compare OmniEdit with 8 other text-guided image editing baselines: MagicBrush [197], InstructPix2Pix [13], UltraEdit(SD3) [207], DiffEdit [29], SDEdit [107], CosXL-Edit [12], HIVE [202] and HQ-Edit [65].

Evaluation Protocol. We conduct both human evaluation and automatic evaluation. For the human evaluation, we follow the procedure from Ku et al. [80] to rate in two criteria: Semantic Consistency (SC) and Perceptual Quality (PQ). Both scores are in $\{0, 0.5, 1\}$. For SC, the human subject is asked to rate the consistency between 1) the edited image and the editing instruction (whether the editing instruction is reflected on the edited image) and between 2) the source image and the edited image (whether the

Table 3.4: Main evaluation results on OmniEdit-Bench. In each column, the highest score is bolded, and the second-highest is underlined.

Models	VIEScore (GPT4o)			VIEScore (Gemini)			Human Evaluation			
	$PQ_{avg}\uparrow$	$SC_{avg}\uparrow$	$O_{avg}\uparrow$	$PQ_{avg}\uparrow$	$SC_{avg}\uparrow$	$O_{avg}\uparrow$	$PQ_{avg}\uparrow$	$SC_{avg}\uparrow$	$O_{avg}\uparrow$	$Acc_{avg}\uparrow$
Inversion-based Methods										
DiffEdit	5.88	2.73	2.79	6.09	2.01	2.39	-	-	-	-
SDEdit	6.71	2.18	2.78	6.31	2.06	2.48	-	-	-	-
End-to-End Methods										
InstructPix2Pix	7.05	3.04	3.45	6.46	1.88	2.31	-	-	-	-
MagicBrush	6.11	3.53	3.60	6.36	2.27	2.61	-	-	-	-
UltraEdit(SD-3)	6.44	4.66	4.86	6.49	4.33	4.45	0.72	0.52	0.57	0.20
HQ-Edit	5.42	2.15	2.25	6.18	1.71	1.96	0.80	0.27	0.29	0.10
CosXL-Edit	<u>8.34</u>	<u>5.81</u>	<u>6.00</u>	<u>7.01</u>	<u>4.90</u>	<u>4.81</u>	<u>0.82</u>	<u>0.56</u>	<u>0.59</u>	<u>0.35</u>
HIVE	5.35	3.65	3.57	5.84	2.84	3.05	-	-	-	-
OmniEdit	**8.38**	**6.66**	**6.98**	**7.06**	**5.82**	**5.78**	**0.83**	**0.71**	**0.69**	**0.55**
Δ - Best baseline	+0.04	+0.85	+0.98	+0.05	+0.92	+0.97	+0.01	+0.15	+0.10	+0.20

model makes the edit that is beyond the editing instruction). For PQ, the subject is asked to rate the quality of the edited image. We then calculate an overall score $O = \sqrt{SC \times PQ}$ that measures the overall quality of the edit. We also calculate the accuracy of the edit, which is defined by the percentage of $SC = 1$ among all examples. We recruit four human raters and require them to evaluate all the editing examples. For LMMs' evaluation, we follow the procedure from Ku et al. [79] where models (in particular, we chose GPT4o and Gemini) are also asked to give SC and PQ scores but on a scale of 0-10.

3.2.5.1 Main Results

We provide a qualitative comparison with baseline models in Figure 3.5. We show the top 4 baselines with OmniEdit on a subset of the OmniEdit-Bench. Our main results are detailed in Table 3.4, where we provide the VIEScore and conduct human evaluation on three baselines and OmniEdit. In Figure 3.1, OmniEdit demonstrates its capability to handle diverse editing tasks across various aspect ratios and resolutions. The addition and object-swap examples preserve source details while incorporating the requested content. Similarly, in Figure 3.5, OmniEdit uniquely adds a clean and distinct NASA logo onto a T-shirt. Table 3.4 corroborates this with OmniEdit achieving the highest Perceptual Quality (PQ) score among the models evaluated.

The specialist-to-generalist framework supplies task-specific supervision. A single synthetic-data pipeline such as Prompt-to-Prompt can introduce changes outside the

requested region; OmniEdit instead uses specialists selected for each operation and filters their outputs for edit quality. Both the VIEScore and human evaluations in Table 3.4 demonstrate that our method significantly outperforms the best baseline in following editing instructions accurately and minimizing over-editing. For instance, baseline models frequently misunderstand the task intent as illustrated in Figure 3.5, where the CosXL-Edit model fails to recognize the removal task and incorrectly interprets a bird addition as a swap between a panda and a bird.

Lastly, baseline models often produce blurry images on the OmniEdit-Bench, as they are trained at resolutions limited to 512x512 or even 256x256, and they perform poorly on non-square aspect ratios. For example, with a 3:4 aspect ratio, the baselines struggle to perform editing. OmniEdit, trained on data with multiple aspect ratios, maintains robust editing capabilities across the diverse aspect ratios encountered on the Omni-Bench, as evidenced in Figure 3.5.

3.2.5.2 Ablation Study

We ablate importance weighting and the EditNet architecture.

Table 3.5: Ablation on importance sampling.

Models	**VIEScore (GPT4o)**			**VIEScore (Gemini)**		
	$PQ_{avg} \uparrow$	$SC_{avg} \uparrow$	$O_{avg} \uparrow$	$PQ_{avg} \uparrow$	$SC_{avg} \uparrow$	$O_{avg} \uparrow$
OmniEdit	8.38	6.66	6.98	7.06	5.82	5.78
OmniEdit w/o importance sampling	6.20	2.95	3.30	6.40	1.80	2.25

Table 3.6: Ablation on OmniEdit architecture design.

Models	**VIEScore (GPT4o)**			**VIEScore (Gemini)**		
	$PQ_{avg} \uparrow$	$SC_{avg} \uparrow$	$O_{avg} \uparrow$	$PQ_{avg} \uparrow$	$SC_{avg} \uparrow$	$O_{avg} \uparrow$
OmniEdit	8.38	6.66	6.98	7.06	5.82	5.78
OmniEdit- ControlNet - TextControl	6.45	4.70	4.89	6.50	4.35	4.48
OmniEdit- ControlNet	6.35	4.60	4.75	6.40	4.25	4.35

Ablation study on the importance sampling. We study a baseline that utilizes the same architecture as OmniEdit, but instead of applying importance scoring and filtering, we sample 1.2M examples directly from the 5M pre-filtering dataset as specified in Table 3.3 and compare it with OmniEdit. As shown in Table 3.5, we observe a significant decrease in VIEScores for both PQ and SC metrics.

Ablation Study on OmniEdit Architecture Design. We conducted an analysis of OmniEdit's architectural design in comparison to two baseline models: OmniEdit-ControlNet and OmniEdit-ControlNet-TextControl and show the result in Table 3.6. OmniEdit-ControlNet represents the SD3-ControlNet architecture trained on the OmniEdit dataset, where the source image serves as the conditioning image for the control branch. OmniEdit-ControlNet-TextControl is a variant of OmniEdit-ControlNet with an added modification: at each layer, we incorporate the text-token output from the control branch into the text-token in the main image generation branch. So this baseline can update the text representation in the main branch but doesn't have the intermediate representation interaction design in EditNet.

Our analysis, as shown in Figure 3.7, reveals that OmniEdit-ControlNet struggled to accurately capture task intent. This is primarily because the ControlNet branch does not update the text representation. For instance, in object removal tasks, prompts like "Remove ObjA" are common, yet the original DiT block remains unchanged, causing it to mistakenly generate an image of "ObjA." On the other hand, although OmniEdit-ControlNet-TextControl successfully updates the text representation, it still encounters difficulties in content removal. The substantial VIEScores gap between OmniEdit-Controlnet-TextControl and OmniEdit in Table 3.6 underscores the importance of the intermediate representation interaction design in EditNet. We also compared OmniEdit with the channel-wise token concatenation method used in InstructPix2Pix (see Figure 3.4). Channel-wise Token concatenation requires fine-tuning the entire network, which can distort the network's original representations. As illustrated in Figure 3.6, after fine-tuning an SD3 channel-wise concatenation model on OmniEdit training set, the representation of Batman is altered. In contrast, EditNet preserves the original representation of Batman while still learning the object swap task.

Generalist and specialist trade-offs. Table 3.7 compares the generalist with individual specialists. Generalization across operations does not imply dominance on every task: the style-transfer specialist scores higher overall under both evaluators, and object-removal rankings differ between evaluators. The benefit is a shared editing model with broad task coverage, with task-specific trade-offs still visible.



(a) Original image (b) Channel-wise concatenation (c) OmniEdit (d) SD3 generation

Figure 3.6: (a) shows the source image. (d) presents images generated by SD3 in response to prompts for "an upper body picture of Batman" and "a shiny red vintage Chevrolet Bel Air car." We use the prompts "Replace the man with Batman" and "Add a shiny red vintage Chevrolet Bel Air car to the right" to OmniEdit and OmniEdit-Channel-Wise-Concatenation, which was trained on OmniEdit training data. From (b) and (c), one can observe that OmniEdit preserves the generation capabilities of SD3, while OmniEdit-Channel-Wise-Concatenation exhibits a notable degradation in generation capability.



(a) Original image (b) ControlNet (c) ControlNet with text control (d) OmniEdit

Figure 3.7: OmniEdit-ControlNet fails to grasp the task intent, while OmniEdit-ControlNet-TextControl—a variant with a text-updating branch—recognizes the intent but struggles with content removal. In contrast, OmniEdit accurately removes content.

Table 3.7: Comparison between OmniEdit and our specialist models.

	VIEScore (GPT4o)			**VIEScore (Gemini)**		
	$PQ_{avg} \uparrow$	$SC_{avg} \uparrow$	$O_{avg} \uparrow$	$PQ_{avg} \uparrow$	$SC_{avg} \uparrow$	$O_{avg} \uparrow$
Obj-Remove-Specialist	9.10	7.76	7.82	7.46	5.39	4.84
OmniEdit	8.45	7.16	7.23	7.37	5.45	5.09
Obj-Replacement-Specialist	8.48	6.92	7.02	7.06	5.68	5.36
OmniEdit	8.95	7.74	8.14	7.00	7.77	7.09
Style-Transfer-Specialist	8.08	7.47	7.37	7.97	6.61	6.76
OmniEdit	7.98	5.77	6.16	8.24	5.24	6.08

3.2.6 Discussion

OmniEdit shows that a generalist editor can acquire a broad range of operations by consolidating supervision from task specialists. The important distinction is not simply the number of training examples, but whether those examples reliably express the intended transformation. The sampling ablation shows that replacing quality-controlled supervision with an equally sized random sample reduces editing performance. The architecture comparisons reveal a related requirement on the model side: source-image conditioning must interact with the instruction representation to distinguish, for example, removing an object from generating it. Together, these findings connect task flexibility to both the quality of supervision and the way the model represents an edit.

This approach broadens image-editing capability by bringing complementary specialist skills into one model. The specialists determine the transformations available for supervision, while the backbone supplies the generative capacity to execute them. OmniEdit's contribution to controllable generation is to connect these two resources through quality-controlled training, allowing users to express different editing operations through a shared instruction interface. AnyV2V, considered next, extends flexible editing to video through a different mechanism: it reuses an image editor at inference time and propagates the edited first frame throughout the video.

3.3 AnyV2V: Tuning-Free Video Editing

3.3.1 Motivation

Section 3.2 develops editing capability through specialist-supervised training. This section asks whether existing image editors can instead be reused for video without tuning a model for each source video. Paired video-editing data are difficult to obtain, and the desired transformation may be easier to specify through a reference image or a manual edit than through text alone.

AnyV2V decomposes the task into editing the first frame and propagating that edit with a pretrained image-to-video model. The edited frame supplies the desired appearance; the source video's inverted latents and intermediate features guide structure and motion. This separates the editing interface from temporal generation: an off-the-shelf image editor or a human can produce the first-frame condition.

The central technical question is which information must be retained from the source video to preserve its motion without suppressing the edit. The method therefore combines DDIM inversion with spatial and temporal feature injection. Experiments evaluate prompt-based editing, reference-based style transfer, subject-driven editing, and identity manipulation, followed by ablations of the guidance components. We also examine inversion of sequences longer than the backbone's training clips. Quantitative and human comparisons are reported in Section 3.3.5.3, with the evaluation protocol in Section 3.3.5.1.

3.3.2 Related Work

Video editors differ in how they preserve source motion and incorporate the desired change. Tune-A-Video and VideoP2P adapt pretrained image generators to a source video [96, 167]. Zero-shot approaches such as FateZero and TokenFlow reuse inverted trajectories, attention, or feature correspondences to improve consistency [42, 122]. ControlVideo adds spatial conditioning, while CoDeF propagates edits through a learned representation of the source video [115, 205].

AnyV2V uses a pretrained image-to-video backbone and supplies the transformation as an edited first frame. UniEdit also uses inversion and feature injection, but operates with a text-to-video backbone [5]. The first-frame interface is the key distinction here: it admits reference-guided or manual edits without requiring every transformation to be expressed through text. Table 3.8 compares the evaluated methods' task interfaces and tuning re-

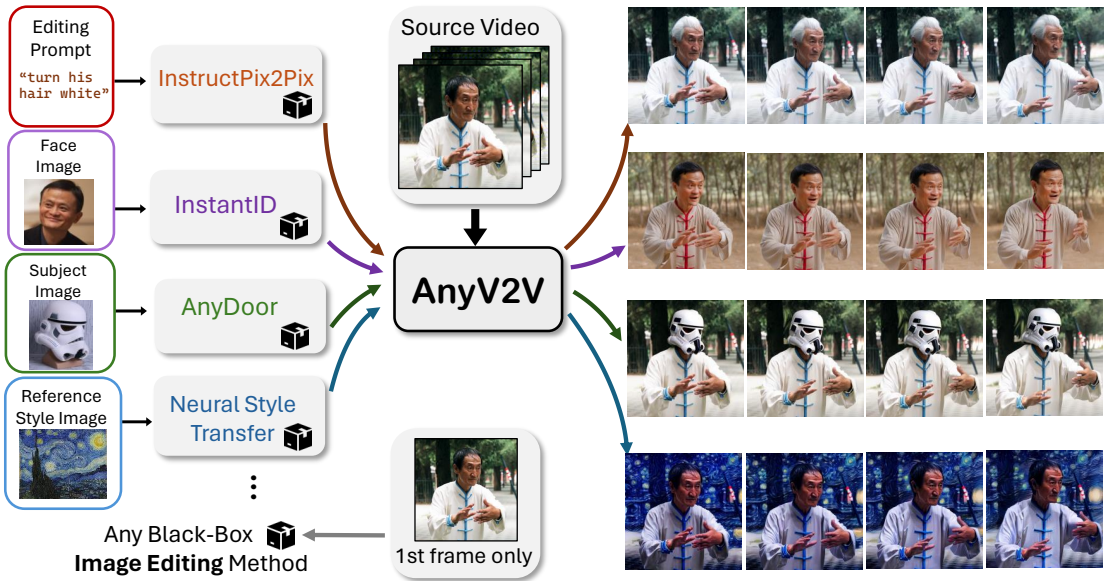


Figure 3.8: AnyV2V supports diverse video-editing tasks—including prompt-based editing, reference-based style transfer, subject-driven editing, and identity manipulation—without parameter tuning. AnyV2V disentangles video editing into two simpler problems: (1) Single image editing and (2) Image-to-video generation with video reference.

quirements. We evaluate AnyV2V with three I2V backbones: I2VGen-XL, ConsistI2V, and SEINE.

Table 3.8: Comparison with different video editing methods and the type of editing tasks.

Method	Prompt-based Editing	Reference-based Style Transfer	Subject-Driven Editing	Identity Manipulation	Tuning-Free?	Backbone
Tune-A-Video [167]	✓	✗	✗	✗	✗	Stable Diffusion
Pix2Video [16]	✓	✗	✗	✗	✓	SD-Depth
Gen-1 [38]	✓	✗	✗	✗	✗	Stable Diffusion
TokenFlow [42]	✓	✗	✗	✗	✓	Stable Diffusion
FLATTEN [27]	✓	✗	✗	✗	✓	Stable Diffusion
Fairy [163]	✓	✗	✗	✗	✓	Stable Diffusion
ControlVideo [205]	✓	✗	✗	✗	✗	ControlNet
CoDeF [115]	✓	✗	✗	✗	✗	ControlNet
VideoSwap [49]	✗	✗	✓	✗	✗	AnimateDiff
UniEdit [5]	✓	✗	✗	✗	✓	T2V diffusion models
AnyV2V (Ours)	✓	✓	✓	✓	✓	I2V diffusion models

3.3.3 Preliminaries

3.3.3.1 Image-to-Video (I2V) Generation Models

In this work, we focus on leveraging latent diffusion-based [129] I2V generation models for video editing. Given an input first frame I_1, a text prompt $\mathbf{s}$ and a noisy video latent $\mathbf{z}_t$ at time step t, I2V generation models recover a less noisy latent $\mathbf{z}_{t-1}$ using a denoising model $\epsilon_\theta(\mathbf{z}_t, I_1, \mathbf{s}, t)$ conditioned on both I_1 and $\mathbf{s}$. The denoising model ϵ_θ contains a set of spatial and temporal self-attention layers, where the self-attention operation can be formulated as:

$$Q = W^Q z, K = W^K z, V = W^V z, \tag{3.7}$$

$$\text{Attention}(Q, K, V) = \text{Softmax}(\frac{QK^\top}{\sqrt{d}})V, \tag{3.8}$$

where z is the input hidden state to the self-attention layer and W^Q, W^K and W^V are learnable projection matrices that map z onto query, key and value vectors, respectively. For spatial self-attention, z represents a sequence of spatial tokens from each frame. For temporal self-attention, z is composed of tokens located at the same spatial position across all frames.

3.3.3.2 DDIM Inversion

The denoising process for I2V generation models from $\mathbf{z}_t$ to $\mathbf{z}_{t-1}$ can be achieved using the DDIM [135] sampling algorithm. DDIM inversion [36, 109] approximately reverses this process using the predicted noise:

$$\mathbf{z}_{t+1} = \sqrt{\frac{\alpha_{t+1}}{\alpha_t}}\mathbf{z}_t + \sqrt{\alpha_{t+1}}\left(\sqrt{\frac{1}{\alpha_{t+1}} - 1} - \sqrt{\frac{1}{\alpha_t} - 1}\right)\epsilon_\theta(\mathbf{z}_t, I_1, \mathbf{s}, t).$$

Here $\alpha_t = \prod_{u=1}^{t}(1 - \beta_u)$ is the cumulative signal coefficient for the diffusion variance schedule $\{\beta_u\}$, and I_1 is the conditioning first frame.

3.3.3.3 Plug-and-Play (PnP) Diffusion Features

Tumanyan et al. [147] proposed PnP diffusion features for image editing, based on the observation that intermediate convolution features f and self-attention scores

$A = \text{Softmax}(\frac{QK^\top}{\sqrt{d}})$ in a text-to-image (T2I) denoising U-Net capture the semantic regions (e.g. legs or torso of a human body) during the image generation process.

Given an input source image I^S and a target prompt P, PnP first performs DDIM inversion to obtain the image's corresponding noise $\{\mathbf{z}_t^S\}_{t=1}^T$ at each time step t. It then collects the convolution features $\{f_t^l\}$ and attention scores $\{A_t^l\}$ from some predefined layers l at each time step t of the backward diffusion process $\mathbf{z}_{t-1}^S = \mathcal{D}_{\theta,t}(\mathbf{z}_t^S, \varnothing)$, where $\mathcal{D}_{\theta,t}$ denotes a DDIM denoising step using the noise predictor ϵ_θ and $\varnothing$ denotes the null text prompt.

To generate the edited image I^*, PnP starts from the initial noise of the source image (i.e. $\mathbf{z}_T^* = \mathbf{z}_T^S$) and performs feature injection during denoising: $\mathbf{z}_{t-1}^* = \mathcal{D}_{\theta,t}(\mathbf{z}_t^*, P; \{f_t^l, A_t^l\})$. The injected features replace the intermediate features and attention scores $\{f_t^{l*}, A_t^{l*}\}$ in ϵ_θ with $\{f_t^l, A_t^l\}$ during this step. This feature injection mechanism ensures that I^* preserves the layout and structure from I^S while reflecting the description in P. To control the feature injection strength, PnP also employs two thresholds τ_f and τ_A such that the feature and attention scores are only injected in the first τ_f and τ_A denoising steps. Our method extends this feature injection mechanism to I2V generation models, where we inject features in convolution, spatial, and temporal attention layers. We show the detailed design of AnyV2V in Section 3.3.4.

3.3.4 Method

Our method presents a two-stage approach to video editing. Given a source video $V^S = \{I_1, I_2, I_3, ..., I_n\}$, where I_i is the frame at time i and n denotes the video length, we first extract the initial frame I_1 and pass it into an image editing model ϕ_{img} to obtain an edited first frame $I_1^* = \phi_{\text{img}}(I_1, C)$. C is the auxiliary conditions for image editing models such as text prompt, mask, style, etc. In the second stage, we feed the edited first frame I_1^* and a target prompt $\mathbf{s}^*$ into an I2V generation model ϵ_θ and employ the inverted latent from the source video V^S to guide the generation process such that the edited video V^* follows the motion of the source video V^S, the semantic information represented in the edited first frame I_1^*, and the target prompt $\mathbf{s}^*$. An overall illustration of our video editing pipeline is shown in Figure 3.9. In this section, we explain each core component of our method.

3.3.4.1 Flexible First Frame Editing

AnyV2V specifies the desired transformation by editing the video's first frame. This interface supports image editors for image style transfer [41, 43, 101, 151], mask-based image

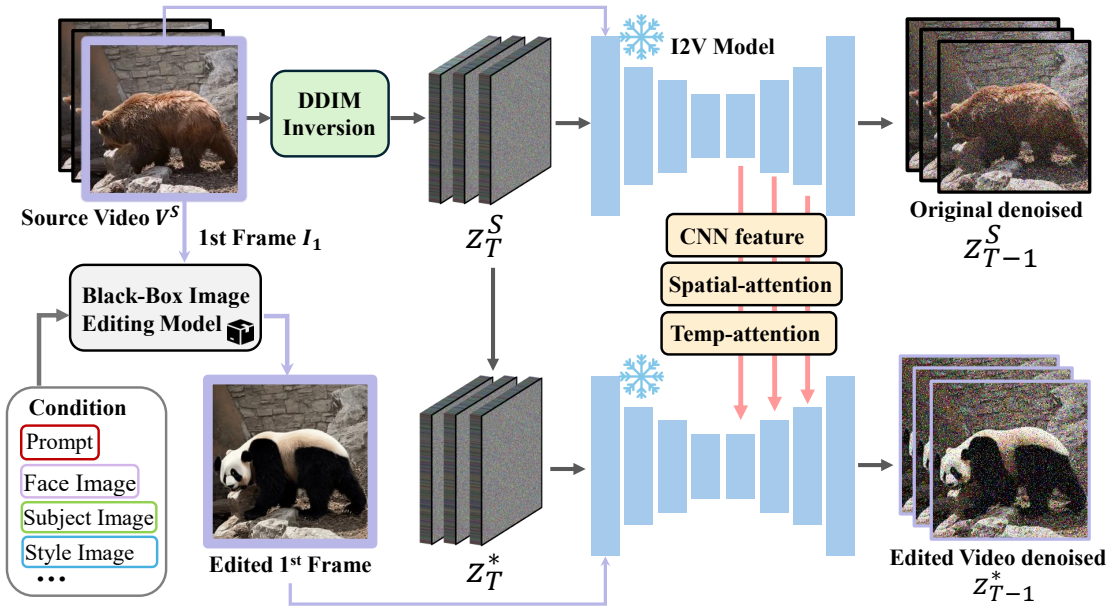


Figure 3.9: AnyV2V takes a source video V^S as input. In the first stage, we apply a black-box image editing method on the first frame I_1 according to the editing task. In the second stage, the source video is inverted to initial noise z_T^S, which is then denoised using DDIM sampling. During the sampling process, we extract spatial convolution, spatial attention, and temporal attention features from the I2V models' decoder layers. To generate the edited video, we initialize $z_T^* = z_T^S$ and perform DDIM sampling conditioned on the edited first frame. During sampling, we inject the features and attention into corresponding layers of the model.

editing [3, 111], image inpainting [82, 138], identity-preserving image editing [156], and subject-driven image editing [21, 48, 86].

3.3.4.2 Structural Guidance using DDIM Inversion

To ensure the generated videos from the I2V generation model follow the general structure as presented in the source video, we employ DDIM inversion to obtain the latent noise of the source video at each time step t. Specifically, we perform the inversion *without* text prompt condition but *with* the first frame condition. Formally, given a source video $V^S = \{I_1, I_2, I_3, ..., I_n\}$, we advance the inverted latent from time step t to $t+1$ as:

$$\mathbf{z}_{t+1}^S = \text{DDIM_Inv}(\mathbf{z}_t^S, I_1, \varnothing, t; \epsilon_\theta), \tag{3.9}$$

where DDIM_Inv$(\cdot)$ denotes the DDIM inversion operation as described in Section 3.3.3. In ideal cases, the latent noise $\mathbf{z}_T^S$ at the final time step T (initial noise of the source video) should be used as the initial noise for sampling the edited videos. In practice, we find that due to the limited capability of certain I2V models, the edited videos denoised from the last time step are sometimes distorted. Following Li et al. [86], we observe that starting the sampling from a previous time step $T' < T$ can be used as a simple workaround to fix this issue.

3.3.4.3 Appearance Guidance via Spatial Feature Injection

Our empirical observation (Section 3.3.5.4) suggests that I2V generation models already have some editing capabilities by only using the edited first frame and DDIM inverted noise as the model input. However, we find that this simple approach is often unable to correctly preserve the background in the edited first frame and the motion in the source video, as the conditional signal from the source video encoded in the inverted noise is limited.

To enforce consistency with the source video, we perform feature injection in both convolution layers and spatial attention layers in the denoising U-Net. During the video sampling process, we simultaneously denoise the source video using the previously collected DDIM inverted latents $\mathbf{z}_t^S$ at each time step t such that $\mathbf{z}_{t-1}^S = \epsilon_\theta(\mathbf{z}_t^S, I_1, \varnothing, t)$. We preserve two types of features during source video denoising: convolution features f^{l_1} before skip connection from the l_1^{th} residual block in the U-Net decoder, and the spatial self-attention scores $\{A_s^{l_2}\}$ from $l_2 = \{l_{low}, l_{low+1}, ..., l_{high}\}$ layers. We collect the queries $\{Q_s^{l_2}\}$ and keys $\{K_s^{l_2}\}$ instead of directly collecting $A_s^{l_2}$ as the attention score matrices are parameterized by the query and key vectors. We then replace the corresponding features during denoising the edited video in both the normal denoising branch and the negative prompt branch for classifier-free guidance [55]. We use two thresholds τ_{conv} and τ_{sa} to control the convolution and spatial attention injection to only happen in the first τ_{conv} and τ_{sa} steps during video sampling.

3.3.4.4 Motion Guidance through Temporal Feature Injection

The spatial feature injection mechanism described in Section 3.3.4.3 significantly enhances the background and overall structure consistency of the edited video. While it also helps maintain the source video motion to some degree, we observe that the edited videos will still have a high chance of containing incorrect motion compared to the source video. On the other hand, we notice that I2V generation models, or video diffusion models in general,

are often initialized from pre-trained T2I models and continue to be trained on video data. During the training process, parameters in the spatial layers are often frozen or set to a lower learning rate such that the pre-trained weights from the T2I model are less affected, and the parameters in the temporal layers are more extensively updated during training. Therefore, it is likely that a large portion of the motion information is encoded in the temporal layers of the I2V generation models. Concurrent work [5] also observes that features in the temporal layers show similar characteristics with optical flow [58], a pattern that is often used to describe the motion of the video.

To better reconstruct the source video motion in the edited video, we propose to also inject the temporal attention features in the video generation process. Similar to spatial attention injection, we collect the source video temporal self-attention queries $Q_t^{l_3}$ and keys $K_t^{l_3}$ from some U-Net decoder layers represented by l_3 and inject them into the edited video denoising branches. We also only apply temporal attention injection in the first τ_{ta} steps during sampling.

3.3.4.5 Putting it Together

Overall, combining the spatial and temporal feature injection mechanisms, we replace the editing branch features $\{f^{*l_1}, Q_s^{*l_2}, K_s^{*l_2}, Q_t^{*l_3}, K_t^{*l_3}\}$ with the features from the source denoising branch:

$$\mathbf{z}_{t-1}^{*} = \epsilon_\theta(\mathbf{z}_t^{*}, I^{*}, \mathbf{s}^{*}, t\ ; \{f^{l_1}, Q_s^{l_2}, K_s^{l_2}, Q_t^{l_3}, K_t^{l_3}\}), \tag{3.10}$$

where $\epsilon_\theta(\cdot\ ; \{f^{l_1}, Q_s^{l_2}, K_s^{l_2}, Q_t^{l_3}, K_t^{l_3}\})$ denotes the feature replacement operation across different layers l_1, l_2, l_3. Our proposed spatial and temporal feature injection scheme enables tuning-free adaptation of I2V generation models for video editing. Our experimental results demonstrate that each component in our design is crucial to the accurate editing of source videos. We showcase more qualitative results for the effectiveness of our model components in Section 3.3.5.

3.3.5 Experiments

3.3.5.1 Implementation Details

We employ AnyV2V on three off-the-shelf I2V generation models: I2VGen-XL[1] [201], ConsistI2V [128] and SEINE [22]. For all I2V models, we use $\tau_{conv} = 0.2T$, $\tau_{sa} = 0.2T$ and

[1]We use the version provided in https://huggingface.co/ali-vilab/i2vgen-xl.

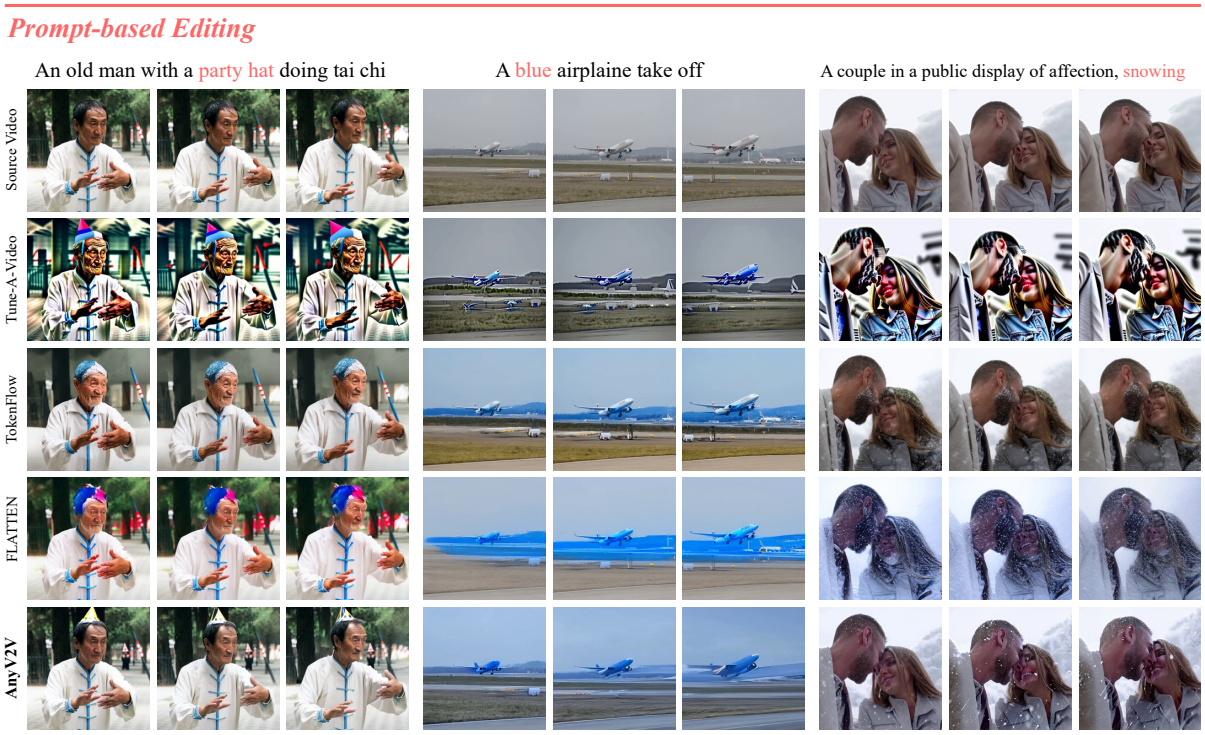


Figure 3.10: AnyV2V is robust in a wide range of prompt-based editing tasks while preserving the background. The results align the most with the text prompt and maintain high motion consistency.

$\tau_{ta} = 0.5T$, where T is the total number of sampling steps. We use the DDIM [135] sampler and set T to the default values of the selected I2V models. Following PnP [147], we set $l_1 = 4$ for convolution feature injection and $l_2 = l_3 = \{4, 5, 6, ..., 11\}$ for spatial and temporal attention injections. During sampling, we apply text classifier-free guidance (CFG) [55] for all models with the same negative prompt *"Distorted, discontinuous, Ugly, blurry, low resolution, motionless, static, disfigured, disconnected limbs, Ugly faces, incomplete arms"* across all edits. To obtain the initial edited frames in our implementation, we use a set of image editing model candidates including prompt-based image editing model InstructPix2Pix [13], style transfer model Neural Style Transfer (NST) [41], subject-driven image editing model AnyDoor [21], and identity-driven image editing model InstantID [156]. We experiment with only the successfully edited frames, which is crucial for our method. We conducted all the experiments on a single Nvidia A6000 GPU. To edit a 16-frame video, it requires around 15G GPU memory and around 100 seconds for the whole inference process.

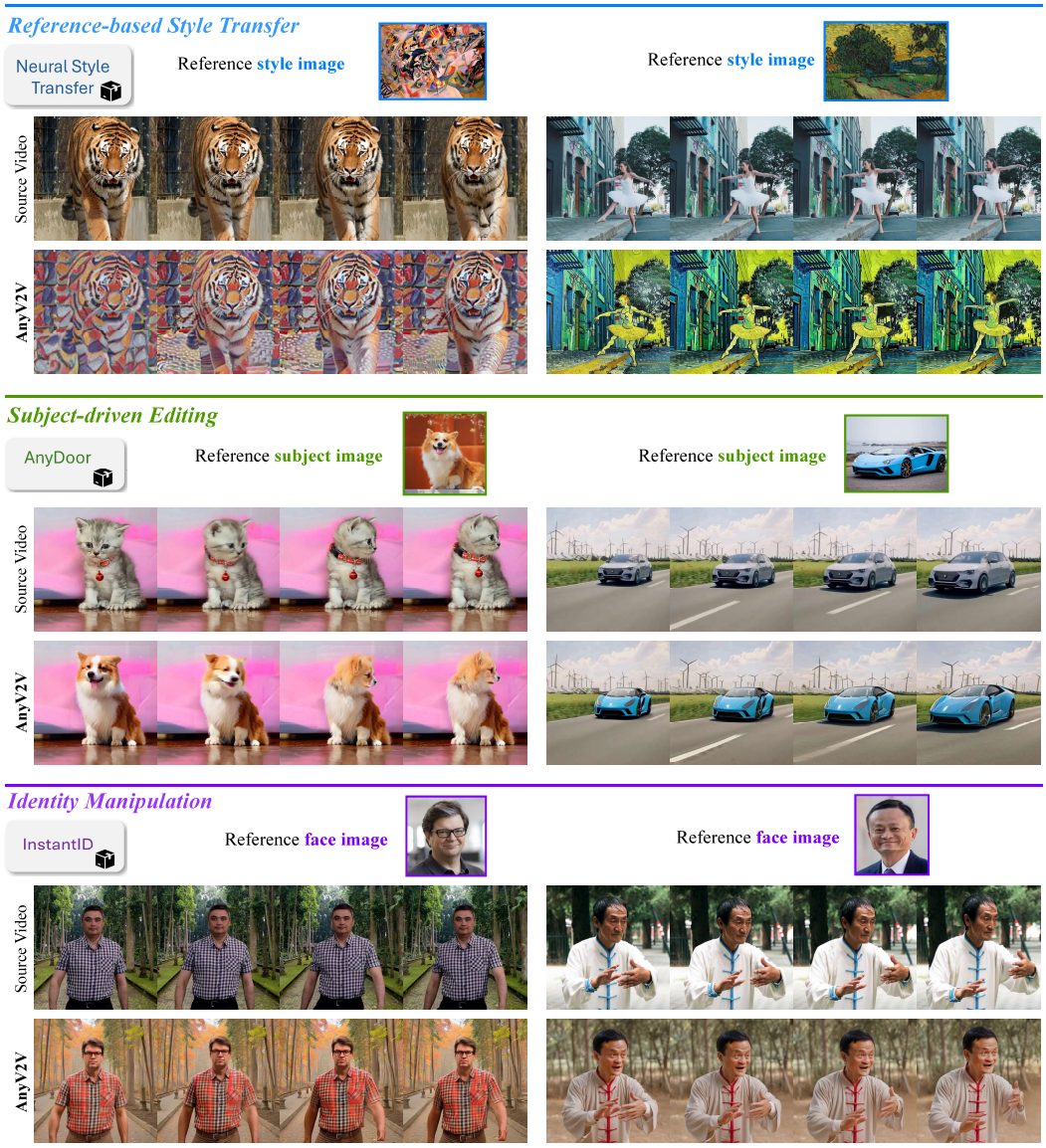


Figure 3.11: With different image editing models, AnyV2V can achieve a wide range of editing tasks, including reference-based style transfer, subject-driven editing, and identity manipulation.

Evaluation protocol. CLIP Text averages frame–target-text cosine similarity; CLIP Image averages similarity between consecutive frame embeddings. The human evaluation uses 89 Pexels videos: 45 prompt-based edits, 20 reference-based style transfers, 13 identity

Table 3.9: Quantitative comparisons for our AnyV2V with baselines on prompt-based video editing. Alignment: prompt alignment; Overall: overall preference. **Bold**: best results; Underline: top-2.

Method	Human Evaluation ↑		CLIP Scores ↑	
	Alignment	Overall	CLIP-Text	CLIP-Image
Tune-A-Video	15.2%	2.1%	0.2902	0.9704
TokenFlow	31.7%	20.7%	0.2858	**0.9783**
FLATTEN	25.5%	16.6%	0.2742	0.9739
AnyV2V (SEINE)	28.9%	8.3%	0.2910	0.9631
AnyV2V (ConsistI2V)	33.8%	11.7%	0.2896	0.9556
AnyV2V (I2VGen-XL)	**69.7%**	**46.2%**	**0.2932**	0.9652

manipulations, and 11 subject-driven edits. Annotators assess instruction/reference alignment and overall preference. The reference-based video comparisons use successfully edited first frames, so the image-editor success rates in Table 3.10 are essential to interpreting their coverage: approximately 90% for NST, 10% for AnyDoor, and 80% for InstantID. They are component success rates, not end-to-end video success rates.

Human participation. Human evaluation was anonymous and voluntary, with withdrawal permitted and no personally identifiable information collected. Participant consent was documented.

3.3.5.2 Task Definitions

1. **Prompt-based Editing:** allows users to manipulate video content using only natural language. This can include descriptive prompts or instructions. With a prompt, users can perform a wide range of edits, such as incorporating accessories, spawning or swapping objects, adding effects, or altering the background.

2. **Reference-based Style Transfer:** A reference artwork specifies the desired appearance. The edited video should reproduce that style while preserving the source content and motion.

3. **Subject-driven Editing:** In subject-driven video editing, we aim at replacing an object in the video with a target subject based on a given subject image while maintaining the video motion and preserving the background.

4. **Identity Manipulation:** Identity manipulation allows the user to manipulate video content by replacing a person with another person's identity in the video based on an input image of the target person.

3.3.5.3 Evaluation Results

As shown in Figure 3.10 and 3.11, AnyV2V can perform the following video editing tasks: (1) prompt-based editing, (2) reference-based style transfer, (3) subject-driven editing, and (4) identity manipulation. We compare AnyV2V against three baseline models Tune-A-Video [167], TokenFlow [42] and FLATTEN [27] for (1) prompt-based editing. Since there exists no publicly available baseline method for task (2) (3) (4), we evaluate the performance of three I2V generation models under AnyV2V.

Prompt-based Editing Unlike the baseline methods [27, 42, 167] that often introduce unwarranted changes not specified in the text commands, AnyV2V utilizes the precision of image editing models to ensure only the targeted areas of the scene are altered, leaving the rest unchanged. Combining AnyV2V with the instruction-guided image editing model InstructPix2Pix [13], AnyV2V accurately places a party hat on an elderly man's head and correctly paints the plane in blue. Additionally, it maintains the original video's background and fidelity, whereas, in comparison, baseline methods often alter the color tone and shape of objects, as illustrated in Figure 3.10. Also, for motion tasks such as adding snowing weather, I2V models from AnyV2V provide inherent support for animating the snowing while the baseline methods would result in flickering. For quantitative evaluations, we conduct a human evaluation to examine the degree of prompt alignment and overall preference of the edited videos based on user voting, and also compute the metrics CLIP-Text for text alignment and CLIP-Image for temporal consistency. CLIP-Text averages the cosine similarity between each frame's image embedding and the target-text embedding from CLIP [126]. CLIP-Image averages the cosine similarity between image embeddings of consecutive frames. Table 3.9 shows that AnyV2V generally achieves high text-alignment and temporal consistency, while AnyV2V with I2VGen-XL backbone is the most preferred method in the human evaluation.

Style Transfer, Subject-Driven Editing and Identity Manipulation For these novel tasks, we stress the alignment with the reference image instead of the text prompt.

As shown in Figure 3.11, we can observe that for task (2), AnyV2V can capture one particular style that is tailor-made, and even if the style is not learned by the text encoder. In the examples, AnyV2V captures the style of Vassily Kandinsky's artwork "Composition VII" and Vincent Van Gogh's artwork "Chateau in Auvers at Sunset" accurately. For task (3), AnyV2V can replace the subject in the video with other subjects even if the new subject differs from the original subject. In the examples, a cat is replaced with a dog according to the reference image and maintains highly aligned motion and background as reflected in the source video. A car is replaced by our desired car while the wheel is still spinning in the edited video. For task (4), AnyV2V can swap a person's identity to anyone. We report human evaluation results in Table 3.10 and find that AnyV2V with I2VGen-XL backbone is the most preferred method in terms of reference alignment and overall performance.

I2V Backbones We find that AnyV2V (I2VGen-XL) tends to be the most robust both qualitatively and quantitatively. It has a good generalization ability to produce consistent motions in the video with high visual quality. AnyV2V (ConsistI2V) can generate consistent motion, but sometimes the watermark would appear due to its training data, thus harming the visual quality. AnyV2V (SEINE)'s generalization ability is relatively weaker but still produces consistent and high-quality video if the motion is simple enough, such as a person walking.

Editing Video beyond Training Frames of I2V model The I2V backbones used in these experiments [22, 128, 201] are trained on 16-frame clips. To edit videos that have length beyond the training frames of the I2V model, an intuitive approach would be generating videos in an auto-regressive manner as used in ConsistI2V [128] and SEINE [22]. However, we find that such an experiment setup cannot maintain semantic consistency in our case. As many works in extending video generation exploit the initial latent to generate longer video [124, 169], we leverage the longer inverted latent as the initial latent and force an I2V model to generate longer frames of output. Our experiments found that the inverted latent contains enough temporal and semantic information to allow the generated video to maintain temporal and semantic consistency, as shown in Figure 3.12.

3.3.5.4 Ablation Study

To verify the effectiveness of our design choices, we conduct an ablation study by iteratively disabling the three core components in our model: temporal feature injection, spatial

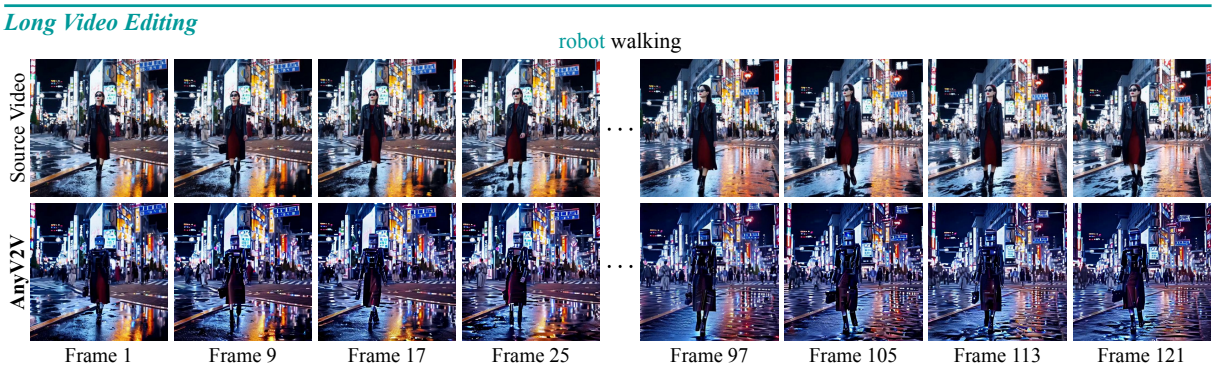


Figure 3.12: AnyV2V can edit video length beyond the training frame while maintaining motion consistency. The first row is the source video frames while the second rows are the edited. The editing prompt of the image was "turn woman into a robot" using image model InstructPix2Pix [13].

Table 3.10: Comparisons for three I2V models under AnyV2V framework on novel video editing tasks. Align: reference alignment; Overall: overall preference. **Bold**: best results; Underline: top-2.

Task	Reference-based Style Transfer		Subject-driven Editing		Identity Manipulation	
Image Editing Method	NST		AnyDoor		InstantID	
Image Editing Success Rate	≈90%		≈10%		≈80%	
Human Evaluation	Align ↑	Overall ↑	Align ↑	Overall ↑	Align ↑	Overall ↑
AnyV2V (SEINE)	92.3%	30.8%	48.4%	15.2%	72.7%	18.2%
AnyV2V (ConsistI2V)	38.4%	10.3%	63.6%	42.4%	72.7%	27.3%
AnyV2V (I2VGen-XL)	**100.0%**	**76.9%**	**93.9%**	**84.8%**	**90.1%**	**45.4%**

feature injection, and DDIM inverted latent as initial noise. We use AnyV2V (I2VGen-XL) and a subset of 20 samples in this ablation study and report both the frame-wise consistency results using CLIP-Image score in Table 3.11 and qualitative comparisons in Figure 3.13.

Effectiveness of Temporal Feature Injection According to the results, after

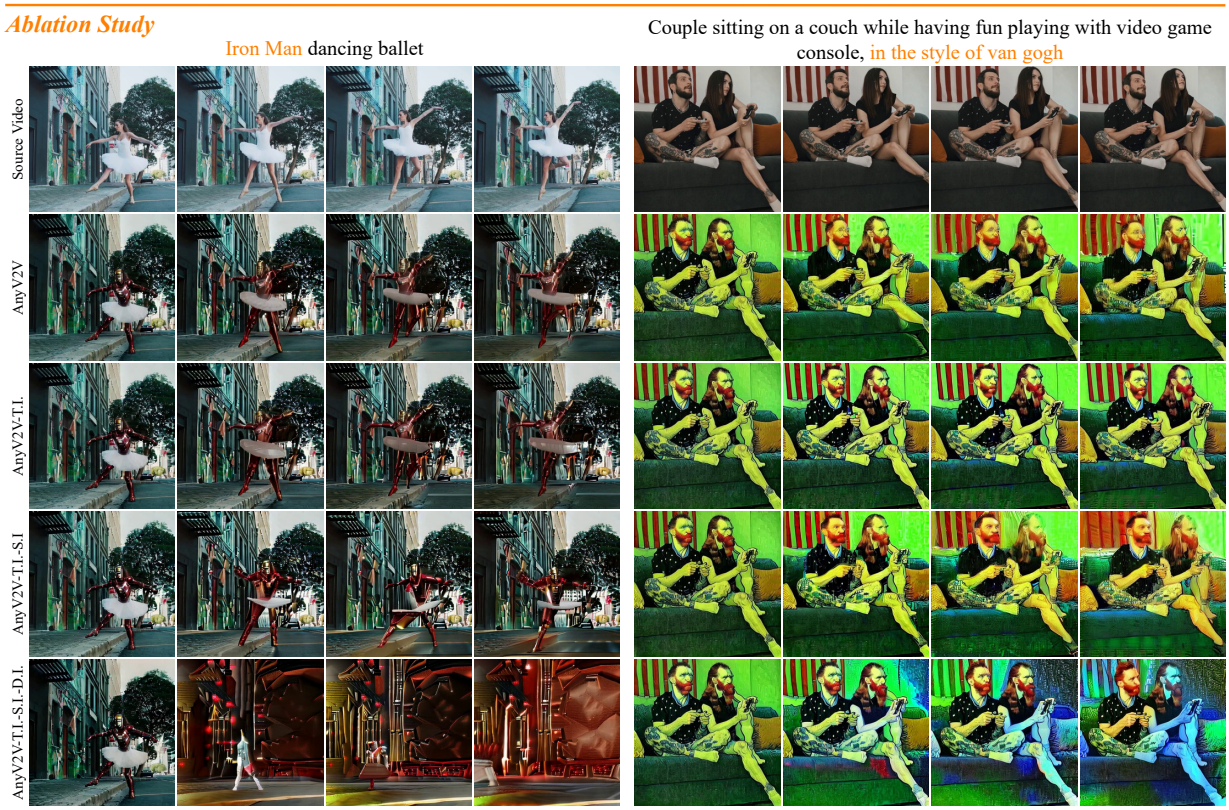


Figure 3.13: Visual comparisons of AnyV2V's editing results after disabling temporal feature injection (T.I.), spatial feature injection (S.I.) and DDIM inverted initial noise (D.I.).

disabling temporal feature injection in AnyV2V (I2VGen-XL), while we observe a slight increase in the CLIP-Image score value, the edited videos often demonstrate less adherence to the motion presented in the source video. For example, in the second frame of the "couple sitting" case (3rd row, 2nd column in the right panel in Figure 3.13), the motion of the woman raising her leg in the source video is not reflected in the edited video without applying temporal injection. On the other hand, even when the style of the video is completely changed, AnyV2V (I2VGen-XL) with temporal injection is still able to capture this nuanced motion in the edited video.

Effectiveness of Spatial Feature Injection As shown in Table 3.11, we observe a drop in the CLIP-Image score after removing the spatial feature injection mechanisms from our model, indicating that the edited videos are not smoothly progressed across

Table 3.11: Ablation study results for AnyV2V (I2VGen-XL). T. Injection and S. Injection correspond to temporal and spatial feature injection mechanisms, respectively.

Model	CLIP-Image ↑
AnyV2V (I2VGen-XL)	0.9648
AnyV2V (I2VGen-XL) w/o T. Injection	0.9652
AnyV2V (I2VGen-XL) w/o T. Injection & S. Injection	0.9637
AnyV2V (I2VGen-XL) w/o T. Injection & S. Injection & DDIM Inversion	0.9607

consecutive frames and contain more appearance and motion inconsistencies. Further illustrated in the fourth row of Figure 3.13, removing spatial feature injection will often result in incorrect subject appearance and pose (as shown in the "ballet dancing" case) and degenerated background appearance (evident in the "couple sitting" case). These observations demonstrate that directly generating edited videos from the DDIM inverted noise is often not enough to fully preserve the source video structures, and the spatial feature injection mechanisms are crucial for achieving better editing results.

DDIM Inverted Noise as Structural Guidance Finally, we observe a further decrease in CLIP-Image scores and a significantly degraded visual appearance in both examples in Figure 3.13 after replacing the initial DDIM inverted noise with random noise during sampling. This indicates that the I2V generation models become less capable of animating the input image when the editing prompt is completely out-of-domain and highlights the importance of the DDIM inverted noise as the structural guidance of the edited videos.

3.3.6 Discussion

AnyV2V extends flexible image editing to the video domain by using an edited first frame to specify the transformation and the source video to guide its propagation. This separation allows prompt-based edits, reference-based style transfer, subject replacement, and identity manipulation to share a video-editing procedure without task-specific fine-tuning. The ablations show why the edited frame alone is insufficient: inverted latents and spatial features help retain source structure, while temporal feature injection preserves motion that a frame-similarity metric may fail to capture. The contribution is thus both a flexible editing interface and a mechanism for carrying that edit through time.

The same decomposition explains the method's dependencies. An incorrect first-frame edit is passed to the video generator, and a satisfactory first frame cannot compensate for a backbone that fails on fast or complex motion. Some reference-based edits also require manual selection of that initial frame. Within these limits, AnyV2V demonstrates a route to broader editing capability through model composition, complementing OmniEdit's route through specialist-supervised training. UniVideo next asks whether understanding, generation, and editing can instead share a learned multimodal interface within one model.

3.4 UniVideo: Unifying Video Understanding, Generation, and Editing

3.4.1 Motivation

OmniEdit consolidates image-editing skills through supervision, whereas AnyV2V composes an image editor with a video generator at inference. UniVideo investigates a third setting: sharing an instruction interface and model across video understanding, generation, and editing. The challenge is to interpret multimodal instructions while retaining the fine visual details needed for source preservation and identity consistency.

UniVideo connects a multimodal large language model (MLLM) to a multimodal diffusion transformer (MMDiT). The MLLM supplies semantic representations of text, images, and videos; a parallel VAE pathway supplies reconstruction-oriented visual features. This dual-stream design avoids relying exclusively on semantic compression when an instruction includes detailed visual references. Related image-domain designs include Qwen-Image and OmniGen2 [164, 166].

Training covers text-to-image, text-to-video, image-to-video, image editing, and in-context video generation and editing. Evaluation distinguishes performance on these trained tasks from generalization to unseen task compositions and transfer from image editing to free-form video editing. Figure 3.17 illustrates the latter cases. Ablations compare multi-task with single-task training and test whether visual inputs should also be fed to the MMDiT branch.

The MLLM remains frozen, so its retained understanding and reasoning abilities are inherited rather than learned from scratch in this work. UniVideo is accordingly a post-trained multimodal generative system. Its contribution is the connection and joint training of these components for the evaluated video tasks.[2]

[2]Project resources and video demonstrations: https://congwei1230.github.io/UniVideo/.

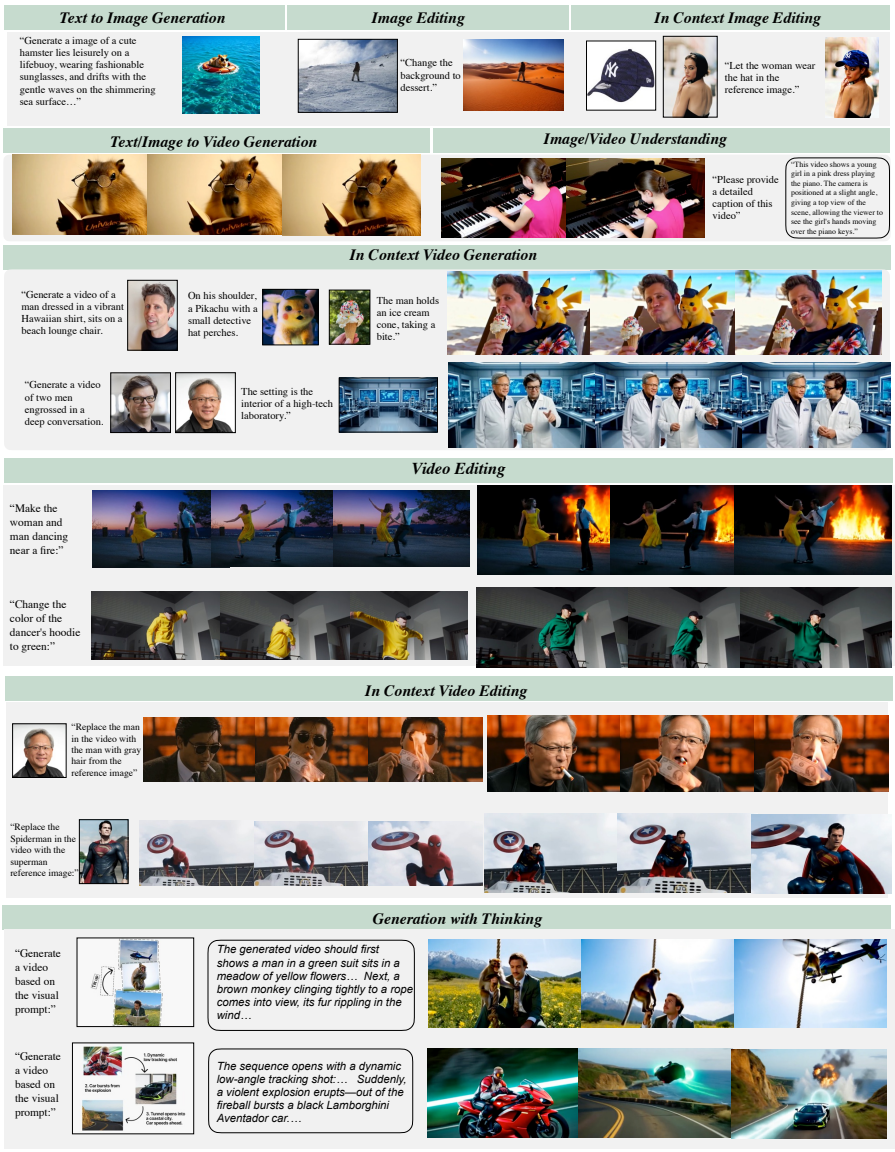


Figure 3.14: UniVideo is a unified system that can **understand** multi-modal instructions and **generate** multi-modal content.

3.4.2 Related Work

Unified understanding and generation. Chameleon and Emu3 use autoregressive multimodal token prediction; Show-o and Transfusion combine autoregressive and

diffusion-based components [140, 158, 173, 210]. MetaQueries and OmniGen2 connect pretrained understanding and generation components [116, 166]. UniVideo studies this connection for video, where detailed visual conditions and temporal structure make the choice of representation important.

Video task interfaces. AnyV2V composes an image editor with video generation, while VACE and UNIC introduce mechanisms for multi-task visual conditioning [71, 81, 187]. Omni-Video and UniVid also connect multimodal understanding with video generation [99, 139]. UniVideo's experiments examine a shared dual-stream architecture across understanding, generation, and editing, including untrained combinations of learned operations.

3.4.3 Architecture and Training

3.4.3.1 Model Architecture

As demonstrated in Figure 3.15, UniVideo consists of two main components: a multimodal large language model (MLLM) and a multimodal DiT (MMDiT). The MLLM handles visual–textual understanding, taking text, image, and video inputs and optionally producing text responses. The MMDiT focuses on visual generation with two branches: one incorporates high-level semantic information from the MLLM, while the other integrates fine-grained reconstruction signals from a VAE. Specifically, we extract the last-layer hidden states of the MLLM, which encode rich semantic features of the multimodal input. These are aligned to the input space of the MMDiT via a trainable connector and fed into its understanding stream. In parallel, visual signals are encoded by the VAE and passed into the MMDiT generation stream to preserve fine details. This design enables strong semantic grounding together with high-fidelity visual detail, which is especially important for video editing and identity-preserving generation.

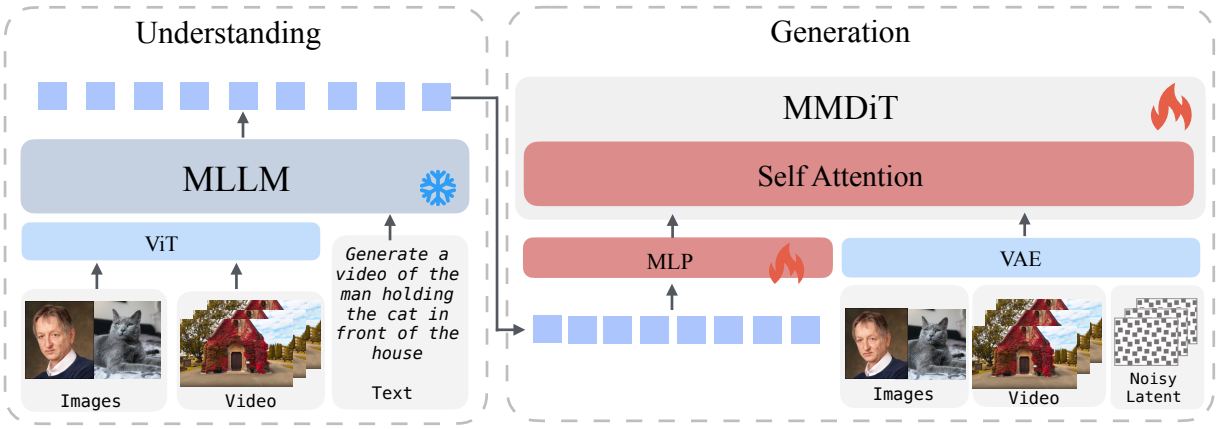


Figure 3.15: **Model architecture.** UniVideo is a dual-stream model consisting of an MLLM for understanding and an MMDiT module for generation. While prior work such as Qwen-Image and OmniGen2 [164, 166], explores a similar idea in the image domain, our model generalizes this design to video.

3.4.3.2 Unifying Multiple Tasks

We unify diverse multimodal tasks through natural language instructions, as illustrated in Figure 3.14. For text-to-video (T2V), the text input is processed by the MLLM, while the noisy video is fed into the MMDiT. For image-to-video (I2V), both the image and text are processed by the MLLM, whereas the image and noisy video are provided to the MMDiT. For in-context video generation (MultiID2V) and in-context video editing (ID-V2V), multiple visual conditions are often available, such as several reference images together with a reference video. Each visual signal is encoded with the VAE, padded to a uniform shape, concatenated along the temporal axis, and then processed with self-attention. Unlike prior approaches that introduce task-specific bias embeddings [187] or context adapter modules [71], we avoid task-specific customization. To help the MMDiT distinguish between condition latents and noisy video latents, we apply 3D positional embeddings, which preserve the spatial indices across frames while incrementing only the temporal dimension. In practice, we find this strategy more effective than Qwen2-VL's MRoPE [155], which offsets all axes whenever a new visual input is introduced.

3.4.3.3 Generation with thinking

UniVideo leverages its MLLM branch to interpret unconventional or hand-crafted prompts, as illustrated in Figure 3.18 and Figure 3.19. For example, users may provide an input image with manual annotations, which the MLLM translates into a structured plan and dense prompt tokens that guide video generation. Unlike agent-based approaches that invoke multiple downstream generators, UniVideo offers a more simplified design: the MMDiT directly integrates embeddings from the dense prompt tokens produced by the MLLM.

3.4.3.4 Training Strategy

Stage 1. Connector alignment between MLLM and MMDiT. In this stage, we train only the MLP connector while keeping both the MLLM and MMDiT frozen. Training is performed on pretraining samples across text-to-image (T2I) and text-to-video (T2V) generation tasks, as well as an image-reconstruction task in which only images from the text-to-image dataset are fed into the MLLM and the MMDiT reconstructs the image using visual features from the MLLM. After this stage, UniVideo can generate images and videos conditioned on text or image inputs from the MLLM.

Stage 2. Fine-tuning on T2I/T2V. In this stage, we keep the MLLM frozen and fine-tune the connector and MMDiT on small-scale, high-quality T2I/T2V samples. After this stage, UniVideo achieves performance comparable to the MMDiT backbone that uses its own text encoder.

Stage 3. Multi-task Training. Finally, we extend training to include in-context generation (multi-ID-to-video), in-context video editing (modifying the input video based on a reference image, such as ID swapping, ID addition, ID deletion, or style transfer), image editing and image-to-video tasks, alongside the previous text-to-image (T2I) and text-to-video (T2V) tasks. We keep the MLLM frozen and only train the connector and MMDiT. This stage enables UniVideo to unify a broad range of video generation and editing tasks under multimodal instruction. The training settings are given in Table 3.15.

3.4.4 Experiments

The experiments examine three aspects of a shared multimodal interface: performance across trained tasks, generalization to unseen tasks and visual prompts, and the architectural choices needed to connect instruction understanding with generation. After the

implementation details in Section 3.4.4.1, Section 3.4.4.2 compares UniVideo with state-of-the-art methods across video understanding and generation tasks. Section 3.4.4.3 studies zero-shot generalization and visual-prompt understanding, and Section 3.4.4.4 tests the design choices.

Table 3.12: **Visual understanding and video generation** results. Best results are shown in **bold**, and second-best are underlined. Following BLIP3-o and MetaQueries [19, 116], *understanding results for UniVideo are reported using its MLLM backbone (Qwen2.5-VL-7B).

Model	**Understanding**			**Video Generation**
	MMB	MMMU	MM-Vet	VBench T2V
Video Understanding Model				
LLaVA-1.5 [92]	67.8	36.4	36.3	×
LLaVA-NeXT [93]	79.3	51.1	57.4	×
Video Generation Model				
CogVideoX (T2V) [183]	×	×	×	81.61
I2VGen-XL [201]	×	×	×	×
HunyuanVideo (T2V) [78]	×	×	×	83.24
Step-Video-(T2V)[102]	×	×	×	81.83
Wan2.1 (T2V) [142]	×	×	×	**84.70**
Unified Understanding & Generation Model				
Emu3 [158]	58.5	31.6	37.2	80.96
TokenFlow-XL [125]	76.8	43.2	48.2	×
Janus [165]	69.4	30.5	34.3	×
JanusFlow [103]	74.9	29.3	30.9	×
OmniGen2 [166]	79.1	53.1	61.8	×
Show-o [173]	-	26.7	-	×
BAGEL [34]	**85.0**	55.3	**67.2**	×
Show-o2 [174]	79.3	48.9	56.6	81.34
UniVideo *	83.5	**58.6**	66.6	83.48

Table 3.13: **Text-to-image generation** results on the GenEval [44] benchmark. † refers to methods using the LLM rewriter.

Type	Method	Single obj.	Two obj.	Counting	Colors	Position	Color attr.	Overall ↑
Image Model	SDv2.1	0.98	0.51	0.44	0.85	0.07	0.17	0.50
	SDXL	0.98	0.74	0.39	0.85	0.15	0.23	0.55
	IF-XL	0.97	0.74	0.66	0.81	0.13	0.35	0.61
	LUMINA-Next	0.92	0.46	0.48	0.70	0.09	0.13	0.46
	SD3-medium	0.99	0.94	0.72	0.89	0.33	0.60	0.74
	FLUX.1-dev	0.99	0.81	0.79	0.74	0.20	0.47	0.67
	NOVA	0.99	0.91	0.62	0.85	0.33	0.56	0.71
Unified Model	OmniGen	0.98	0.84	0.66	0.74	0.40	0.43	0.68
	TokenFlow-XL	0.95	0.60	0.41	0.81	0.16	0.24	0.55
	Janus	0.97	0.68	0.30	0.84	0.46	0.42	0.61
	Janus Pro	0.99	0.89	0.59	0.90	0.79	0.66	0.80
	Emu3-Gen†	0.99	0.81	0.42	0.80	0.49	0.45	0.66
	Show-o	0.98	0.80	0.66	0.84	0.31	0.50	0.68
	MetaQuery-XL†	–	–	–	–	–	–	0.80
	BLIP3-o† 4B	–	–	–	–	–	–	0.81
	BLIP3-o† 8B	–	–	–	–	–	–	0.84
	BAGEL	0.99	0.94	0.81	0.88	0.64	0.63	0.82
	UniWorld-V1	0.99	0.93	0.79	0.89	0.49	0.70	0.80
	OmniGen2	1.00	0.95	0.64	0.88	0.55	0.76	0.80
	UniVideo	0.93	0.63	0.23	0.76	0.21	0.41	0.53
	UniVideo†	0.98	0.85	0.36	0.90	0.45	0.64	0.69

Table 3.14: **Image editing** results on ImgEdit-Bench [186] and GEdit-Bench-EN [97] (SC: Semantic Consistency, PQ: Perceptual Quality, O: Overall). "Overall" is calculated by averaging all scores across tasks. The best results are highlighted in **bold**, and the second-best results are underlined.

	ImgEdit-Bench										GEdit-Bench		
Model	**Add**	**Adj.**	**Ext.**	**Rep.**	**Rm.**	**Bg.**	**Sty.**	**Hyb.**	**Act.**	**Overall**	**G-SC**	**G-PQ**	**G-O**
GPT-4o	4.61	4.33	2.90	4.35	3.66	4.57	4.93	3.96	4.89	4.20	7.85	7.62	7.53
MagicBrush	2.84	1.58	1.51	1.97	1.58	1.75	2.38	1.62	1.22	1.90	4.68	5.66	4.52
Instruct-P2P	2.45	1.83	1.44	2.01	1.50	1.44	3.55	1.20	1.46	1.88	3.58	5.49	3.68
AnyEdit	3.18	2.95	1.88	2.47	2.23	2.24	2.85	1.56	2.65	2.45	3.18	5.82	3.21
OmniGen	3.47	3.04	1.71	2.94	2.43	3.21	4.19	2.24	3.38	2.96	5.96	5.89	5.06
ICEdit	3.58	3.39	1.73	3.15	2.93	3.08	3.84	2.04	3.68	3.05	5.11	6.85	4.84
Step1X-Edit	3.88	3.14	1.76	3.40	2.41	3.16	4.63	2.64	2.52	3.06	7.09	6.76	**6.70**
BAGEL	3.56	3.31	1.70	3.30	2.62	3.24	4.49	2.38	4.17	3.20	**7.36**	6.83	6.52
UniWorld-V1	3.82	3.64	**2.27**	3.47	3.24	2.99	4.21	2.96	2.74	3.26	4.93	**7.43**	4.85
OmniGen2	3.57	3.06	1.77	3.74	3.20	3.57	4.81	2.52	4.68	3.44	7.16	6.77	6.41
UniVideo	**4.22**	**3.90**	1.66	**3.96**	**3.73**	**4.11**	4.54	**3.55**	**4.81**	**3.83**	7.08	7.08	6.41

3.4.4.1 Implementation Details

We adopt Qwen2.5-VL-7B [7] as the MLLM backbone and HunyuanVideo-T2V-13B [78] as the MMDiT backbone. The original HunyuanVideo uses two text encoders; we remove them and instead use Qwen2.5-VL as the unified multi-modal embedder. To align feature dimensions between Qwen2.5-VL and HunyuanVideo, we apply an MLP with a 4× expansion. The training configuration and data construction are described below.

Training configuration and data. We remove the original CLIP conditioning and classifier-free-guidance distillation embeddings. Table 3.15 reports the three stages.

Table 3.15: Training hyperparameters across different stages. Stage 1: Connector alignment, Stage 2: Fine-tuning, Stage 3: Multi-task training.

Hyperparameters	**Stages**		
	Stage 1 (Connector Alignment)	**Stage 2** (Fine-tuning)	**Stage 3** (Multi-task)
Learning rate	1×10^{-4}	2.0×10^{-5}	2.0×10^{-5}
LR scheduler	Constant	Constant	Constant
Weight decay	0.0	0.0	0.0
Gradient norm clip	1.0	1.0	1.0
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.95, \epsilon = 1.0 \times 10^{-15}$)		
Warm-up steps	50	50	50
Training steps	15K	5K	15K
EMA ratio	-	0.9999	0.9999
Gen resolution (min, max)	(240, 480)	(480, 854)	(480, 854)
Gen frames (min, max)	(1, 1)	(1, 129)	(1, 129)
Und resolution (min, max)	(240, 480)	(480, 854)	(480, 854)
Und frames (min, max)	(1, 1)	(1, 4)	(1, 4)
Diffusion timestep shift	5.0	5.0	5.0

Identity-conditioned video tasks. Following the ConceptMaster pipeline, Qwen2.5-VL-7B proposes subject tags from keyframes. GroundingDINO localizes subjects occupying

10–60% of the first frame, and SAM2 tracks them through the video; mask-area visibility filtering removes unsuitable tracks. A one-billion-parameter, Wan-like inpainter copies interleaved half-blocks to a control branch and injects residuals conditioned on the masked video, mask, and text. It is trained on VIVID-10M.

For swapping, an inpainted video and a subject reference condition reconstruction of the original video. Addition similarly uses the original as target, but inpainting omits the subject from its caption to remove it from the source. Deletion reverses this direction, mapping the original to the inpainted video. Qwen2.5-VL-72B generates instructions from first-frame pairs. Human filtering checks visual quality, instruction fulfillment, and consistency with the source, rejecting artifacts, flicker, partial edits, and distortions. Because swapping targets are the original videos, a separate target-identity check is unnecessary.

Stylization. Stylized text-to-video samples are converted into realistic counterparts using grayscale, eightfold downsampled and upsampled tile conditions. The one-billion-parameter control model copies half of the backbone blocks and is trained on 10K real-to-real pairs. The realistic counterpart becomes the editing input and the stylized video the target.

Image editing and generation. Image-editing data combine FLUX-Kontext, OmniEdit, ImgEdit, and ShareGPT4o sources. A Qwen evaluator assigns 0–10 scores for quality, instruction fulfillment, and source consistency; samples failing any filtering threshold are discarded, yielding approximately 500K pairs. Text-to-image and text-to-video training also use internal data.

3.4.4.2 Main Results

3.4.4.2.1 Visual Understanding and Generation UniVideo's visual understanding is powered by a frozen pretrained MLLM. Freezing the MLLM preserves its strong native understanding ability and prevents performance degradation from joint training with generative tasks. As shown in Table 3.12, UniVideo achieves competitive scores of 83.5 on MMBench [98], 58.6 on MMMU[195], and 66.6 on MM-Vet[193] for understanding tasks. At the same time, it retains strong generation ability, supporting both I2V and T2V within a single unified model. In contrast, baseline models rely on different variants for different tasks, whereas UniVideo reaches performance comparable to the HunyuanVideo backbone on the VBench[64] benchmarks.

3.4.4.2.2 Image Generation tasks.

Image Generation. We evaluate UniVideo on text-to-image generation using the GenEval benchmark [44]. Methods marked with † denote approaches that employ an LLM-based prompt rewriter. Despite being an image–video generalist model and using only a small proportion of T2I samples during Stage 3 training, UniVideo retains competitive text-to-image performance and outperforms many specialized text-to-image models and unified text–image multimodal models (Table 3.13).

Image Editing. We further evaluate UniVideo on image editing tasks using ImgEdit-Bench [186] and GEdit-Bench-EN [97]. We report Semantic Consistency (SC) as the primary evaluation metric. Despite being an image–video generalist model, UniVideo achieves strong editing performance and surpasses several specialized image editing models (Table 3.14).

3.4.4.2.3 Video Generation tasks. For the **text-to-video generation task**, we use the prompt suite provided in VBench [64][3] which contains 946 prompts covering 16 dimensions, including *subject consistency, background consistency, aesthetic quality, imaging quality, object class, multiple objects, color, spatial relationship, scene, temporal style, overall consistency, human action, temporal flickering, motion smoothness, dynamic degree, appearance style*. Following VBench, we obtain the semantic score and the quality score based on the scores on these dimensions. The aggregate text-to-video result is reported in Table 3.12.

3.4.4.2.4 In-Context Video Generation Benchmark: Following FullDiT [74] and OmniGen2 [166], we construct a test set covering both single-ID and multi-ID video generation scenarios. In the single-ID setting, a subject may have multiple reference images (e.g., different viewpoints of a person or object). In the multi-ID setting, the references include 2–4 distinct identities. The test set contains 20 cases: ten single-identity cases and ten with two to four identities, covering objects, people, animals, and cartoon characters.

Metrics: We conduct both human evaluations and automatic metric assessments. For human evaluation, we follow the protocols of Instruct-Imagen [59] and OmniGen2 [166] to perform a systematic study. Each sample is rated by at least three annotators on (i) subject

[3]VBench GPT-enhanced prompts: https://github.com/Vchitect/VBench/blob/master/prompts/augmented_prompts/gpt_enhanced_prompts/all_dimension_longer.txt.

consistency (SC), (ii) prompt following (PF), and (iii) overall video quality (VQ). Scores in each category are drawn from $\{0, 0.5, 1\}$, where 0 indicates inconsistency or extremely poor quality, and 1 indicates full consistency or high quality. For automatic evaluation, we adopt two metrics from VBench [64]: smoothness and aesthetics.

Baselines: We compare UniVideo with the state-of-the-art open-source model VACE, given the scarcity of video models capable of in-context generation. We also include commercial baselines such as Pika2.2 and Kling1.6.

Results: Quantitative comparisons are presented in Table 3.16. UniVideo achieves superior or competitive performance across all metrics compared to the baselines. Additional results are shown in Figure 3.16. Notably, baseline models often struggle with complex instructions involving multiple identities (e.g., when the number of reference images is 4), whereas UniVideo can accurately follow instructions while preserving identity.

Table 3.16: Quantitative comparison on **In-Context Generation**. Human evaluation includes Subject Consistency (SC), Prompt Following (PF), and Overall Video Quality (VQ). Automatic metrics measure Smoothness and Aesthetics. Best results in **bold**, second-best underlined.

	Single Reference					**Multi Reference (≥2)**				
Model	SC↑	PF↑	VQ↑	Smooth↑	Aesth↑	SC↑	PF↑	VQ↑	Smooth↑	Aesth↑
VACE	0.31	0.65	0.42	0.922	5.426	0.48	0.53	0.48	0.530	5.941
Kling1.6	0.68	**0.95**	0.88	0.938	**5.896**	0.73	0.45	**0.95**	0.916	6.034
Pika2.2	0.45	0.43	0.15	0.928	5.125	0.71	0.48	0.43	0.898	5.176
UniVideo	**0.88**	0.93	**0.95**	**0.943**	5.740	**0.81**	**0.75**	0.85	**0.942**	**6.128**

3.4.4.2.5 In-Context Video Editing **Benchmark:** Following UNIC [187], we construct a test set covering four editing types: swap, delete, addition, and style transfer. Each example consists of a source video and a reference image, together with a natural language instruction. We adapt mask-based UNICBench specifications to text instructions; stylization is assessed for preservation of source content and motion.

Metrics: We adopt the evaluation protocol of UNIC [187] and conduct automatic metric assessments. Specifically, we use CLIP-I and DINO-I to measure identity consistency, and CLIP-Score to measure prompt following.

Baselines: We compare UniVideo with state-of-the-art task-specific expert models, including UNIC, AnyV2V, and VideoPainter. We also evaluate against commercial models

Table 3.17: Quantitative comparison with task-specific expert models on **In-Context Video Editing**. Our model is the **only mask-free approach**, capable of performing edits solely based on instructions without requiring explicit mask inputs to indicate editing regions. Despite this more challenging setting, it achieves superior or competitive performance across most metrics compared to state-of-the-art task-specific expert baselines. Best scores are shown in **bold**, and second-best are underlined.

In Context Insert					
Model	Identity		Alignment	Video Quality	
	CLIP-I↑	DINO-I↑	CLIP-score↑	Smoothness↑	Aesthetic↑
VACE	0.513	0.105	0.103	0.947	5.693
UNIC	0.598	0.245	0.216	0.961	5.627
Kling1.6	0.632	0.287	0.246	**0.993**	5.798
Pika2.2	0.692	**0.399**	0.253	0.951	5.591
UniVideo *(Mask Free)*	**0.693**	0.398	**0.259**	0.943	**6.031**

In Context Swap					
Model	Identity		Alignment	Video Quality	
	CLIP-I↑	DINO-I↑	CLIP-score↑	Smoothness↑	Aesthetic↑
VACE	0.703	0.391	0.218	0.960	5.961
UNIC	0.725	0.429	0.242	0.971	6.056
Kling1.6	0.707	**0.437**	0.211	**0.995**	6.042
Pika2.2	0.704	0.406	0.211	0.967	5.097
AnyV2V	0.605	0.229	0.218	0.917	4.842
UniVideo *(Mask Free)*	**0.728**	0.427	**0.244**	0.973	**6.190**

In Context Delete					
Model	Video Reconstruction		Alignment	Video Quality	
	PSNR↑	RefVideo-CLIP↑	CLIP-score↑	Smoothness↑	Aesthetic↑
VACE	20.601	0.874	0.206	0.968	**5.637**
UNIC	19.171	0.817	**0.217**	0.970	5.493
Kling1.6	15.476	0.888	0.208	**0.998**	4.965
AnyV2V	19.504	0.869	0.205	0.964	5.325
VideoPainter	**22.987**	**0.920**	0.212	0.957	5.403
UniVideo *(Mask Free)*	17.980	0.888	0.214	0.971	5.498

In Context Stylization					
Model	Style & Content		Alignment	Video Quality	
	CSD-Score↑	ArtFID↓	CLIP-score↑	Smoothness↑	Aesthetic↑
AnyV2V	0.207	43.299	0.195	0.937	4.640
StyleMaster	**0.306**	38.213	0.188	0.952	5.121
UNIC	0.197	**36.198**	0.215	0.932	5.045
UniVideo *(Mask Free)*	0.228	37.877	**0.226**	**0.963**	**6.281**



Figure 3.16: **Qualitative comparison** of UniVideo with SoTA Task Specific Experts on **In Context Generation** and **In Context Editing** tasks.

such as Pika2.2 and Kling1.6. **Note** that the baselines are given explicit masks to localize the editing region, whereas UniVideo operates without masks.

Results: Quantitative comparisons are presented in Table 3.17. Although UniVideo is evaluated under the more challenging mask-free setting, it still achieves superior or competitive performance across most metrics compared to the baselines. Additional results are shown in Figure 3.16. UniVideo can accurately follow instructions while preserving the identity of the reference images.

3.4.4.3 Model Analysis

3.4.4.3.1 Zero Shot Generalization We observe two forms of generalization in UniVideo. Although the training data of UniVideo does not include general free-form video editing tasks, it transfers this ability from diverse image editing data and in-context video editing data (limited to ID deletion, swapping, addition, and stylization) to the video domain, enabling it to handle free-form video editing instructions (e.g., changing material or

environment). We also observe that UniVideo is capable of handling task compositions. It can combine in-context editing with style transfer, or perform multiple edits simultaneously (e.g., deleting one identity while adding another). Figure 3.17 shows examples of both forms of generalization.



Figure 3.17: **Zero-Shot Generalization**. We demonstrate two types of generalization. (i) UniVideo was not trained on General Free-form Video Editing data. It transfers this ability from diverse image editing data to the video domain through joint training with in-context video generation and editing data (limited to ID deletion, swapping, addition, and stylization), enabling it to handle previously unseen video editing instructions. (ii) UniVideo can also generalize to novel task compositions, even though it was not explicitly trained on such compositions.

3.4.4.3.2 Generation With Thinking We demonstrate UniVideo's *generation-with-thinking* capability in Figure 3.19, focusing on visual prompt understanding that requires structured multimodal reasoning. We consider two representative types of visual prompts. In the first setting, users draw reference images and story plans on a canvas. Here, the model can interpret the plan and generate corresponding videos. In the second setting, annotations are drawn directly on an input image, which the model treats as an I2V task; in this case, UniVideo can interpret the motion or new events described by the visual prompt. These results highlight the advantages of UniVideo in handling complex multimodal instructions. Although the qualitative results are obtained in a zero-shot setting,

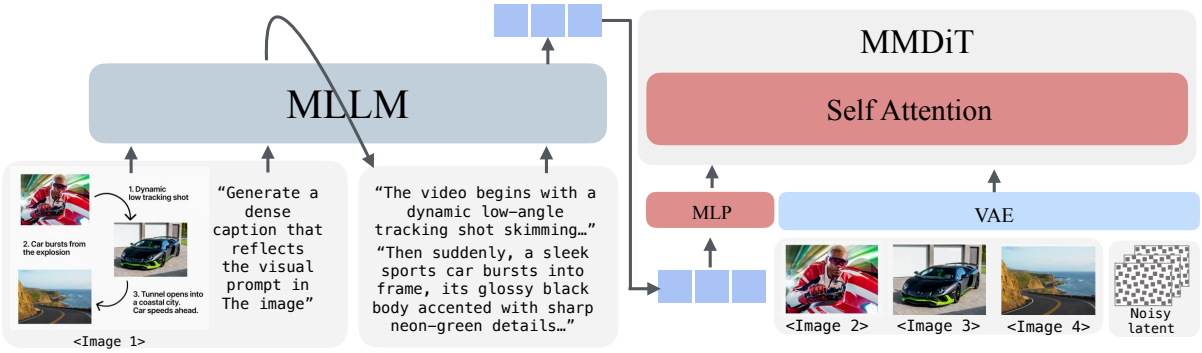


Figure 3.18: **Generation with thinking.** UniVideo leverages the MLLM stream to understand and interpret user intent from complex multimodal prompts that cannot be handled by the DiT alone. For example, users can provide diagrams or visual annotations to guide video generation without writing dense textual prompts.



Figure 3.19: **Qualitative results of generation with thinking.** We illustrate two types of visual prompts: in the first three examples, annotations are drawn on a canvas, while in the last example, the annotation is drawn directly on an input image.

future end-to-end training on task-specific data may further improve performance.

3.4.4.4 Ablation Study

Our ablation studies address the following two questions: (i) *Does multi-task learning enhance performance compared with single-task learning?* (ii) *Is our two-branch design effective? Specifically, should visual embeddings be streamed to both the MLLM and MMDiT branches?*

We conduct human evaluations on In-Context Video Editing and In-Context Video Generation, using the same evaluation protocol as in Section 3.4.4.2.4. (i) To study multi-task learning, we compare UniVideo with a single-task baseline. The single-task baseline shares the same architecture as UniVideo but requires an independent model for each task and has access only to task-specific data. Results in Table 3.18 demonstrate the effectiveness of multi-task learning, especially for the editing task, where UniVideo benefits from large-scale image editing data during joint learning. (ii) To evaluate the impact of streaming visual inputs, we compare UniVideo with a variant that shares the same architecture: **w/o visual for MMDiT:** visual inputs are fed only to the MLLM branch.

As shown in Table 3.18, feeding visual inputs exclusively to the MLLM results in a dramatic drop in identity preservation.

Table 3.18: Ablation study across single-task model, UniVideo, and UniVideo w/o Visual for MMDiT across different In-Context tasks.

		Single-task			UniVideo			UniVideo w/o Visual for MMDiT		
		PF↑	SC↑	VQ↑	PF↑	SC↑	VQ↑	PF↑	SC↑	VQ↑
In-context generation	singleid	0.85	0.83	0.93	0.93	0.88	0.95	0.75	0.32	0.86
	multiid	0.75	0.79	0.73	0.75	0.81	0.85	0.81	0.23	0.83
In-context editing	insert	0.81	0.85	0.86	0.92	0.92	0.91	0.68	0.18	0.75
	swap	0.53	0.78	0.68	0.91	0.85	0.85	0.63	0.15	0.62
	delete	0.32	0.42	0.89	0.52	0.58	0.92	0.21	0.13	0.63
	stylization	0.56	0.43	0.63	0.79	0.64	0.64	0.86	0.11	0.57
Average		0.64	0.67	0.79	0.80	0.78	0.85	0.66	0.18	0.71

3.4.5 Discussion

UniVideo shows how a shared multimodal interface can connect instruction understanding with video generation and editing. Its dual-stream design addresses two different demands

of a creative request: interpreting what the user wants and retaining the visual details that the output must preserve. The identity-preservation ablation makes this distinction concrete: supplying visual inputs only through the MLLM loses information that the parallel VAE pathway retains. Joint training further allows image-editing supervision to support free-form video editing, while the generalization examples show untrained combinations of learned operations. These findings support sharing representations and supervision across tasks, beyond merely placing separate task interfaces in one system.

The distinction between understanding a request and executing it nevertheless remains important. UniVideo inherits understanding from its frozen MLLM, while the connector and diffusion generator learn to use that representation for synthesis. Over-editing can still alter source content or motion beyond the requested change. Relative to OmniEdit's consolidation of specialist supervision and AnyV2V's inference-time composition, UniVideo advances task flexibility through jointly learned generation capabilities. Its results also show why a common interface must be evaluated for both the changes it enables and the content it preserves.

3.5 Summary

This chapter addressed limited task flexibility in image and video generation and editing. OmniEdit consolidated supervision from task specialists into a generalist image editor, supporting seven categories of edits within a shared model. Its data curation and architecture studies showed the importance of both reliable supervision and effective interaction between instruction and image features. AnyV2V extended flexible editing to the video domain by propagating an edited first frame throughout a video. This composition supported prompt-based edits, style transfer, identity manipulation, and subject-driven editing without fine-tuning for each task. UniVideo brought understanding, generation, and editing into a shared model that interprets multimodal instructions and distinguishes the intended task. Its nine-task coverage and generalization experiments demonstrated that jointly learned capabilities can support requests beyond the training tasks. Together, these studies broaden the ways users can specify a request and the range of tasks a system can perform: through specialist supervision, the composition of existing models, or unified multimodal learning. They address task flexibility through distinct mechanisms, rather than requiring one model to be trained separately for every operation. Chapter 4 turns to the additional control requirements introduced by video's temporal dimension, focusing on speech and historical context.

Chapter 4

Temporal Control through Speech and Historical Context

This chapter is based on the following works. Author contributions are described in the Statement of Contributions.

- Cong Wei, Bo Sun, Haoyu Ma, Ji Hou, Felix Juefei-Xu, Zecheng He, Xiaoliang Dai, Luxin Zhang, Kunpeng Li, Tingbo Hou, Animesh Sinha, Peter Vajda, and Wenhu Chen. *MoCha: Towards Movie-Grade Talking Character Generation*. In *Advances in Neural Information Processing Systems 38* (NeurIPS), 2025. Selected for a Spotlight presentation. (Section 4.2)
- Shuo Chen, Cong Wei, Sun Sun, Tiancheng Shen, Ping Nie, Kai Zou, Ge Zhang, Ming-Hsuan Yang, and Wenhu Chen. *Context Forcing: Consistent Autoregressive Video Generation with Long Context*. In Proceedings of the *International Conference on Machine Learning* (ICML), 2026. (Section 4.3)

4.1 Overview

The preceding chapter considers instruction-guided tasks across images and videos, including consistency when an edit is carried through a clip. This chapter focuses on the additional temporal dimension of control: how speech and earlier context guide the unfolding video. A speech track should guide visible performance and the allocation of dialogue

turns. Historical context should constrain a video's continuation so that subjects, scenes, and events remain coherent over time. Neither form of temporal control is guaranteed by the ability to synthesize plausible individual frames.

This chapter studies two forms of temporal control. MoCha (Section 4.2) addresses speech-conditioned video generation. It incorporates audio into a video diffusion transformer and coordinates multi-character dialogue, connecting speech timing with visible performance to support movie-grade storytelling. Context Forcing (Section 4.3) addresses the use of historical context in long-video generation. Its long-context teacher guides the learning of longer-range dependencies, while Slow-Fast Memory balances recent detail with longer-term context. The resulting system supports real-time long-video generation while mitigating forgetting and drift.

The shared question is how conditioning information can remain effective across time. Speech alignment, dialogue behavior, history use, and continuation quality assess temporal control, while throughput measures the additional practical requirement of real-time generation.

4.2 MoCha: Speech-Conditioned Video Generation

4.2.1 Motivation

Text-conditioned video generation does not by itself specify how visible speech should align with an audio track. Conversely, animating a reference portrait does not fully address dialogue scenes with body motion, changing cameras, and multiple speakers. This section studies *Talking Characters*: generating video from text and speech while coordinating lip motion, expression, action, and speaker identity.

MoCha conditions a video diffusion transformer directly on text and audio, without requiring a reference image, pose skeleton, or facial-keypoint sequence. The key difficulty is temporal alignment: compressed video tokens and audio features have different resolutions, while natural motion depends on information beyond a single instant. Localized audio attention exposes the relevant temporal neighborhood to each video token. A modality-aware curriculum combines speech-labeled video with text-only video, and structured character tags associate dialogue turns with speakers.

The following sections define the task, develop the conditioning and training mechanisms, and evaluate them on MoCha-Bench. Automatic synchronization measures and

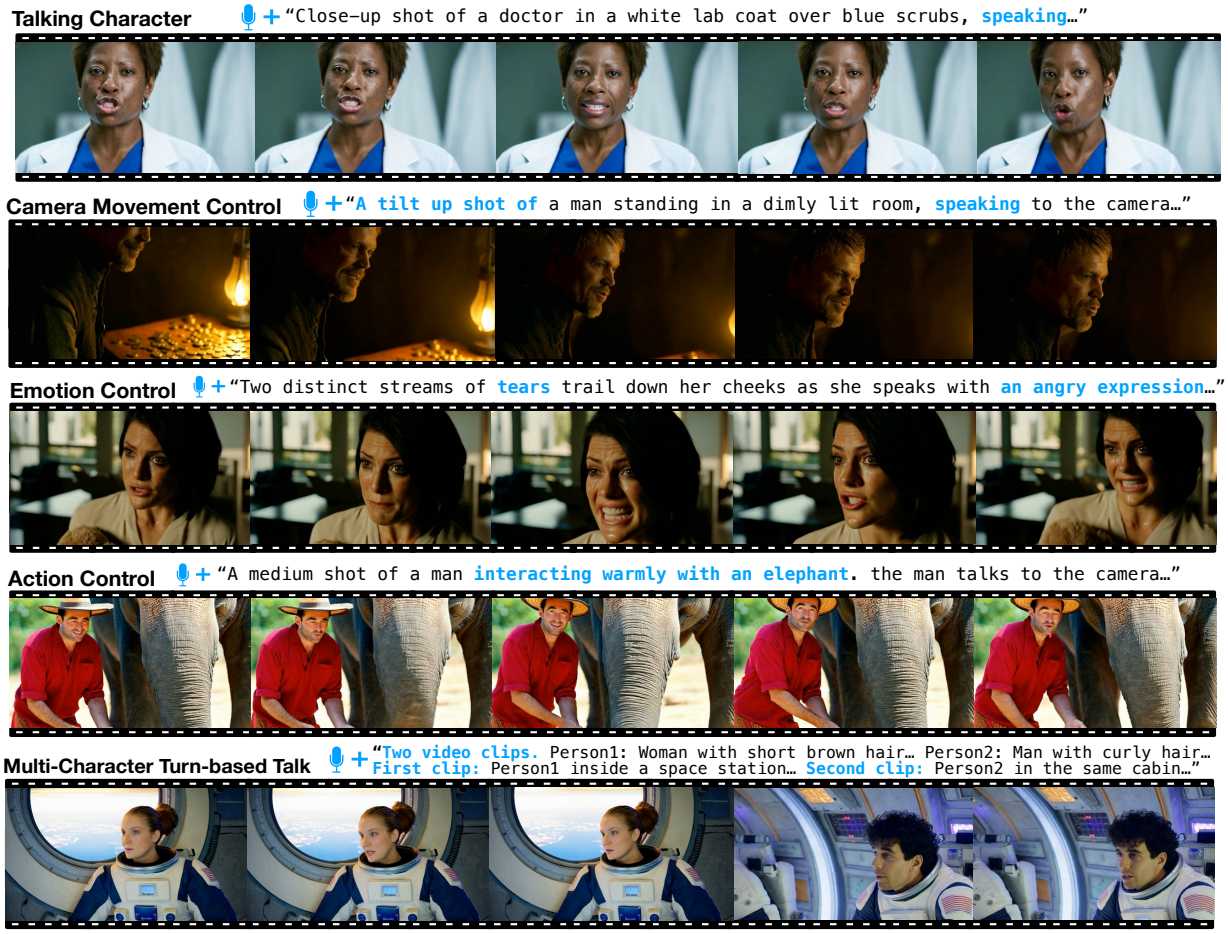


Figure 4.1: MoCha is an end-to-end dialogue-centric video generation model that takes only **speech** and **text** as input, without requiring any auxiliary conditions.

human judgments assess different aspects of the output; ablations isolate the effects of audio attention, training, and character tagging.[1]

4.2.2 Related Work

Speech-conditioned animation. Talking-head systems commonly animate a reference face through facial landmarks or 3D motion representations. SadTalker, AniPortrait, and VLOGGER illustrate these approaches [28, 160, 203]. Diffusion-based methods such

[1]Project resources and video demonstrations: https://congwei1230.github.io/MoCha/.

as EMO, Hallo, and Hallo3 learn richer audio-driven motion while retaining portrait or reference-based inputs [30, 144, 177]. MoCha instead targets text-and-speech-conditioned character generation, including body action, camera motion, and speaker assignment.

Video-generation backbones. Video LDM, CogVideoX, and MovieGen develop the motion modeling and visual fidelity needed by general video generators [11, 120, 183]. The general diffusion background appears in Chapter 2. This section's distinction is the conditioning and supervision required to align speech with visible character behavior; a strong video prior alone does not define that alignment.

4.2.3 Task Definition

We introduce a novel task, *Talking Characters*, which focuses on generating digital characters that exhibit realistic, human-like behaviors from natural language and speech input. The task is motivated by the goal of automating dialogue-centric film production, going beyond *traditional video generation* that typically focuses on non-dialogue, narration-based scenes.

Talking Characters differs from conventional *talking-head* generation—which is restricted to a single face, fixed camera, and square crop—by enabling full-body character synthesis across a range of shot sizes (e.g., close-up, medium, wide) with dynamic camera motion. It supports the generation of one or more characters situated within a contextually appropriate scene.

The task is formally defined by the input-output specification and evaluation protocol, in Table 4.1.

4.2.4 Speech-Conditioned Video Generation

In this section, we introduce the MoCha model, the first model to generate talking characters. We begin by outlining its architecture in Section 4.2.4.1, followed by the localized audio attention mechanism in Section 4.2.4.2. Next, we describe the method of generating multiple clips in Section 4.2.4.3. Finally, we explain the training strategy in Section 4.2.4.4.

4.2.4.1 Audio + Text to Video Diffusion Transformers

Figure 4.2 presents the overall architecture of MoCha. Unlike prior works that employ text-to-image (T2I) U-Net [31, 144, 145, 177] for talking head generation, MoCha is a

Aspect	Description
Input	*Text Prompt* y: Natural language description including (1) environment, (2) character appearance and (3) actions, (4) facing direction and emotion, (5) frame position, (6) camera movement, and (7) shot size. *Speech Audio* $a \in \mathbb{R}^L$: Raw waveform signal that drives characters' lip movements, facial expressions, and body motions.
Output	*Video* $\nu \in \mathbb{R}^{T\times H\times W\times 3}$: A rendered video of one or more talking characters (human, 3D cartoon, or animal), where T is the number of frames and (H, W) is the spatial resolution.
Eval	The generated characters are expected to perform well across the following five axes: *(1) Lip-Sync Quality:* Accurate and temporally aligned lip movements with respect to α. *(2) Expression Naturalness:* Expressive and coherent facial emotions that align with both y and a. *(3) Action Naturalness:* Realistic body gestures that reflect described actions in y, synchronized with a. *(4) Text Alignment:* Coherence between visual content and y, including visual layout, character appearance, and camera motion. *(5) Visual Quality:* High fidelity and temporal consistency, free from visual artifacts.

Table 4.1: Task definition of Talking Characters: formalized inputs (y, a), output ν, and evaluation criteria.

diffusion transformer (DiT) [118].

Model Architecture. MoCha adopts a fully tokenized design, where both text and speech inputs are projected into token sequences and integrated with video tokens through cross-attention. Given a video $\nu \in \mathbb{R}^{T\times H\times W\times 3}$ with T frames, we encode it into a latent representation $x_0 \in \mathbb{R}^{\tau\times h\times w\times c}$ using a 3D VAE, which down-samples the video spatially and temporally. We define the temporal down-sampling ratio as $r = \frac{T}{\tau}$. Next, x_0 is flattened into a sequence of tokens of size $(\tau \times h \times w) \times c$ and passed to the DiT model $f_\theta(\cdot)$. Within each DiT block, the model first applies self-attention to the video tokens x_0, followed by sequential cross-attention with the text condition tokens y and audio condition tokens α. The audio tokens $\alpha \in \mathbb{R}^{T\times c}$ are derived from raw waveforms a using Wav2Vec2 [4] and processed through MLPs to align its feature dimension with the video tokens.

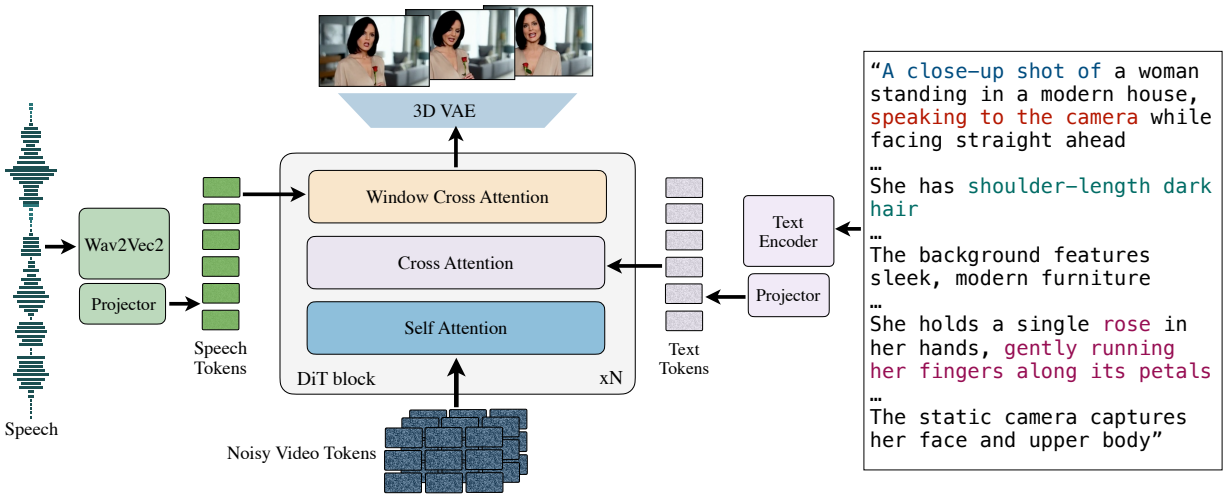


Figure 4.2: **MoCha Architecture.** MoCha is an end-to-end Diffusion Transformer (DiT) that generates video frames from the joint conditioning of **speech** and **text**, without relying on any auxiliary signals. Both speech and text inputs are projected into token representations and aligned with video tokens through cross-attention.

Training Objective. We adopt Flow Matching [91], which enables efficient simulation of continuous-time dynamics, to train our model. Given a latent video representation $x_1 \in \mathbb{R}^{\tau \times h \times w \times c}$ (encoded from the input video), random noise $\epsilon \sim \mathcal{N}(0, I)$, and a continuous time step $t \in [0, 1]$, we construct an intermediate latent x_t by interpolating between ϵ and x_1:

$$x_t = (1 - t)\,\epsilon + t\,x_1. \tag{4.1}$$

The model is trained to predict the velocity, defined as the difference between the data and noise:

$$v_t = \frac{dx_t}{dt} = x_1 - \epsilon. \tag{4.2}$$

The training loss is then:

$$L = \mathbb{E}_{\epsilon \sim \mathcal{N}(0,I),\, x_1,\, y,\, \alpha,\, t \in [0,1]} \left\| f_\theta\big(x_t, y, \alpha, t\big) - (x_1 - \epsilon) \right\|_2^2, \tag{4.3}$$

where x_1 is the latent video, y and α are text and audio conditions, and $f_\theta(\cdot)$ is the DiT model. Unlike prior works [68, 70, 78, 90, 108, 144, 145, 177], MoCha does not rely on auxiliary objectives such as face or body masking. Instead, it learns speech-video correlations purely from data, using a fully tokenized and end-to-end training approach.

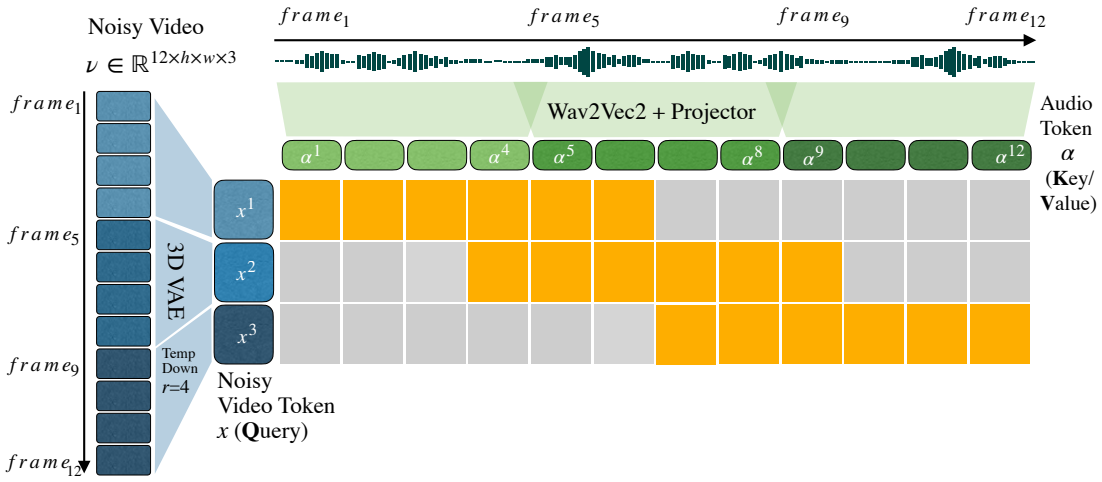


Figure 4.3: **MoCha's Localized Audio Attention.** To address the mismatch between compressed video tokens and high-resolution audio, MoCha employs a localized attention strategy where each video token attends only to a narrow window of nearby audio tokens. This promotes precise lip-sync and improves temporal coherence.

4.2.4.2 Localized Audio Attention

Most existing talking head generation models rely on 2D diffusion architectures (e.g., U-Net) that generate T frames autoregressively, with each frame ν_i conditioned directly on its corresponding audio token $\alpha_i \in \mathbb{R}^c$. This one-to-one mapping between audio and video timesteps naturally ensures tight synchronization. In contrast, diffusion transformer (DiT) architectures depart from this design in two critical ways that complicate temporal alignment:

1. *Temporal Resolution Mismatch:* DiT-based models operate on latent representations produced by a 3D VAE, which temporally compresses videos by a factor of r (commonly $r = 4$ or 8 in recent T2V models [78, 120]). As a result, video tokens span only $\tau = T/r$ steps, while audio remains at the original resolution T, eliminating direct alignment.
2. *Fully Parallel Decoding:* Unlike autoregressive designs, DiT generates all τ latent frames in parallel. Without constraints, naïve cross-attention permits each video token to attend globally to the full audio sequence, which can lead to incorrect associations.

To address these challenges, we propose a *Localized Audio Attention* mechanism that introduces a temporal inductive bias. Inspired by the observation that lip movements depend on short-term phonetic patterns while body gestures and expressions reflect longer-term

textual semantics, we constrain each video token's attention to a limited audio span.

Specifically, for a latent video frame $x^{(i)} \in \mathbb{R}^{h \times w \times c}$ at timestep $i \in \{1, \ldots, \tau\}$, we restrict its cross-attention to audio tokens $\alpha^{(j)}$ in the window:

$$j \in [\max(1, (i-1)r),\ \min(T, ir+1)]\,. \tag{4.4}$$

With one-based, inclusive indices, the uncompressed temporal segment is $[(i-1)r+1, ir]$. Adding one context token on each side gives $r+2 = 6$ tokens away from sequence boundaries; clipping to $[1, T]$ can shorten the window at the boundaries. This simple yet effective constraint encourages alignment between speech and video while preserving local continuity across frames.

4.2.4.3 Multi-character Turn-based Conversation

With the tokenized architecture described in Section 4.2.4.1, the model supports multi-clip video generation in exactly the same way as single-clip generation—**without any additional architectural modifications**. As illustrated in Figure 4.4, unlike video extension methods that rely on autoregressive generation conditioned on previously generated content, MoCha generates all clips in parallel. It leverages self-attention across video tokens to maintain character consistency across clips and preserve coherence in the surrounding environment.

Assuming only one character speaks at a time, we observe that speaker changes in the audio implicitly guide MoCha to transition between clips—without requiring any explicit indicators such as clip tokens [170]. Simultaneously, the text condition y specifies the content of each clip.

Formally, given an audio sequence $\alpha = \{\alpha^1, \alpha^2, \ldots, \alpha^T\}$, where the tokens correspond to two speakers segmented as $\alpha = [\alpha^1, \ldots, \alpha^{rk}] \,\|\, [\alpha^{rk+1}, \ldots, \alpha^T]$, with r being the VAE temporal downsampling ratio and k being an integer index. The model generates the latent video sequence in parallel as: $x = f_\theta(y, \alpha)$, where $x = \{x^1, x^2, \ldots, x^\tau\}, \tau = T/r$. The output sequence x can be segmented into two clips aligned with the respective speaker turns:

$$x = [x^1, \ldots, x^k] \,\|\, [x^{k+1}, \ldots, x^\tau].$$

Multi-clip generation requires prompts that associate each character with the same attributes across clips. This becomes especially difficult when characters interact or reappear in different clips. Naive captioning models typically rely on visual descriptions to refer to characters. As a result, they must repeat detailed appearance descriptions each time a

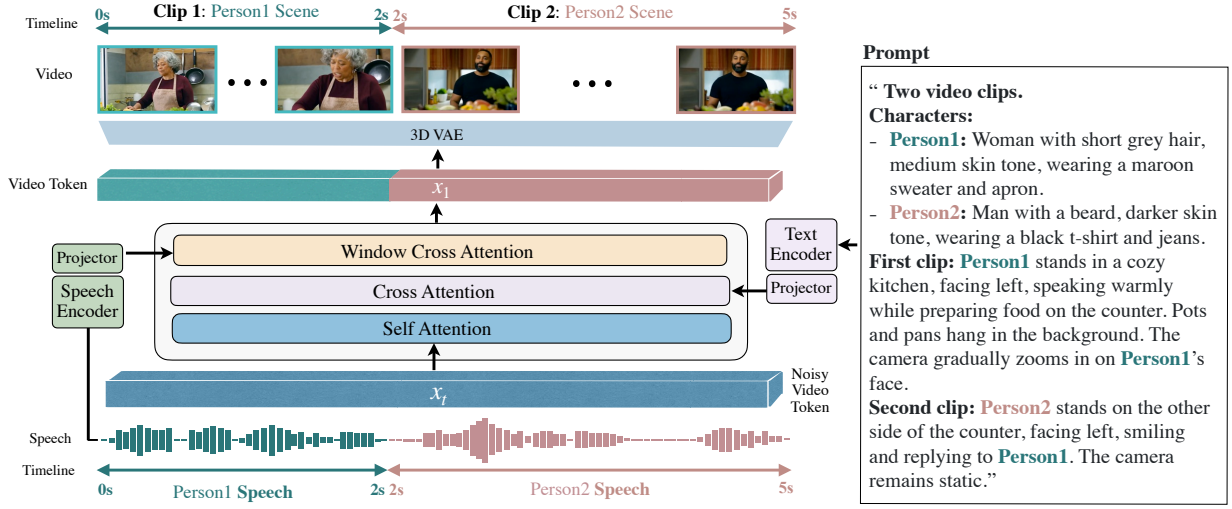


Figure 4.4: **Multi-character Conversation and Character Tagging.** MoCha supports generating multi-character conversations with scene cuts. We design a specialized prompt template: it first specifies the number of clips, then introduces the characters along with their descriptions and tags. Each clip is subsequently described using only the **character tags**, simplifying the prompt while preserving clarity. MoCha leverages self-attention across video tokens to ensure character and environment consistency. The audio signal implicitly guides MoCha when the transition between clips happens.

character is mentioned, leading to long, redundant, and confusing prompts. Figure 4.5 illustrates this repetition. This verbosity not only increases the risk of exceeding token limits (e.g., 256 tokens) but also confuses the model during generation—especially in multi-clip scenarios.

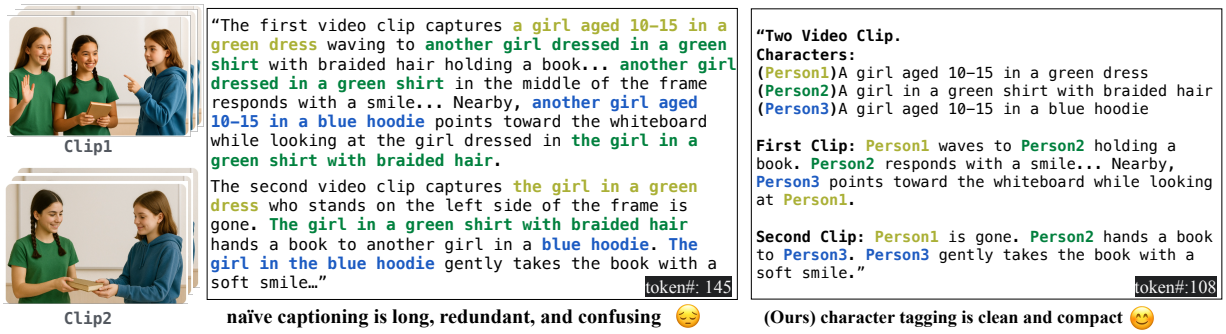


Figure 4.5: Character tagging provides more compact and structured prompts compared to naïve captioning.

We address this by introducing a structured prompt template with fixed keywords and

a character tagging mechanism that promotes clarity, compactness, and consistency:

- **"Two Video Clip. Characters:"** Defines a list of characters, each described by visual attributes and assigned a unique tag [89] (e.g., `Person1`, `Person2`).
- **"First clip", "Second clip"** Each video segment is described using only the defined character tags.

This design significantly reduces redundancy and helps the model reliably associate visual attributes with character actions, even across multiple clips.

4.2.4.4 MoCha Training Strategy

We identify two key challenges in training high-quality talking character models: **(i)** *Data Scarcity and Limited Diversity:* Speech-annotated video datasets are relatively scarce and often lack sufficient visual and semantic diversity, making it difficult to directly train a MoCha model on them. **(ii)** *Varying Speech Influence Across Shot Types:* The impact of speech on video generation differs between spatial scales: it strongly governs lip and facial movements in close-up shots but has a diminishing influence in medium or wide shots involving full-body motion. Training across all shot types simultaneously may lead to slow convergence. To address both issues, we propose a **Curriculum-based Multimodal Training strategy** that integrates modality-aware supervision with progressive visual complexity.

Mixed-Modal Sampling. Our data consists of a mix of multimodal and unimodal data:

- 80% Multimodal (speech+text): The majority enables fine-grained audiovisual grounding.
- 20% Unimodal (text-only): Text-only video samples offer broader visual diversity and varied camera movements, helping the model retain strong generalization capabilities. In this setting, the speech embedding is replaced with a zero vector before the audio projector.

Shot-Type-Based Curriculum. We organize training into multiple stages based on shot complexity:

- Stage 0: We pretrain on large-scale text-only datasets to establish strong visual priors.
- Stages 1–N: We begin with close-up shots, which have high speech-visual correlation, and progressively incorporate more challenging scenarios such as medium/wide shots and multi-character scenes. At each stage, we halve the share of easier examples from the previous stage while maintaining the 80%/20% multimodal-unimodal ratio.

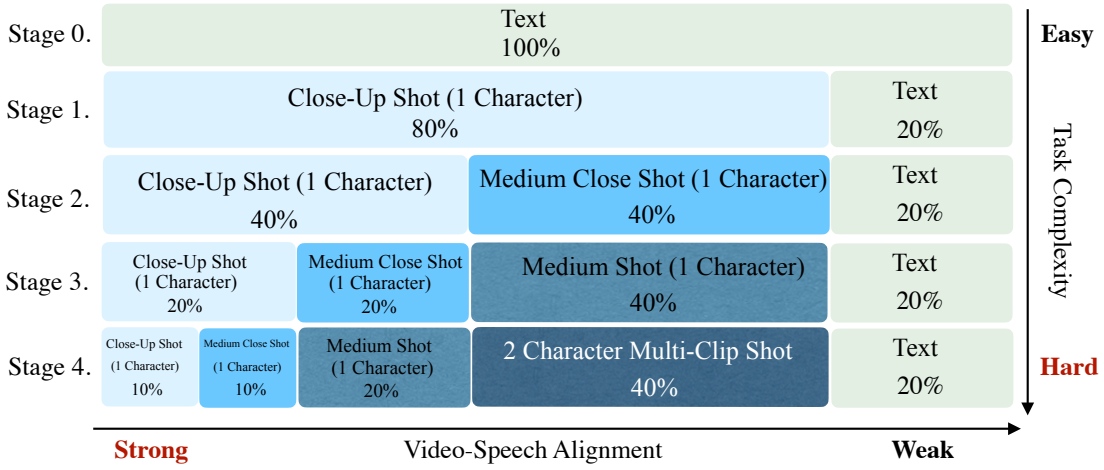


Figure 4.6: **Progressive Curriculum for Multimodal Training in MoCha.** MoCha is first pretrained on text-only video data to acquire general visual generation capability. The model is then progressively exposed to speech-conditioned data across different task difficulties.

This combined strategy allows MoCha to benefit from both abundant text-only data and limited multimodal supervision while progressively mastering harder generation tasks.

4.2.5 Experiments

The evaluation examines speech synchronization, visual performance, and the assignment of dialogue turns to speakers. After the model details in Section 4.2.5.1, we introduce MoCha-Bench and compare MoCha with baseline methods in Section 4.2.5.2. The ablations in Section 4.2.5.3 examine how audio attention, training, and structured prompts support these forms of temporal control.

4.2.5.1 Implementation Details

MoCha builds upon a pretrained 30B-parameter MovieGen backbone [78, 120], which we extend for speech-conditioned video generation in a fully tokenized, end-to-end setting. The model is configured to produce 128 frames at 24 frames per second, resulting in 5.3-second video clips. Training data is standardized to a spatial resolution of approximately 720×720, with flexible aspect ratios to support various shot types. To enable multimodal

Method	Lip-Sync Quality	Facial Expression Naturalness	Action Naturalness	Text Alignment	Visual Quality
Hallo3 [30]	<u>2.45</u>	<u>2.25</u>	<u>2.13</u>	<u>2.35</u>	2.36
SadTalker [203]	1.21	1.14	1.00	N/A	<u>2.95</u>
AniPortrait [160]	1.16	1.12	1.00	N/A	1.45
MoCha (Ours)	**3.85** (+1.40)	**3.82** (+1.57)	**3.82** (+1.69)	**3.85** (+1.50)	**3.72** (+0.77)

Table 4.2: **Human evaluation scores on MoCha-Bench.** Scores range from 1 to 4 across five evaluation axes, where a score of 4 reflects performance that is nearly indistinguishable from real video or cinematic production. Participants rated each method on five aspects: lip-sync quality, facial expression naturalness, action naturalness, text-prompt alignment, and visual quality. MoCha significantly outperforms prior methods across all categories. Green numbers indicate absolute improvements (Δ) over the second-best method (underlined). SadTalker and AniPortrait consistently received a score of 1 for action naturalness, as these methods only perform head movements.

learning, we curate two complementary datasets: (1) a large-scale collection of $\mathcal{O}(100)$ million text-captioned videos and (2) a smaller set of $\mathcal{O}(800)$ thousand speech-annotated samples [30]. Training is distributed across 64 compute nodes.

Architecture and training settings. The 30B DiT uses 48 layers, model dimension 6144, FFN dimension 16384, 48 attention heads, SwiGLU, and RMSNorm. The VAE downsamples space by 8 and time by 4, with 16 latent channels. UL2, ByT5, and MetaCLIP each supply 256 prompt tokens. Speech features concatenate twelve 768-dimensional Wav2Vec2 layers and use a two-layer MLP (hidden dimension 512) to project to 6144. Training uses learning rate 10^{-5} with 2,000 warm-up steps and global batch size 512. The four speech-training stages use 200K, 200K, 100K, and 100K iterations; simpler examples retain half of the preceding mixture at each transition. Encoders and the text projector remain frozen while the generator is fine-tuned. Scene segmentation, speech denoising, motion and lip-sync filtering, and structured captioning produce approximately 800K speech samples.

4.2.5.2 Evaluation

Baselines. We compare our method with state-of-the-art audio-driven talking face generation models, including SadTalker [203], AniPortrait [160], and Hallo3 [30]. These models



Figure 4.7: **Qualitative comparison with SoTA on MoCha-Bench.** MoCha not only produces lip movements that align closely with the input speech—enhancing the clarity and naturalness of articulation—but also generates expressive facial animations and realistic, complex actions that faithfully follow the textual prompt. In contrast, SadTalker and AniPortrait exhibit minimal head motion and limited lip synchronization. Hallo3 mostly follows the lip-syncing but suffers from inaccurate articulation, erratic head movements, and noticeable visual artifacts. Since the baselines operate in an image-to-video (I2V) setting, we provide them with the first frame generated by MoCha as input for comparison. The first frame is cropped and resized as needed to meet the requirements of each baseline.

generate talking faces conditioned on audio and auxiliary signals, such as the first frame, facial keypoints, or pose skeletons. In contrast, MoCha generates talking characters directly from raw speech and text.

Benchmark. We introduce **MoCha-Bench**, a benchmark tailored for the Talking Characters generation task. It contains 200 diverse samples, each comprising a text prompt and corresponding audio clip. The dataset spans various camera shot angles and camera movement—for example, close-up shots emphasize facial expressions and lip-sync, while medium shots highlight hand gestures and body movement. Scenes cover a wide range of human activities and object interactions (e.g., `woman holding a coffee cup`, `professor talking to student`), with characters speaking with various emotions and facing directions. All prompts were manually curated and further enriched using the publicly released LLaMA-3 [37] model to enhance expressiveness and diversity.

Method	Sync-C ↑	Sync-D ↓
SadTalker	4.727	9.239
AniPortrait	1.740	11.383
Hallo3	4.866	8.963
Ours	**6.333** (+1.47)	**8.185** (−0.78)

Table 4.3: **Comparison with SoTA on MoCha-Bench.** We report lip-sync metrics, and MoCha outperforms baselines with superior lip-sync quality. Sync-C is SyncNet's confidence that mouth motion and speech are aligned (higher is better); Sync-D is the SyncNet embedding distance between them at the best-matching offset (lower is better). Both are defined under Quantitative Experiments.

Ablation	Sync-C ↑	Sync-D ↓
Ours	**6.333**	**8.185**
w/o Curriculum Training	5.659	8.435
w/o Localized Audio Attn.	5.103	8.851

Table 4.4: **Ablation on MoCha-Bench.** Removing either Localized Audio Attn. or Curriculum Training degrades lip-sync.

Qualitative Experiments. We present qualitative results of MoCha in Figure 4.1, demonstrating its ability to generate diverse and realistic human motion while maintaining precise speech synchronization, even during complex actions. Contrary to the traditional view that treats audio as a weak conditioning signal requiring auxiliary supervision (e.g., pose annotations), our results show that strong audio-visual alignment can emerge purely from end-to-end training.

Figure 4.7 illustrates a side-by-side evaluation of MoCha against baseline methods on MoCha-Bench. Since MoCha generates video directly from speech and text, while baselines operate in an image-to-video (I2V) setup, we ensure fair comparison by providing each baseline with the first frame generated by MoCha (cropped/resized to focus on the head region as needed). MoCha not only produces lip movements that closely align with the speech—enhancing both articulation and naturalness—but also generates expressive facial movements that accurately follow the textual prompt. In contrast, SadTalker and AniPortrait exhibit minimal head motion and limited lip synchronization. While Hallo3 achieves mostly consistent lip-syncing, it suffers from inaccurate articulation and erratic head movements.

Quantitative Experiments. We measure lip synchronization with Sync-C and Sync-D, two scores computed with SyncNet [25]. SyncNet maps a short clip of the mouth region and the corresponding audio to nearby embeddings when they are in sync, and to distant embeddings when they are not. For a generated video and its input speech, the audio is shifted against the video over a range of frame offsets, and the embedding distance at

each offset is averaged over the clip. Sync-D is the smallest of these averaged distances, so a lower value means closer agreement between mouth motion and speech. Sync-C is the median of these averaged distances minus the smallest one. A higher value means the best offset stands out clearly from the others, that is, SyncNet is more confident that the video and the speech are aligned. Table 4.3 presents a comparison on MoCha-Bench.

Our model achieves the best scores across all lip-sync metrics, demonstrating the effectiveness of MoCha's end-to-end DiT design. These results further confirm that strong audio-visual alignment can emerge purely from audio conditioning without any auxiliary signal.

Human Evaluations. We conduct a comprehensive human evaluation to compare MoCha against baseline methods on the MoCha-Bench dataset. The evaluation is based on five axes tailored for the Talking Characters task (see Section 4.2.3), with scores ranging from 1 to 4. Each model output received 5 independent ratings per example, resulting in over 1000 responses per model. As illustrated in Table 4.2, MoCha significantly outperforms all baselines across all five axes, with average scores approaching 4—indicating performance that is nearly indistinguishable from real video or cinematic production.

4.2.5.3 Ablation Studies

We conduct ablation experiments to assess the individual contributions of MoCha's core components. Table 4.4 presents the impact of each component. **(i)** We disable our localized audio attention mechanism during training, which results in a noticeable drop in Sync-C and increased Sync-D. A separate controlled 4B comparison below tests alternative audio-attention designs. **(ii)** We remove the shot-type curriculum and train MoCha directly on the mixed speech- and text-labeled data with all shot types from the beginning, for the same number of steps. Sync-C drops from 6.333 to 5.659 and Sync-D rises from 8.185 to 8.435, showing that the staged training improves lip-sync quality.

Controlled audio and dialogue ablations. The audio-conditioning ablation uses a 4B DiT pretrained on MovieGen [120], fine-tuned for 200K steps on 400K close-up, single-character speech clips. All settings are shared except the audio-attention module. Video queries attend to audio keys and values. The six variants are unrestricted attention; zero-initialized learned key-position embeddings; sinusoidal key-position embeddings; RoPE on queries and keys; localized attention; and localized attention with RoPE. Localized attention restricts each video frame to the audio interval described in Section 4.2.4.2. Table 4.5 reports Sync-C and Sync-D.

Method	Sync-C ↑	Sync-D ↓
DiT-4B + Localized Audio Attention (**MoCha-4B**)	**5.692**	**8.403**
DiT-4B + Localized Audio Attention + RoPE	5.027	9.038
DiT-4B + Naive Audio Attention + RoPE	4.872	8.893
DiT-4B + Naive Audio Attention + Sinusoidal Embedding	4.747	8.986
DiT-4B + Naive Audio Attention + Learnable Embedding	2.540	10.363
DiT-4B + Naive Audio Attention	2.364	10.385

Table 4.5: **Comparison of Audio Cross-Attention Variants on MoCha-Bench.** We report lip-sync metrics. The proposed Localized Audio Attention achieves the best performance.

The 4B experiment above is separate from the main 30B ablation and holds training conditions fixed across audio modules. For turn-based dialogue, replacing unique person identifiers with repeated descriptions worsens speaker-conditioned generation, as shown in Table 4.6.

Method	Sync-C ↑	Sync-D ↓	Text alignment ↑	Visual quality ↑	Action naturalness ↑
MoCha-30B + tagging	4.951	8.601	**3.81**	**3.64**	**3.69**
w/o tagging	4.465	8.792	2.01	2.15	3.03

Table 4.6: **Ablation of Character Tagging on MoCha-Bench (Turn-Based Dialogue Category).** We report lip-sync metrics (Sync-C ↑, Sync-D ↓) and human evaluation scores. Character tagging improves semantic consistency and reduces scene confusion in multi-clip dialogue scenarios.

4.2.6 Discussion

MoCha extends speech-conditioned generation beyond the face to full-body scenes and multi-character dialogue. In this setting, temporal control involves not only matching mouth motion to speech, but also associating an utterance with the intended character and maintaining the scene across dialogue turns. The ablations connect these requirements to different parts of the method: localized audio attention supports audio-visual alignment, the shot-type curriculum improves lip-sync quality, and character tagging helps

preserve semantic consistency in turn-based dialogue. The human evaluation complements synchronization metrics by assessing the broader quality of the generated character scenes.

These results show how speech can guide video generation at both the level of local timing and the level of dialogue structure. Rather than treating audio as an additional global condition, MoCha organizes how it is associated with visual events and character identities. This provides one form of temporal control within the thesis. Context Forcing, considered next, addresses another: how a generator can use its own history to keep a continuation consistent with what has already occurred.

4.3 Context Forcing: Real-Time Long-Video Generation

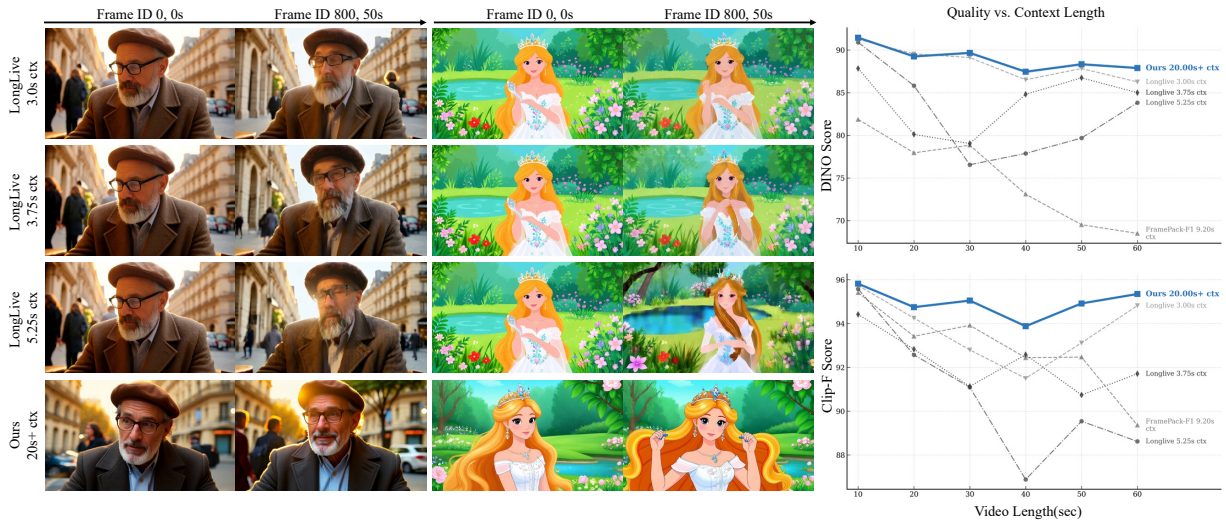


Figure 4.8: **Context Forcing mitigates the forgetting–drifting dilemma.** (1) State-of-the-art models are limited by short context windows (3.0–9.2 s), which leads to poor long-term consistency (*Forgetting*). (2) For streaming long-context tuning baselines (e.g., LongLive), enlarging the context window during inference (3.0 → 5.25 s) causes error accumulation and distribution shift (*Drifting*). In contrast, **Context Forcing** supports **20s+** context while maintaining strong long-term consistency.

4.3.1 Motivation

Real-time long-video generation requires more than repeatedly applying a short-clip generator. It must sustain useful temporal context while producing frames efficiently enough for continuous playback. As introduced in Chapter 2, an autoregressive model conditions on its own previous outputs, so both useful history and earlier errors enter later predictions. A short memory can lose subject and scene information; retaining a longer history can propagate artifacts.

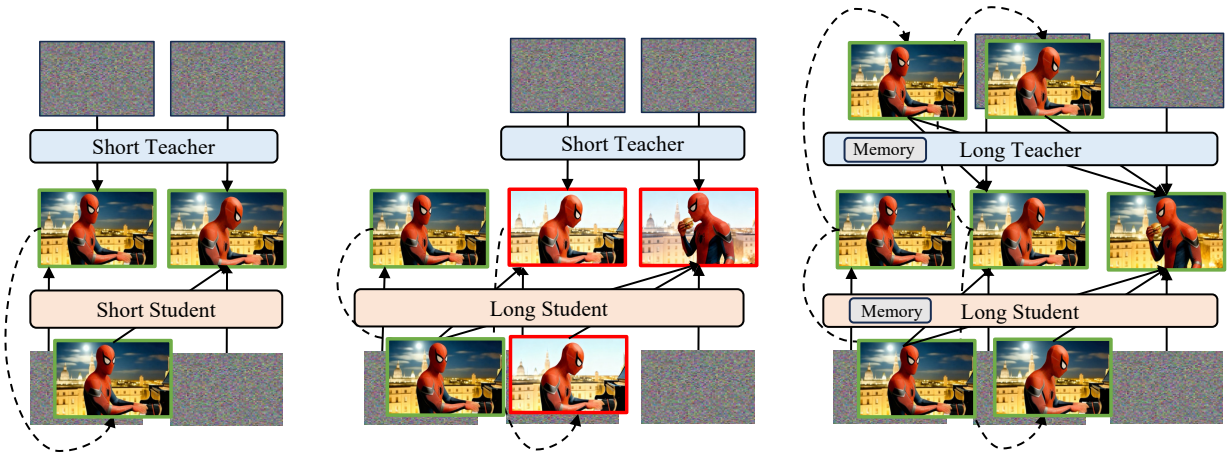
Context Forcing studies a mismatch in distillation: a student may generate a long rollout while receiving supervision from a teacher that sees only a short window. Figure 4.9 contrasts this setting with a teacher that can inspect the preceding context. A short-window teacher cannot directly judge dependencies outside its input, even when those dependencies are available to the student.

The proposed framework combines a long-context teacher, contextual distribution matching, and training on imperfect generated histories. A Slow-Fast Memory mechanism manages the growing context by retaining recent detail and compressing older information. The experiments separately examine continuation, short- and long-video quality, generation throughput, context length, and memory design. This distinguishes the duration a system can generate from the history it can use effectively, and tests whether longer teacher context improves consistency rather than merely increasing storage.

4.3.2 Related Work

Causal generation and rollout training. CausVid combines diffusion with causal attention, while Self Forcing and LongLive expose the student to generated histories to reduce the training–inference mismatch [63, 182, 192]. Context Forcing addresses a related but different constraint: whether the teacher itself sees enough history to supervise long-range dependencies. Figure 4.9 illustrates this distinction.

Efficiency and memory. Few-step distillation reduces sampling cost [136, 190, 191], while windowed attention, positional modifications, and memory compression address the cost of long histories [77, 199, 200, 208]. Sparse context routing and adaptive memory provide further approaches to long-video generation [14, 67]. ConsistI2V attends explicitly to the first frame [128]; WorldMem and RELIC study persistent memory or recurrent states [57, 172], while WorldPlay targets interactive world modeling with long-term geometric consistency [137]. Slow-Fast Memory complements these directions by managing



(a) Self-Forcing Training
short teacher → short student

(b) Memoryless Long Tuning
short teacher → long student

(c) **Context Long Tuning(ours)**
long teacher → long student

Figure 4.9: **Training paradigms for AR video diffusion models.** (a) Self Forcing: A student matches a teacher capable of generating only 5s video using a 5s self-rollout. (b) LongLive [182]: The student performs long rollouts supervised by a memoryless 5s teacher on random chunks. The teacher's inability to see beyond its 5s window creates a student-teacher mismatch. (c) **Context Forcing (Ours)**: The student is supervised by a long-context teacher aware of the full generation history, resolving the mismatch in (b).

context in a student trained with a context-aware teacher. The experiments test the roles of teacher context and memory design separately.

4.3.3 Method

We operate within the causal autoregressive framework, where the generation of a long video $X_{1:N}$ is decomposed into a sequence of conditional steps over frames or short chunks X_t. State-of-the-art methods, such as CausVid [192] and Self Forcing [63], enforce strict temporal causality via block-wise attention, modeling the distribution as $\prod_t p(X_t \mid X_{<t})$. These approaches typically employ Distribution Matching Distillation (DMD) [191] to distill a high-quality bidirectional teacher into a causal student. Building on these foundations, we introduce **Context Forcing**.

Our goal is to train a causal video diffusion model, parameterized by θ, whose induced distribution over *long videos* $p_\theta(X_{1:N})$ matches the real data distribution $p_{\text{data}}(X_{1:N})$. Here,

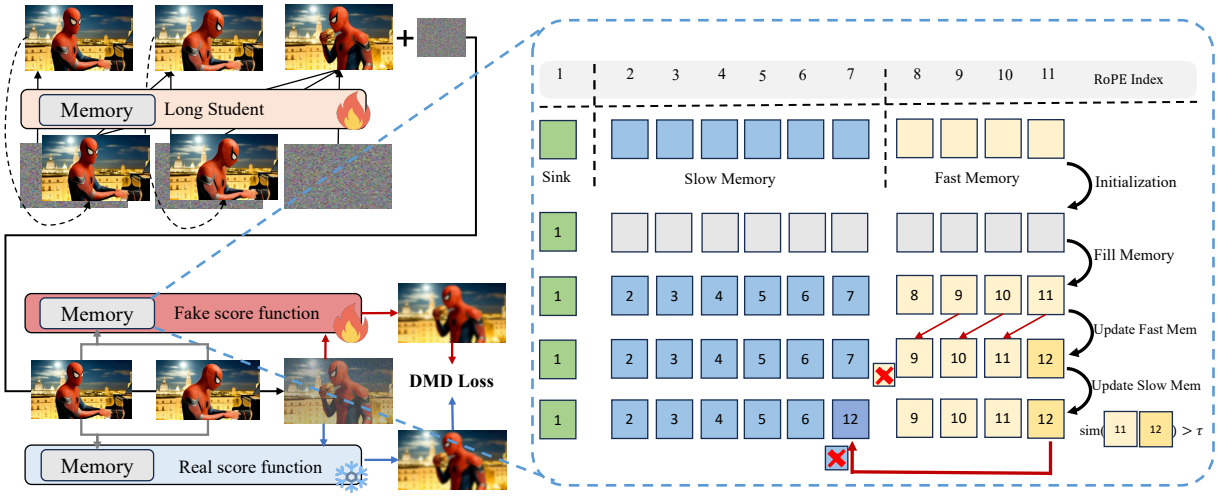


Figure 4.10: **Context Forcing and Context Management System.** We use KV Cache as the context memory, and we organize it into three parts: sink, slow memory and fast memory. During contextual DMD training, the long teacher provides supervision to the long student by utilizing the same context memory mechanism.

N represents a duration spanning tens or hundreds of seconds. The objective is to minimize the global long-horizon KL divergence:

$$\mathcal{L}_{\text{global}} = \min_{\theta} \text{KL}\Big(p_\theta(X_{1:N}) \parallel p_{\text{data}}(X_{1:N})\Big). \tag{4.5}$$

Directly optimizing Eq. (4.5) ensures long-term coherence but is computationally intractable for large N. By applying the chain rule of KL divergence, we decompose the global objective into two components:

$$\begin{aligned}\mathcal{L}_{\text{global}} = & \underbrace{\text{KL}\Big(p_\theta(X_{1:k}) \parallel p_{\text{data}}(X_{1:k})\Big)}_{\mathcal{L}_{\text{local}}\text{: Local Dynamics}} \\ & + \underbrace{\mathbb{E}_{X_{1:k}\sim p_\theta}\Big[\text{KL}\Big(p_\theta(X_{k+1:N}|X_{1:k}) \parallel p_{\text{data}}(X_{k+1:N}|X_{1:k})\Big)\Big]}_{\mathcal{L}_{\text{context}}\text{: Global Continuation Dynamics}}\end{aligned} \tag{4.6}$$

This decomposition motivates our two-stage curriculum:

- **Stage 1 (Optimizing $\mathcal{L}_{\text{local}}$):** We match the distribution of short windows ($X_{1:k}$) to the real data distribution to learn local dynamics.

- **Stage 2 (Optimizing $\mathcal{L}_{\text{context}}$):** We match the model's continuation predictions ($X_{k+1:N}$) with the temporal evolution of real data to learn long-term dependencies.

4.3.3.1 Stage 1: Local Distribution Matching

The first stage warms up the causal student by minimizing $\mathcal{L}_{\text{local}}$. Given a teacher distribution $p_T(X_{1:k})$ (approximately the real data), we optimize:

$$\mathcal{L}_{\text{local}} = \text{KL}\Big(p_\theta(X_{1:k}) \,\|\, p_T(X_{1:k})\Big), \tag{4.7}$$

where k corresponds to a 1–5 second window. We estimate the distribution matching gradient following DMD [191]. Let $x = G_\theta(z)$ for noise z, and let x_t be the diffused version of x at timestep t. The gradient is given by:

$$\nabla_\theta \mathcal{L}_{\text{local}} \approx \mathbb{E}_{z,t,x_t}\Big[w_t \alpha_t \left(s_\theta(x_t, t) - s_T(x_t, t)\right) \frac{\partial G_\theta(z)}{\partial \theta}\Big], \tag{4.8}$$

where s_θ and s_T are the student and teacher scores, respectively, and w_t is a weighting function. This stage ensures $p_\theta(X_{1:k}) \approx p_{\text{data}}(X_{1:k})$, providing high-quality contexts for the subsequent stage.

4.3.3.2 Stage 2: Contextual Distribution Matching

Stage 2 targets $\mathcal{L}_{\text{context}}$, the second term of Eq. (4.6). This term requires minimizing the divergence between the student's continuation $p_\theta(\cdot|X_{1:k})$ and the true data continuation $p_{\text{data}}(\cdot|X_{1:k})$.

However, p_{data} is not directly accessible for arbitrary contexts generated by the student. To solve this, we employ a pretrained **Context Teacher** T, which provides a reliable proxy distribution $p_T(X_{k+1:N} \mid X_{1:k})$. We rely on two key assumptions to justify using the teacher as a target:

Assumption 1 (Teacher reliability near student contexts). *Whenever the student context $X_{1:k} \sim p_\theta(X_{1:k})$ remains close to the real data manifold, the teacher's continuation $p_T(X_{k+1:N} \mid X_{1:k})$ is accurate.* This holds whenever the teacher is well-trained on real video prefixes.

Assumption 2 (Approximate real prefixes). *Stage 1 successfully aligns $p_\theta(X_{1:k})$ with $p_{data}(X_{1:k})$.* This ensures that student rollouts remain within the teacher's reliable region during Stage 2 training.

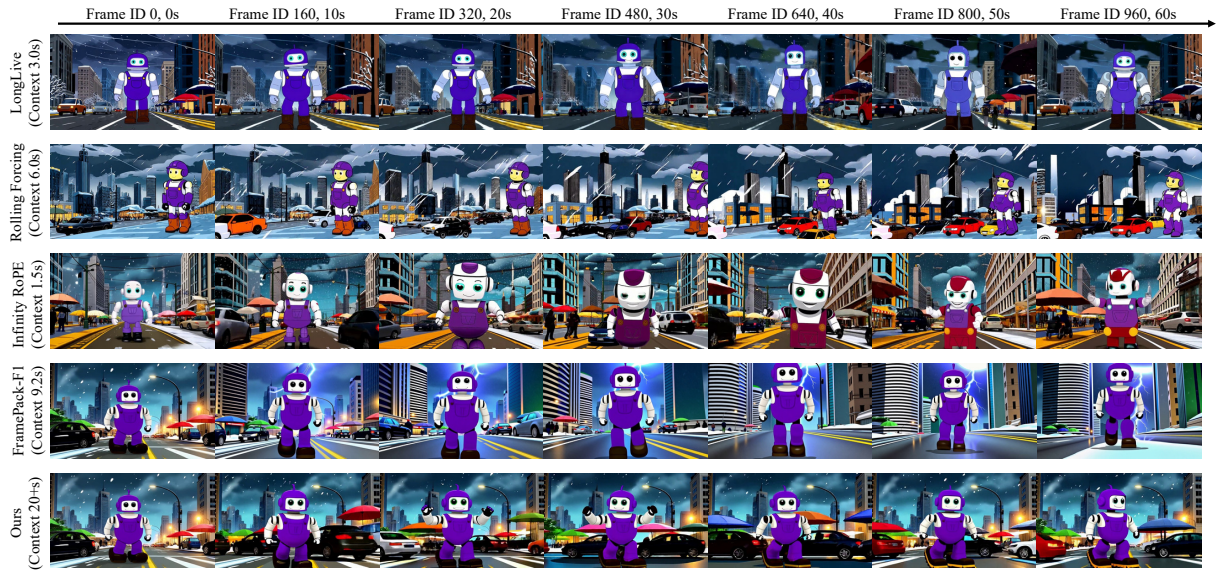


Figure 4.11: **Comparison on 1-min Video Generation.** Our method keeps both the background and subject consistent across the 1-min video, while other baselines show different levels of drifting or identity shift.

Under these assumptions, we approximate $p_{\text{data}} \approx p_T$ in the second term of Eq. (4.6), yielding the **Contextual DMD (CDMD)** objective:

$$\mathcal{L}_{\text{CDMD}} = \mathbb{E}_{X_{1:k} \sim p_\theta(X_{1:k})} \Big[\text{KL}\Big(p_\theta(X_{k+1:N} \mid X_{1:k}) \,\|\, p_T(X_{k+1:N} \mid X_{1:k}) \Big) \Big] \tag{4.9}$$

Crucially, the expectation is over $X_{1:k} \sim p_\theta$, ensuring the student is trained on its *own* rollouts, thereby mitigating exposure bias.

Score-based CDMD Gradient. We estimate the gradient of Eq. (4.9) using a conditional variant of the DMD gradient. Let $x_{\text{cont}} = G_\theta(z_{\text{cont}} \mid X_{1:k})$ be the generated continuation, and $x_{t,\text{cont}}$ be its diffused version. Running both fake score and real score models on the *same* student-generated context produces scores $s_\theta(\cdot \mid X_{1:k})$ and $s_T(\cdot \mid X_{1:k})$. The gradient is:

$$\nabla_\theta \mathcal{L}_{\text{CDMD}} \approx \mathbb{E}_{\substack{X_{1:k} \sim p_\theta \\ z_{\text{cont}}, t}} \Big[w_t \alpha_t \Big(s_\theta(x_{t,\text{cont}}, t \mid X_{1:k}) - s_T(x_{t,\text{cont}}, t \mid X_{1:k}) \Big) \frac{\partial G_\theta(z_{\text{cont}} \mid X_{1:k})}{\partial \theta} \Big]. \tag{4.10}$$

By descending Eq. (4.10), we align the student's long-term autoregressive dynamics with the teacher's robust priors.

Long Self-Rollout Curriculum. Minimizing $\mathcal{L}_{\text{context}}$ requires the context horizon k to approach the full sequence length N. However, sampling $X_{1:k} \sim p_\theta$ for large k early in training causes severe distribution shift due to accumulated drift. To mitigate this, we employ a dynamic horizon schedule $N_{\max}^{(t)}$ that grows linearly with training step t. At each iteration, the rollout length is sampled as $k \sim \mathcal{U}(k_{\min}, N_{\max}^{(t)})$. This curriculum initializes training in the stable Stage 1 regime ($k \approx k_{\min}$) and progressively exposes the model to long-range dependencies.

Clean Context Policy. Self Forcing [63] typically generates rollouts using a random timestep selection strategy to ensure supervision across all diffusion steps. We retain this random exit policy for the *target* frames $X_{k+1:N}$ to preserve gradient coverage, but enforce that the *context* frames $X_{1:k}$ are fully denoised. We apply a complete few-step denoising process to the context. This decoupling ensures the context remains informative and aligned with the teacher's training distribution but also maintains supervision for every diffusion step.

4.3.3.3 Context Management System

Teacher and student share an autoregressive architecture and a fixed-capacity KV cache representing $X_{1:k}$. The cache separates stable initial context, selected historical information, and recent local detail.

Cache Partitioning. The total cache is defined as the union of disjoint sets:

$$\mathcal{M} = \mathcal{S} \cup \mathcal{C}_{\text{slow}} \cup \mathcal{L}_{\text{fast}}.$$

- *Attention Sink ($\mathcal{S}$)*: Retains initial N_s tokens to stabilize attention, following StreamingLLM [171], as in recent streaming video generators [132, 182].
- *Slow Memory ($\mathcal{C}_{slow}$)*: A long-term buffer of up to N_c tokens, storing high-entropy keyframes and updating only with significant new information.
- *Fast Memory ($\mathcal{L}_{fast}$)*: A rolling FIFO queue of size N_l, capturing immediate local context with short-term persistence.

Surprisal-Based Consolidation. After enqueuing x_t in Fast Memory, we compare its key k_t with the preceding key k_{t-1}. Low similarity indicates potentially informative visual changes rather than redundant history, motivating promotion to Slow Memory:

$$\pi(x_t) = \begin{cases} \text{Consolidate} & \text{if } \text{sim}(k_t, k_{t-1}) < \tau, \\ \text{Discard} & \text{otherwise,} \end{cases} \tag{4.11}$$

where τ is a similarity threshold. If $|\mathcal{C}_{\text{slow}}| > N_c$, the oldest entry is evicted to keep memory bounded.

Bounded Positional Encoding. Unlike standard autoregressive video models [32, 63], where positional indices grow unbounded ($p_t = t \to \infty$), leading to distribution shifts on long sequences, we adopt *Bounded Positional Indexing*. All tokens' temporal RoPE positions are constrained to a fixed range $\Phi = [0, N_s + N_c + N_l - 1]$ regardless of generation step t:

$$\phi(x) = \begin{cases} i \in [0, N_s - 1] & \text{if } x \in \mathcal{S}, \\ j \in [N_s, N_s + N_c - 1] & \text{if } x \in \mathcal{C}_{\text{slow}}, \\ k \in [N_s + N_c, N_s + N_c + N_l - 1] & \text{if } x \in \mathcal{L}_{\text{fast}}. \end{cases} \tag{4.12}$$

This creates a static attention window where recent history (Fast) slides through high indices, while salient history (Slow) is compressed into lower indices, stabilizing attention over long sequences.

4.3.3.4 Robust Context Teacher Training

Standard training conditions the model on ground-truth context, but inference relies on self-generated history, creating a distribution shift known as exposure bias. To ensure our Context Teacher provides robust guidance even when the student drifts, we adopt Error-Recycling Fine-Tuning (ERFT) [87].

Rather than training on clean history $X_{1:k}$, we inject realistic accumulated errors into the teacher's context. We construct a perturbed context $\tilde{X}_{1:k} = X_{1:k} + \mathbb{I} \cdot e_{\text{drift}}$, where e_{drift} is sampled from a bank of past model residuals and $\mathbb{I}$ is a Bernoulli indicator. The teacher is optimized to recover the correct velocity v_{target} from $\tilde{X}_{1:k}$. This active correction capability ensures $p_T(\cdot \mid X_{1:k})$ remains a reliable proxy for p_{data} even when the student's context $X_{1:k}$ degrades.

For ground-truth videos longer than 10 seconds, the teacher is prefilled with 21 latent frames (3 sink, 12 slow, 6 fast), matching the student's memory; the next 21 frames are

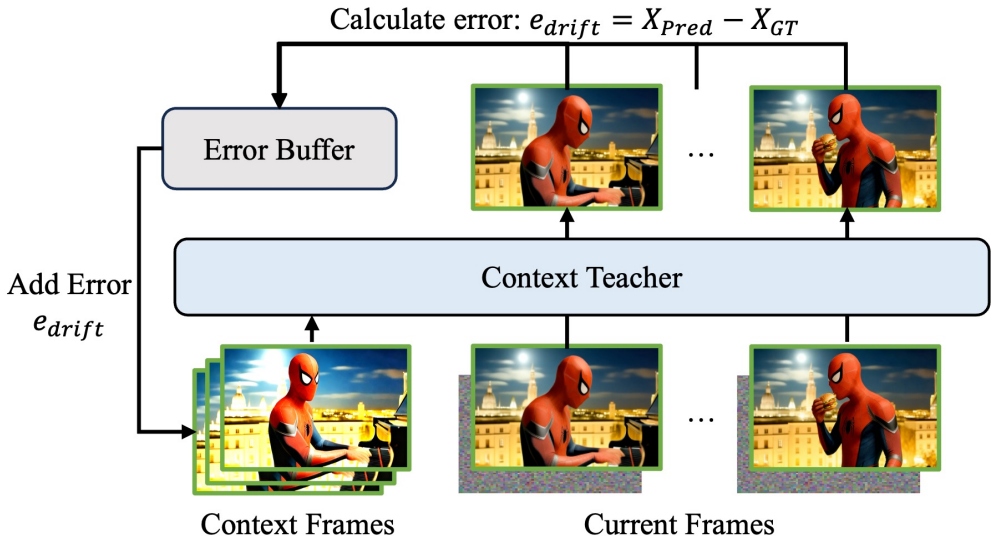


Figure 4.12: **Robust context teacher training with error recycling.** Past latent prediction errors are stored in an error buffer and probabilistically injected into the context and noisy latents. The input sequence uses the same context management system as the student.

prediction targets. Concretely, at each timestep, we obtain the predicted latent via one-step ODE integration. ERFT computes the teacher's latent prediction residuals online, stores them in a timestep-indexed shared buffer, then uniformly resamples and probabilistically injects them into context (Figure 4.12). These are model errors, not hand-crafted perturbations.

4.3.4 Experiments

The evaluation separates long-horizon consistency from generation throughput: effective use of history must preserve earlier content without making continued generation impractical. Comparisons and human judgments assess the resulting videos, while ablations examine the contributions of contextual distillation, teacher robustness, memory selection, and positional indexing.

Implementation Details. We implement the robust context teacher using Wan2.1-T2V-1.3B [142] as the base model. To construct the training dataset, we filter the Sekai [88] and Ultravideo [179] collections to retain high-quality videos exceeding 10 seconds in duration,

yielding a total of 40k clips. The robust context teacher is trained for 8k steps with a batch size of 8. During training, frames are sampled uniformly from the 5–20 second interval of the video data to serve as context.

The student model also utilizes the Wan2.1-T2V-1.3B model. In Stage 1, we employ 81-frame video clips from the VidProM [157] dataset and train for 600 iterations with a batch size of 64. In Stage 2, which focuses on context distillation, we extend the rollout horizon to video lengths of 10–30 seconds to address short-term memory limitations. This phase is similarly trained on the VidProM dataset for 500 iterations using the same batch size. For both teacher and student models, we set the KV cache size to 21 latent frames, and set $N_s = 3, N_c = 12, N_l = 6, \tau = 0.95$. We implement Surprisal-Based Consolidation at 2-chunk intervals. Upon chunk consolidation, we retain only the first latent, effectively extending the context beyond 20s.

Baselines. We evaluate our method against three distinct categories of baselines. The first category comprises bidirectional diffusion models, specifically LTX-Video [52] and Wan2.1 [142]. The second category includes autoregressive models such as SkyReels-V2 [17], MAGI-1 [143], CausVid [192], NOVA [35], Pyramid Flow [73], and Self Forcing [63]. The third category consists of recent methods targeting long video generation within autoregressive frameworks. These include LongLive [182] with a context length of 3 seconds, Self Forcing++ [32], Rolling Forcing [95] with a context length of 6 seconds, and Infinity-RoPE [188] with a context length of 1.5 seconds. Finally, we include a long context baseline FramePack [199] with a context length of 9.2 seconds.

Evaluation. We report performance on VBench [64] following [63, 182]. Beyond standard benchmarks, we assess fine-grained consistency using DINOv2 [114] (structural identity), CLIP-F [126] (semantic context), and CLIP-T (prompt alignment). To improve robustness against temporal artifacts, we implement window-based sampling: for any timestamp t, we compute the average cosine similarity between the first frame (V_0) and frames within $[t - 0.5s, t + 0.5s]$. We average results over five random seeds per prompt to ensure statistical reliability. This approach effectively measures long-term subject and background consistency.

4.3.4.1 Video Continuation with Robust Context Teacher

We test the teacher on Stage 1 student-generated contexts using 100 randomly sampled MovieGenBench prompts [120]. Table 4.7 reports subject/background consistency and DINOv2, CLIP-F, and CLIP-T scores for the complete 10-second sequences. These results

Table 4.7: Single-prompt 60-second long video consistency evaluation. Higher scores indicate greater consistency.

DINOv2 similarity

Model	Context	10s	20s	30s	40s	50s	60s
FramePack-F1	9.2s	81.86	77.95	78.84	73.10	69.52	68.50
LongLive	3.0s	<u>91.25</u>	**89.55**	<u>89.12</u>	<u>86.51</u>	<u>87.83</u>	<u>86.26</u>
Infinity-RoPE	1.5s	91.18	88.17	85.37	79.80	81.10	83.72
Ours, teacher	20+s	87.61	-	-	-	-	-
Ours, student	20+s	**91.45**	<u>89.25</u>	**89.66**	**87.45**	**88.33**	**87.89**

CLIP-F similarity

Model	Context	10s	20s	30s	40s	50s	60s
FramePack-F1	9.2s	95.41	93.42	<u>93.92</u>	<u>92.44</u>	92.47	89.36
LongLive	3.0s	<u>95.74</u>	<u>94.25</u>	92.80	91.50	<u>93.12</u>	<u>94.82</u>
Infinity-RoPE	1.5s	94.09	91.71	91.13	89.71	86.11	88.88
Ours, teacher	20+s	95.52	-	-	-	-	-
Ours, student	20+s	**95.82**	**94.75**	**95.05**	**93.88**	**94.92**	**95.35**

CLIP-T similarity

Model	Context	10s	20s	30s	40s	50s	60s
FramePack-F1	9.2s	36.36	34.67	33.75	33.84	34.77	32.30
LongLive	3.0s	<u>36.95</u>	<u>35.80</u>	<u>36.17</u>	<u>36.58</u>	<u>35.92</u>	<u>37.13</u>
Infinity-RoPE	1.5s	35.26	35.03	35.88	32.56	32.29	32.28
Ours, teacher	20+s	35.93	-	-	-	-	-
Ours, student	20+s	**37.12**	**36.25**	**36.72**	**37.15**	**37.08**	**37.66**

Background and subject consistency

Model	Context	Background	Subject
FramePack-F1	9.2s	91.61	89.15
LongLive	3.0s	94.92	93.05
Infinity-RoPE	1.5s	92.42	90.11
Ours, teacher	20+s	<u>95.24</u>	<u>94.87</u>
Ours, student	20+s	**95.95**	**95.68**

Table 4.8: Comparison of video generation models across architecture families. Results are shown for 5-second and 60-second generation. Higher scores and throughput are better.

5-second generation

Model	Params	FPS	Total	Quality	Semantic	Background consistency	Subject consistency
Bidirectional models							
LTX-Video	1.9B	8.98	80.00	82.30	70.79	95.30	95.01
Wan2.1	1.3B	0.78	<u>84.26</u>	<u>85.30</u>	80.09	<u>96.96</u>	95.99
Autoregressive models							
SkyReels-V2	1.3B	0.49	82.67	84.70	74.53	96.83	96.07
MAGI-1	4.5B	0.19	79.18	82.04	67.74	96.83	95.83
CausVid	1.3B	17.0	81.20	84.05	69.80	95.12	95.96
NOVA	0.6B	0.88	80.12	80.39	79.05	95.16	93.38
Pyramid Flow	2B	6.7	81.72	84.74	69.62	96.09	96.08
Self Forcing, chunk-wise	1.3B	17.0	84.31	85.07	**81.28**	95.98	<u>96.29</u>
Long autoregressive models							
LongLive	1.3B	20.7	**84.87**	**86.97**	76.47	96.55	95.82
Self Forcing++	1.3B	17.0	83.11	83.79	<u>80.37</u>	-	-
Rolling Forcing	1.3B	15.8	81.22	84.08	69.78	96.11	96.02
Infinity-RoPE	1.3B	17.0	81.79	83.27	75.87	96.34	95.14
Ours, student model	1.3B	17.0	83.44	84.98	77.29	**97.38**	**96.84**

support the teacher's ability to continue student-generated contexts under the evaluated conditions, as required by the Stage 2 assumptions.

4.3.4.2 Text-to-Short Video Generation

On 5-second VBench generation with official extended prompts, Context Forcing retains competitive short-video performance (Table 4.8).

4.3.4.3 Text-to-Long Video Generation

Qualitative Results. We evaluate our proposed method against baseline models on 60-second video generation, with qualitative results illustrated in Figure 4.11. By leveraging a slow-fast memory architecture with a KV cache size of 21 and a context span exceeding

Table 4.8: Video generation quality and throughput across architecture families (continued): 60-second generation.

60-second generation							
Model	Params	FPS	Total	Quality	Semantic	Background consistency	Subject consistency
Bidirectional models							
LTX-Video	1.9B	8.98	-	-	-	-	-
Wan2.1	1.3B	0.78	-	-	-	-	-
Autoregressive models							
SkyReels-V2	1.3B	0.49	70.47	75.30	51.15	89.95	84.99
MAGI-1	4.5B	0.19	69.87	76.12	44.87	87.76	79.46
CausVid	1.3B	17.0	71.04	76.80	48.01	89.85	86.75
NOVA	0.6B	0.88	65.25	70.25	45.24	88.06	77.50
Pyramid Flow	2B	6.7	-	-	-	-	-
Self Forcing, chunk-wise	1.3B	17.0	71.86	77.20	50.51	87.84	83.60
Long autoregressive models							
LongLive	1.3B	20.7	**83.64**	**84.53**	74.97	94.62	93.88
Self Forcing++	1.3B	17.0	-	-	-	-	-
Rolling Forcing	1.3B	15.8	79.31	81.87	67.69	94.12	93.10
Infinity-RoPE	1.3B	17.0	79.99	80.81	74.30	94.21	93.05
Ours, student model	1.3B	17.0	82.45	83.55	**76.10**	**95.34**	**94.88**

20s, our method achieves superior consistency and effectively mitigates content drifting compared to the baselines.

Quantitative Results. We evaluate 60-second video generation performance on VBench with results summarized in Table 4.8, using its official extended prompts. Additionally, we report DINOv2, CLIP-F, and CLIP-T scores in Table 4.7, using 100 text prompts randomly sampled from MovieGenBench [120], following the same experimental protocol as in Section 4.3.4.1. Both tables demonstrate that our method achieves high consistency, particularly during extended video sequences. Notably, while LongLive also achieves competitive scores, qualitative inspection reveals that it frequently exhibits abrupt scene resets and cyclic motion patterns, shown in Figure 4.13.



Figure 4.13: **Visual artifacts in LongLive.** After 524 frames, the video resets to its initial frame, disrupting temporal continuity.

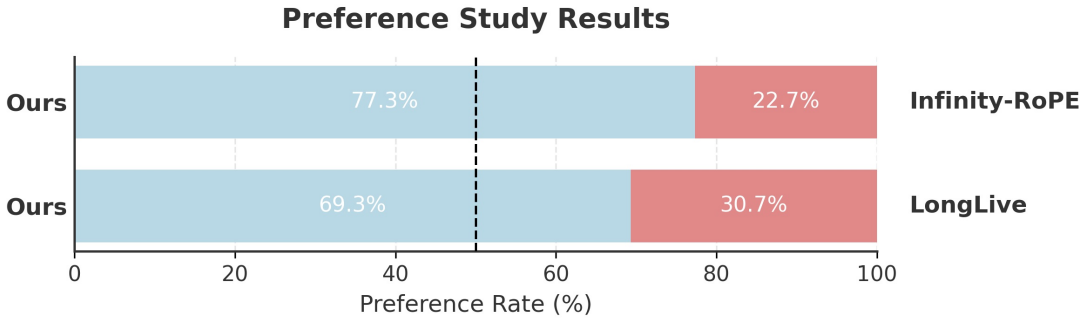


Figure 4.14: **Human preference for Context Forcing against long-video baselines.** In the blind A/B study, Context Forcing is preferred in 77.3% of comparisons with Infinity-RoPE and 69.3% with LongLive.

4.3.4.4 User Study

We conducted a blind A/B study with 15 participants using long videos generated from 20 randomly selected text prompts. Participants compared Context Forcing against LongLive and Infinity-RoPE, with paired videos presented in randomized order. They selected a single preferred result after considering overall visual quality, long-term temporal consistency, and semantic alignment. Context Forcing received 77.3% preference against Infinity-RoPE and 69.3% against LongLive (Figure 4.14).

4.3.4.5 Ablation Studies

Slow Memory Sampling Strategy Similarity-based selection outperforms uniform sampling at intervals of 1 and 2 chunks in Table 4.9, supporting the value of selecting informative history rather than retaining it at fixed intervals.

Context DMD Distillation Applying the context management system directly after Stage 1, without contextual DMD, reduces semantic and temporal consistency (Table 4.9).

Table 4.9: Ablation study on Slow Memory Sampling Strategy, Context DMD, and Bounded Positional Encoding (evaluated on 60s).

Model	Total Score ↑	Quality Score ↑	Semantic Score ↑	Background Consistency ↑	Subject Consistency ↑	Dynamic Degree ↑
Slow Memory Sampling Strategy						
Uniform sample, interval 1	80.82	82.20	<u>75.32</u>	92.45	92.10	52.15
Uniform sample, interval 2	<u>81.11</u>	<u>82.61</u>	75.12	93.12	92.85	<u>55.30</u>
Contextual Distillation						
w/o. Contextual Distillation	80.36	82.28	72.70	<u>93.55</u>	<u>93.20</u>	48.12
Bounded Positional Encoding						
w/o. Bounded Positional Encoding	73.52	75.44	65.82	84.68	79.24	27.45
Ours	**82.45**	**83.55**	**76.10**	**95.34**	**94.88**	**58.26**



Figure 4.15: **Ablation on Error-Recycling Fine-Tuning (ERFT)**. With ERFT, the context teacher is more robust to accumulated errors.

Enlarging accessible memory alone therefore does not replace training with context-aware supervision.

Error-Recycling Fine-Tuning (ERFT). Given 5s dataset videos as context, the teacher produces fewer artifacts during 30s autoregressive generation with ERFT (Figure 4.15), improving the continuation distribution used for distillation.

Bounded Positional Encoding. Removing bounded positions at inference reduces background and subject consistency (Table 4.9), demonstrating the importance of stable positional indexing during long rollouts.

4.3.5 Discussion

Context Forcing shows that maintaining a longer video history requires more than enlarging the context available at inference. The student must also learn from supervision that accounts for that history. Applying the memory system without contextual distillation reduces consistency, while error-recycling fine-tuning improves the teacher's continuations from imperfect generated contexts. Slow-Fast Memory and bounded positional indexing then make this context usable during extended rollouts. Together, these results explain how the method combines an effective context of more than 20 seconds with a student throughput of 17.0 frames per second in the reported setting (Table 4.8).

The observed drift and loss of fine detail reveal a trade-off in retaining history through a compressed representation: bounded memory supports continued generation while selecting which past information constrains later frames. The consistency results and memory ablations therefore assess how effectively the method uses history, not merely how long it can generate. Alongside MoCha's alignment of speech and dialogue with visual events, Context Forcing establishes historical consistency as a distinct temporal-control requirement. Its contribution is to align training supervision and memory design so that earlier generated content can guide subsequent video continuation.

4.4 Summary

This chapter addressed limited temporal control by studying how speech and historical context guide video generation. MoCha incorporated speech conditioning into a video diffusion transformer, aligning visible performance with audio while using character tagging to associate dialogue turns with speakers. Its synchronization, human-evaluation, and

ablation results demonstrated the value of these mechanisms for speech-conditioned video generation and movie-grade storytelling. Context Forcing addressed a different temporal requirement: retaining earlier context as generation continues. A long-context teacher supplied supervision for longer-range dependencies, while Slow-Fast Memory combined recent detail with longer-term context under a bounded memory budget. The resulting system supported real-time long-video generation while reducing forgetting and drift in the evaluated continuations. Together, the studies show that temporal control involves more than adding another input modality or extending output duration. The model must align conditions with the moments they govern and retain information that remains relevant to later generation. Chapter 5 next considers how feedback can identify failures to meet a request and guide improvements in image generation and editing.

Chapter 5

Improving Instruction Adherence through Reward Feedback

This chapter is based on the following works. Author contributions are described in the Statement of Contributions.

- Haozhe Wang, Cong Wei, Weiming Ren, Jiaming Liu, Fangzhen Lin, and Wenhu Chen. *RationalRewards: Reasoning Rewards Scale Visual Generation Both Training and Test Time*. In Proceedings of the *Conference on Language Modeling* (COLM), 2026. (Section 5.2)
- Yuxuan Zhang, Cong Wei, Penghui Du, Bo Li, Huaisong Zhang, Songcheng Cai, Yubo Wang, Dongfu Jiang, Yuyu Zhang, Changqian Yu, Ping Nie, Wenhu Chen, and Kelsey R. Allen. *RewardHarness: Learning Human Preferences for Image Editing with Only 100 Demonstrations*. In Proceedings of the *Conference on Language Modeling* (COLM), 2026. (Section 5.3)

5.1 Overview

The preceding chapters address ways to control generation, but providing instructions and reference inputs does not ensure that a generator follows them. An edit may omit a requested change, alter an unrelated region, or trade source preservation for visual appeal. The third challenge is therefore to obtain feedback that identifies such deviations and helps

improve the output. An evaluator must account for the instruction, source content, and generated result together. This chapter evaluates its feedback through both preference prediction and downstream optimization.

This chapter develops two approaches to learning that feedback, with downstream validation in image generation and editing. RationalRewards (Section 5.2) learns structured critiques from preference data. Preference-Anchored Rationalization provides supervision for the evaluator, whose scores and critiques can then guide generator training and inference-time refinement. RewardHarness (Section 5.3) asks whether an evaluator can instead adapt through an external context while its model parameters remain fixed. It learns reusable evaluation guidelines and tools from a small set of preference demonstrations and tests their usefulness as feedback for image editing.

The studies differ in what is learned: evaluator parameters in RationalRewards, and evaluation procedures and tools in RewardHarness. Both test whether better preference agreement translates into improved image generation and editing, rather than merely more plausible scores.

5.2 RationalRewards: Reasoning-Based Rewards for Visual Generation

5.2.1 Motivation

Image-generation and editing rewards must account for instruction adherence, visual quality, physical plausibility, and other task-dependent criteria. A scalar score summarizes a judgment but does not explain the errors behind it. Building on the preference-learning setting in Chapter 2, this section asks whether structured rationales can support both evaluation and output improvement.

RationalRewards produces a multi-dimensional critique before deriving scores. Preference-Anchored Rationalization (PARROT) uses existing pairwise preference labels to obtain training rationales without requiring human-written explanations for every pair. A teacher generates label-conditioned rationales, a consistency check filters them, and a student learns to predict rationales without observing the preference label. The method treats rationales as latent variables and connects these stages to an evidence lower bound.

The resulting evaluator supports two interventions. Its scores can train a generator in parameter space, while its critiques can revise a prompt after inspecting a generated



Figure 5.1: RL (LoRA) training on Qwen-Image using scalar rewards encounters reward hacking (bottom row): as training reward continues to grow, generation quality starts to degenerate, because black-box rewards mislead visual generators with biases. In contrast, RationalRewards (top row) sustains generation quality with stable reward growth. Figures 5.6 and 5.7 provide additional training curves and visualizations.

image in a Generate–Critique–Refine loop. Experiments compare preference prediction, downstream reinforcement learning, and test-time refinement.

5.2.2 Related Work

Preference models. ImageReward, PickScore, UnifiedReward, and EditReward learn visual preference judgments from annotated comparisons [76, 159, 168, 175]. Generative reward modeling and STaR motivate using intermediate reasoning as part of prediction [50, 105, 196]. PARROT focuses on obtaining such reasoning from existing preference labels and testing its usefulness for visual evaluation.

Output improvement. FlowGRPO, DanceGRPO, and DiffusionNFT use feedback to update visual generators [94, 178, 209]. Test-time approaches instead revise conditions or iterate on an output: PromptEnhancer rewrites prompts before synthesis, whereas Reason-Edit includes reflection on edits [153, 189]. RationalRewards compares parameter updates with post-generation critique and prompt revision using the same preference-calibrated evaluator.

5.2.3 Method

5.2.3.1 Preference-Anchored Rationalization

Let $x = (I_A, I_B, c)$ contain two candidates and the user request, including a source image for editing; y is the observed preference and z a latent rationale. PARROT learns a student that produces a rationale and preference without observing y [152, 196]. Its variational objective is

$$\mathcal{L}_{\text{ELBO}} = \mathbb{E}_{q_\phi(z|x,y)}[\log P_\theta(y \mid x, z)] - D_{\text{KL}}(q_\phi(z \mid x, y) \,\|\, P_\theta(z \mid x)) \,. \tag{5.1}$$

For an input comparison x, observed preference y, and latent rationale z, introduce $q_\phi(z \mid x, y)$ and factor the student as $P_\theta(y, z \mid x) = P_\theta(y \mid x, z)P_\theta(z \mid x)$. Jensen's inequality gives

$$\log P_\theta(y \mid x) = \log \mathbb{E}_{q_\phi}\left[\frac{P_\theta(y, z \mid x)}{q_\phi(z \mid x, y)}\right] \tag{5.2}$$

$$\geq \mathbb{E}_{q_\phi}[\log P_\theta(y \mid x, z)] - D_{\text{KL}}(q_\phi(z \mid x, y) \,\|\, P_\theta(z \mid x)) = \mathcal{L}_{\text{ELBO}}. \tag{5.3}$$

The gap is exactly

$$\log P_\theta(y \mid x) - \mathcal{L}_{\text{ELBO}} = D_{\text{KL}}(q_\phi(z \mid x, y) \,\|\, P_\theta(z \mid x, y)) \,. \tag{5.4}$$

Thus the bound is tight when the variational and true posteriors agree. PARROT uses a prompted teacher as an approximate posterior, filters samples by preference recovery, and fits the label-free student to retained rationales. With fixed q_ϕ, the trainable part of the KL is the negative expected student log-likelihood. Filtering is a practical approximation to favoring predictive rationales, not exact variational optimization; this factorization alone does not establish a causal explanation of preference.

Anchoring and filtering. Qwen3-VL-32B-Instruct generates a comparison rationale conditioned on the human preference. It assesses text faithfulness, physical/visual quality, text rendering, and image faithfulness for editing, following the dimensions of Hu et al. [61]. The label focuses the teacher on explaining an observed preference rather than predicting it unaided. A second query receives the rationale but not the label; a sample is retained only when the recovered preference matches:

$$\mathcal{C}(x, y, z) = \mathbb{I}\left[\arg\max_{y'} P_{\text{Teacher}}(y' \mid x, z) = y\right] . \tag{5.5}$$

This check rejects non-predictive rationales but does not establish that every retained explanation is visually faithful.

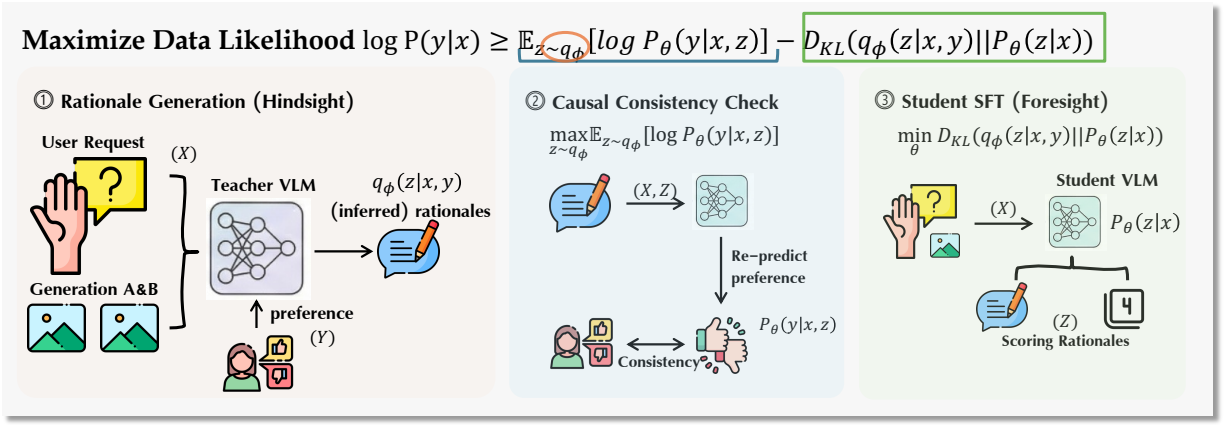


Figure 5.2: **We implement Preference-Anchored Rationalization as a practical three-phase pipeline.**

Distillation and pointwise deployment. The student learns to generate retained rationales without label access. For fixed teacher samples, minimizing the trainable KL term reduces to supervised likelihood maximization. Because RL and refinement require single-image feedback, the teacher also assesses each candidate independently using the validated comparison as a reference hint. This produces absolute 1–4 scores and justifications; generation omits image faithfulness, and text rendering is N/A when not required. The student is trained jointly on pairwise and pointwise samples.

Pointwise projection extends beyond the pairwise bound. It assumes shared evaluation criteria and can suffer from absolute-score miscalibration or dependence on the comparison partner. Rubric anchors and instructions to assess independently mitigate these risks without eliminating them.

The following summarizes the inputs and outputs of the PARROT and Generate–Critique–Refine prompts. The scoring anchors appear in Table 5.1.

Pairwise Rationale Generation The teacher receives the instruction, source image, two candidate edits, and the human preference label as an anchor. It compares both candidates on text faithfulness, image faithfulness, physical and visual quality, and text rendering, returning per-dimension justifications, floating-point scores on the 1–4 scale, winners (A, B, or tie), and a summary. For text-to-image generation, the source image is absent and image faithfulness is marked N/A. Text rendering is N/A when not required by the instruction.

Consistency Check The teacher receives the instruction and the complete generated rationale, including its scores and summary, without the preference label. It must return only A, B, or tie. The rationale is retained when this predicted preference matches the human label.

Pointwise Projection A validated pairwise rationale is supplied as a reference hint together with the instruction, source image, and one candidate edit. The teacher produces an absolute assessment with per-dimension scores, justifications, and a summary. The prompt identifies which candidate the reference describes and instructs the teacher not to mention the reference hint in the resulting assessment.

Generate–Critique–Refine At inference time, the trained reward model assesses a generated output using the same dimensions and produces both a critique and a revised request in a single forward pass. The response contains per-dimension scores and justifications, an overall summary, refinement comments, and the refined request. If any applicable dimension scores below 3.0, the revised request is sent to the generator; otherwise the output is accepted. The reported experiments permit at most one refinement per image.

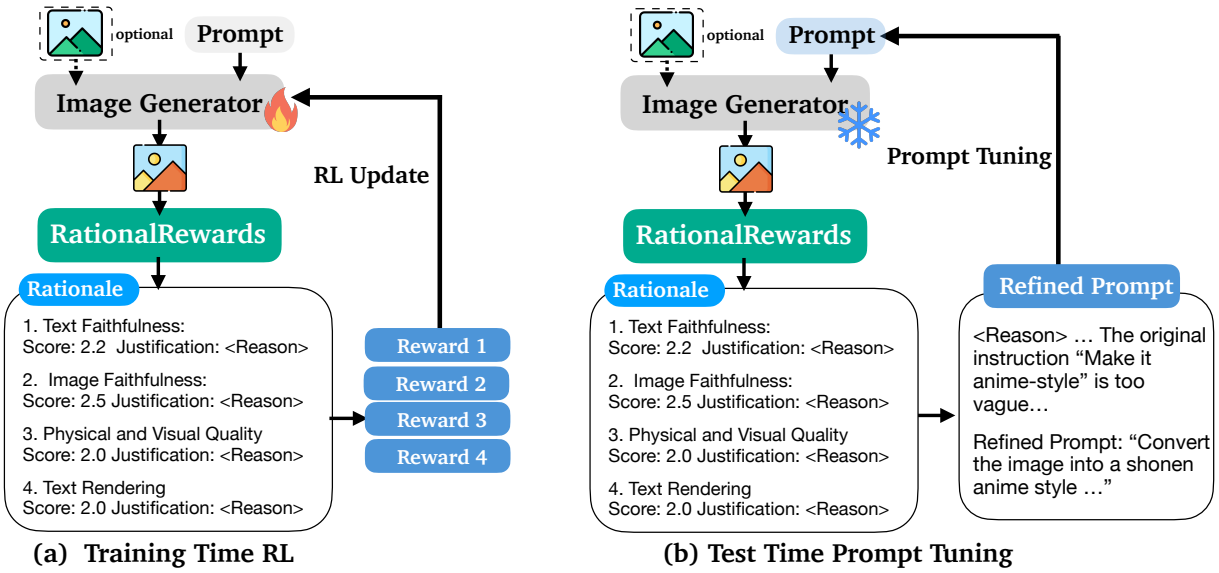
Scoring anchors. Scores interpolate between integer anchors on a 1–4 scale. Image faithfulness is N/A without a source image; text rendering is N/A when not requested. Applicable dimensions receive equal weight when producing the scalar RL reward.

5.2.3.2 Feedback at Training and Inference Time

Figure 5.3 contrasts how the same evaluator supplies feedback in parameter space and prompt space: scores guide generator updates, whereas critiques guide revisions to the generation request. Scores are averaged over applicable dimensions to supply the reward for DiffusionNFT-based generator updates [209]. The same evaluator can instead critique an output and revise its prompt in a Generate–Critique–Refine loop, leaving generator weights fixed. In the reported protocol, any applicable score below 3.0 triggers at most one refinement. This trades an additional critique and generation for potentially improved output, providing an alternative or complement to parameter updates [134].

Table 5.1: Scoring anchors used for pointwise assessment. Fractional scores interpolate between adjacent descriptions; N/A dimensions are excluded from reward averaging.

Dimension	Anchors (4 to 1)
Text faithfulness	4: all requested elements, no unrequested changes; 3: minor missing or incorrect details; 2: some key elements missing or misinterpreted; 1: major omissions or contradictions.
Image faithfulness	4: relevant source content preserved or transformed as requested; 3: minor losses or mismatches; 2: key source aspects lost or distorted; 1: source content ignored or destroyed.
Physical/visual quality	4: plausible lighting, geometry and anatomy without visible artifacts; 3: minor non-disruptive flaws; 2: clear disruptive physical or rendering errors; 1: severe geometry, anatomy or artifact failures.
Text rendering	4: correct, legible and well integrated; 3: minor spelling/capitalization errors; 2: major spelling errors or distorted text; 1: unreadable, severely distorted or missing text.



(a) Training Time RL **(b) Test Time Prompt Tuning**

Figure 5.3: RationalRewards provides feedback for two forms of optimization. (a) Scores derived from structured rationales guide RL updates to the generator. (b) Critiques guide prompt refinement in a Generate–Critique–Refine loop while generator weights remain fixed. The optional image input enables image editing as well as text-to-image generation.

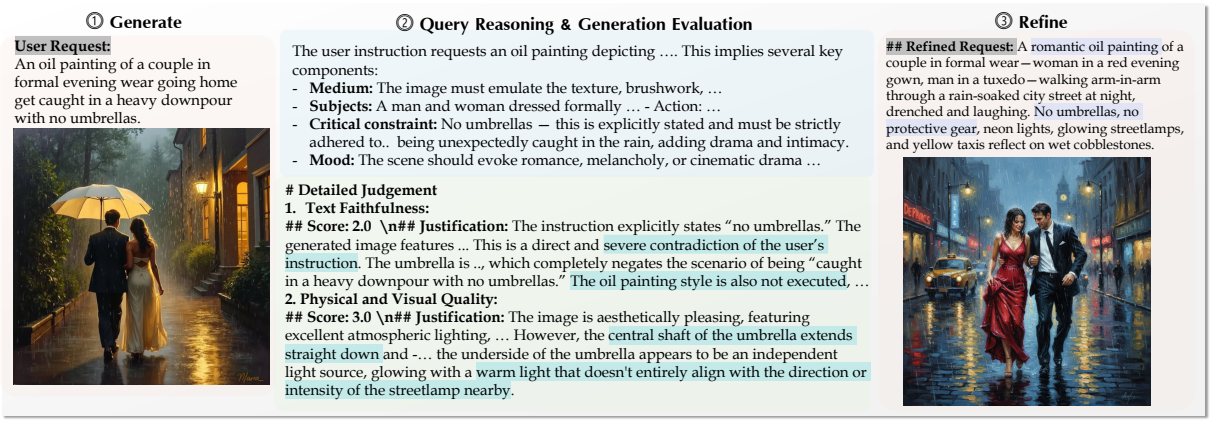


Figure 5.4: Test-Time Prompt Refinement via "Generate–Critique–Refine" loop with RationalRewards.

5.2.4 Experiments

Training Data. We evaluate RationalRewards on both image generation and image editing tasks. Our training data derives from existing preference datasets: 30K query-preference pairs from EditReward [168] for image editing, and 50K pairs from HPDv3 and RapidData [104] for text-to-image generation. These datasets provide only binary or ranked preference labels without explanations. We apply the PARROT pipeline (§ 5.2.3.1) with Qwen3-VL-32B-Instruct as the teacher model to transform these raw preference pairs into reasoning-annotated training data. Our data scale is 10-20 times smaller – this efficiency stems from the teacher model's pre-trained knowledge, which PARROT distills through structured rationales rather than raw labels; the ablation in § 5.2.4.1 isolates this factor. During Phase 2 (consistency filtering), approximately 72% of generated rationales survive the predictive consistency check, indicating that preference anchoring produces largely coherent rationales while the filter removes a meaningful fraction of hallucinated or insufficiently informative samples.[1]

Source Dataset	Task	Raw Pairs	Post-Filtering Pairs	Final Pointwise Samples
EditReward	Image Editing	30K	∼21.6K	∼43.2K
HPDv3	Text-to-Image	50K	∼36K	∼55K
RapidData	Text-to-Image			

Table 5.2: Training data composition before and after consistency filtering. Each pairwise sample yields two candidate pointwise projection samples (one per image). The final pointwise counts reflect additional filtering; approximately 55K text-to-image samples are retained from the candidates derived from approximately 36K pairs.

The rejected rationales commonly hallucinate image content, favor the non-preferred image despite the label anchor, or provide generic judgments insufficient to recover the preference. Preference recovery is a filtering criterion, not an independent audit of explanation faithfulness. For preference evaluation, MMRB2 uses 1,000 examples for each of T2I and editing, EditReward-Bench uses 133, and GenAI-Bench uses 1,700 T2I and 900 editing examples; the metric is pairwise accuracy.

5.2.4.1 Accuracy in Preference Modeling

We first evaluate whether RationalRewards produces human-aligned preference judgments. We report pairwise comparison accuracy on three established benchmarks: Multimodal

[1]Project resources: https://tiger-ai-lab.github.io/RationalRewards/.

Table 5.3: Comparison of reward models as evaluators. We include Multimodal Reward Bench 2 (MMRB2), EditReward-Bench, and GenAI-Bench. T2I and Edit denote text-to-image generation and image editing, respectively.

Judge	**MMRB2**		**EditReward**	**GenAI-Bench**	
	T2I	**Edit**		**T2I**	**Edit**
Qwen2.5-VL-7B [7]	50.4	57.1	52.69	–	40.48
Qwen2.5-VL-72B	59.1	64.6	63.9	66.6	74.3
Qwen3-VL-8B [180]	59.4	61.7	51.9	55.1	50.1
Qwen3-VL-32B	64.1	67.3	64.2	66.9	76.3
EditReward-7B [168]	–	67.2	56.99	–	65.72
UnifiedReward-7B [159]	59.8	–	–	67.9	–
RationalRewards (Qwen2.5-VL-7B)	62.3	68.5	63.6	66.4	75.7
RationalRewards (Qwen3-VL-8B)	**64.2**	**70.3**	**66.2**	**69.8**	**80.1**
Qwen3-VL-32B-Instruct Distillation	57.4	65.6	56.8	59.3	62.8
Commercial Models					
GPT-4.1	65.8	68.2	58.3	60.5	69.3
Gemini 2.5 Flash [26]	63.1	66.5	58.6	65.8	73.0
Gemini 2.5 Pro	70.5	71.3	71.3	66.2	78.9
Gemini 3 Pro [33]	74.4	74.9	72.2	73.1	80.5

Reward Bench 2 [61], GenAI-Bench [69], and EditReward-Bench [168] for both text- and image-to-image generation.

Main Results. As shown in Table 5.3, our 8B-parameter RationalRewards surpasses all open-source scalar reward models by a substantial margin across all three benchmarks, without requiring complex loss designs to handle label noise or annotation ambiguities. Notably, RationalRewards outperforms commercial models including Gemini-2.5-Flash and approaches the performance of Gemini-2.5-Pro on preference prediction, offering a cost-effective alternative for quality assessment and evaluation in visual generation.

Ablation of PARROT versus Direct Distillation. To isolate the contribution of PARROT from generic knowledge distillation, we include a baseline that performs direct SFT distillation from Qwen3-VL-32B-Instruct to the same 8B backbone, using the same data volume but without preference-anchored rationalization (marked "Qwen3-VL-32B-Instruct Distillation" in Table 5.3). This baseline underperforms RationalRewards on all benchmarks—by 6.8 points on MMRB2 (T2I) and 17.3 points on GenAI (Edit)—confirming

that the structured rationalization process, not simply access to a larger teacher, drives the performance gains. We also replace the backbone with Qwen2.5-VL-7B-Instruct; the results still exceed prior scalar reward models on four of the five benchmarks, clarifying that improvements are attributable to PARROT rather than the specific choice of backbone.

5.2.4.2 Optimization in Dual Spaces

Given the strong discriminative performance of RationalRewards, we now investigate its utility for improving downstream generation. We explore two complementary optimization strategies: *parameter-space* tuning via RL and *prompt-space* tuning via test-time critique-and-refinement. We evaluate on ImgEdit-Bench [186] and GEdit-Bench-EN [97] for image editing, and UniGenBench++ for text-to-image generation.

Parameter Space Tuning (RL). We experiment with the recent Diffusion RL approach, DiffusionNFT [209], which samples a group of generations for the same user prompt and optimizes with a weighted diffusion loss.

Optimization settings. The scalar reward is the arithmetic mean of applicable dimension scores, excluding N/A. RL prompts are drawn from EditReward and HPDv3 examples with initial mean RationalRewards scores below 3.0. Table 5.4 gives the training configuration; the test-time threshold, one-refinement budget and VLM overhead are reported alongside the PT results.

We use RationalRewards to provide dense, per-dimension reward signals for RL fine-tuning and systematically compare against alternative reward models spanning two axes: *scalar vs. reasoning-based* and *generic vs. preference-trained*:

1. **Scalar reward models**: EditReward [168] for image editing and MultiReward (used by DiffusionNFT [209]) for text-to-image generation. These output a single scalar score without natural language reasoning.
2. **Generic reasoning model**: Qwen3-VL-32B-Instruct used directly as a judge. This model can produce natural language critiques but has *not* been trained on preference data via PARROT, isolating the contribution of our training pipeline from raw model scale.

As shown in Tables 5.6 and 5.5, RL with RationalRewards yields consistent improvements over both base models across nearly all subcategories, surpassing both scalar reward baselines and the generic reasoning baseline. For image editing, RationalRewards-guided RL improves Flux.1 Kontext from 3.52 to 3.84 overall on ImgEdit-Bench, outperforming

Hyperparameter	Value
Resolution	512 × 512
Guidance Scale (Flux.1 Kontext Dev)	2.5
Sampling Steps (Training)	15 (DPM solver)
Sampling Steps (Evaluation)	20
Noise Level	0.7
Learning Rate	2e-4
β (guidance strength)	0.0001
Batch Size (per GPU)	8
Group Size K	16 (across 16 process groups)
Quality Filtering (mean threshold)	0.9
Quality Filtering (std threshold)	0.05
LoRA Rank	64
LoRA Alpha	128
GPU Configuration	8 × A100-80GB (training) + 8 × A100-80GB (reward)
RL Training Compute	∼384 GPU-hours per generator

Table 5.4: Hyperparameters for RL fine-tuning via DiffusionNFT.

Table 5.5: Ablation of RationalRewards for Text-to-image RL on UniGenBench++. We compare scalar reward model *MultiReward* and generic reasoning reward *Qwen3-VL-32B*.

Model	Action	Attribute	Compound	Layout	Grammar	Logic	Relation	Style	Text	World Know.	Overall
FLUX.1-dev	62.24	67.20	45.75	70.84	62.30	29.77	66.88	85.00	32.18	87.50	60.97
+MultiReward	59.78	68.23	44.21	74.37	59.33	28.25	68.35	76.05	36.21	86.03	60.12
+Qwen3-VL-32B	65.47	72.68	53.28	71.82	60.78	33.24	71.85	85.53	42.15	89.47	64.63
+RationalRewards	67.40	76.36	57.67	72.15	60.29	40.53	74.59	87.20	52.57	90.61	**67.94**
SD-3.5-Medium	60.41	66.99	53.35	70.31	59.89	37.73	68.78	89.80	15.23	84.34	60.71
+RationalRewards	64.36	81.49	67.98	75.88	58.68	42.37	75.60	89.60	10.05	91.77	**65.78**
+MultiReward	57.03	66.67	51.03	75.37	57.22	34.86	67.51	77.60	21.84	86.71	59.58
+Qwen3-VL-32B	61.23	74.48	63.85	75.34	59.67	31.23	72.84	84.73	14.87	88.86	62.71
Qwen-Image	82.49	87.93	72.94	86.56	60.96	51.59	80.08	94.70	72.13	94.15	78.36
+MultiReward	79.52	86.45	70.91	88.53	58.43	48.62	80.55	83.75	67.18	92.17	75.61
+Qwen3-VL-32B	81.95	87.45	76.42	87.73	62.93	51.14	81.55	95.20	75.67	95.63	79.57
+RationalRewards	82.11	87.82	78.82	88.07	66.21	52.88	82.21	96.60	79.76	96.57	**81.11**

EditReward-guided RL (3.66) by a clear margin. For text-to-image generation, Table 5.5 reports the per-generator comparison with both scalar and generic reasoning rewards. Notably, the 8B RationalRewards outperforms Qwen3-VL-32B used as a direct judge, confirming that PARROT's structured preference training provides value beyond raw model capacity.

Test-Time Prompt Space Tuning. We leverage the generative nature of Rational-

Table 5.6: Ablation of RationalRewards as dual-space optimizer on editing tasks. For prompt space tuning, we compare pre-generation PromptEnhancer [153]. For parameter space tuning, we compare SFT and RL with different rewards. We include OOD physics-aware editing, PICA-Bench with representative aspects (Left), and generic editing benchmarks (Right). Left: +PT means base+PT, independently of RL. Right: +RL+PT applies PT to the corresponding RL-tuned generator.

Model	Representative Aspects			Overall
	Light	Reflec.	Deform.	
Flux.1 Kontext [dev]	53.64	43.84	33.74	41.07
+PromptEnhancer	55.53	45.87	38.14	45.28
+PT (RationalRewards)	56.87	51.43	41.08	**48.12**
+PICA SFT	51.21	47.22	33.99	41.93
+RL (RationalRewards)	51.75	54.81	39.36	44.25
Qwen-Image-Edit	52.02	49.07	38.14	49.71
+PromptEnhancer	58.49	50.42	42.30	50.97
+PT (RationalRewards)	63.34	61.55	43.28	**55.65**
+PICA SFT	60.47	55.19	40.99	52.06
+RL (RationalRewards)	63.07	60.71	41.32	54.11

Model	ImgEdit	GEdit-Bench-EN		
	Overall	G_SC	G_PQ	G_O
Flux.1 Kontext [dev]	3.52	7.16	7.37	6.51
+RL (EditReward)	3.66	7.38	7.53	6.88
+RL (Qwen3-VL-32B)	3.67	7.42	7.48	6.82
+RL (RationalRewards)	3.84	7.75	8.24	**7.37**
+RL+PT (RationalRewards)	**4.01**	7.77	7.61	7.23
Qwen-Image-Edit	4.27	8.00	7.86	7.56
+RL (EditReward)	4.25	8.36	7.91	7.77
+RL (Qwen3-VL-32B)	4.25	8.42	7.83	7.79
+RL (RationalRewards)	4.38	8.74	8.43	8.29
+RL+PT (RationalRewards)	**4.43**	8.94	8.20	**8.33**

Rewards in a *Generate–Critique–Refine* protocol: the generator produces an initial image; RationalRewards evaluates it across four dimensions with natural language critique and refinement suggestions; if any dimension score falls below a threshold of 3.0, the refined request is fed back to the generator. This single-iteration loop adds approximately 0.4 seconds of VLM inference overhead per image (via vLLM prefix caching and paged attention), compared to ∼384 GPU-hours for RL fine-tuning of a single base model.

Prompt Tuning on Base and RL-Tuned Generators. For ImgEdit-Bench and GEdit-Bench-EN (Table 5.6, right), +RL+PT denotes test-time prompt tuning applied to the corresponding RL-tuned generator, with its weights fixed during refinement. On ImgEdit-Bench, prompt tuning boosts the RL-tuned Flux model from 3.84 to 4.01 overall. For Qwen-Image-Edit, prompt tuning applied on top of RL yields the best overall score of 4.43, with the two methods proving complementary. On GEdit-Bench-EN Overall, RL+PT (8.33) slightly exceeds RL alone (8.29).

In contrast, the PICA-Bench comparisons (Table 5.6, left) apply prompt tuning directly to the base generators, without prior RL fine-tuning. Here, +PT means base+PT: Flux.1 Kontext reaches 48.12 overall, compared with 44.25 for base+RL, while Qwen-Image-Edit reaches 55.65, compared with 54.11 for base+RL. Thus, the PICA-Bench results compare the two interventions separately, whereas the ImgEdit-Bench and GEdit-Bench-EN results test PT on top of RL.

The RL performance ceiling is partly structural: LoRA-based fine-tuning constrains pa-

rameter update capacity, and the RL query distribution may not fully cover the evaluation distribution. In contrast, prompt tuning performs per-instance optimization without risk of catastrophic forgetting. These results motivate a *latent capability hypothesis*: generators already possess the capacity for high-quality outputs, but this capacity is under-elicited by suboptimal prompts. RationalRewards's critique bridges user intent and model capability without weight modification. This interpretation concerns how prompts elicit existing capabilities; the experiments measure output quality rather than internal representations.

Qualitative Results. Figure 5.5 illustrates the two interventions. In image editing, the anime-style example shows stronger stylization after optimization, while the snowy-forest example shows more extensive changes to the requested background. The text-to-image examples compare the base and RL-tuned generators. The Base+PT column applies prompt tuning directly to the base generator, distinct from the RL+PT setting evaluated on ImgEdit-Bench and GEdit-Bench-EN.

5.2.4.3 Training Dynamics and Reward Hacking

Figure 5.6 compares reward trajectories and their variability with scalar-reward alternatives. The observed training behavior supports the optimization comparisons, but smooth reward curves alone are not evidence of faithful feedback. Figure 5.7 illustrates why: in scalar-reward training, increasing scores can accompany accumulating texture artifacts and changes in depicted content. These examples motivate checking generated images as well as the reward; they do not establish immunity to reward hacking for RationalRewards.

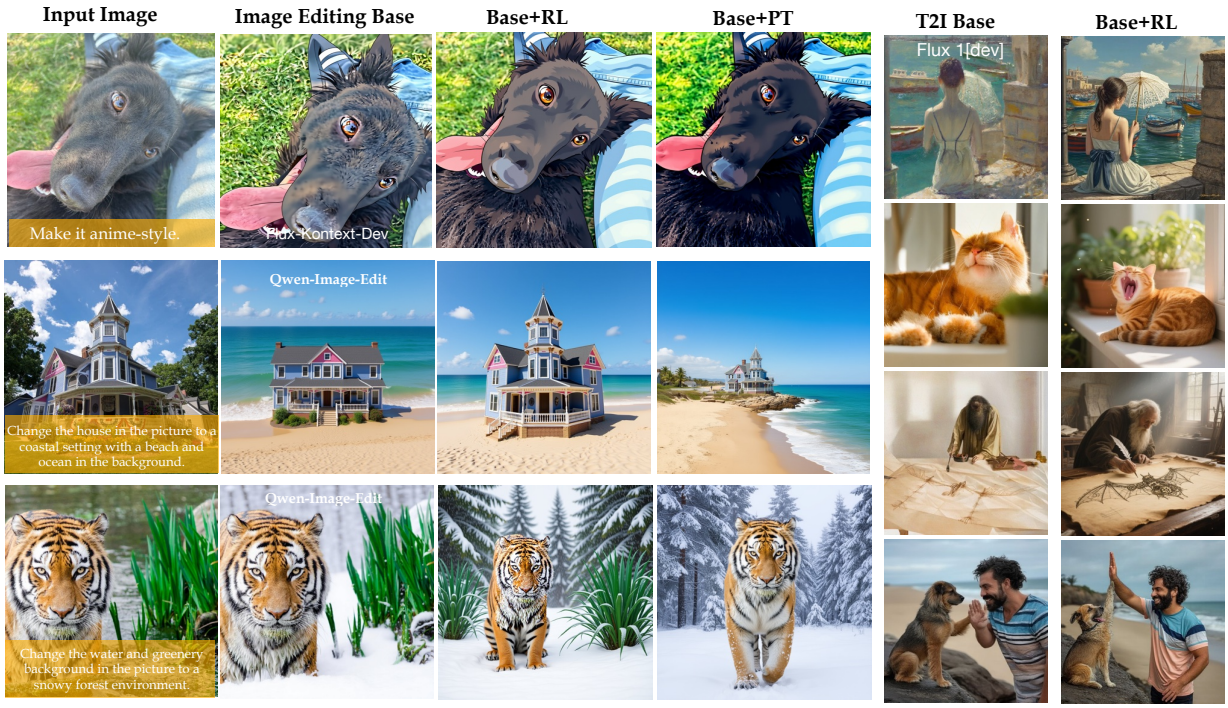


Figure 5.5: Qualitative results using RationalRewards. Left: image editing with the base generator, RL fine-tuning (Base+RL), and prompt tuning of the base generator (Base+PT). Right: text-to-image generation before and after RL fine-tuning. These selected examples illustrate changes in style, requested scene content, and visual detail; aggregate comparisons are reported in Tables 5.6 and 5.5.

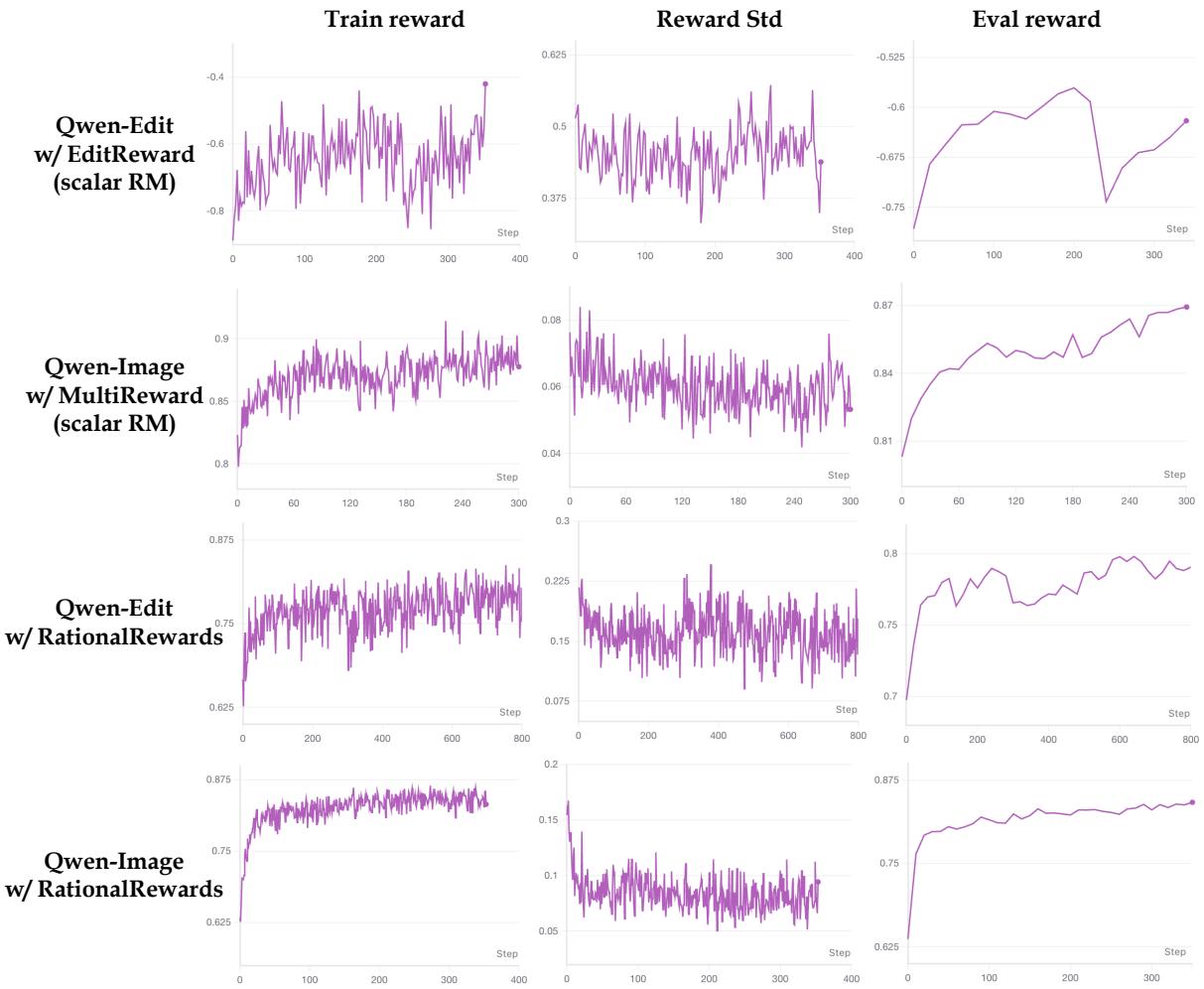


Figure 5.6: Training curves comparison between RationalRewards and two scalar reward models: EditReward [168] and MultiReward, as used in DiffusionNFT [209].



Figure 5.7: Two selected trajectories from scalar-reward optimization, shown from 0 to 300 RL steps. Rising reward can accompany visual degradation. The two rows illustrate drawing and cat examples.

5.2.5 Discussion

RationalRewards turns preference supervision into feedback for improving outputs, rather than only ranking candidates. PARROT anchors rationales to observed preferences and filters them before distillation. Its advantage over direct teacher distillation indicates that how evaluation knowledge is transferred matters, beyond access to a larger teacher. The resulting critiques connect the user's request to specific aspects of an image, supporting both generator training and prompt revision.

The downstream experiments show that feedback can improve instruction adherence through changes to either parameters or inputs. Prompt tuning improves base generators on PICA-Bench and adds overall gains to RL-tuned generators on ImgEdit-Bench and to Qwen-Image-Edit on GEdit-Bench-EN. These settings distinguish refinement as an alternative to, or a complement to, parameter updates. Inspecting an output can thus help elicit a better response, although the experiments do not isolate under-elicited capabilities from other explanations, such as added prompt context.

These findings connect preference-calibrated evaluation to generator training and prompt refinement. Feedback reliability depends on the teacher and preference data; preference alignment alone does not guarantee faithful rationales or immunity to reward hacking. Whereas RationalRewards learns feedback through evaluator training, RewardHarness next adapts reusable evaluation skills and tools with the evaluator's parameters fixed.

5.3 RewardHarness: Learning Evaluation Skills from Preference Demonstrations

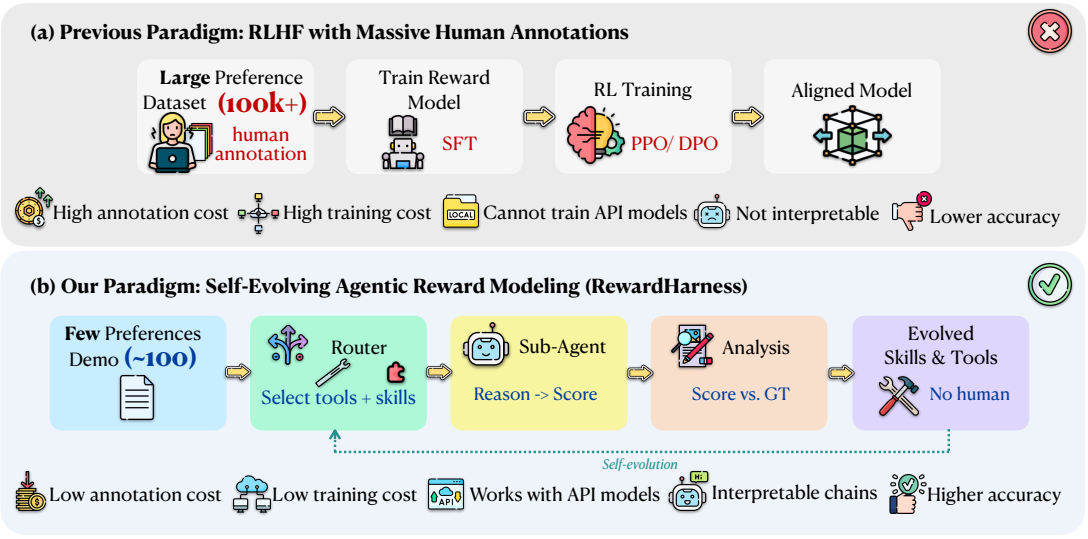


Figure 5.8: **Paradigm comparison.** The conventional paradigm collects large-scale human preference data, trains a reward model, and uses it as the reward signal for RL alignment. In contrast, RewardHarness starts from a small set of preference demonstrations and self-evolves a Skills-and-Tools Library through iterative evaluation and analysis, yielding an interpretable reward system.

5.3.1 Motivation

RationalRewards learns to evaluate visual outputs from tens of thousands of preference pairs: its training pipeline starts with 30,000 image-editing pairs and another 50,000 text-to-image pairs before filtering (Table 5.2). RewardHarness asks whether image-editing preferences can instead be learned from just 100 labeled preference demonstrations, without updating the evaluator's parameters. The central idea is to use these examples to improve reusable evaluation procedures around a frozen vision-language model, rather than to train a new reward model on a large preference dataset.

RewardHarness stores these procedures in an evolving library of *Skills* and *Tools*. Skills specify evaluation criteria; tools define targeted visual checks. An Orchestrator selects

relevant entries, and a frozen Sub-Agent applies them to an instruction, source image, and candidate edits. Prediction errors on the demonstration set guide changes to the library, allowing a small set of examples to shape how the evaluator inspects and compares previously unseen edits.

Figure 5.8 summarizes the distinction from conventional reward-model training. The method section specifies the library, selection procedure, and self-evolution loop. Experiments assess held-out preference prediction, the library's evolution, and its use as a reward for training a separate image generator. The last use still involves generator parameter updates: freezing the evaluator does not make the entire downstream workflow training-free.

5.3.2 Related Work

Learning visual preferences. ImageReward, PickScore, VisionReward, and EditReward train evaluators from preference data [76, 168, 175, 176]. RationalRewards (Section 5.2) additionally trains the evaluator to produce structured rationales. RewardHarness instead adapts an external evaluation library using a small demonstration set, while freezing the evaluator's parameters.

Context evolution and tools. Reflexion, ExpeL, and Voyager use feedback to refine memories, instructions, or reusable skills [133, 150, 206]. ReAct, Gorilla, and ToolLLM study reasoning with external tool calls [117, 123, 184]. RewardHarness specializes these ideas to preference evaluation: it evolves both the guidelines and targeted checks supplied to a frozen VLM, then evaluates the resulting system as a reward for a separate generator.

5.3.3 Method

We present RewardHarness, a self-evolving agentic reward system that acquires human evaluation preferences through context evolution alone, without updating any evaluator model parameters. RewardHarness consists of two main components: an **Orchestrator** agent and a shared **Library** of interpretable evaluation artifacts. At inference time, the Orchestrator retrieves relevant artifacts from the Library and injects them into the context of a frozen Sub-Agent vision-language model (VLM), which performs the preference judgment. At evolution time, the Orchestrator drives iterative Library refinement using a small calibration set of human preference demonstrations. Figure 5.9 provides an overview of the full pipeline. We describe each component in turn: the problem formulation (§5.3.3.1), the

Skills and Tools Library (§5.3.3.2), the Orchestrator (§5.3.3.3), the Sub-Agent (§5.3.3.4), and the self-evolution loop (§5.3.3.5).

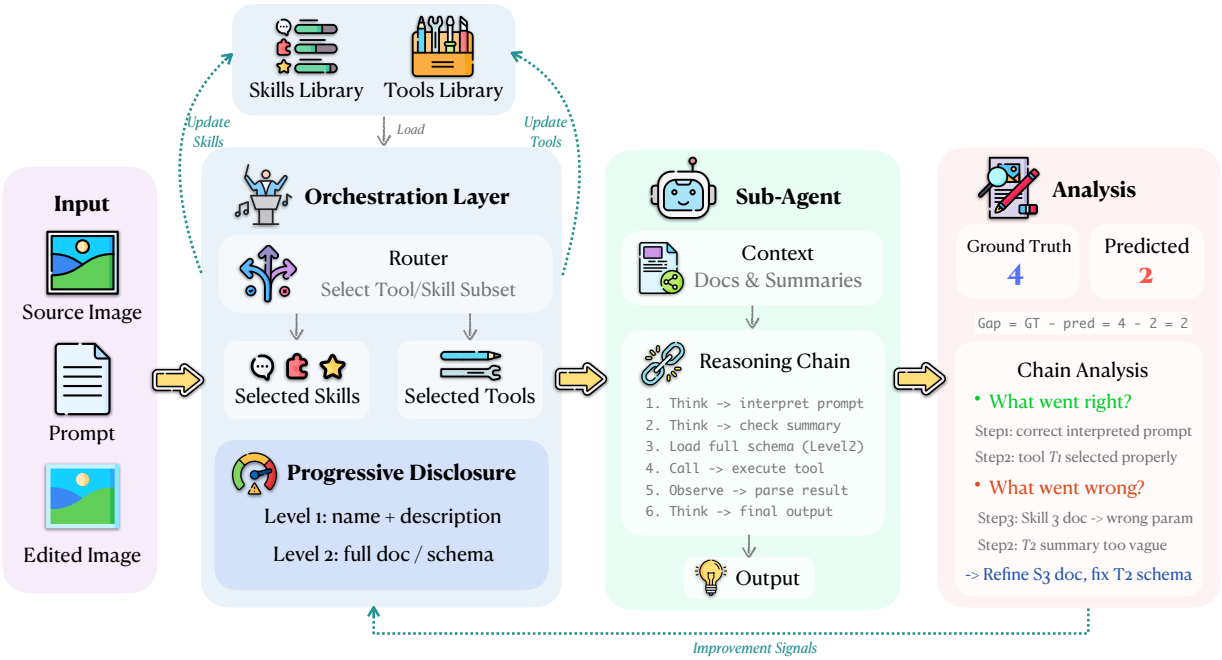


Figure 5.9: **Overview of the RewardHarness self-evolution pipeline.** Multimodal inputs enter the Orchestrator, which selects Skills and Tools through progressive disclosure routing. A frozen Sub-Agent applies the selected context to produce a preference judgment. The Chain Analyzer compares predictions with human labels and proposes changes to a candidate Library; the validation gate determines whether to commit them.

5.3.3.1 Problem Formulation

Given a source image I_s, an editing instruction p, and K candidate edited images $\{I_1, \ldots, I_K\}$, the task is to produce a preference ranking π over $\{1, \ldots, K\}$ such that $I_{\pi(1)} \succ I_{\pi(2)} \succ \cdots \succ I_{\pi(K)}$. In RewardHarness, ranking is realized by a frozen VLM $\mathcal{M}$ steered entirely by a context $\mathcal{C}$ assembled at inference time:

$$\pi = \mathcal{M}\Big(I_s,\ \{I_k\}_{k=1}^{K},\ p,\ \mathcal{C}\Big), \tag{5.6}$$

where $\mathcal{C}$ comprises the Skill documents and Tool specifications selected by the Orchestrator; the parameters of $\mathcal{M}$ are never updated.

5.3.3.2 Skills and Tools Library

RewardHarness maintains a **Library**, a versioned collection of Skills and Tools that encodes accumulated evaluation knowledge. The Library is initialized empty and grows through self-evolution (§5.3.3.5). Representative examples of both components are shown in Figure 5.10.

Skills. A *Skill* is a structured Markdown evaluation guideline containing: a **name**, a one-line **description**, a **scoring rubric** decomposing quality into assessable criteria, and **examples** illustrating correct application. For instance, the skill *realism-and-artifact-penalties* provides rubrics that distinguish visual artifacts (always penalized) from conceptual unrealism (acceptable when explicitly requested by the editing instruction).

Tools. A *Tool* is a structured Markdown document that specifies a targeted visual analysis procedure: it defines the tool's **name**, **purpose**, expected **inputs** and **outputs**, **invocation conditions**, and a step-by-step **execution protocol**. Unlike Skills (which provide declarative evaluation criteria), Tools provide *procedural* in-context specifications rather than standalone learned modules: by reading a Tool document, a general-purpose VLM can temporarily act as a specialized expert for a particular visual analysis task, either by performing the targeted analysis directly or by issuing a structured secondary VLM query defined by the Tool schema, without any parameter updates. For example, the *text-and-ocr-analyzer* tool instructs the Sub-Agent to extract, compare, and verify text content in source and edited images, catching typos and placement errors that holistic evaluation routinely misses.

5.3.3.3 Orchestrator Layer

The Orchestrator is a Claude-based LLM with two roles. During *inference*, it examines the editing instruction and source image, then selects Skills and Tools to assemble the evaluation context $\mathcal{C}$. Figure 5.9 illustrates progressive disclosure routing. During *evolution*, it analyzes reasoning chains against human labels and proposes Library updates (§5.3.3.5).

5.3.3.4 Sub-Agent

The Sub-Agent is a frozen, pluggable VLM that receives the multimodal inputs I_s, $\{I_k\}_{k=1}^{K}$, p, and the assembled context $\mathcal{C}$ from the Orchestrator. By reading the Skill and Tool

(a) Skill: realism-and-artifact-penalties

```
name: realism-and-artifact-penalties

type: skill

description: Guidance on penalizing artifacts while allowing conceptual unrealism if
requested by the prompt.

# Realism and Artifact Penalties

1. Conceptual Unrealism vs. Visual Artifacts: If the prompt requests an inherently
surreal ... scenario, DO NOT penalize the image for being conceptually unrealistic. The
goal is to follow the prompt.

2. Penalize Visual Artifacts: Only penalize for actual image processing artifacts, such
as: bad blending or visible seams ..., floating objects with missing shadows, distorted
textures ...

3. Prioritize Execution Quality: Between two images that follow the instruction, prefer
the one with fewer visual artifacts ...
```

(b) Tool: text-and-ocr-analyzer

```
name: text-and-ocr-analyzer

type: tool

description: Extracts and analyzes text within images to verify spelling, placement, and
clarity.

input_schema: {images: list[base64_str], query: str}

output_schema: {image_1_text: str, image_1_spelling_correct: bool, ...}

system_prompt: “You are an expert OCR and text analysis AI. Extract all visible text …
Pay strict attention to spelling, typos, and placement ...”

Use this tool whenever the prompt asks to add or modify text. It will read the exact
spelling from the images so you can penalize typos or hallucinations. Example query:
‘What text is written on the bottom left of the images? Is it spelled exactly as
`begging` ?’
```

Figure 5.10: Examples of a **Skill** and a **Tool** sampled from the Library at evolution iteration 69. Skills are declarative rubrics guiding the Sub-Agent's assessment criteria; Tools are procedural specifications instructing the Sub-Agent to perform targeted visual analysis.

documents in $\mathcal{C}$, the Sub-Agent temporarily adopts the role of a specialized evaluator and constructs a structured reasoning chain. Our default configuration uses Qwen2.5-VL-7B-Instruct, but the Sub-Agent is fully pluggable: we also evaluate Gemini as a drop-in replacement (Table 5.7). The reasoning chain proceeds in three steps:

1. **Rubric application.** For each Skill in $\mathcal{C}$, the Sub-Agent applies its scoring rubric to every candidate image, producing per-criterion assessments grounded in the skill's guidelines and examples.
2. **Tool-guided analysis (optional).** For each Tool in $\mathcal{C}$ whose invocation conditions are met, the Sub-Agent follows the tool's execution protocol to perform a targeted visual analysis (e.g., OCR extraction, spatial relationship verification, object counting) and appends the structured result to the reasoning chain.
3. **Aggregation and ranking.** The Sub-Agent synthesizes all per-criterion assessments and tool outputs into a final preference ranking π over the K candidates.

5.3.3.5 Self-Evolution Loop

The self-evolution loop takes as input $N = 100$ human preference demonstrations $\mathcal{D} = \{(I_s^{(i)}, p^{(i)}, \{I_k^{(i)}\}, \pi^{*(i)})\}_{i=1}^{N}$. The Orchestrator partitions $\mathcal{D}$ into a training split $\mathcal{D}_{\text{train}}$ ($\approx$70 examples) and a held-out validation split $\mathcal{D}_{\text{val}}$ ($\approx$30 examples). Each iteration of the loop proceeds through five stages:

Step 1: Evaluation. For each example in $\mathcal{D}_{\text{train}}$, the Orchestrator retrieves the most relevant Skills and Tools from the current Library and assigns them to a Sub-Agent. The Sub-Agent constructs a reasoning chain and produces a predicted preference ranking $\hat{\pi}^{(i)}$ following §5.3.3.4.

Step 2: Scoring. Predicted rankings are compared with human preferences to identify correct predictions and errors.

Step 3: Chain analysis. The Orchestrator examines reasoning chains from both correct and incorrect predictions. For errors, it performs root-cause analysis: identifying whether the failure stems from a missing evaluation criterion (suggesting a new Skill), an incorrect

rubric application (suggesting a Skill modification), or a perceptual hallucination (suggesting a new or improved Tool). For correct predictions, it identifies which Skills and Tools were instrumental, reinforcing their retention. The Chain Analyzer produces a structured proposal specifying a target artifact, an operation (create, modify, or deprecate), and a justification grounded in the observed reasoning traces. It changes only a candidate Library snapshot, not the committed Library.

Step 4: Library update. Based on the analysis, the Orchestrator proposes one of three actions: (i) *creating* a new Skill or Tool, (ii) *modifying* an existing entry, or (iii) *deprecating* an entry that consistently leads to incorrect reasoning. In addition to incremental updates, the system can also perform aggressive pruning to remove accumulated artifacts from the exploration phase. In our experiments, the pruning phase begins around iteration 50 after the library peaks at 13 entries (8 Skills + 5 Tools), eventually producing a compact final library with 7 entries (3 Skills + 4 Tools).

Step 5: Validation and gating. The candidate Library is evaluated on $\mathcal{D}_{\text{val}}$ and committed only if its accuracy strictly exceeds that of the current committed Library; otherwise the candidate is discarded and the previous snapshot restored. Thus, an Orchestrator recommendation alone cannot make an edit persist. Skill and Tool proposals had acceptance rates of 8.1% and 41.4%, respectively, illustrating the greater difficulty of changing declarative rubrics without validation regression. After the iteration budget, the final committed Library is frozen for benchmark evaluation. Its 3 Skills and 4 Tools achieved 62.5% validation accuracy, compared with 42.5% for the empty library (a 47% relative improvement).

5.3.4 Experiments

We evaluate RewardHarness on editing reward benchmarks and downstream RL applications. These tests distinguish agreement with human preferences from the usefulness of the evolved evaluator as a training signal. The evolution analysis then examines how the learned Skills and Tools change as the Library adapts. The default open-source Sub-Agent is a frozen Qwen2.5-VL-7B-Instruct backbone served via vLLM; no evaluator or Sub-Agent parameters are updated during reward-system evolution. We also evaluate Gemini-2.0-Flash as a closed-source Sub-Agent, evolving its Library separately using the same framework (Table 5.7).

Table 5.7: Comparison of image editing evaluators on editing reward benchmarks. **Best** and second-best results are highlighted. Δ measures the average improvement over the GPT-4o baseline. Scores for Gemini 2.5 Flash follow EditReward [168]; Table 5.3 instead reports the RationalRewards study's own evaluation runs for API models, so the GenAI-Bench score of Gemini 2.5 Flash differs between the two tables (57.0 here vs. 73.0).

Method	**EditReward-Bench**			**GenAI-Bench**	**Avg.**	**Δ**
	K=2	K=3	K=4			
Proprietary Models						
GPT-4o	45.7	27.3	7.3	53.5	33.5	–
GPT-5	57.5	38.5	12.8	59.6	42.1	+8.6
Gemini-2.0-Flash	52.4	33.3	**13.5**	53.3	38.1	+4.6
Gemini-2.5-Flash	58.6	39.9	12.2	57.0	41.9	+8.4
Claude-Haiku-4.5	57.9	30.7	7.4	47.1	35.8	+2.3
Open-Source Models						
Qwen2.5-VL-7B	52.7	24.7	3.4	40.5	30.3	−3.2
Qwen2.5-VL-32B	50.5	25.3	4.1	39.3	29.8	−3.7
MiMo-VL-7B	49.5	30.4	9.5	57.9	36.8	+3.3
EditReward (Qwen)	57.0	36.0	10.8	64.0	42.0	+8.5
EditReward (MiMo)	56.5	42.7	11.5	65.7	44.1	+10.6
Models with Evolving Skills and Tools (Ours)						
RewardHarness (Qwen)	57.9	**46.7**	10.8	**67.5**	45.7	+12.2
RewardHarness (Gemini-2.0-Flash)	**66.2**	45.3	**13.5**	64.4	**47.4**	+13.9

5.3.4.1 Main Results on Image-Editing Evaluation

We evaluate on EditReward-Bench [168] and GenAI-Bench [69]. The 100 demonstrations come exclusively from the EditReward-Data training split; EditReward-Bench is disjoint and reserved for final evaluation. Each predicted K-way ranking is converted into all pairwise comparisons: $K = 2, 3, 4$ contribute 1, 3, and 6 comparisons per example. The EditReward-Bench subsets contain 133, 450, and 888 comparisons, respectively, totaling 1,471. No additional human feedback or evaluator weight updates are used.

Main results. Table 5.7 compares RewardHarness against proprietary models (GPT-4o, GPT-5, Gemini, Claude) and open-source baselines (Qwen2.5-VL, MiMo-VL, EditReward) on EditReward-Bench (K=2/3/4) and GenAI-Bench. With a frozen Qwen2.5-VL-7B Sub-Agent, RewardHarness achieves 45.7 average accuracy, outperforming all listed baselines on average, including the strongest open-source reward model EditReward (MiMo) at 44.1 and the strongest proprietary baseline GPT-5 at 42.1. Importantly, this is not simply a backbone advantage: compared under the same Qwen2.5-VL-7B backbone, RewardHarness (Qwen) still outperforms EditReward (Qwen) by 3.7 points on average (45.7 vs. 42.0). Crucially, this result is obtained without any parameter updates to the underlying VLM and using only 100 preference examples sampled from the EditReward training set for evolution.

The frozen Qwen2.5-VL-7B model scores only 30.3 by itself, so the full system improves it by +15.4 points through the evolved Skills and Tools applied to each evaluation example. RewardHarness also generalizes well beyond its evolution data: although each Library is evolved only from 100 examples sampled from the EditReward training split, the Qwen-based RewardHarness achieves the best GenAI-Bench accuracy of 67.5, suggesting that the learned Skills and Tools capture general editing-quality criteria rather than benchmark-specific heuristics.

Pluggable Sub-Agent. With a Library separately evolved using Gemini-2.0-Flash, the framework achieves the best average accuracy of 47.4, the best EditReward-Bench result at K=2, and tied-best performance at K=4. This demonstrates applicability to both open and API-based Sub-Agents, rather than direct transfer of the same Library between backends.

5.3.4.2 Performance as Reward Modeling

We next test whether the learned evaluator improves image editing when used as a training reward. We use RewardHarness as the reward signal in FlowGRPO [94] to fine-tune

FLUX.2-klein-base-4B, evaluating the resulting editor on ImgEdit-Bench [186] against the base model and an EditReward-trained counterpart under the same FlowGRPO setup.

Reward-driven editing improvement. As shown in Table 5.8, FlowGRPO fine-tuning with RewardHarness improves the base model overall on ImgEdit-Bench (3.32 → 3.52), reaching the same overall score as Flux.1 Kontext [dev] despite using a significantly smaller 4B backbone.

Comparison under the same FlowGRPO setup. Both reward signals use the same FlowGRPO pipeline: RewardHarness improves the overall score from 3.32 to 3.52, compared with 3.45 for EditReward. The gains are not uniform across categories. EditReward improves Add and Replace more, while RewardHarness performs better on Adjust, Extract, Background, and retains the base-model Compose score. This comparison isolates the reward choice within the shared training setup.

Table 5.8: To validate the effectiveness of RewardHarness as a reward model, we use it to RL-tune FLUX.2-klein-base-4B and evaluate on downstream image editing benchmarks (ImgEdit-Bench). Compared to EditReward, RewardHarness yields more substantial improvements in editing performance.

Method	**ImgEdit-Bench**									
	Add	Adjust	Extract	Replace	Remove	Bg.	Style	Compose	Action	Overall
Existing Image Editing Models										
AnyEdit	3.18	2.95	1.88	2.47	2.23	2.23	2.85	1.56	2.65	2.45
UltraEdit	3.44	2.81	2.13	2.96	1.45	2.86	3.76	1.91	2.98	2.70
Step1X-Edit	3.88	3.14	1.76	3.40	2.41	3.16	4.63	2.64	2.52	3.06
BAGEL	3.56	3.31	1.70	3.30	2.62	3.24	4.49	2.38	4.17	3.20
OmniGen2	3.57	3.06	1.77	3.74	3.20	3.57	4.81	2.52	4.68	3.44
Ovis-U1	4.13	3.62	2.98	4.45	4.06	4.22	4.69	3.45	4.61	4.00
GPT-Image-1	4.61	4.33	2.90	4.35	3.66	4.57	4.93	3.96	4.89	4.20
Flux.1 Kontext [dev]	3.76	3.45	2.15	3.98	2.94	3.78	4.38	2.96	4.26	3.52
Models Tuned with RL										
FLUX.2-klein-base-4B	3.27	4.00	2.04	3.07	2.90	3.47	4.26	3.19	4.44	3.32
+RL (EditReward)	4.21	4.01	2.02	3.21	2.58	3.69	4.32	2.85	4.34	3.45
+RL (RewardHarness)	4.03	4.27	2.24	3.12	2.19	4.22	4.44	3.19	4.22	**3.52**

5.3.4.3 Analysis

Figure 5.13 shows the self-evolution dynamics over 77 iterations for the Gemini-2.0-Flash Sub-Agent, corresponding to the configuration with the best average accuracy in Table 5.7. Validation accuracy plateaus at 52.5% as the library grows to 13 entries (8 Skills + 5 Tools), then improves after the pruning phase begins around iteration 50. The final selected library at iteration 69 reaches 62.5% validation accuracy with 7 entries (3 Skills + 4 Tools). In the separately evolved Gemini Library, iteration 10 contains six skills and three tools, whereas the exported iteration-69 checkpoint contains three skills and four tools; validation accuracy changes from 0.525 to 0.625. Early updates introduce OCR and visual-question-answering tools after text-reading and hallucination failures; later guidance strengthens verification and distinguishes intentional surrealism from rendering artifacts. A spatial/object tool supports counting and orientation checks through structured output. These observations make the learned feedback concrete: the Library changes both what to check and when to consult a tool. They describe one run, not a controlled attribution of the accuracy gain to any individual update. We further examine RewardHarness's behavior qualitatively. Figure 5.11 shows a representative preference-scoring example from EditReward-Bench: RewardHarness assigns the higher score to the human-preferred candidate, while EditReward fails. Figure 5.12 compares RL-tuned editing outputs, showing that the RewardHarness-trained variant faithfully executes editing instructions while the base model and EditReward-trained variant frequently fail.

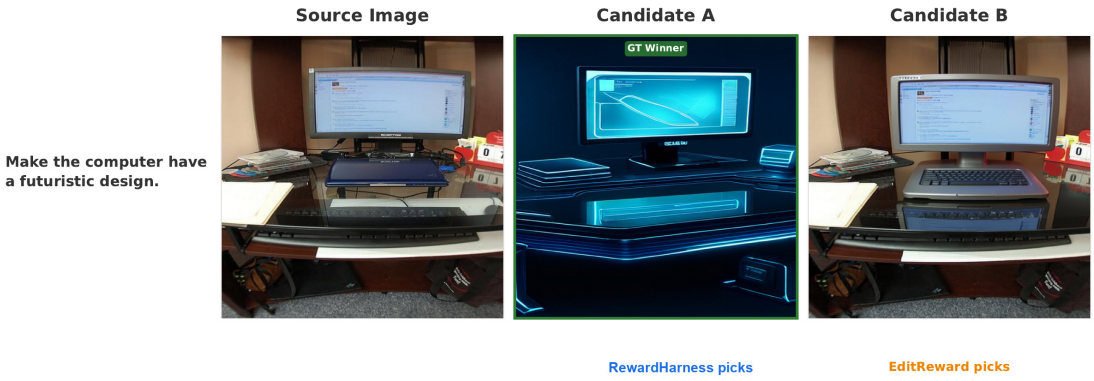


Figure 5.11: **Preference-scoring comparison on EditReward-Bench.** The figure shows a source image, an editing instruction, and two candidate edits (A and B). GT denotes the ground-truth human preference label, RewardHarness denotes our predicted preference score, and ER denotes the EditReward score. RewardHarness assigns the higher score to the human-preferred candidate (marked "GT Winner"), while EditReward fails.

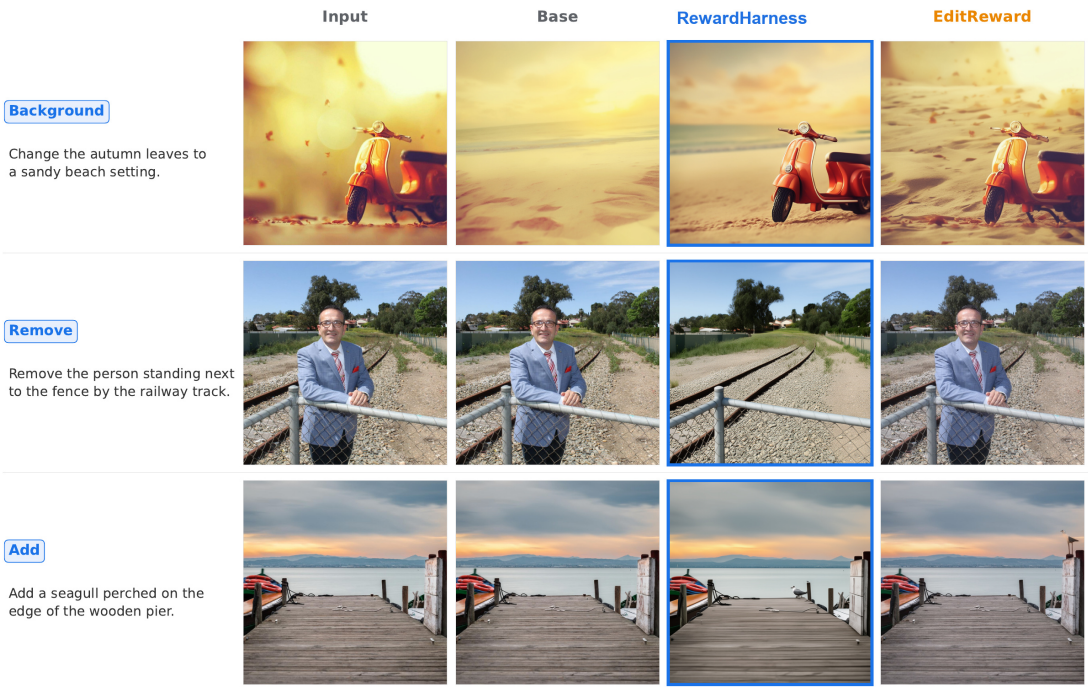


Figure 5.12: **Qualitative comparison on ImgEdit-Bench.** Each row presents a different editing task with the source image, the base model output (FLUX.2-klein-base-4B), and two RL-fine-tuned variants: RewardHarness and EditReward. RewardHarness consistently produces edits that faithfully follow the instruction while preserving visual quality and physical consistency, whereas both the base model and the EditReward-trained variant frequently fail to execute the intended edit or introduce artifacts.

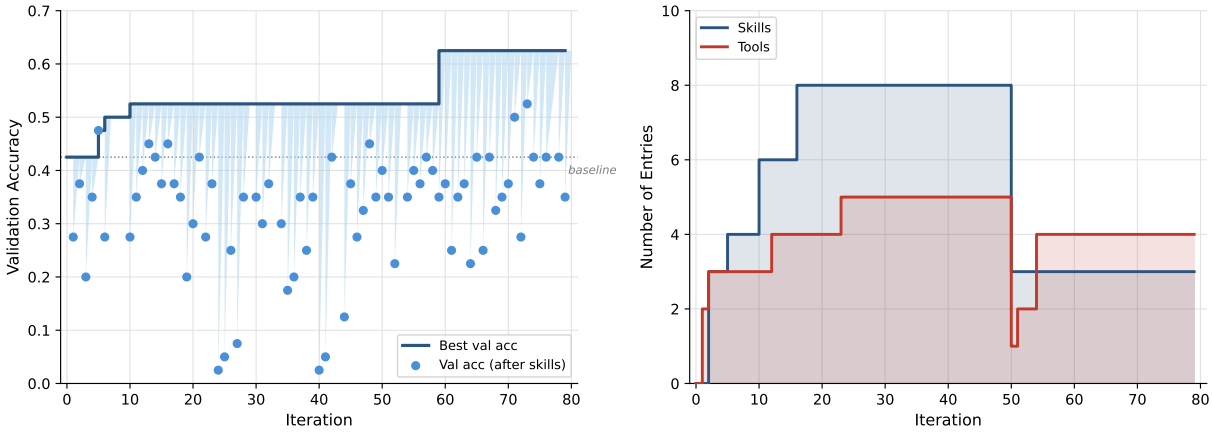


Figure 5.13: **Self-evolution dynamics over 77 iterations.** *Left:* Per-iteration (dots) and best (solid line) validation accuracy; the gating mechanism rejects proposals that fail to improve the current best, while the shaded region shows the gap between proposals and the running best. *Right:* Numbers of Skills and Tools over time. After peaking at 13 total entries (8 Skills + 5 Tools), the pruning phase begins around iter 50 and the final selected library reaches 62.5% accuracy with 7 entries (3 Skills + 4 Tools), a 47% relative improvement over the 42.5% baseline.

5.3.5 Discussion

RewardHarness uses 100 preference demonstrations to adapt how an evaluator performs an assessment without changing its parameters. Its evolved Skills define editing criteria, while Tools target details that holistic judgments may miss. With the Qwen Sub-Agent held fixed, average preference accuracy rises from 30.3 to 45.7 through the evolved evaluation context. Under the same FlowGRPO setup, it also yields a higher overall editing score than EditReward, connecting improved preference prediction to useful generator supervision.

Relative to RationalRewards, the contribution is an alternative place to store and refine evaluation knowledge: an external Library rather than evaluator weights. The Gemini results extend the approach to an API-based Sub-Agent, using a separately evolved Library rather than transferring the Qwen Library unchanged. The demonstrations supply the preference criteria, the Orchestrator selects relevant evaluation procedures, and validation gating retains Library updates that improve calibration performance. Together, these results show how a small demonstration set can organize a frozen model's capabilities into task-specific reward feedback, complementing RationalRewards' parameter-based adaptation with context-based learning.

5.4 Summary

This chapter addressed the need for robust and informative feedback when generated images fail to satisfy an instruction. RationalRewards learned structured critiques from preference data, connecting scores to explanations of instruction adherence, source preservation, and visual quality. Its experiments showed how this feedback can improve generation through both reinforcement learning and inference-time prompt refinement, allowing the same evaluator to support changes in model parameters or in the generation request. RewardHarness explored a complementary route: learning reusable evaluation skills and tools from only 100 preference demonstrations while keeping the underlying evaluator frozen. Its preference-modeling and downstream reinforcement-learning results showed that adapting evaluation procedures can provide useful rewards without retraining the evaluator itself. Across both studies, feedback was assessed not only by agreement with human preferences but also by its ability to improve generated outputs. This distinction is central to controllable generation: a plausible score or explanation is useful only insofar as it helps identify and correct a failure to meet the request. Effective feedback therefore depends on evaluation criteria that capture the requirements of the request. Chapter 6 develops this point for world knowledge and visual reasoning, where evaluating success requires checking knowledge accuracy and task-specific correctness alongside visual quality.

Chapter 6

Benchmarking World Knowledge and Visual Reasoning in Visual Generation

This chapter is based on the following works. Author contributions are described in the Statement of Contributions.

- Haozhe Wang, Weijia Feng, Jinpeng Yu, Che Liu, Ping Nie, Fangzhen Lin, Jiaming Liu, Ruihua Huang, Jimmy Lin, Wenhu Chen, and Cong Wei. *Search Beyond What Can Be Taught: Evolving the Knowledge Boundary in Agentic Visual Generation.* arXiv preprint arXiv:2607.05382, 2026. (Section 6.2)
- Xuan He, Cong Wei, Yuhao Cheng, Linrui Ma, Yuxuan Zhang, Zuojun Li, Yuhao Wen, Jize Jiang, Zeyi Liu, Yuren Hao, Songcheng Cai, Keming Wu, Penghui Du, Kai Zou, Rui Yang, Chenkai Sun, Ke Yang, Ping Nie, Kelsey R. Allen, Chenglong Wang, Michel Galley, Jianfeng Gao, and ChengXiang Zhai. *VGI-Bench: Probing Visual Intelligence in Video Generation Models.* Accepted to the Main Conference of the *Conference on Empirical Methods in Natural Language Processing* (EMNLP), 2026. (Section 6.3)

6.1 Overview

Chapter 5 showed that learned feedback can improve image generation and editing. For such feedback to guide a model toward satisfying a request, the evaluation must distinguish

successful fulfillment from a merely plausible-looking output. This distinction becomes especially important when requests depend on world knowledge or visual reasoning: an attractive image can depict the wrong entity or cultural detail, and a convincing video can violate the rules of a reasoning task or fail to reach its goal. Broad preference judgments summarize overall quality, but do not by themselves identify which of these requirements have been met. The fourth challenge is therefore to make world knowledge and visual reasoning explicit, measurable criteria for evaluating generated content.

This chapter presents two complementary benchmarks. SearchGen-Bench (Section 6.2) examines long-tailed and evolving world-knowledge requirements in image generation. Its task taxonomy, visual checklists, and controlled retrieval setting diagnose both missing knowledge and failures to use external evidence. Selective retrieval and co-training of a search agent and generator are studied as interventions motivated by these findings. VGI-Bench (Section 6.3) evaluates visual reasoning in video generation models. Given an initial image, a goal, and task constraints, it examines whether models can solve visually grounded problems through video generation. Its analyses consider task completion, rule violations, sensitivity to input conditions, and the reliability of the measured behavior.

The benchmarks thus examine two aspects of satisfying a request: using world knowledge to depict requested content and solving visual reasoning problems through video generation. Their task-specific criteria expose limitations that can be obscured by visually convincing outputs. SearchGen further investigates retrieval and co-training as responses to its benchmark findings, while VGI-Bench centers on evaluating and diagnosing visual reasoning.

6.2 SearchGen: Benchmarking World Knowledge in Visual Generation

6.2.1 Motivation

A visually plausible image can still depict the wrong entity, historical object, or cultural detail. This section studies such failures through SearchGen-20K and SearchGen-Bench, constructed from real user requests across twenty-two domains and twelve failure categories. Their visual checklists evaluate knowledge-grounded correctness rather than treating perceptual quality as sufficient evidence of task success.

The benchmark also provides a setting for testing external search. Retrieved references can supply missing information, but can introduce irrelevant structure, stylistic contami-

nation, and identity blending. The question is therefore not simply whether a generator has access to search, but when to invoke it and how to use its results.

SearchGen pairs the generator with a gate–filter–integrate reasoner that selects when to search, which content to retain, and how to construct the generation condition. It then co-trains the components: preference optimization updates the generator from search-augmented examples, and rejection-sampling fine-tuning recalibrates the reasoner to the updated generator. This tests a generator-specific, evolving boundary between information handled without retrieval and information for which retrieval remains useful.

The section first presents the benchmark and diagnoses failures with and without search. It then develops the reasoner and co-training procedure and evaluates the reported training recipe. The experiments follow one generator-update and reasoner-recalibration cycle on two generator backbones.

6.2.2 Related Work

Retrieval and visual conditioning. RAG grounds language generation in retrieved documents [84]. In visual generation, Re-Imagen and retrieval-augmented diffusion condition on external image–text evidence [10, 20]; Gen-Searcher trains an agentic searcher for a frozen image generator [40]. SearchGen studies both retrieval noise and changes to the search policy when the generator is updated.

Selective search and planning. Self-RAG, FLARE, and Adaptive-RAG make retrieval conditional on the query or generation process [2, 66, 72]. Agentic visual methods such as RPG-DiffusionMaster and IterComp use planning or feedback to improve composition [181, 204]. The present study distinguishes arranging known concepts from obtaining missing reference information, while recognizing that a request can require both. Its gate–filter–integrate pipeline also controls visual contamination after retrieval.

Evaluation and co-training. GenAI-Bench and T2I-CompBench assess prompt adherence and composition, while TIFA and Gecko develop scalable faithfulness evaluation [60, 62, 85, 162]. SearchGen-Bench complements these settings with culturally specific, long-tailed, and time-sensitive requests. It measures generation failures on these requests without assuming access to a model's complete training corpus. Building on feedback-based improvement [24, 194], the co-training experiment first updates the generator and then recalibrates the reasoner. This ordered update, together with evaluation of selective search, distinguishes the studied recipe from improving either component in isolation.

6.2.3 SearchGen-Bench: Construction and Evaluation

We examine requests whose correctness depends on specific entities, cultural conventions, or factual relationships. The benchmark distinguishes performance across knowledge demands and tests whether external references help, rather than assuming that visual plausibility establishes correctness.

6.2.3.1 What Users Actually Ask For

To ground the benchmark in real-user demand, we analyzed 20,840 prompts from a production text-to-image platform. The landscape is far more diverse, long-tailed, and rapidly evolving than existing benchmarks suggest. We discover twelve recurring failure categories distilled from real-world user requests (Table 6.1).

The taxonomy separates requests that need precise visual identity, factual text, or both. A game character may require a costume reference; a dynastic timeline requires names and dates; a culturally specific object may require visual conventions and their meaning. These differences motivate separate image and web-text search tools rather than a single retrieval modality.

6.2.3.2 Dataset Construction

From the failure taxonomy (Table 6.1), we construct SearchGen-20K spanning all twelve categories and twenty-two domains. The dataset is intentionally bilingual: English prompts (58%) average 266 characters and tend toward descriptive elaboration, while Chinese prompts (42%) average 89 characters and favor dense, implicit specification, capturing authentic cross-lingual prompt behavior rather than translated templates. The available training pool comprises 20,188 rows (20,187 unique prompts) with an average of 5.2 multimodal knowledge gaps per prompt (90.5% carry three or more), annotated with 34,694 visual reference slots, and 16,345 textual knowledge slots. For supervised training and model selection, we use a subset of this pool: 20,000 training rows and 128 held-out validation rows. The separate test set contains 751 prompts; not all available training-pool rows are used.

To construct this dataset, we first extract a seed database of 31,537 entities from production-scale user requests, each annotated with canonical names, estimated training-set frequency, ground-truth visual references (where available), and distinguishing attributes; the distribution is intentionally long-tailed, mirroring real user demand (Figure 6.1a). In this collection, 93.1% of entities appear in only a single prompt, illustrating

Table 6.1: **Taxonomy of search-intensive visual generation.** Twelve categories organized by failure source, each with a representative example prompt. The *Modality* column indicates the dominant knowledge type required.

Category	Modality	Example Prompt
Temporal – Recent	Both	"The mascot for the 2025 Osaka Expo in its official pose and colors"
Temporal – Current	Both	"Current FIFA World Cup group stage rankings as a scoreboard graphic"
Entity & IP	Visual	"Jingliu from Honkai: Star Rail wielding her ice sword"
Concept & Symbol	Both	"The national flag of Bhutan with the correct Druk dragon design"
Factual & Historical	Both	"Spartan phalanx at Thermopylae with historically accurate bronze armor"
Cultural Specificity	Both	"A traditional Oaxacan alebrije dragon with authentic chromatic patterns"
Visual / UI / UX	Visual	"iOS 17 Weather app screenshot showing a thunderstorm animation"
Data Visualization	Textual	"China dynastic timeline with accurate dates and founding emperors"
Text / Typography	Textual	"Art Nouveau concert poster with period-authentic display lettering"
Complex Composite	Both	"Aztec-style infographic of DNA replication with labeled process stages"
Vague / Abstract	Textual	"The feeling of nostalgia on a rainy afternoon in a small Japanese town"
Implicit Reasoning	Both	"Cozy mountain cabin interior in the style of a Miyazaki film"

the long tail of user demand; this statistic does not by itself establish their absence from a generator's pretraining data. From this seed database, we synthesize the SearchGen-20K dataset via human-brainstormed template-based generation and LLM-assisted rewriting.

During prompt synthesis, we deliberately employ an answer-first strategy. We ask a frontier model to select self-contained world-knowledge gaps based on the seed context. Therefore, each prompt carries *knowledge gap references* by construction, structured slots specifying what visual or textual knowledge the generator is likely to lack, each with a severity label (critical, important, moderate or minimal) and a justification field. These

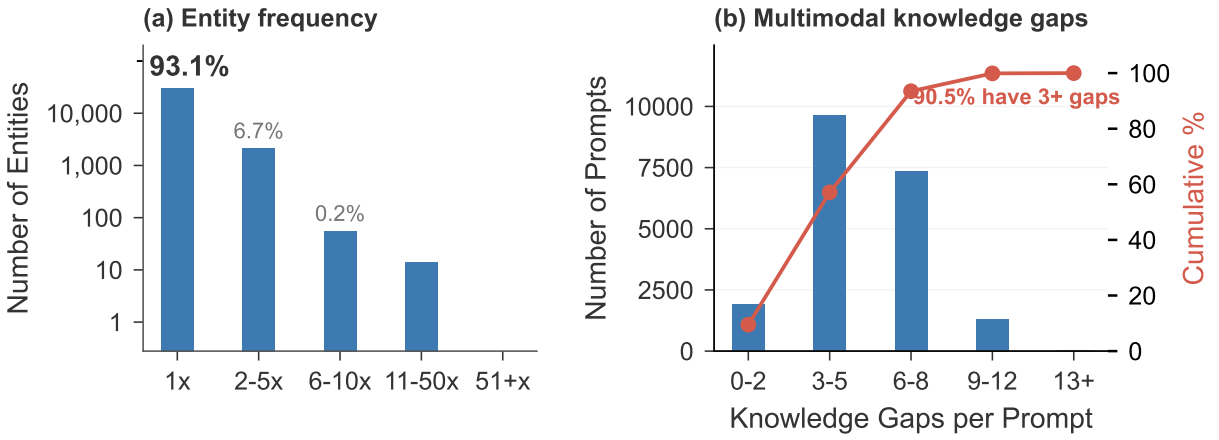


Figure 6.1: **Dataset composition: entity long-tail and multimodal knowledge gaps.** **(a)** Entity frequency distribution: 93.1% of the 31,537 unique visual entities appear in only a single prompt, confirming the extreme long-tailed nature of real-world image generation requests. **(b)** Multimodal knowledge gaps: prompts carry a mean of 5.2 simultaneous knowledge gaps (reference slots + text knowledge slots + failure modes); 90.5% carry three or more, confirming the multi-constrained nature of the dataset.

annotations ease the burden of automated evaluation.

Replayable evaluation. Table 6.2 records the released prompts, reasoning traces, generated images, and cached search corpus. Freezing the retrieved evidence allows offline comparisons without live search-result drift or repeated search API calls. It also supports reusing the same examples for preference learning and search-policy evaluation; replayability does not remove possible biases in the original retrieval or annotations.

Two annotators independently label 2,000 sampled failure cases and refine the taxonomy to Cohen's $\kappa > 0.85$ before applying it to the production prompts. Entity extraction yields 31,537 seeds across 22 domains, with canonical names, estimated frequency, visual references, and distinguishing attributes. Prompt construction combines parameterized templates, language-model rewriting, and human checks for ambiguity, cultural sensitivity, failure-category alignment, and evidence-grounded checklist items. Accepted prompts carry 2–4 search queries, 3–8 key visual elements, and category-specific rubrics.

The collection spans long-tail entities in East and South Asia, the Middle East, Africa, and Latin America. Supervised experiments select 20,000 training and 128 validation rows from the 20,188-row pool, with 751 separate test prompts; other non-test prompts support

Table 6.2: **We release SearchGen-20K, SearchGen-Corpus-1M and SearchGen-Bench which compose a replayable environment for agentic visual generation.** These assets provide agentic reasoning and generation traces collected across multiple reasoners and generators for the 20,188 training rows (20,187 unique prompt strings). **SearchGen-Corpus-1M** provides cached search sessions and downloads which emulate live web search offline, relieving the burden of costly search APIs.

Asset	**Scale**	**Enables**
Training Trajectories *(20,188 rows)*		
Reasoning traces across various frontier reasoners	96,848	search-policy / reward learning
Image Generations using different reasoning trace x generators	283,493	preference & distillation data
Search corpus *(archived, indexed)*		
Search sessions	159,027	reproducible retrieval, no live API
image / web	120,828 / 38,199	
SERP hits (unique URLs)	1,097,961	retrieval / grounding studies
Cached downloads	705,838	fully offline replay
image / web	599,859 / 105,979	

cached search and co-training.

Judge–human agreement. The human study uses 500 prompt–image pairs. Raters receive the same prompts, checklists, and category rubrics as the automated judge, score checklist items, rubric dimensions, and overall quality, and adjudicate high-disagreement items. Gemini-3-Flash achieves Spearman $\rho = 0.87$ against the consolidated ratings. Agreement is stronger for object-centric NoSearch examples than for TextualSearch cases involving glyph correctness and fine layout; automated scores on the latter should therefore be interpreted cautiously.

6.2.3.3 Evaluation Protocols

The automated evaluation separates knowledge-grounded correctness from rendering quality using two groups of score components. As Table 6.3 details, four components are *knowledge-sensitive*: *Checklist verification* tests 3–10 binary questions about key visual elements. *Rubric-adaptive scoring* applies 2–5 category-tailored weighted dimensions. *Prompt*

faithfulness measures overall adherence to the textual request, and *visual reference fidelity* measures faithfulness to provided reference images (when applicable). These evaluation criteria tailor to prompt-specific knowledge demands.

The remaining five components are *knowledge-invariant* dimensions that assess *rendering quality* (bottom half of Table 6.3): image quality, text rendering, AI naturalness, composition & aesthetics, and physical plausibility. The following example illustrates how the components assess different aspects of one generation request.

Example: Prompt-Specific Evaluation Context

Prompt: *"Paint Yang Chaoyue (a celebrity) on a 1970s Chinese rural field ridge; long-handled hoe; earth tones; 1970s film grain."*

Checklist (binary verification questions, each targeting one knowledge gap):

- Is the person recognizable as Yang Chaoyue (facial features and overall bearing)?
- Do the clothing and setting match a 1970s Chinese rural commune worker rather than modern workwear?
- Does the farm tool match the form of a traditional long-handled hoe?
- Does the image show the requested earth-tone palette and 1970s film-grain texture?

Rubric (category-specific weighted dimensions; the justification states what each dimension scores):

- `celebrity_likeness` (0.4) — how well the model preserves Yang Chaoyue's face when placed in a non-standard context.
- `historical_authenticity` (0.4) — accuracy of the 1970s attire and farming environment.
- `atmospheric_tone` (0.2) — consistency of the film grain and earth-tone color palette.

Every prompt additionally receives generic dimensions (prompt faithfulness, image quality, AI naturalness, composition).

Knowledge Gap References: Textual knowledge of the background of 1970s Chinese history. Two reference images (period labor clothing; celebrity likeness) supplied alongside the generated image.

Reasoner stages and outputs: Section 6.2.5.1.

Scoring and difficulty partitions. The judge receives the prompt, verification checklist, category rubric, and annotated reference images. References supply evidence and are not themselves scored. Each component uses a 0–3 scale in half-point increments, from contradicted or unmet to fully met. The response records an initial evidence-based anchor, justified signed adjustments, and a clamped final score. The overall score averages present components and rescales the result to 0–100; absent conditional components are

Table 6.3: **Evaluation components of the SearchGen-Bench judge.** Each component is scored 0–100; the overall score is the mean of all applicable components. Knowledge-sensitive components (top) vary per prompt and measure knowledge presence; rendering-quality components (bottom) are fixed across all prompts and measure generation competence independently of knowledge.

Component	**What it evaluates**
Knowledge-sensitive (prompt-adaptive)	
Checklist Verification	3–10 per-prompt checks on key visual elements, each scored on visual evidence
Rubric Scoring	2–5 weighted, prompt-adaptive dimensions that grade quality
Prompt Faithfulness	Presence and accuracy of all requested subjects, attributes, actions, scene, and style
Visual Reference Fidelity	Identity, attribute, and style fidelity to the prompt-requested entities (if applicable)
Rendering quality (knowledge-invariant)	
Image Quality	Clarity and sharpness; freedom from artifacts; style, lighting, and perspective coherence
Text Rendering	Accuracy, readability, spelling, and placement of in-image text; scored only when text is required
AI Naturalness	Texture realism vs. AI smoothness and uncanny uniformity; grounded vs. generic look
Composition & Aesthetics	Framing and balance; depth and spatial arrangement; color harmony; overall appeal
Physical Plausibility	Anatomy, object physics, spatial consistency, and material/lighting coherence

skipped rather than assigned zero. The component definitions are given in Table 6.3; the conditional and aggregation rules here govern their use.

The 651 search-intensive prompts are divided into difficulty terciles using Nano Banana Pro's no-search overall scores. These fixed Set I–III partitions are shared across experiments; the 100-prompt NoSearch subset is reported separately.

Conditional components are omitted rather than scored zero: visual reference fidelity requires an attached reference; text rendering is left empty when text is neither required nor present; physical plausibility is left empty for abstract or stylized content. Rubric-adaptive scoring excludes the fixed generic dimensions to avoid duplicating them. Reference images provide evidence and are not independently scored.

Third-party systems. Commercial APIs may internally use retrieval or proprietary knowledge stores. Their benchmark scores describe end-to-end products, not necessarily generators under conditioning assumptions identical to the open-weight models.

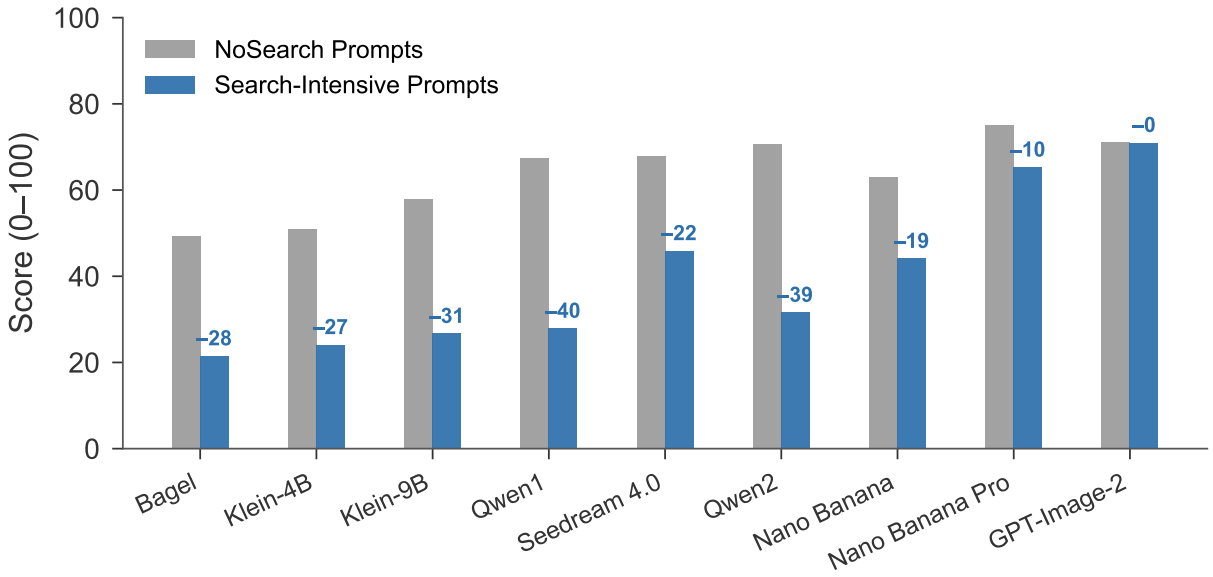


Figure 6.2: **Performance across knowledge-demand subsets for nine generators.** Grouped bars show the nine-component mean on NoSearch vs. Search-Intensive strata. All nine evaluated generators score lower on the Search-Intensive subset. Score changes range from −0.1 (GPT-Image-2) to −39.5 (Qwen-Image).

6.2.4 Benchmark Findings

With the task taxonomy and evaluation protocol established, we distinguish failures to generate the requested knowledge-grounded content from failures to use retrieved evidence. This distinction motivates the selective-search and co-training interventions that follow.

6.2.4.1 What Makes World-Knowledge-Grounded Image Generation Hard?

To examine how performance varies with knowledge demand, we curate two complementary test sets: *NoSearch* (100 prompts selected where generators perform well with parametric knowledge alone) and *Search-Intensive* (651 prompts where external world knowledge is required). Table 6.4 reports six key evaluation dimensions on the Search-Intensive slice, grouped into knowledge-sensitive components (which measure knowledge presence) and knowledge-invariant components (which measure rendering competence). Tables 6.5 and 6.6 report all nine components for both subsets.

The component breakdown helps distinguish visually plausible generation from

Table 6.4: **The knowledge bottleneck on SearchGen-Bench: Search-Intensive subset (651 prompts).** Scores on a 0–100 scale; higher is better. *Knowledge-sensitive* components (Checklist, Rubric, Visual Reference) vary per prompt and directly measure knowledge presence; *knowledge-invariant* components (Text Rendering, Physical Plausibility, Image Quality) measure rendering competence. Lower knowledge-sensitive scores than rendering-quality scores indicate difficulties depicting the requested knowledge even when visual quality is maintained.

	Knowledge-Sensitive			*Knowledge-Invariant*		
Generator	**Checklist**	**Rubric**	**Visual Reference**	**Text**	**Physical**	**Img Quality**
Open-weight generators						
Bagel [34]	18.2	17.6	13.5	2.5	36.8	30.5
Flux.2-Klein-4B [9]	19.8	18.4	11.9	4.2	46.2	37.2
Flux.2-Klein-9B [9]	24.2	23.1	16.9	7.2	48.6	36.8
Qwen-Image [164]	24.8	24.3	17.7	8.7	44.6	40.1
Commercial systems						
Qwen-Image-2 [164]	28.5	27.1	21.0	12.7	48.2	42.2
Seedream-4.0 [141]	44.2	43.6	35.1	35.9	64.0	57.0
Nano Banana [47]	41.0	40.4	33.2	28.0	65.5	57.1
Nano Banana Pro [47]	64.4	63.1	58.3	65.0	78.5	71.4
GPT-Image-2 [112]	71.2	70.1	66.0	75.9	77.3	75.1

knowledge-grounded correctness. For Flux.2-Klein-9B on the Search-Intensive subset, Checklist and Visual Reference scores are 24.2 and 16.9, compared with Image Quality 36.8 and Physical Plausibility 48.6. Tables 6.5 and 6.6 report the component scores for both subsets across the evaluated systems. These comparisons diagnose failures under the benchmark's conditions; they do not reveal proprietary model internals or isolate retrieval from all other differences between commercial systems.

Table 6.5: SearchGen-Bench: Overall and knowledge-sensitive scores. Results are reported for the NoSearch and Search-Intensive subsets on a 0–100 scale.

Generator	Overall	Checklist	Rubric	Prompt	Visual ref.
NoSearch					
Bagel	49.3	52.3	49.1	39.2	34.8
Flux.2-Klein-4B	51.0	54.7	51.8	39.3	32.6
Flux.2-Klein-9B	57.8	63.8	59.8	51.3	43.7

Continued on next page

Table 6.5 (continued)

Generator	Overall	Checklist	Rubric	Prompt	Visual ref.
Qwen-Image	67.4	74.8	70.8	62.8	56.7
Qwen-Image-2	70.7	78.7	73.8	68.4	60.0
Seedream-4.0	67.9	74.9	70.6	61.7	56.7
Nano Banana	63.1	71.1	67.5	59.7	53.8
Nano Banana Pro	75.0	82.8	78.1	72.8	67.8
GPT-Image-2	71.1	78.6	75.6	70.8	64.3
Search-Intensive					
Bagel	21.5	18.2	17.6	13.3	13.5
Flux.2-Klein-4B	24.1	19.8	18.4	12.4	11.9
Flux.2-Klein-9B	26.7	24.2	23.1	17.2	16.9
Qwen-Image	27.9	24.8	24.3	18.6	17.7
Qwen-Image-2	31.6	28.5	27.1	22.1	21.0
Seedream-4.0	45.9	44.2	43.6	38.5	35.1
Nano Banana	44.1	41.0	40.4	36.0	33.2
Nano Banana Pro	65.3	64.4	63.1	60.7	58.3
GPT-Image-2	71.0	71.2	70.1	69.2	66.0

Table 6.6: SearchGen-Bench: Rendering and appearance scores. Results are reported for the NoSearch and Search-Intensive subsets on a 0–100 scale.

Generator	Image quality	Text rendering	AI naturalness	Composition	Physical
NoSearch					
Bagel	58.0	18.4	48.3	65.3	60.4
Flux.2-Klein-4B	61.3	13.2	49.5	67.2	63.5
Flux.2-Klein-9B	63.8	31.3	52.5	72.3	66.7
Qwen-Image	68.3	63.0	61.0	76.8	73.8
Qwen-Image-2	70.2	71.7	60.8	80.3	75.6
Seedream-4.0	69.5	71.1	59.7	80.3	73.9
Nano Banana	66.3	42.8	56.5	76.3	68.5
Nano Banana Pro	71.5	85.9	65.0	83.3	78.2
GPT-Image-2	69.0	67.7	60.8	77.7	73.9
Search-Intensive					

Continued on next page

Table 6.6 (continued)

Generator	Image quality	Text rendering	AI naturalness	Composition	Physical
Bagel	30.5	2.5	29.3	33.6	36.8
Flux.2-Klein-4B	37.2	4.2	33.6	39.3	46.2
Flux.2-Klein-9B	36.8	7.2	32.9	40.4	48.6
Qwen-Image	40.1	8.7	31.6	42.8	44.6
Qwen-Image-2	42.2	12.7	36.3	45.5	48.2
Seedream-4.0	57.0	35.9	47.1	58.7	64.0
Nano Banana	57.1	28.0	47.4	61.5	65.5
Nano Banana Pro	71.4	65.0	62.0	75.9	78.5
GPT-Image-2	75.1	75.9	64.7	80.4	77.3

6.2.4.2 Naive Search Is a Second Problem

External references may supply missing information, but also introduce unwanted appearance or content. We therefore test search as an intervention on the benchmark, rather than assuming that retrieval must help.

We stress-test this pipeline with two search policies. *BlindSearch*: the reasoner searches for every nominated gap regardless of whether search actually helps. *ReasonedSearch*: a frontier VLM (Gemini-3-Flash) selectively searches for quality-relevant gaps, then picks the better of the augmented and unaugmented outputs per prompt.

To diagnose when and where each policy helps or harms, we further partition the *Search-Intensive* set into two subsets: *VisualSearch* prompt set that benefits substantially from visual references, e.g., celebrities, or *TextualSearch* that benefits from textual references, e.g., scientific infographics.

If search were uniformly beneficial, the problem would reduce to engineering better queries. Table 6.7 shows otherwise. On the *NoSearch* stratum, *BlindSearch* strictly degrades every generator: Qwen-Image-2 drops from 70.7 to 60.4, a 14.6% relative loss on prompts it already handles well. *ReasonedSearch* improves Qwen-Image-2 on *NoSearch* ($70.7 \rightarrow 76.5$) while strongly lifting *TextualSearch* from 22.9 to 34.1. Selective search improves knowledge-intensive generation while avoiding unnecessary retrieval on requests the generator already handles.

Figure 6.3 illustrates two structurally distinct failure modes. *Concept corruption* (top): search fires on a prompt the generator already handles; the returned reference overrides

Need Search?	Generator	Baseline	Reasoned	Blind
NoSearch	Qwen-Image-2	70.7	76.5	60.4
	Qwen-Image	67.4	75.0	59.5
	Flux.2-Klein-9B	57.8	66.4	52.3
VisualSearch	Qwen-Image-2	37.2	49.1	45.3
	Qwen-Image	32.8	44.4	39.7
	Flux.2-Klein-9B	29.6	39.4	36.0
TextualSearch	Qwen-Image-2	22.9	34.1	32.1
	Qwen-Image	20.5	28.5	25.3
	Flux.2-Klein-9B	18.9	26.6	24.0

Table 6.7: **Search is a double-edged sword.** We analyze three search policies on three knowledge settings.

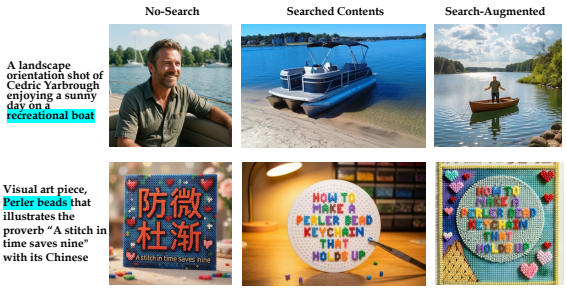


Figure 6.3: **Failure examples of search.** Search queries executed are colored. Searched contents introduce noise and may degrade generation quality.

accurate internal knowledge, a gating failure. *Copy effect* (bottom): a reference carries too much raw information and becomes a copying template rather than a knowledge supplement, a filtering failure. These two modes directly motivate the gate–filter–integrate protocol in §6.2.5.1. The generator's inability to filter search noise mirrors text-domain evidence that parametric memory and external lookup trade off by query type [106], but the visual setting adds modality-specific failure modes, including spatial distortion, identity blending, and stylistic leakage, that demand different triage and integration strategies.

6.2.5 Method

6.2.5.1 Selective Search

The benchmark's search-hurt cases motivate three stages. The *gate* identifies gaps and issues at most three image or web queries for gaps rated critical or important; otherwise it skips search. The *filter* selects evidence relevant to each gap while limiting extraneous content. The *integration* stage combines the request with retrieved facts and names the attributes to borrow from attached visual references, reducing incentives to copy unrelated backgrounds or layouts.

The gate identifies knowledge gaps, classifies severity as critical, important, moderate, or minimal, and returns search-or-skip decisions with image or web queries. The filter selects a relevant candidate by zero-based index while limiting extraneous content. Integration specifies which reference attributes to borrow and which content to create, then returns a refined prompt identifying the relevant references. Exact JSON trajectories are retained in the co-training corpus.

Operational knowledge boundary. For a fixed generator, prompt distribution, retrieval protocol, and quality measure, the knowledge boundary separates requests for which external evidence provides a meaningful quality gain from those handled without it. This is an operational comparison of outputs, not direct access to stored knowledge. A low search gain can also reflect poor retrieval or an inability to use references; updates need not improve every request. We therefore test changes through per-prompt and per-stratum measurements rather than assuming a universally expanding boundary.

6.2.5.2 Generator Update and Reasoner Recalibration

The intervention has three stages:

1. **Warm-start.** Supervised training teaches Qwen3-VL-8B the structured gate, filter, and integration outputs using approximately 10K expert trajectories selected from an available pool of approximately 20K [6].

2. **Generator update.** With the reasoner supplying search-augmented conditions, the generator samples candidate images, forms preference pairs from their evaluation scores, and undergoes online diffusion preference optimization [127, 149].

3. **Reasoner recalibration.** With the updated generator fixed, search trajectories are scored by their resulting images. Positive group-relative-advantage trajectories are retained for rejection-sampling fine-tuning of the reasoner. This can reinforce both useful retrieval and abstention.

The training settings are given below. The experiment asks whether updating both components improves knowledge-intensive prompts without harming those already handled, and whether recalibration is specific to the paired generator.

Training configuration. Reasoner SFT uses Qwen3-VL-8B-Instruct, approximately 10K trajectories selected from a roughly 20K pool, a 1% held-out split, and full-parameter training with frozen ViT. It runs for 5–6 epochs at learning rate 10^{-5} with 0.05 warm-up ratio, effective batch size 64, bfloat16, and maximum sequence length 12,288 on eight H20-80GB GPUs. The input image budget is 512^2 pixels.

Generator DPO uses Flux.2-Klein-4B with LoRA rank/alpha 64/128, learning rate 10^{-4}, AdamW, weight decay 10^{-4}, gradient norm limit 1.0, and 512×512 images. Each prompt produces five candidates; highest- and lowest-scored outputs form preference pairs.

Table 6.8: **Compute budget overview.** Training GPU-hours are approximate.

Phase	GPUs	Wall-clock	GPU-hours
Phase 0: Reasoner SFT	8× H20	<4 h	∼32
Phase 1: Generator DPO	8× H20	24 h	∼192
Phase 2: Reasoner RFT	8× H20	∼4 h	∼32
VLM Judge serving (vLLM)	1× H20	concurrent	—
Total (training)	—	—	∼256

DPO uses $\beta = 100$, guidance 4.0, 50 sampling steps, logit-normal timestep sampling, and EMA decay 0.99 updated every four steps. The training set contains approximately 7K preference pairs. A separate Qwen3-VL-8B judge service supplies scores. Flow-SDE uses noise level/time shift 1.0 and timestep truncation 0.1; a condition-SSIM penalty of 0.95 discourages near-identical pairs. Reasoner RFT retains positive group-relative-advantage trajectories against the updated generator.

6.2.6 Co-Training Experiments

We evaluate co-training along three dimensions: performance across training stages and difficulty tiers, selective use of retrieval on prompts already handled by the generator, and compatibility between the calibrated reasoner and its paired generator. These comparisons examine how generator updates and reasoner recalibration contribute to knowledge-grounded generation. Table 6.5 provides the broader component comparison.

Generators. Co-training experiments use two architecturally distinct open-weight generators: Flux.2-Klein-4B (**Klein-4B**), a flow-matching model, and Bagel-7B (**Bagel**), a unified vision–language model. Both support text-to-image and reference-conditioned generation. Using two architectures tests co-training under different conditioning mechanisms. Each has a DPO-finetuned variant (**Klein-4B-DPO** and **Bagel-DPO**) whose knowledge boundary has shifted outward through Phase 1 training.

Reasoner variants. The progression from weaker to stronger reasoners isolates the effect of generator-aware calibration: **No Search**: generator alone (no retrieval baseline). **Blind Search** (SFT-8B): Qwen3-VL-8B after supervised warm-up; the reasoner follows the protocol to perform reasoning and search but is generator-agnostic, i.e., not aware of knowledge boundary of specific generators. **Generator-Adaptive Search**: undergoes co-training based on Blind Search; the reasoner is calibrated to specific generator's spe-

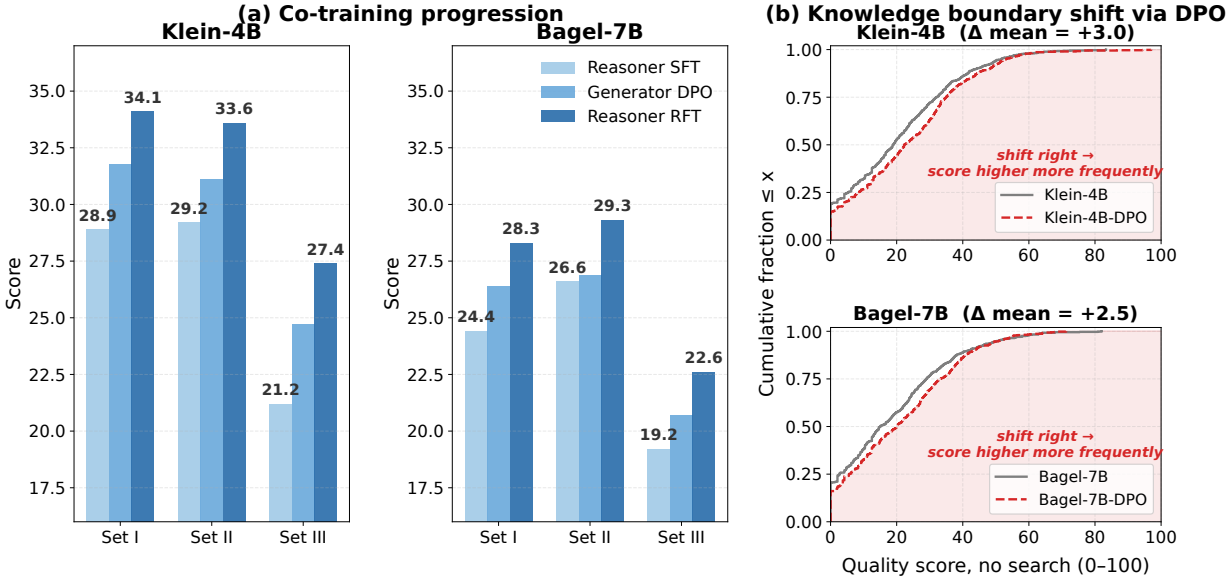


Figure 6.4: **Co-training progression and knowledge boundary shift. (a)** Grouped bars show the three co-training stages (Reasoner SFT, Generator DPO, Reasoner RFT) for Set I (easiest), Set II, and Set III (hardest) search-intensive tiers for Klein-4B and Bagel-7B. All three tiers show monotonic improvement for both generators. **(b)** Cumulative distribution of per-prompt no-search quality scores for base vs. DPO-finetuned generators on the 751-prompt test set. The DPO curve sits below the base curve, indicating a rightward shift: fewer prompts receive low scores and more receive high scores from parametric knowledge alone. The shaded region marks the difference between the score distributions. Both Klein-4B (top) and Bagel-7B (bottom) obtain higher no-search scores after DPO.

cific boundary. **Oracle (Frontier API)**: Gemini-3-Flash serves as a generator-agnostic reference reasoner using the same reasoning and search protocol.

All reasoners are equipped with search tools for world-knowledge grounding. We use Google Image and Web Search via SERP API.

Evaluation. All scores use the Gemini-3-Flash judge on the 751-prompt test set (0–100 scale, 9-component rubric averaged to an overall score; Section 6.2.3.3). The search-intensive prompts are partitioned into three difficulty sets by Nano Banana Pro no-search quality (tercile boundaries): Set I (easiest), Set II, and Set III (hardest).

6.2.6.1 Co-Training Produces Monotonic, Selective Improvement

Figure 6.4 shows how performance changes through co-training. In panel (a), Reasoner SFT, Generator DPO, and Reasoner RFT yield successive gains within each difficulty tier. Panel (b) compares the distributions of per-prompt no-search quality before and after DPO: the rightward shift indicates that both Klein-4B and Bagel-7B achieve higher scores without retrieval. Under the operational definition in Section 6.2.5.1, these results support a change in the generator's knowledge boundary. They measure changes in output quality, rather than directly identifying facts stored in model parameters.

Table 6.9 quantifies this progression. The top section traces three phases in order: Phase 0 pairs a generator-agnostic SFT reasoner with the base generator; Phase 1 keeps the same reasoner but pairs it with the DPO-strengthened generator (isolating the generator improvement); Phase 2 recalibrates the reasoner to the strengthened generator via RFT (isolating the reasoner calibration effect). The bottom section provides reference baselines that bound interpretation: no-search baselines measure performance without retrieval, frontier-reasoner baselines measure performance with a stronger generic reasoner, and a cross-check (calibrated reasoner on base generator) tests generator-specificity directly.

Three empirical patterns emerge.

Monotonicity. Across both generators, scores on all three search-intensive difficulty tiers increase through the co-training stages. On Klein-4B, Phase 2 reaches 31.8 overall on the Search partition, slightly exceeding the frontier oracle on the same generator (31.2). Generator DPO (Phase 0→1) yields +2.8; subsequent reasoner RFT (Phase 1→2) adds +2.6. The same pattern holds for Bagel (Table 6.9). On the NoSearch subset, the final recalibrated system also improves over the initial system, reaching 56.9 versus 54.6 for Klein-4B and 54.3 versus 52.6 for Bagel.

Figure 6.4(a) illustrates the successive gains within the three search-intensive difficulty tiers for Klein-4B, with the largest total gain on Set III (hardest).

Selectivity. Recall from §6.2.4.2 that naive search *degrades* prompts the generator already handles. We therefore evaluate the calibrated reasoner on the NoSearch subset as well as the search-intensive prompts. The NoSearch column in Table 6.9 reports a Phase 2 score of 56.9 for Klein-4B, a +7.0 gain over the no-search DPO baseline (49.9). This improvement is consistent with better selection of when to search on requests the strengthened generator already handles. Within the Search partition, Phase 2 improves over the no-search DPO baseline in all three difficulty sets and matches or closely approaches the frontier oracle. The same selectivity pattern holds for Bagel (Phase 2: 54.3 on NoSearch vs. 48.8 for no-search DPO; 26.8 overall).

Table 6.9: **Main results: the co-training progression.** The **Progression** section traces the three co-training phases for both Klein-4B and Bagel. The **Reference baselines** section reports the no-search and frontier-reasoner baselines, together with the cross-generator check for each architecture. Difficulty terciles (Set I–III) by Nano Banana Pro no-search quality; NoSearch subset (100 prompts) reported separately.

		NoSearch	Search			
			Set I	Set II	Set III	Overall
Co-training progression						
Klein-4B						
Phase 0	Blind Search (SFT-8B) + Klein-4B	54.6	28.9	29.2	21.2	26.4
Phase 1	Blind Search (SFT-8B) + Klein-4B-DPO	54.0	31.8	31.1	24.7	29.2
Phase 2	Generator-Adaptive Search (RFT-8B) + Klein-4B-DPO	**56.9**	**34.1**	**33.6**	**27.4**	**31.8**
Bagel						
Phase 0	Blind Search (SFT-8B) + Bagel	52.6	24.4	26.6	19.2	23.4
Phase 1	Blind Search (SFT-8B) + Bagel-DPO	52.4	26.4	26.9	20.7	24.7
Phase 2	Generator-Adaptive Search (RFT-8B) + Bagel-DPO	**54.3**	**28.3**	**29.3**	**22.6**	**26.8**
Reference baselines						
Klein-4B						
	No Search + Klein-4B-DPO	49.9	28.2	26.3	20.6	25.0
	Oracle + Klein-4B-DPO	55.7	33.7	33.9	26.0	31.2
	Generator-Adaptive Search (RFT-8B) + Klein-4B	54.6	29.0	29.8	21.5	26.8
Bagel						
	No Search + Bagel-DPO	48.8	24.3	24.0	18.8	22.4
	Oracle + Bagel-DPO	53.2	28.1	29.0	21.1	26.1
	Generator-Adaptive Search (RFT-8B) + Bagel	52.5	25.1	27.3	19.8	24.1

Generator-specificity. The cross-check pairs the same calibrated reasoner with the base and DPO-strengthened generators to examine generator–reasoner compatibility. In Table 6.9, Generator-Adaptive Search paired with *base* Klein-4B scores 26.8 overall on the Search partition (vs. 31.8 when paired with Klein-4B-DPO, the generator it was calibrated for). The corresponding scores for Bagel are 24.1 and 26.8. Within the DPO-strengthened generator, reasoner recalibration raises Search overall performance from 29.2 to 31.8 for Klein-4B and from 24.7 to 26.8 for Bagel. With the base generator, the same reasoner yields smaller gains over the SFT reasoner: from 26.4 to 26.8 for Klein-4B and from 23.4 to 24.1 for Bagel. These results support adapting the search policy to the generator with which it is used.

Scope of the evidence. The phase comparisons, NoSearch recovery, and cross-generator pairing examine complementary aspects of recalibration. They demonstrate generator-specific calibration across the two evaluated backbones, with comparisons designed to test search selection, adaptation, and generator–reasoner compatibility.

6.2.7 Discussion

SearchGen shows why knowledge-grounded generation requires evaluating more than visual plausibility or the availability of search. A generator may lack the requested knowledge, but retrieved references can also disrupt content that it would otherwise render correctly. The benchmark's knowledge strata and replayable search environment make both effects visible. Selective search responds to this distinction by deciding when external evidence is useful, while the co-training study tests how that decision changes after the generator improves. The cross-generator pairing results are particularly informative: a reasoner calibrated to an updated generator loses effectiveness when paired with the base generator. Search policy must therefore account for the capabilities of the generator that will use its evidence.

This finding makes the benchmark a diagnostic tool for controllable generation, connecting knowledge failures to specific retrieval and training interventions. The co-training evidence comes from one generator-update and reasoner-recalibration cycle on the tested 4B and 7B models. Difficult textual knowledge, reference contamination, and automated-judge errors delimit the interpretation of these results. SearchGen's contribution is to make the generator's knowledge boundary measurable and to use that boundary to guide selective search and adaptation. VGI-Bench next extends the evaluation question from whether an output depicts the requested knowledge correctly to whether a generated video reaches a goal through a valid process.

6.3 VGI-Bench: Probing Visual Intelligence in Video Generation Models

6.3.1 Motivation

A video can depict a plausible final configuration without showing a valid way to reach it. A maze solver may cross a wall, a folding sequence may change its inventory of faces, or

an object may disappear during manipulation. Such failures matter when video generation expresses a solution, not just convincing motion. Recent studies of generative visual reasoning motivate testing this distinction [18, 146, 161].

VGI-Bench evaluates goal-directed procedures using realistic inputs, explicit rules, and graded difficulty. It separates progress toward a goal from violations along the trajectory. Tasks are calibrated to short video generation and reviewed for clarity and partial feasibility. Table 6.10 situates these choices among existing benchmarks.

Bench	Reasoning Demand	Appearance Abst., Real.	Process-sensitive No, Yes	Difficulty Control
PhysGenBench	Low / Mid			
WorldSimBench	Low / Mid			
TiViBench†	High			
V-ReasonBench	High			
VBVR-Bench	High			
Ours	High			

†Data has not been publicly released and statistics are inferred from the paper.

Table 6.10: Comparison with related video benchmarks. Each pie encodes the fraction of tasks that satisfy a desideratum versus do not. For input appearance, Real. denotes photorealistic-style inputs, while Abst. denotes synthetic inputs, like line-art or schematic images. Section 6.3.2 details the ratio assignment.

Beyond aggregate performance, we ask whether supplying a solution or changing visual style alters execution, how synthetic-task fine-tuning transfers to realistic tasks, and whether denoising revisions correct visible solutions. These are behavioral tests under specified conditions, not evidence that plausible videos establish a general reasoning mechanism.

6.3.2 Related Work

Video generation has progressed from content synthesis toward tests of visual simulation and goal-directed behavior [78, 142, 146, 161, 183]. TiViBench, V-ReasonBench, and MMGR assess generative reasoning across spatial, physical, and logical tasks [15, 18, 100];

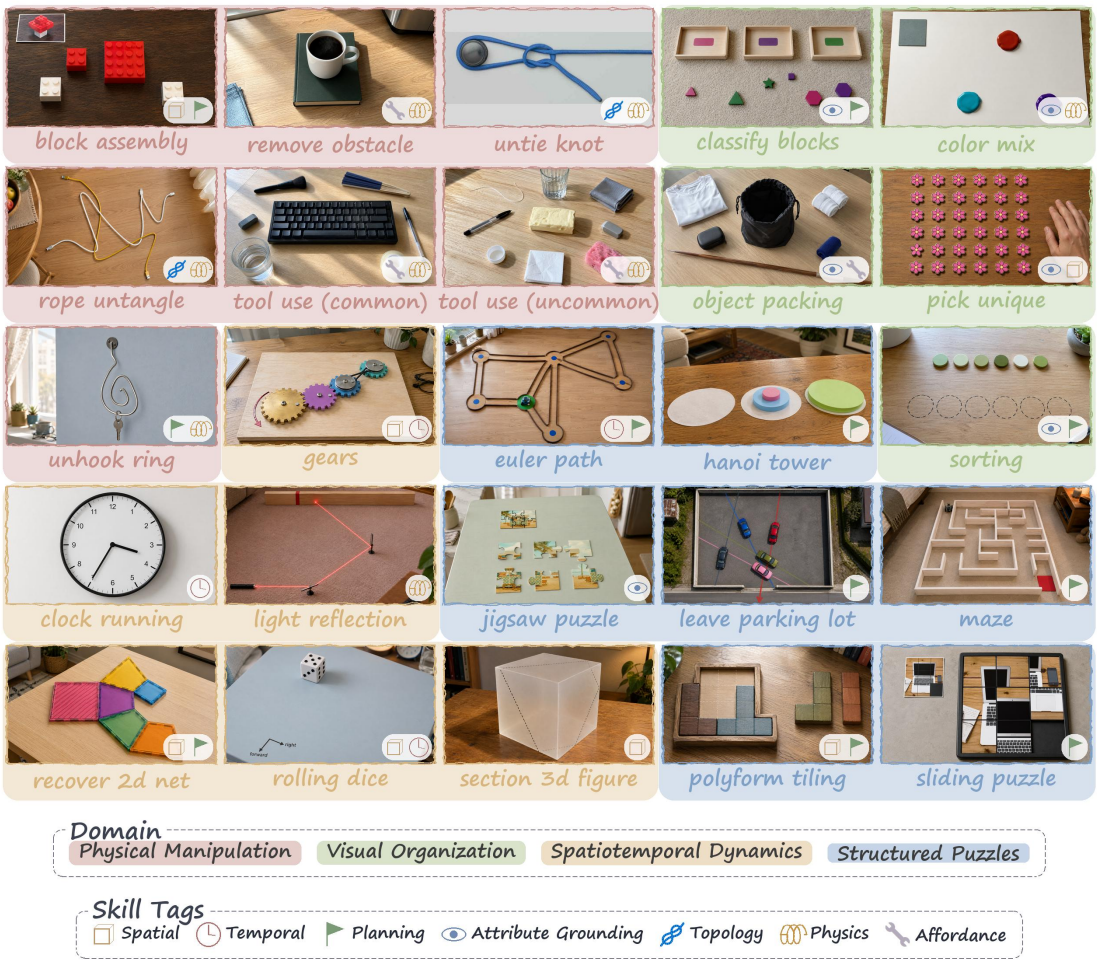


Figure 6.5: Overview of representative tasks in **VGI-Bench**. The benchmark adopts a double-level taxonomy: each task belongs to one mutually exclusive task domain and receives one or more non-exclusive skill tags. Each panel shows a representative real-scene input image, with icons indicating its skill tags.

VBVR also supplies synthetic task supervision for adaptation [154]. VGI-Bench emphasizes realistic, process-sensitive procedures with feasibility checks and graded difficulty.

Table 6.10 compares input appearance, process sensitivity, and difficulty control. Released datasets are inspected directly; unreleased datasets are characterized from their papers. Process sensitivity requires nontrivial valid transitions, even when the endpoint is easy to check. Feasibility calibration and explicit difficulty levels are distinct criteria.

6.3.3 Benchmark Design and Evaluation

6.3.3.1 Taxonomy and Task Collection

The 27 tasks belong to four mutually exclusive domains: Visual Organization (arranging, grouping, or selecting objects), Physical Manipulation (plausible object interactions), Structured Puzzles (rule-governed transformations), and Spatiotemporal Dynamics (state evolution and temporal dependencies). Non-exclusive tags identify Spatial, Temporal, Planning, Attribute Grounding, Physics, Topology, and Affordance skills.

Each instance supplies a first-frame image, a textual goal and constraints, and evaluation standards. Tasks have three difficulty levels with roughly ten instances per level. Inputs are adapted from web images and datasets or generated with human review [46, 113]; all use a 16:9 aspect ratio. Prompts specify permitted actions, prohibited shortcuts, state changes, and background/camera constraints. Image or textual references describe acceptable outcomes where available.

Quality control. Two easiest-level instances of each proposed task are tested on several video models. Acceptance requires evidence of both success and failure among tested models; otherwise materials or task design are revised. This filters for partly feasible, nontrivial procedures, rather than proving full solvability. Manual review checks goal fidelity, prompt clarity, and compatibility with roughly 5–10-second clips.

Coverage and an example standard. The 27 tasks comprise eight Structured Puzzles, five Visual Organization, seven Spatiotemporal Dynamics, and seven Physical Manipulation tasks. Skill tags are non-exclusive. Input review checks objects, attributes, layout and artifacts before accepting each instance. For maze navigation, raw completeness tiers 0, 1 and 2 map to 0, 0.5 and 1: no valid progress, legal partial progress without reaching the goal, and reaching/stopping at the goal. The rubric separately checks unchanged walls and

layout, continuous corridor motion without teleportation or duplicate cars, valid turns, no wall crossing, and stopping on the goal after travelling from the start. Touching or briefly overlapping a wall outline without crossing is not counted as a crossing. Thus reaching a goal and following the task rules are assessed separately; the final score combines them per instance. Other tasks use task-specific standards.

6.3.3.2 Goal Completion and Process Validity

Completeness maps a task-specific failed, partial, or complete judgment to 0, 0.5, or 1. **Rubric Score** checks local constraints, assigning item i with x_i violations the score $1/(x_i + 1)$ and averaging over all N items:

$$\text{Rub} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{x_i + 1}, \qquad \text{Final} = \text{Comp} \times \text{Rub}. \tag{6.1}$$

A static output may preserve rules but make no progress; an apparent solution may reach its goal through forbidden moves. Multiplication requires both completion and process adherence. Reported scores average per-instance products, not products of separately averaged metrics.

Completeness uses a global view; the rubric judge inspects local windows and revisits suspicious intervals at a higher sampling rate. An image-free aggregation pass deduplicates evidence before counting violations.

Both passes use `gemini-3-flash-preview`. Available image/text references are routed per task; absent references are omitted. Completeness receives the input, available ground truth, ordered frames, and tiered standard, returning a tier and rationale without inventing unobserved events. The global pass samples at 2 fps.

Local inspection and counting. The rubric pass receives the checklist and timestamped frames, returning localized comments and suspicious frame indices rather than final counts. Coarse inspection samples the video at 4 fps using 10-frame sliding windows with a one-frame overlap between consecutive windows. Flagged intervals are reinspected at 8 fps.

An image-free polish pass receives all coarse/refined comments and the checklist, merges persistent events, overlapping-window duplicates, and adjacent reports within 0.5 seconds, and returns nonnegative per-item counts with evidence. Counts yield the rubric score defined in Section 6.3.3.2. Domain scores pool all relevant instances. Strict success requires both component scores to be perfect.

6.3.4 Experiments

The experiments assess task completion and rule violations separately, then check the reliability of the automatic evaluation. The diagnostic studies that follow examine how input conditions, fine-tuning, and denoising affect the observed reasoning behavior.

6.3.4.1 Setup

Our evaluation covers closed-source and open-source video models. Due to the evaluation cost, we evaluate half of the instances for each task using a fixed random seed. A full-instance check on five models and six tasks across all four domains changes average scores by less than five percentage points and preserves model ranking on that slice; this does not establish stability for every model–task pair. Following the criteria in Section 6.3.3.2, we use Gemini-3-Flash [45] for our automatic evaluator. Table 6.12 reports the generation configurations.

Category	**Level**	*Commercial*						*Open Source*		
		Sdce2.0	**Sora2**	**Veo3.1**	**Kling3.0**	**Wan2.7**	**Gen4.5**	**Mnx-H3**	**HY1.5**	**Wan2.2**
Visual Organization	Easy	72.6	55.6	50.3	67.6	37.0	60.5	42.8	26.6	30.4
	Mid	56.4	37.0	43.3	47.5	32.2	38.3	42.3	17.9	15.3
	Hard	53.5	39.1	44.0	43.7	32.5	41.9	39.8	24.7	18.5
	Avg.	60.8	43.9	45.9	52.9	33.9	46.9	41.6	23.1	21.4
Spatiotemporal Dynamics	Easy	54.6	45.3	28.1	45.2	38.9	39.6	44.3	34.7	38.7
	Mid	47.4	34.6	21.2	35.2	42.6	34.6	40.7	27.2	32.0
	Hard	33.4	24.4	16.9	28.2	28.9	29.5	35.9	23.9	19.9
	Avg.	45.3	34.8	22.3	36.5	36.8	34.8	40.3	28.7	30.2
Structured Puzzles	Easy	46.9	40.5	31.8	45.3	25.6	29.2	51.9	9.9	17.1
	Mid	45.1	25.8	21.8	38.3	23.3	20.7	52.3	8.8	8.4
	Hard	41.8	22.4	13.5	28.9	24.7	18.7	47.6	7.9	5.6
	Avg.	44.6	29.5	22.4	37.5	24.5	22.9	50.6	8.9	10.4
Physical Manipulation	Easy	64.4	47.9	49.6	60.6	55.8	52.1	58.6	22.5	33.7
	Mid	59.2	41.8	40.4	51.2	48.2	43.4	41.5	16.7	22.1
	Hard	44.4	32.6	37.3	45.6	37.1	39.5	33.0	11.9	17.5
	Avg.	56.0	40.8	42.5	52.5	47.1	45.0	44.4	17.0	24.4
Overall		51.0	36.7	32.0	44.0	35.7	36.6	44.4	19.1	21.6

Table 6.11: Evaluation results of video generation models, the best and second-best scores are highlighted. HY1.5 for HunyuanVideo-1.5, Sdce2.0 for Seedance2.0, Mnx-H3 for MiniMax-H3.

Model	Resolution	Duration (s)	FPS	Price (USD / s)
		Open Source		
HY1.5	1280×720	5/10	24	–
Wan2.2	1280×720	5/10	16	–
Mnx-H3	960×544	5/10	24	–
		Commercial		
Sora2	1280×720	8	30	0.10
Veo3.1	1280×720	8	24	0.20
Sdce2.0	1280×720	5/10	24	0.15
Kling3.0	1280×720	5/10	24	0.08
Wan2.7	1280×720	5/10	30	0.10
Gen4.5	1280×720	5/10	24	0.12

Table 6.12: Generation configuration of the video models we evaluate, resolution defaults to 1280 × 720. Price is the per-second generation cost. HY1.5 for HunyuanVideo-1.5, Sdce2.0 for Seedance2.0, Mnx-H3 for MiniMax-H3.

6.3.4.2 Results

Commercial video models generally lead, but all remain far from solved. Table 6.11 reports the video generation model results. Commercial models generally outperform open-source models, with Seedance-2.0 achieving the best overall score (51.0), although the open-weight MiniMax-H3 ranks second (44.4); nevertheless, all tasks remain far from solved. **Structured Puzzles** is the most challenging domain, exposing failures in multi-step rules and state tracking. Figure 6.6 further shows that **Topology** and **Temporal** are the weakest skill dimensions (one task may have several skill tags), reflecting failures in connectivity preservation and multi-step state tracking. Strict success requires perfect completeness and rubric scores. Table 6.13 reports both component means, the mean of per-instance products, and strict success. Multiplying component means need not reproduce the final score. Human task success is 97.1%, while the best video-model strict success is 14.0%; these are not identical response protocols. The human participants annotate images or explain solutions orally after brief instructions and examples. Their time budget uses the 90th percentile of average pilot response times across task families; half of each difficulty is sampled. Human responses are judged for binary task success rather than video-specific violations. The result estimates solvability under these conditions, not a population-wide ceiling for video generation.

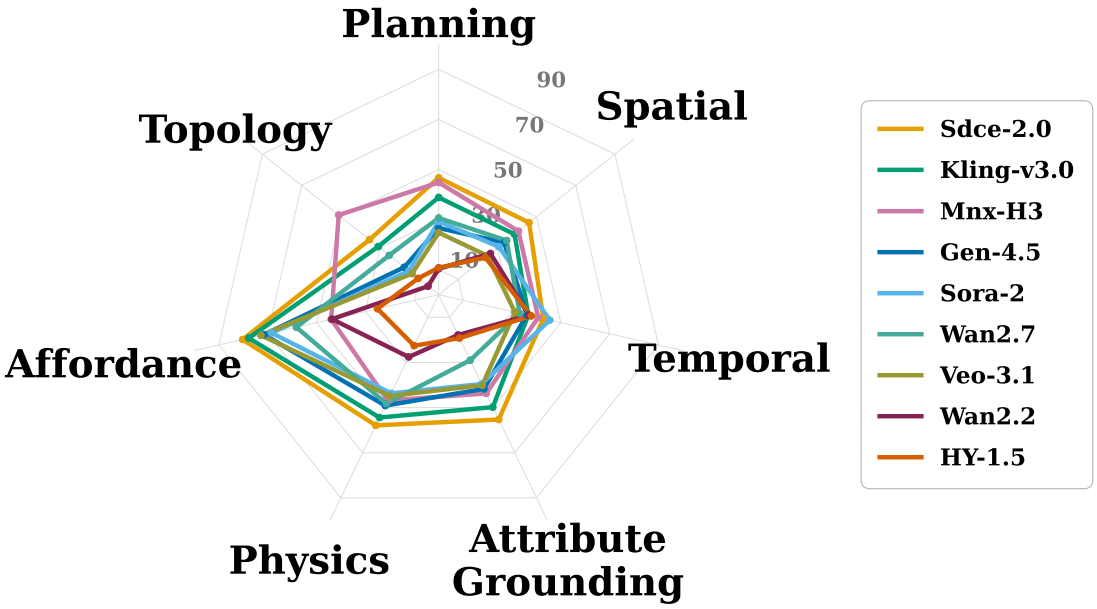


Figure 6.6: Per-model performance across the skill tags.

Table 6.13: Overall VGI-Bench component means, final score and strict success (%). Final is the mean of per-instance products, not the product of component means.

Model	Completeness	Rubric	Final	Strict success
Seedance2.0	70.5	68.6	51.0	14.0
Sora2	52.2	64.0	36.7	6.2
Veo3.1	46.9	60.6	32.0	5.3
Kling3.0	63.0	66.3	44.0	9.5
Wan2.7	50.8	64.3	35.7	7.9
Gen4.5	54.1	61.9	36.6	5.3
MiniMax-H3	56.5	74.2	44.4	12.1
HunyuanVideo-1.5	29.0	61.8	19.1	1.0
Wan2.2	32.8	59.8	21.6	2.3
VBVR-Wan2.2	52.1	73.3	40.2	11.0

6.3.4.3 Evaluation Reliability

The reliability of our VLM-based evaluator is assessed by comparing it against human annotations and ablating the key components: adaptive frame sampling and the sliding focus window. Also, we report the ablation results of base model with comparable pricing

level. Table 6.14 reports AUC and pairwise accuracy against human preferences for the full method and its two ablations. The full evaluator achieves the strongest agreement with human judgments, while removing either component leads to degradation, confirming the necessity of both.

Four annotators rank model-video triplets for task progress, rule compliance, and correctness, with two annotators per triplet. Ties and skips are allowed. Triplets become pairwise preferences; contradictory annotations and preference cycles are removed, leaving 1100+ pairs. These filtering choices define the evaluated pool.

Table 6.14 reports the aggregate judge and component ablations. Pairwise agreement treats score differences below 0.05 on the normalized scale as ties. ROC-AUC uses score differences on strict human preferences without that threshold.

Pairs are grouped by task domain and all assigned skill tags; non-exclusive tags cause some pairs to enter multiple skill pools. Agreement varies across slices, with roughly 0.73–0.85 AUC and 69–80% pairwise accuracy. Physical Manipulation, Topology, and Physics warrant particular caution: sampled frames can miss subtle interactions or connectivity changes. Aggregate agreement does not eliminate these judge limitations.

Judge Design	**AUC**	**Pairwise Acc.**
Main (w/ Gemini-3-Flash)	**0.803**	**73.2%**
w/o adaptive fps	0.772	69.5%
w/o focus window	0.753	68.3%
w/ GPT-5-mini	0.690	64.3%
w/ Claude-Haiku-4.5	0.478	47.9%
w/ Qwen3.6-Plus	0.624	59.1%

Table 6.14: Reliability of our VLM-based evaluator compared with human annotations, including key component and base model ablations.

6.3.5 Diagnostic Findings

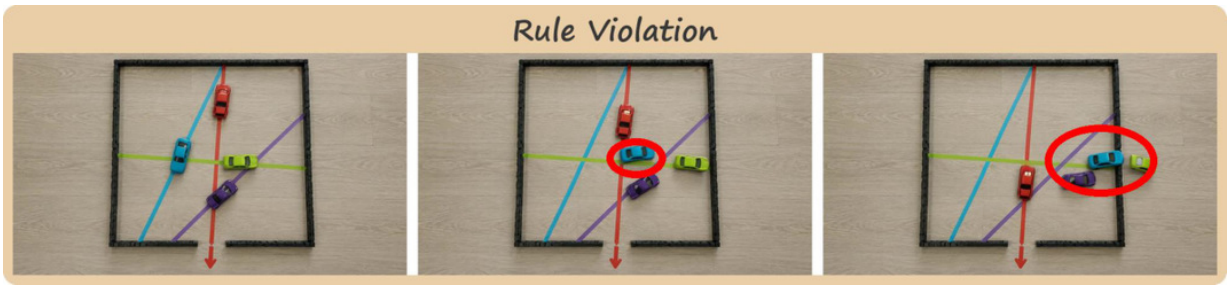
6.3.5.1 Failure Modes

Figure 6.7 distinguishes three recurring failures. *Physical collapse* violates plausible dynamics; *rule violations* reach a goal through prohibited actions or shortcuts; *object/state inconsistency* loses identities, duplicates objects, or resets intermediate states. Smooth

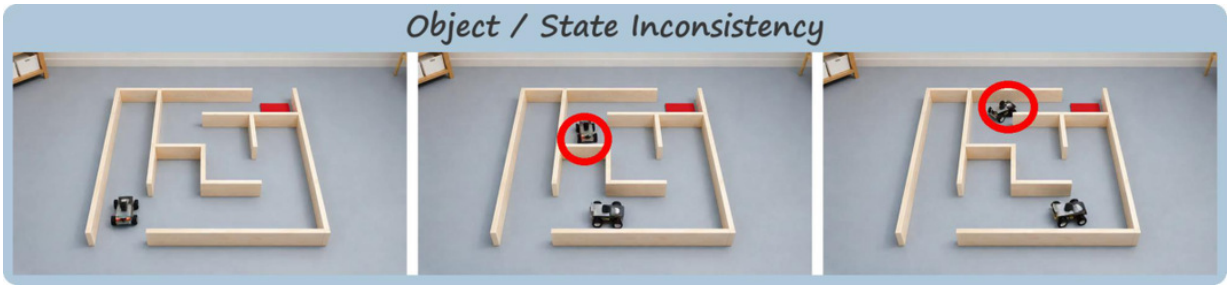
motion or a plausible endpoint can coexist with these errors. The distinction motivates evaluating the trajectory rather than accepting final-frame resemblance.



(a) **Physical collapse.** The laptop is tilted up, yet **the cup lying on its lid does not roll down**.



(b) **Rule violation.** Each car must stay on its own colored track and may not cross walls; **the video violates both rules**.



(c) **Object/state inconsistency.** While the car drives through the maze, **a spurious second car appears partway through**.

Figure 6.7: Representative cases in the failure modes.

6.3.5.2 Input Conditions: A Solution Is Not Enough

Oracle prompts supply a ground-truth step-by-step solution while retaining the task rules. On the tested seven-task subset, gains are limited even when the intended solution is supplied, especially for HunyuanVideo-1.5. Models can still fail to execute valid trajectories: specifying the answer does not ensure precise spatiotemporal instruction following. The experiment does not fully isolate reasoning from rendering because some solutions are difficult to express unambiguously in language.

Paired realistic and line-art inputs yield different rankings, particularly among open-source models (Figure 6.8). Visual-domain mismatch can therefore affect apparent reasoning performance.

Oracle prompts prepend an executable ground-truth solution to the canonical task goal and rules for seven tasks. Controlled comparisons use HunyuanVideo-1.5, Veo3.1, and Wan2.7; proprietary prompt rewriting complicates comparisons where enhancement cannot be configured. This reduces the burden of finding a solution without removing constraints on its visual execution.

Style comparisons hold tasks fixed and replace realistic inputs with line-art variants. The five models are Sora2, Veo3.1, Kling3.0, Wan2.2, and HunyuanVideo-1.5. The seven tasks are CLOCK_RUNNING, HANOI_TOWER, JIGSAW_PUZZLE, LEAVE_PARKING_LOT, MAZE, PICK_UNIQUE, and POLYFORM_TILING.

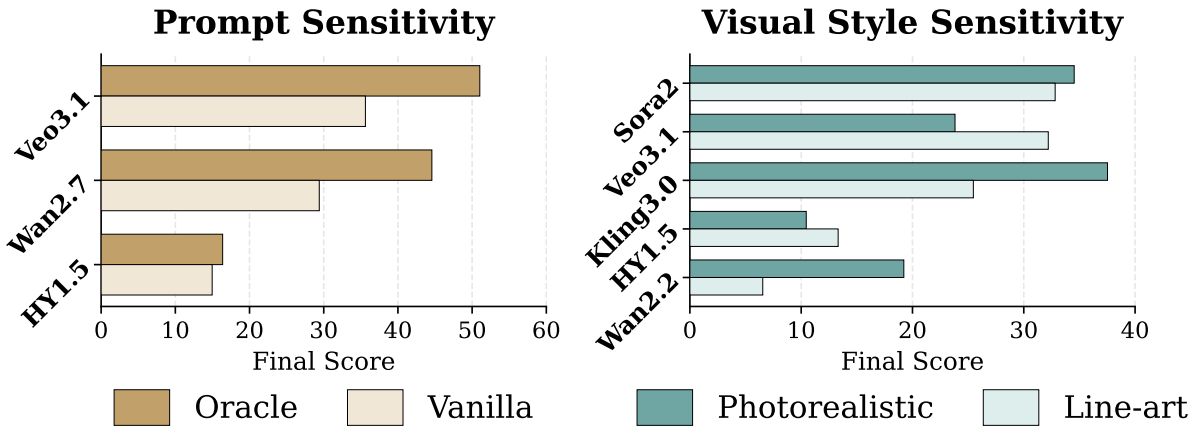


Figure 6.8: Performance comparison in oracle prompting and in varying input visual styles.

6.3.5.3 Synthetic Fine-Tuning: Transfer Depends on Coverage

We compare released VBVR models fine-tuned on a 1M-sample abstract-style dataset with their Wan2.2-I2V-A14B, Wan2.1-I2V-14B, and LTX-2.3 bases [154]. Tasks are grouped by structural overlap with the training distribution.

Model	Overlap	Semi-overlap	Non-overlap
Wan2.2-I2V (base)	15.3	17.8	29.2
VBVR-Wan2.2	**55.8** ↑40.5	**34.9** ↑17.1	**35.4** ↑ 6.2
Wan2.1-I2V (base)	11.3	17.0	25.6
VBVR-Wan2.1	**25.1** ↑13.8	**20.2** ↑ 3.2	22.0 ↓ 3.6
LTX-2.3 (base)	15.0	18.2	15.5
VBVR-LTX2.3	**24.1** ↑ 9.1	**23.6** ↑ 5.4	**19.2** ↑ 3.7

Table 6.15: Performance of the three released VBVR models and their respective base models, grouped by structural overlap with training distribution. Per-cell values are mean final score across tasks in the group.

Group-level gains decrease as structural overlap with the training tasks decreases, but overlap does not guarantee improvement on an individual task. Planning and spatial skills benefit more than physical interaction or strong temporal dependencies, which can degrade. For UNTIE_KNOT, rubric score rises from 0.02 to 0.34 with little change in strict success: fewer violations need not mean that the goal is solved. Overlap tasks share goals and structures with VBVR but differ in visual domain; semi-overlap tasks share only some structures or action patterns; non-overlap tasks have no clear counterpart. Individual counterexamples remain: clock-running scores fall from 32.5 to 10.0 despite overlap, whereas untie-knot scores rise from 1.8 to 34.0 despite non-overlap. The findings support bounded transfer, not uniform scaling of general visual reasoning.

6.3.5.4 Denoising Revisions Rarely Correct the Solution

To distinguish revision from self-correction, we decode intermediate states of four open-source models and label transitions as unrecognizable, stable, or changed. Changed states are divided into correct-to-wrong, wrong-to-correct, and wrong-to-another-wrong; only wrong-to-correct counts as self-correction.

We examine Wan2.2-I2V, VBVR-Wan2.2, HunyuanVideo-1.5, and LTX2.3 with their default 40-step settings. The sampled intervals are $1 \to 2$, $2 \to 3$, $3 \to 4$, $4 \to 10$, $10 \to 20$,

Model	Structured Puzzles	Visual Organization	Spatiotemporal Dynamics	Physical Manipulation	Overall
Wan2.2-I2V	11.6	21.4	30.2	24.4	21.5
VBVR-Wan2.2	53.2 ↑41.6	37.6 ↑16.2	29.0 ↓1.2	42.6 ↑18.1	41.2 ↑19.7

Model	Planning	Spatial	Temporal	Attribute Grounding
Wan2.2-I2V	10.4	26.4	37.0	17.8
VBVR-Wan2.2	44.1 ↑33.7	37.3 ↑10.9	25.8 ↓11.2	37.9 ↑20.1

Model	Physics	Affordance	Topology
Wan2.2-I2V	27.6	43.7	5.4
VBVR-Wan2.2	39.9 ↑12.3	55.2 ↑11.5	26.3 ↑20.9

Table 6.16: Per-domain (top) and per-skill (bottom) performance (final score) breakdown for VBVR-Wan2.2 vs. base Wan2.2-I2V.

and $20 \rightarrow 40$. Decoding 117 task instances produces $117 \times 6 = 702$ annotated pairs; each reported interval column sums to 100%.

Each pair has one mutually exclusive label: unrecognizable if either state is too unresolved to identify a solution; stable if the readable solution is unchanged; otherwise correct-to-wrong, wrong-to-correct, or wrong-to-another-wrong. Only wrong-to-correct is self-correction. These labels concern decoded behavior rather than latent representations. Early unresolved states and low overall task success limit what can be inferred about internal reasoning from the observed transitions.

Self-correction remains below 1% in every interval and is absent after step 4. In contrast, wrong-to-another-wrong transitions reach 23.1% at $4 \rightarrow 10$ and 24.8% at $10 \rightarrow 20$, while stability reaches 90.6% in the second half. Later changes therefore need not improve a solution and often refine an already incorrect hypothesis. This cautions against equating video denoising with reliable self-correction [110, 211]. The observation concerns visible decoded states on these models and tasks, not every possible computation in latent space.

Transition	1→2	2→3	3→4	4→10	10→20	20→40
♠ Unrecognizable	69.2	41.9	22.2	6.8	0.0	0.0
♣ Stable	17.9	49.6	70.1	69.2	75.2	90.6
■ (a) Correct → Wrong	2.6	0.9	0.0	0.9	0.0	0.0
■ (b) Wrong → Correct	0.9	0.0	0.9	0.0	0.0	0.0
■ (c) Wrong → Wrong′	9.4	7.7	6.8	23.1	24.8	9.4

Table 6.17: Distribution (%) of solution-state transitions between consecutive decoded denoising steps. Row ■ (b) (shaded) is self-correction. Protocols are in Section 6.3.5.4.

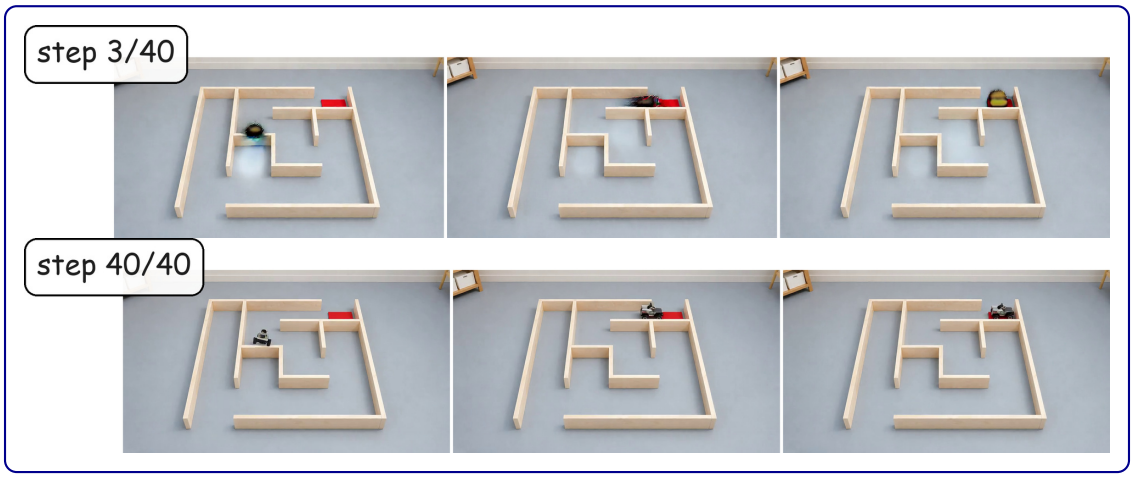


Figure 6.9: Decoded maze frames at different denoising stages.

6.3.6 Discussion

VGI-Bench makes the validity of a generated process part of the evaluation target. A plausible final frame can conceal a wall crossing, an impossible physical event, or an object that disappears along the way. Separating goal progress from rule violations exposes these failures even when the video looks coherent. The diagnostic studies reinforce this distinction: supplying a solution does not ensure its execution, synthetic-task fine-tuning transfers unevenly, and visible revisions during denoising rarely correct an incorrect so-

lution in the tested models. Together, these findings show why visual fluency alone is insufficient evidence of goal-directed reasoning.

The benchmark turns that distinction into task-specific criteria for assessing whether a generator fulfills a request. Its representative task suite evaluates short, roughly 5–10-second image-to-video procedures using fixed 16:9 inputs and English prompts and rubrics. Within this defined setting, separating task progress from rule compliance makes it possible to distinguish an incomplete solution from an apparently successful outcome reached through invalid steps. VGI-Bench thus complements SearchGen's tests of knowledge-grounded content with tests of procedural correctness.

6.4 Summary

This chapter addressed evaluation blind spots beyond visual quality by examining world knowledge and visual reasoning directly. SearchGen-Bench translated long-tailed and evolving knowledge requirements into task-specific checklists and reference-grounded criteria. Its evaluations distinguished missing knowledge from rendering limitations and showed why external search must be selective: retrieved evidence can help knowledge-intensive requests but can also introduce irrelevant or misleading information. SearchGen further studied retrieval and co-training of a search agent and generator as responses to these findings. VGI-Bench examined whether video generation models can solve visually grounded reasoning problems while satisfying task constraints. Its completion and rubric-based evaluation revealed a substantial gap between producing a plausible-looking sequence and generating a successful solution. Analyses of input conditions, fine-tuning, and denoising behavior further identified where apparent reasoning ability remained fragile. Together, the benchmarks make specific requirements of a generation request measurable, rather than treating visual appeal or general preference as sufficient evidence of success. They complement the earlier chapters' evaluations of task flexibility, temporal control, and feedback with tests of what models know and how they reason visually. The concluding chapter synthesizes these contributions and develops future research directions for controllable visual generation.

Chapter 7

Conclusion and Future Work

This thesis studied how foundation models can support controllable image and video generation beyond traditional text conditioning. Chapter 3 addressed limited task flexibility: OmniEdit consolidated specialist supervision into a generalist image editor, AnyV2V extended image editing to video without task-specific fine-tuning, and UniVideo unified understanding, generation, and editing in a shared model that interprets multimodal instructions and distinguishes among diverse tasks. Chapter 4 addressed the additional control requirements introduced by video's temporal dimension: MoCha enabled speech-conditioned video generation, while Context Forcing combined a long-context teacher with Slow-Fast Memory to support real-time long video generation while retaining historical context. Chapter 5 developed more informative feedback for improving generation: RationalRewards used reasoning-based scores and critiques to improve outputs through generator training and test-time prompt refinement, and RewardHarness evolved evaluation skills and tools around a fixed evaluator. Chapter 6 examined requirements that visual quality alone does not capture: SearchGen established a benchmark for knowledge-intensive generation and explored search–generation co-training, while VGI-Bench evaluated visual reasoning through tasks with explicit goals and rules. Together, these contributions broaden the tasks and conditions that visual generators can support, improve their responses through feedback, and expose remaining gaps in knowledge and reasoning. A central conclusion is that controllability requires more than accepting additional inputs. The studies in this thesis show the importance of interpreting those inputs in relation to the intended task, keeping relevant conditions effective throughout generation, and using feedback that identifies failures against the requested requirements. Progress toward controllable visual generation therefore depends on developing these capabilities alongside visual quality, rather than treating realistic appearance as sufficient evidence of success.

7.1 Practical Considerations and Limitations

The methods developed here offer several ways to build on stronger foundation models without treating each new backbone as an entirely new task-specific system. OmniEdit's supervision strategy could draw on more capable specialists, and AnyV2V's composition strategy could connect improved image editors and video generators. UniVideo offers a framework for bringing stronger multimodal understanding into generation, while RationalRewards and RewardHarness provide feedback outside the generator itself. Adapting these designs to new backbones entails matching component interfaces, maintaining supervision for the intended tasks, and calibrating evaluators to the generators they assess. Their reusable contribution lies in how they organize capabilities and feedback, linking advances in the underlying models to task-specific control.

These contributions motivate three extensions: unified multimodal models for interactive creation, persistent world simulation, and dense feedback for visual simulation. UniVideo supplies a shared interface for diverse tasks; MoCha and Context Forcing provide mechanisms for speech alignment and historical consistency; and RationalRewards, RewardHarness, SearchGen, and VGI-Bench connect evaluation to user intent and task-specific correctness. Each direction builds on these mechanisms to study longer and more complex interactions.

7.2 Unified Multimodal Models for Interactive Visual Creation

UniVideo connects multimodal understanding with generation and editing, allowing a shared model to interpret instructions and distinguish among diverse tasks. OmniEdit broadens the available image-editing operations, and AnyV2V extends image-editing interfaces to video through composition. These advances provide a foundation for extending task-level control to the management of an evolving creative project. A user developing a long video may supply many reference images and clips, revisit earlier versions, and revise requirements across a session lasting twenty minutes or several hours. Supporting this workflow requires understanding not only the latest instruction, but also which references it concerns, which earlier decisions remain valid, and which have been superseded.

A natural next step is to extend UniVideo's unified framework through interleaved, multi-turn understanding, generation, and editing. Selective memory could retain the roles of reference assets and the current creative requirements, while retrieving detailed

visual history when a revision needs it. OmniEdit's supervision strategy and AnyV2V's composition strategy offer complementary ways to supply editing capabilities within this framework. RationalRewards' critique-and-refine loop could likewise be extended to incorporate user corrections, with the model asking for clarification when requirements conflict. Training would need to expose the system to revisions and changing instructions, rather than treating each request as independent. Evaluation should then follow complete creation sessions, measuring whether new requests are fulfilled without losing still-valid decisions.

7.3 Persistent Interactive World Simulation

Context Forcing supports real-time long-video generation by combining long-context teacher supervision with Slow-Fast Memory, while MoCha coordinates visual performance with speech and multi-character dialogue. These methods extend the duration and conditioning of generated video, but an interactive world requires more than a coherent continuation. Users should be able to act through keyboard commands or text and observe persistent consequences: an object moved in one room should remain there when the user returns, and a character should retain its identity after leaving view. Extending our temporal-control methods to minutes or hours of interaction therefore raises a state-maintenance problem. Recent frames can describe the current view, but lasting changes, off-screen objects, and previously visited places must remain available even when they are no longer visually prominent.

Building on Slow-Fast Memory, future work could separate short-term visual context from persistent representations of entities, locations, and action outcomes. Context Forcing's teacher–student training could be extended to action-conditioned trajectories containing scene revisits and lasting changes, teaching the generator when to recover earlier state and when to update it. MoCha's audio–visual alignment mechanisms provide a further starting point for jointly generating video, speech, and environmental sound; this would extend its current speech-conditioned setting to audiovisual world simulation. The research challenge is to keep these streams consistent with both the evolving world and the user's actions under interactive latency constraints. Evaluation should include revisiting locations, checking whether action effects persist, and measuring identity drift and audiovisual synchronization over sustained interaction.

7.4 Reliable Evaluation and Dense Feedback for Visual Simulation

SearchGen-Bench makes knowledge-dependent errors observable through task-specific criteria, while VGI-Bench diagnoses failures in visual reasoning, rule following, and state consistency. RationalRewards and RewardHarness show how learned evaluation can guide improvements in image generation and editing. A further direction is to extend this feedback from assessing completed outputs to locating errors within a complex visual simulation. An overall score may fail to distinguish an incorrect entity from an impossible interaction, or a convincing final state from a trajectory that violates the task's rules. Longer creation sessions and interactive worlds amplify this problem because errors can propagate before the completed output is judged. Progress toward human-level evaluation requires identifying these specific failures and grounding the judgment in the available evidence, rather than producing increasingly fluent but unsupported critiques.

RationalRewards' structured rationales could be extended to localize feedback to objects, regions, events, and state transitions. RewardHarness's external skills and tools could support targeted checks of knowledge, physical relationships, and task rules, with uncertainty retained when the evidence is insufficient. The error categories exposed by SearchGen-Bench and VGI-Bench provide a basis for constructing such supervision and testing its coverage. Dense feedback would then mean more informative, localized judgments, not simply an independent score for every frame: many violations become apparent only by comparing states over time. These signals could guide training or trigger selective revision during inference. Their value should be verified with independent human judgments and task-specific checks, including whether optimization reduces the diagnosed errors rather than exploiting the evaluator.

7.5 Closing Remarks

This thesis advances visual generation toward understanding and fulfilling users' creative intent. Its methods broaden how intentions can be expressed and acted upon, its feedback systems guide improvements in generated outputs, and its benchmarks make knowledge and reasoning requirements measurable. Together, these contributions connect flexible interfaces, temporal control, and task-specific evaluation into a research framework for controllable visual generation.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=hSyW5goOZv.

[3] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18208–18218, 2022.

[4] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460, 2020.

[5] Jianhong Bai, Tianyu He, Yuchi Wang, Junliang Guo, Haoji Hu, Zuozhu Liu, and Jiang Bian. Uniedit: A unified tuning-free framework for video motion and appearance editing. *arXiv preprint arXiv:2402.13185*, 2024.

[6] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang,

Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report, 2025. URL https://arxiv.org/abs/2511.21631.

[7] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[8] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. *arXiv preprint arXiv:2305.13301*, 2023. URL https://arxiv.org/abs/2305.13301.

[9] Black Forest Labs. FLUX.2: Frontier visual intelligence. Model release and technical documentation, 2026. URL https://bfl.ai/. We evaluate the open-weight FLUX.2 [klein] 4B and 9B text-to-image checkpoints.

[10] Andreas Blattmann, Robin Rombach, Kaan Oktay, Jonas Müller, and Björn Ommer. Retrieval-augmented diffusion models. In *Advances in Neural Information Processing Systems*, volume 35, pages 15309–15324, 2022.

[11] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22563–22575, 2023.

[12] Frederic Boesel and Robin Rombach. Improving image editing models with generative data refinement. In *Tiny Papers @ ICLR 2024*, 2024. URL https://openreview.net/forum?id=q5UrA58oyY.

[13] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, 2023.

[14] Shengqu Cai, Ceyuan Yang, Lvmin Zhang, Yuwei Guo, Junfei Xiao, Ziyan Yang, Yinghao Xu, Zhenheng Yang, Alan Yuille, Leonidas Guibas, et al. Mixture of contexts for long video generation. *arXiv preprint arXiv:2508.21058*, 2025.

[15] Zefan Cai, Haoyi Qiu, Tianyi Ma, Haozhe Zhao, Gengze Zhou, Kung-Hsiang Huang, Parisa Kordjamshidi, Minjia Zhang, Wen Xiao, Jiuxiang Gu, Nanyun Peng, and Junjie Hu. Mmgr: Multi-modal generative reasoning, 2025. URL https://arxiv.org/abs/2512.14691.

[16] Duygu Ceylan, Chun-Hao P Huang, and Niloy J Mitra. Pix2video: Video editing using image diffusion. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 23149–23160. IEEE, 2023.

[17] Guibin Chen, Dixuan Lin, Jiangping Yang, Chunze Lin, Junchen Zhu, Mingyuan Fan, Hao Zhang, Sheng Chen, Zheng Chen, Chengcheng Ma, et al. Skyreels-v2: Infinite-length film generative model. *arXiv preprint arXiv:2504.13074*, 2025.

[18] Harold Haodong Chen, Disen Lan, Wen-Jie Shu, Qingyang Liu, Zihan Wang, Sirui Chen, Wenkai Cheng, Kanghao Chen, Hongfei Zhang, Zixin Zhang, Rongjin Guo, Yu Cheng, and Ying-Cong Chen. Tivibench: Benchmarking think-in-video reasoning for video generative models. *arXiv preprint arXiv:2511.13704*, 2025.

[19] Jiuhai Chen, Zhiyang Xu, Xichen Pan, Yushi Hu, Can Qin, Tom Goldstein, Lifu Huang, Tianyi Zhou, Saining Xie, Silvio Savarese, et al. Blip3-o: A family of fully open unified multimodal models-architecture, training and dataset. *arXiv preprint arXiv:2505.09568*, 2025.

[20] Wenhu Chen, Hexiang Hu, Chitwan Saharia, and William W. Cohen. Re-Imagen: Retrieval-augmented text-to-image generator. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XSEBxOiSjFQ.

[21] Xi Chen, Lianghua Huang, Yu Liu, Yujun Shen, Deli Zhao, and Hengshuang Zhao. Anydoor: Zero-shot object-level image customization. *arXiv preprint arXiv:2307.09481*, 2023.

[22] Xinyuan Chen, Yaohui Wang, Lingjun Zhang, Shaobin Zhuang, Xin Ma, Jiashuo Yu, Yali Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. Seine: Short-to-long video diffusion model for generative transition and prediction. *arXiv preprint arXiv:2310.20700*, 2023.

[23] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24185–24198, 2024.

[24] Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models. *arXiv preprint arXiv:2401.01335*, 2024.

[25] Joon Son Chung and Andrew Zisserman. Out of time: Automated lip sync in the wild. In *Computer Vision – ACCV 2016 Workshops*, volume 10117 of *Lecture Notes in Computer Science*, pages 251–263. Springer, 2017. doi: 10.1007/978-3-319-54427-4_19.

[26] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

[27] Yuren Cong, Mengmeng Xu, Christian Simon, Shoufa Chen, Jiawei Ren, Yanping Xie, Juan-Manuel Perez-Rua, Bodo Rosenhahn, Tao Xiang, and Sen He. Flatten: optical flow-guided attention for consistent text-to-video editing. *arXiv preprint arXiv:2310.05922*, 2023.

[28] Enric Corona, Andrei Zanfir, Eduard Gabriel Bazavan, Nikos Kolotouros, Thiemo Alldieck, and Cristian Sminchisescu. Vlogger: Multimodal diffusion for embodied avatar synthesis. *arXiv preprint arXiv:2403.08764*, 2024.

[29] Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord. Diffedit: Diffusion-based semantic image editing with mask guidance. *arXiv preprint arXiv:2210.11427*, 2022.

[30] Jiahao Cui, Hui Li, Yun Zhan, Hanlin Shang, Kaihui Cheng, Yuqi Ma, Shan Mu, Hang Zhou, Jingdong Wang, and Siyu Zhu. Hallo3: Highly dynamic and realistic portrait image animation with diffusion transformer networks. *arXiv preprint arXiv:2412.00733*, 2024.

[31] Jiahao Cui, Hui Li, Yao Yao, Hao Zhu, Hanlin Shang, Kaihui Cheng, Hang Zhou, Siyu Zhu, and Jingdong Wang. Hallo2: Long-duration and high-resolution audio-driven portrait image animation. *ICLR*, 2025.

[32] Justin Cui, Jie Wu, Ming Li, Tao Yang, Xiaojie Li, Rui Wang, Andrew Bai, Yuanhao Ban, and Cho-Jui Hsieh. Self-forcing++: Towards minute-scale high-quality video generation. *arXiv preprint arXiv:2510.02283*, 2025.

[33] Google DeepMind. Google gemini-3 system card, 2025.

[34] Chaorui Deng, Deyao Zhu, Kunchang Li, Chenhui Gou, Feng Li, Zeyu Wang, Shu Zhong, Weihao Yu, Xiaonan Nie, Ziang Song, Guang Shi, and Haoqi Fan. Emerging

properties in unified multimodal pretraining. *arXiv preprint arXiv:2505.14683*, 2025. URL https://arxiv.org/abs/2505.14683.

[35] Haoge Deng, Ting Pan, Haiwen Diao, Zhengxiong Luo, Yufeng Cui, Huchuan Lu, Shiguang Shan, Yonggang Qi, and Xinlong Wang. Autoregressive video generation without vector quantization. *arXiv preprint arXiv:2412.14169*, 2024.

[36] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[37] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[38] Patrick Esser, Johnathan Chiu, Parmida Atighehchian, Jonathan Granskog, and Anastasis Germanidis. Structure and content-guided video synthesis with diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7346–7356, 2023.

[39] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024.

[40] Kaituo Feng, Manyuan Zhang, Shuang Chen, Yunlong Lin, Kaixuan Fan, Yilei Jiang, Hongyu Li, Dian Zheng, Chenyang Wang, and Xiangyu Yue. Gen-searcher: Reinforcing agentic search for image generation. *arXiv preprint arXiv:2603.28767*, 2026.

[41] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of artistic style. *CoRR*, abs/1508.06576, 2015. URL http://arxiv.org/abs/1508.06576.

[42] Michal Geyer, Omer Bar-Tal, Shai Bagon, and Tali Dekel. Tokenflow: Consistent diffusion features for consistent video editing. *arXiv preprint arxiv:2307.10373*, 2023.

[43] Golnaz Ghiasi, Honglak Lee, Manjunath Kudlur, Vincent Dumoulin, and Jonathon Shlens. Exploring the structure of a real-time, arbitrary neural artistic stylization network. *CoRR*, abs/1705.06830, 2017. URL http://arxiv.org/abs/1705.06830.

[44] Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36:52132–52152, 2023.

[45] Google. Gemini 3 Flash: Frontier intelligence built for speed, 2025. URL https://blog.google/products-and-platforms/products/gemini/gemini-3-flash/.

[46] Google. Introducing Nano Banana Pro, 2025. URL https://blog.google/innovation-and-ai/products/nano-banana-pro/.

[47] Google DeepMind. Gemini image generation (gemini 2.5 Flash / gemini 3 Pro). Product and API documentation, 2025. URL https://ai.google.dev/gemini-api/docs. Consumer-facing names Nano Banana and Nano Banana Pro refer to Gemini 2.5 Flash Image and Gemini 3 Pro Image endpoints accessed via the Gemini developer API in our experiments.

[48] Jing Gu, Yilin Wang, Nanxuan Zhao, Tsu-Jui Fu, Wei Xiong, Qing Liu, Zhifei Zhang, He Zhang, Jianming Zhang, HyunJoon Jung, and Xin Eric Wang. Photoswap: Personalized subject swapping in images. arXiv preprint arXiv:2305.18286, 2023.

[49] Yuchao Gu, Yipin Zhou, Bichen Wu, Licheng Yu, Jia-Wei Liu, Rui Zhao, Jay Zhangjie Wu, David Junhao Zhang, Mike Zheng Shou, and Kevin Tang. Videoswap: Customized video subject swapping with interactive semantic point correspondence. *arXiv preprint arXiv:2312.02087*, 2023.

[50] Jiaxin Guo, Zewen Chi, Li Dong, Qingxiu Dong, Xun Wu, Shaohan Huang, and Furu Wei. Reward reasoning model, 2025. URL https://arxiv.org/abs/2505.14674.

[51] Yuwei Guo, Ceyuan Yang, Anyi Rao, Yaohui Wang, Yu Qiao, Dahua Lin, and Bo Dai. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. *arXiv preprint arXiv:2307.04725*, 2023.

[52] Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, et al. Ltx-video: Realtime video latent diffusion. *arXiv preprint arXiv:2501.00103*, 2024.

[53] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-or. Prompt-to-prompt image editing with cross-attention control. In *The Eleventh International Conference on Learning Representations*, 2023.

[54] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://arxiv.org/abs/1706.08500.

[55] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[56] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[57] Yicong Hong, Yiqun Mei, Chongjian Ge, Yiran Xu, Yang Zhou, Sai Bi, Yannick Hold-Geoffroy, Mike Roberts, Matthew Fisher, Eli Shechtman, et al. Relic: Interactive video world model with long-horizon memory. *arXiv preprint arXiv:2512.04040*, 2025.

[58] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981.

[59] Hexiang Hu, Kelvin CK Chan, Yu-Chuan Su, Wenhu Chen, Yandong Li, Kihyuk Sohn, Yang Zhao, Xue Ben, Boqing Gong, William Cohen, et al. Instruct-imagen: Image generation with multi-modal instruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4754–4763, 2024.

[60] Yushi Hu, Benlin Liu, Jungo Kasai, Yizhong Wang, Mari Ostendorf, Ranjay Krishna, and Noah A. Smith. TIFA: Accurate and interpretable text-to-image faithfulness evaluation with question answering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20406–20417, 2023.

[61] Yushi Hu, Reyhane Askari-Hemmat, Melissa Hall, Emily Dinan, Luke Zettlemoyer, and Marjan Ghazvininejad. Multimodal rewardbench 2: Evaluating omni reward models for interleaved text and image. *arXiv preprint arXiv:2512.16899*, 2025.

[62] Kaiyi Huang, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. T2I-CompBench: A comprehensive benchmark for open-world compositional text-to-image generation. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[63] Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *arXiv preprint arXiv:2506.08009*, 2025.

[64] Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages

21807–21818, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Huang_VBench_Comprehensive_Benchmark_Suite_for_Video_Generative_Models_CVPR_2024_paper.html.

[65] Mude Hui, Siwei Yang, Bingchen Zhao, Yichun Shi, Heng Wang, Peng Wang, Yuyin Zhou, and Cihang Xie. Hq-edit: A high-quality dataset for instruction-based image editing. *arXiv preprint arXiv:2404.09990*, 2024.

[66] Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong Park. Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. *arXiv preprint arXiv:2403.14403*, 2024.

[67] Sihui Ji, Xi Chen, Shuai Yang, Xin Tao, Pengfei Wan, and Hengshuang Zhao. Memflow: Flowing adaptive memory for consistent and efficient long video narratives. *arXiv preprint arXiv:2512.14699*, 2025.

[68] Xiaozhong Ji, Xiaobin Hu, Zhihong Xu, Junwei Zhu, Chuming Lin, Qingdong He, Jiangning Zhang, Donghao Luo, Yi Chen, Qin Lin, et al. Sonic: Shifting focus to global audio perception in portrait animation. *arXiv preprint arXiv:2411.16331*, 2024.

[69] Dongfu Jiang, Max Ku, Tianle Li, Yuansheng Ni, Shizhuo Sun, Rongqi Fan, and Wenhu Chen. Genai arena: An open evaluation platform for generative models. *arXiv preprint arXiv:2406.04485*, 2024.

[70] Jianwen Jiang, Chao Liang, Jiaqi Yang, Gaojie Lin, Tianyun Zhong, and Yanbo Zheng. Loopy: Taming audio-driven portrait avatar with long-term motion dependency. In *The Thirteenth International Conference on Learning Representations*, 2025.

[71] Zeyinzi Jiang, Zhen Han, Chaojie Mao, Jingfeng Zhang, Yulin Pan, and Yu Liu. Vace: All-in-one video creation and editing. *arXiv preprint arXiv:2503.07598*, 2025.

[72] Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992. Association for Computational Linguistics, 2023.

[73] Yang Jin, Zhicheng Sun, Ningyuan Li, Kun Xu, Kun Xu, Hao Jiang, Nan Zhuang, Quzhe Huang, Yang Song, Yadong Mu, and Zhouchen Lin. Pyramidal flow matching for efficient video generative modeling. *arXiv preprint arXiv:2410.05954*, 2024.

[74] Xuan Ju, Weicai Ye, Quande Liu, Qiulin Wang, Xintao Wang, Pengfei Wan, Di Zhang, Kun Gai, and Qiang Xu. Fulldit: Multi-task video generative foundation model with full attention. *arXiv preprint arXiv:2503.19907*, 2025.

[75] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

[76] Yuval Kirstain, Adam Polyak, Uriel Singer, Shahbuland Matiana, Joe Penna, and Omer Levy. Pick-a-pic: An open dataset of user preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36:36652–36663, 2023.

[77] Akio Kodaira, Tingbo Hou, Ji Hou, Masayoshi Tomizuka, and Yue Zhao. Streamdit: Real-time streaming text-to-video generation. *arXiv preprint arXiv:2507.03745*, 2025.

[78] Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, Kathrina Wu, Qin Lin, Junkun Yuan, Yanxin Long, Aladdin Wang, Andong Wang, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Hongmei Wang, Jacob Song, Jiawang Bai, Jianbing Wu, Jinbao Xue, Joey Wang, Kai Wang, Mengyang Liu, Pengyu Li, Shuai Li, Weiyan Wang, Wenqing Yu, Xinchi Deng, Yang Li, Yi Chen, Yutao Cui, Yuanbo Peng, Zhentao Yu, Zhiyu He, Zhiyong Xu, Zixiang Zhou, Zunnan Xu, Yangyu Tao, Qinglin Lu, Songtao Liu, Dax Zhou, Hongfa Wang, Yong Yang, Di Wang, Yuhong Liu, Jie Jiang, and Caesar Zhong. HunyuanVideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024. URL https://arxiv.org/abs/2412.03603.

[79] Max Ku, Dongfu Jiang, Cong Wei, Xiang Yue, and Wenhu Chen. VIEScore: Towards explainable metrics for conditional image synthesis evaluation. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12268–12290, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.663. URL https://aclanthology.org/2024.acl-long.663.

[80] Max Ku, Tianle Li, Kai Zhang, Yujie Lu, Xingyu Fu, Wenwen Zhuang, and Wenhu Chen. ImagenHub: Standardizing the evaluation of conditional image generation models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=OuV9ZrkQlc.

[81] Max Ku, Cong Wei, Weiming Ren, Harry Yang, and Wenhu Chen. Anyv2v: A tuning-free framework for any video-to-video editing tasks. *arXiv preprint arXiv:2403.14468*, 2024.

[82] Wing-Fung Ku, Wan-Chi Siu, Xi Cheng, and H. Anthony Chan. Intelligent painter: Picture composition with resampling diffusion model. arXiv preprint arXiv:2210.17106, 2022.

[83] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *International journal of computer vision*, 128(7):1956–1981, 2020.

[84] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474, 2020.

[85] Baiqi Li, Zhiqiu Lin, Deepak Pathak, Jiayao Li, Yixin Fei, Kewen Wu, Tiffany Ling, Xide Xia, Pengchuan Zhang, Graham Neubig, and Deva Ramanan. GenAI-Bench: Evaluating and improving compositional text-to-visual generation. *arXiv preprint arXiv:2406.13743*, 2024. URL https://arxiv.org/abs/2406.13743.

[86] Tianle Li, Max Ku, Cong Wei, and Wenhu Chen. Dreamedit: Subject-driven image editing. arXiv preprint arXiv:2306.12624, 2023.

[87] Wuyang Li, Wentao Pan, Po-Chien Luan, Yang Gao, and Alexandre Alahi. Stable video infinity: Infinite-length video generation with error recycling. *arXiv preprint arXiv:2510.09212*, 2025.

[88] Zhen Li, Chuanhao Li, Xiaofeng Mao, Shaoheng Lin, Ming Li, Shitian Zhao, Zhaopan Xu, Xinyue Li, Yukang Feng, Jianwen Sun, et al. Sekai: A video dataset towards world exploration. *arXiv preprint arXiv:2506.15675*, 2025.

[89] Feng Liang, Haoyu Ma, Zecheng He, Tingbo Hou, Ji Hou, Kunpeng Li, Xiaoliang Dai, Felix Juefei-Xu, Samaneh Azadi, Animesh Sinha, et al. Movie weaver: Tuning-free multi-concept video personalization with anchored prompts. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 13146–13156, 2025.

[90] Gaojie Lin, Jianwen Jiang, Jiaqi Yang, Zerong Zheng, and Chao Liang. Omnihuman-1: Rethinking the scaling-up of one-stage conditioned human animation models. *arXiv preprint arXiv:2502.01061*, 2025.

[91] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[92] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 26296–26306, 2024.

[93] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llavanext: Improved reasoning, ocr, and world knowledge, 2024. URL https://llava-vl.github.io/blog/2024-01-30-llava-next/.

[94] Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-grpo: Training flow matching models via online rl. *arXiv preprint arXiv:2505.05470*, 2025.

[95] Kunhao Liu, Wenbo Hu, Jiale Xu, Ying Shan, and Shijian Lu. Rolling forcing: Autoregressive long video diffusion in real time. *arXiv preprint arXiv:2509.25161*, 2025.

[96] Shaoteng Liu, Yuechen Zhang, Wenbo Li, Zhe Lin, and Jiaya Jia. Video-P2P: Video editing with cross-attention control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8599–8608, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Liu_Video-P2P_Video_Editing_with_Cross-attention_Control_CVPR_2024_paper.html.

[97] Shiyu Liu, Yucheng Han, Peng Xing, Fukun Yin, Rui Wang, Wei Cheng, Jiaqi Liao, Yingming Wang, Honghao Fu, Chunrui Han, Guopeng Li, Yuang Peng, Quan Sun, Jingwei Wu, Yan Cai, Zheng Ge, Ranchen Ming, Lei Xia, Xianfang Zeng, Yibo Zhu, Binxing Jiao, Xiangyu Zhang, Gang Yu, and Daxin Jiang. Step1X-Edit: A practical framework for general image editing. *arXiv preprint arXiv:2504.17761*, 2025. URL https://arxiv.org/abs/2504.17761.

[98] Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. Mmbench: Is your multi-modal model an all-around player? In *European conference on computer vision*, pages 216–233. Springer, 2024.

[99] Jiabin Luo, Junhui Lin, Zeyu Zhang, Biao Wu, Meng Fang, Ling Chen, and Hao Tang. Univid: The open-source unified video model. *arXiv preprint arXiv:2509.24200*, 2025.

[100] Yang Luo, Xuanlei Zhao, Baijiong Lin, Lingting Zhu, Liyao Tang, Yuqi Liu, Ying-Cong Chen, Shengju Qian, Xin Wang, and Yang You. V-ReasonBench: Toward unified reasoning benchmark suite for video generation models. *arXiv preprint arXiv:2511.16668*, 2025. URL https://arxiv.org/abs/2511.16668.

[101] Winfried Lötzsch, Max Reimann, Martin Büßemeyer, Amir Semmo, Jürgen Döllner, and Matthias Trapp. Wise: Whitebox image stylization by example-based learning. *ECCV*, 2022.

[102] Guoqing Ma, Haoyang Huang, Kun Yan, Liangyu Chen, Nan Duan, Shengming Yin, Changyi Wan, Ranchen Ming, Xiaoniu Song, Xing Chen, Yu Zhou, Deshan Sun, Deyu Zhou, Jian Zhou, Kaijun Tan, Kang An, Mei Chen, Wei Ji, Qiling Wu, Wen Sun, Xin Han, Yanan Wei, Zheng Ge, Aojie Li, Bin Wang, Bizhu Huang, Bo Wang, Brian Li, Changxing Miao, Chen Xu, Chenfei Wu, Chenguang Yu, Dapeng Shi, Dingyuan Hu, Enle Liu, Gang Yu, Ge Yang, Guanzhe Huang, Gulin Yan, Haiyang Feng, Hao Nie, Haonan Jia, Hanpeng Hu, Hanqi Chen, Haolong Yan, Heng Wang, Hongcheng Guo, Huilin Xiong, Huixin Xiong, Jiahao Gong, Jianchang Wu, Jiaoren Wu, Jie Wu, Jie Yang, Jiashuai Liu, Jiashuo Li, Jingyang Zhang, Junjing Guo, Junzhe Lin, Kaixiang Li, Lei Liu, Lei Xia, Liang Zhao, Liguo Tan, Liwen Huang, Liying Shi, Ming Li, Mingliang Li, Muhua Cheng, Na Wang, Qiaohui Chen, Qinglin He, Qiuyan Liang, Quan Sun, Ran Sun, Rui Wang, Shaoliang Pang, Shiliang Yang, Sitong Liu, Siqi Liu, Shuli Gao, Tiancheng Cao, Tianyu Wang, Weipeng Ming, Wenqing He, Xu Zhao, Xuelin Zhang, Xianfang Zeng, Xiaojia Liu, Xuan Yang, Yaqi Dai, Yanbo Yu, Yang Li, Yineng Deng, Yingming Wang, Yilei Wang, Yuanwei Lu, Yu Chen, Yu Luo, Yuchu Luo, Yuhe Yin, Yuheng Feng, Yuxiang Yang, Zecheng Tang, Zekai Zhang, Zidong Yang, Binxing Jiao, Jiansheng Chen, Jing Li, Shuchang Zhou, Xiangyu Zhang, Xinhao Zhang, Yibo Zhu, Heung-Yeung Shum, and Daxin Jiang. Step-Video-T2V technical report: The practice, challenges, and future of video foundation model. *arXiv preprint arXiv:2502.10248*, 2025. URL https://arxiv.org/abs/2502.10248.

[103] Yiyang Ma, Xingchao Liu, Xiaokang Chen, Wen Liu, Chengyue Wu, Zhiyu Wu, Zizheng Pan, Zhenda Xie, Haowei Zhang, Xingkai Yu, et al. Janusflow: Harmonizing autoregression and rectified flow for unified multimodal understanding and generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 7739–7751, 2025.

[104] Yuhang Ma, Xiaoshi Wu, Keqiang Sun, and Hongsheng Li. Hpsv3: Towards wide-spectrum human preference score. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15086–15095, 2025.

[105] Dakota Mahan, Duy Van Phung, Rafael Rafailov, Chase Blagden, Nathan Lile, Louis Castricato, Jan-Philipp Fränken, Chelsea Finn, and Alon Albalak. Generative reward models, 2024. URL https://arxiv.org/abs/2410.12832.

[106] Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822, 2023.

[107] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022.

[108] Rang Meng, Xingyu Zhang, Yuming Li, and Chenguang Ma. Echomimicv2: Towards striking, simplified, and semi-body human animation. *arXiv preprint arXiv:2411.10061*, 2024.

[109] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6038–6047, 2023.

[110] Kaleb Newman, Tyler Zhu, and Olga Russakovsky. Video models reason early: Exploiting plan commitment for maze solving. *arXiv preprint arXiv:2603.30043*, 2026.

[111] Alexander Quinn Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob Mcgrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. In *International Conference on Machine Learning*, pages 16784–16804. PMLR, 2022.

[112] OpenAI. GPT Image: Openai image generation. Product and API documentation, 2025. URL https://platform.openai.com/docs/guides/image-generation. We refer to the OpenAI image-generation endpoint as GPT-Image-2 in our experiments.

[113] OpenAI. GPT Image 2 (gpt-image-2). https://developers.openai.com/api/docs/models/gpt-image-2, 2026.

[114] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2023.

[115] Hao Ouyang, Qiuyu Wang, Yuxi Xiao, Qingyan Bai, Juntao Zhang, Kecheng Zheng, Xiaowei Zhou, Qifeng Chen, and Yujun Shen. Codef: Content deformation fields for temporally consistent video processing. *arXiv preprint arXiv:2308.07926*, 2023.

[116] Xichen Pan, Satya Narayan Shukla, Aashu Singh, Zhuokai Zhao, Shlok Kumar Mishra, Jialiang Wang, Zhiyang Xu, Jiuhai Chen, Kunpeng Li, Felix Juefei-Xu, et al. Transfer between modalities with metaqueries. *arXiv preprint arXiv:2504.06256*, 2025.

[117] Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. Gorilla: Large language model connected with massive apis. *arXiv preprint arXiv:2305.15334*, 2023.

[118] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Peebles_Scalable_Diffusion_Models_with_Transformers_ICCV_2023_paper.html.

[119] Pexels. Pexels, 2024. URL www.pexels.com.

[120] Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, et al. Movie gen: A cast of media foundation models. *arXiv preprint arXiv:2410.13720*, 2024.

[121] KR Prajwal, Rudrabha Mukhopadhyay, Vinay P Namboodiri, and CV Jawahar. A lip sync expert is all you need for speech to lip generation in the wild. In *Proceedings of the 28th ACM international conference on multimedia*, pages 484–492, 2020.

[122] Chenyang Qi, Xiaodong Cun, Yong Zhang, Chenyang Lei, Xintao Wang, Ying Shan, and Qifeng Chen. Fatezero: Fusing attentions for zero-shot text-based video editing. *arXiv preprint arXiv:2303.09535*, 2023.

[123] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. Toolllm: Facilitating large language models to master 16000+ real-world apis. *arXiv preprint arXiv:2307.16789*, 2023.

[124] Haonan Qiu, Menghan Xia, Yong Zhang, Yingqing He, Xintao Wang, Ying Shan, and Ziwei Liu. Freenoise: Tuning-free longer video diffusion via noise rescheduling. arXiv preprint arXiv:2310.15169, 2023.

[125] Liao Qu, Huichao Zhang, Yiheng Liu, Xu Wang, Yi Jiang, Yiming Gao, Hu Ye, Daniel K Du, Zehuan Yuan, and Xinglong Wu. Tokenflow: Unified image tokenizer for multimodal understanding and generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 2545–2555, 2025.

[126] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

[127] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[128] Weiming Ren, Harry Yang, Ge Zhang, Cong Wei, Xinrun Du, Stephen Huang, and Wenhu Chen. Consisti2v: Enhancing visual consistency for image-to-video generation. *arXiv preprint arXiv:2402.04324*, 2024.

[129] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[130] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell

Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35: 25278–25294, 2022.

[131] Shelly Sheynin, Adam Polyak, Uriel Singer, Yuval Kirstain, Amit Zohar, Oron Ashual, Devi Parikh, and Yaniv Taigman. Emu edit: Precise image editing via recognition and generation tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8871–8879, 2024.

[132] Joonghyuk Shin, Zhengqi Li, Richard Zhang, Jun-Yan Zhu, Jaesik Park, Eli Shechtman, and Xun Huang. Motionstream: Real-time video generation with interactive motion controls. *arXiv preprint arXiv:2511.01266*, 2025.

[133] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

[134] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters, 2024. URL https://arxiv.org/abs/2408.03314.

[135] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=St1giarCHLP.

[136] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Proceedings of the 40th International Conference on Machine Learning*, pages 32211–32252, 2023.

[137] Wenqiang Sun, Haiyu Zhang, Haoyuan Wang, Junta Wu, Zehan Wang, Zhenwei Wang, Yunhong Wang, Jun Zhang, Tengfei Wang, and Chunchao Guo. Worldplay: Towards long-term geometric consistency for real-time interactive world modeling. *arXiv preprint arXiv:2512.14614*, 2025.

[138] Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor Lempitsky. Resolution-robust large mask inpainting with fourier convolutions. *arXiv preprint arXiv:2109.07161*, 2021.

[139] Zhiyu Tan, Hao Yang, Luozheng Qin, Jia Gong, Mengping Yang, and Hao Li. Omni-video: Democratizing unified video understanding and generation. *arXiv preprint arXiv:2507.06119*, 2025.

[140] Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*, 2024.

[141] Team Seedream, Yunpeng Chen, Yu Gao, Lixue Gong, Meng Guo, Qiushan Guo, Zhiyao Guo, Xiaoxia Hou, Weilin Huang, Yixuan Huang, Xiaowen Jian, Huafeng Kuang, Zhichao Lai, Fanshi Li, Liang Li, Xiaochen Lian, Chao Liao, Liyang Liu, Wei Liu, Yanzuo Lu, Zhengxiong Luo, Tongtong Ou, Guang Shi, Yichun Shi, Shiqi Sun, Yu Tian, Zhi Tian, Peng Wang, Rui Wang, Xun Wang, Ye Wang, Guofeng Wu, Jie Wu, Wenxu Wu, Yonghui Wu, Xin Xia, Xuefeng Xiao, Shuang Xu, Xin Yan, Ceyuan Yang, Jianchao Yang, Zhonghua Zhai, Chenlin Zhang, Heng Zhang, Qi Zhang, Xinyu Zhang, Yuwei Zhang, Shijia Zhao, Wenliang Zhao, and Wenjia Zhu. Seedream 4.0: Toward next-generation multimodal image generation. *arXiv preprint arXiv:2509.20427*, 2025. URL https://arxiv.org/abs/2509.20427.

[142] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. URL https://arxiv.org/abs/2503.20314.

[143] Hansi Teng, Hongyu Jia, Lei Sun, Lingzhi Li, Maolin Li, Mingqiu Tang, Shuai Han, Tianning Zhang, WQ Zhang, Weifeng Luo, et al. Magi-1: Autoregressive video generation at scale. *arXiv preprint arXiv:2505.13211*, 2025.

[144] Linrui Tian, Qi Wang, Bang Zhang, and Liefeng Bo. Emo: Emote portrait alive generating expressive portrait videos with audio2video diffusion model under weak conditions. In *European Conference on Computer Vision*, pages 244–260. Springer, 2024.

[145] Linrui Tian, Siqi Hu, Qi Wang, Bang Zhang, and Liefeng Bo. Emo2: End-effector guided audio-driven avatar video generation. *arXiv preprint arXiv:2501.10687*, 2025.

[146] Jingqi Tong, Yurong Mou, Hangcheng Li, Mingzhe Li, Yongzhuo Yang, Ming Zhang, Qiguang Chen, Tianyi Liang, Xiaomeng Hu, Yining Zheng, Xinchi Chen, Jun Zhao, Xuanjing Huang, and Xipeng Qiu. Thinking with video: Video generation as a promising multimodal reasoning paradigm. *arXiv preprint arXiv:2511.04570*, 2025.

[147] Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1921–1930, June 2023.

[148] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018. URL https://arxiv.org/abs/1812.01717.

[149] Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8228–8238, 2024.

[150] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

[151] Haofan Wang, Qixun Wang, Xu Bai, Zekui Qin, and Anthony Chen. Instantstyle: Free lunch towards style-preserving in text-to-image generation. *arXiv preprint arXiv:2404.02733*, 2024.

[152] Haozhe Wang, Haoran Que, Qixin Xu, Minghao Liu, Wangchunshu Zhou, Jiazhan Feng, Wanjun Zhong, Wei Ye, Tong Yang, Wenhao Huang, et al. Reverse-engineered reasoning for open-ended generation. *arXiv preprint arXiv:2509.06160*, 2025.

[153] Linqing Wang, Ximing Xing, Yiji Cheng, Zhiyuan Zhao, Donghao Li, Tiankai Hang, Jiale Tao, Qixun Wang, Ruihuang Li, Comi Chen, et al. Promptenhancer: A simple approach to enhance text-to-image models via chain-of-thought prompt rewriting. *arXiv preprint arXiv:2509.04545*, 2025.

[154] Maijunxian Wang, Ruisi Wang, Juyi Lin, Ran Ji, Thaddäus Wiedemer, Qingying Gao, Dezhi Luo, Yaoyao Qian, Lianyu Huang, Zelong Hong, Jiahui Ge, Qianli Ma, Hang He, Yifan Zhou, Lingzi Guo, Lantao Mei, Jiachen Li, Hanwen Xing, Tianqi Zhao, Fengyuan Yu, Weihang Xiao, Yizheng Jiao, Jianheng Hou, Danyang Zhang, Pengcheng Xu, Boyang Zhong, Zehong Zhao, Gaoyun Fang, John Kitaoka, Yile Xu, Hua Xu, Kenton Blacutt, Tin Nguyen, Siyuan Song, Haoran Sun, Shaoyue Wen, Linyang He, Runming Wang, Yanzhi Wang, Mengyue Yang, Ziqiao Ma, Raphaël Millière, Freda Shi, Nuno Vasconcelos, Daniel Khashabi, Alan Yuille, Yilun Du, Ziming Liu, Bo Li, Dahua Lin, Ziwei Liu, Vikash Kumar, Yijiang Li, Lei Yang, Zhongang Cai, and Hokin Deng. A very big video reasoning suite. *arXiv preprint arXiv:2602.20159*, 2026. URL https://arxiv.org/abs/2602.20159.

[155] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[156] Qixun Wang, Xu Bai, Haofan Wang, Zekui Qin, and Anthony Chen. Instantid: Zero-shot identity-preserving generation in seconds. *arXiv preprint arXiv:2401.07519*, 2024.

[157] Wenhao Wang and Yi Yang. Vidprom: A million-scale real prompt-gallery dataset for text-to-video diffusion models. *Advances in Neural Information Processing Systems*, pages 65618–65642, 2024.

[158] Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yueze Wang, Zhen Li, Qiying Yu, et al. Emu3: Next-token prediction is all you need. *arXiv preprint arXiv:2409.18869*, 2024.

[159] Yibin Wang, Yuhang Zang, Hao Li, Cheng Jin, and Jiaqi Wang. Unified reward model for multimodal understanding and generation. *arXiv preprint arXiv:2503.05236*, 2025.

[160] Huawei Wei, Zejun Yang, and Zhisheng Wang. Aniportrait: Audio-driven synthesis of photorealistic portrait animation. *arXiv preprint arXiv:2403.17694*, 2024.

[161] Thaddäus Wiedemer, Yuxuan Li, Paul Vicol, Shixiang Shane Gu, Nick Matarese, Kevin Swersky, Been Kim, Priyank Jaini, and Robert Geirhos. Video models are zero-shot learners and reasoners. *arXiv preprint arXiv:2509.20328*, 2025.

[162] Olivia Wiles, Chuhan Zhang, Isabela Albuquerque, Ivana Kajic, Su Wang, Emanuele Bugliarello, Yasumasa Onoe, Chris Knutsen, Cyrus Rashtchian, Jordi Pont-Tuset, and Aida Nematzadeh. Revisiting text-to-image evaluation with Gecko: On metrics, prompts, and human ratings. *arXiv preprint arXiv:2404.16820*, 2024.

[163] Bichen Wu, Ching-Yao Chuang, Xiaoyan Wang, Yichen Jia, Kapil Krishnakumar, Tong Xiao, Feng Liang, Licheng Yu, and Peter Vajda. Fairy: Fast parallelized instruction-guided video-to-video synthesis. *arXiv preprint arXiv:2312.13834*, 2023.

[164] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Shengming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-Image technical report. *arXiv preprint arXiv:2508.02324*, 2025. URL https://arxiv.org/abs/2508.02324.

[165] Chengyue Wu, Xiaokang Chen, Zhiyu Wu, Yiyang Ma, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, Chong Ruan, et al. Janus: Decoupling visual encoding for unified multimodal understanding and generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 12966–12977, 2025.

[166] Chenyuan Wu, Pengfei Zheng, Ruiran Yan, Shitao Xiao, Xin Luo, Yueze Wang, Wanli Li, Xiyan Jiang, Yexin Liu, Junjie Zhou, et al. Omnigen2: Exploration to advanced multimodal generation. *arXiv preprint arXiv:2506.18871*, 2025.

[167] Jay Zhangjie Wu, Yixiao Ge, Xintao Wang, Stan Weixian Lei, Yuchao Gu, Yufei Shi, Wynne Hsu, Ying Shan, Xiaohu Qie, and Mike Zheng Shou. Tune-A-Video: One-shot tuning of image diffusion models for text-to-video generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7623–7633, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Wu_Tune-A-Video_One-Shot_Tuning_of_Image_Diffusion_Models_for_Text-to-Video_Generation_ICCV_2023_paper.html.

[168] Keming Wu, Sicong Jiang, Max Ku, Ping Nie, Minghao Liu, and Wenhu Chen. Editreward: A human-aligned reward model for instruction-guided image editing. *arXiv preprint arXiv:2509.26346*, 2025.

[169] Tianxing Wu, Chenyang Si, Yuming Jiang, Ziqi Huang, and Ziwei Liu. Freeinit: Bridging initialization gap in video diffusion models. *arXiv preprint arXiv:2312.07537*, 2023.

[170] Ziyi Wu, Aliaksandr Siarohin, Willi Menapace, Ivan Skorokhodov, Yuwei Fang, Varnith Chordia, Igor Gilitschenski, and Sergey Tulyakov. Mind the time: Temporally-controlled multi-event video generation. In *CVPR*, 2025.

[171] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2309.17453.

[172] Zeqi Xiao, Yushi Lan, Yifan Zhou, Wenqi Ouyang, Shuai Yang, Yanhong Zeng, and Xingang Pan. Worldmem: Long-term consistent world simulation with memory. *arXiv preprint arXiv:2504.12369*, 2025.

[173] Jinheng Xie, Weijia Mao, Zechen Bai, David Junhao Zhang, Weihao Wang, Kevin Qinghong Lin, Yuchao Gu, Zhijie Chen, Zhenheng Yang, and Mike Zheng Shou. Show-o: One single transformer to unify multimodal understanding and generation. *arXiv preprint arXiv:2408.12528*, 2024.

[174] Jinheng Xie, Zhenheng Yang, and Mike Zheng Shou. Show-o2: Improved native unified multimodal models. *arXiv preprint arXiv:2506.15564*, 2025.

[175] Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imagereward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36: 15903–15935, 2023.

[176] Jiazheng Xu, Yu Huang, Jiale Cheng, Yuanming Yang, Jiajun Xu, Yuan Wang, Wenbo Duan, Shen Yang, Qunlin Jin, Shurun Li, et al. Visionreward: Fine-grained multi-dimensional human preference learning for image and video generation. *arXiv preprint arXiv:2412.21059*, 2024.

[177] Mingwang Xu, Hui Li, Qingkun Su, Hanlin Shang, Liwei Zhang, Ce Liu, Jingdong Wang, Yao Yao, and Siyu Zhu. Hallo: Hierarchical audio-driven visual synthesis for portrait image animation. *arXiv preprint arXiv:2406.08801*, 2024.

[178] Zeyue Xue, Jie Wu, Yu Gao, Fangyuan Kong, Lingting Zhu, Mengzhao Chen, Zhiheng Liu, Wei Liu, Qiushan Guo, Weilin Huang, et al. Dancegrpo: Unleashing grpo on visual generation. *arXiv preprint arXiv:2505.07818*, 2025.

[179] Zhucun Xue, Jiangning Zhang, Teng Hu, Haoyang He, Yinan Chen, Yuxuan Cai, Yabiao Wang, Chengjie Wang, Yong Liu, Xiangtai Li, et al. Ultravideo: High-quality uhd video dataset with comprehensive captions. *arXiv preprint arXiv:2506.13691*, 2025.

[180] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[181] Ling Yang, Zhaochen Yu, Chenlin Meng, Minkai Xu, Stefano Ermon, and Bin Cui. Mastering text-to-image diffusion: Recaptioning, planning, and generating with multimodal LLMs. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 56704–56721. PMLR, 2024. URL https://proceedings.mlr.press/v235/yang24ai.html.

[182] Shuai Yang, Wei Huang, Ruihang Chu, Yicheng Xiao, Yuyang Zhao, Xianbang Wang, Muyang Li, Enze Xie, Yingcong Chen, Yao Lu, et al. Longlive: Real-time interactive long video generation. *arXiv preprint arXiv:2509.22622*, 2025.

[183] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, Da Yin, Yuxuan Zhang, Weihan Wang, Yean Cheng, Bin Xu, Xiaotao Gu, Yuxiao Dong, and Jie Tang. CogVideoX: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=LQzN6TRFg9.

[184] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023.

[185] Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. IP-Adapter: Text compatible image prompt adapter for text-to-image diffusion models. *arXiv preprint arXiv:2308.06721*, 2023. URL https://arxiv.org/abs/2308.06721.

[186] Yang Ye, Xianyi He, Zongjian Li, Bin Lin, Shenghai Yuan, Zhiyuan Yan, Bohan Hou, and Li Yuan. ImgEdit: A unified image editing dataset and

benchmark. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/bec783f119b969772c35243b1c801090-Abstract-Datasets_and_Benchmarks_Track.html.

[187] Zixuan Ye, Xuanhua He, Quande Liu, Qiulin Wang, Xintao Wang, Pengfei Wan, Di Zhang, Kun Gai, Qifeng Chen, and Wenhan Luo. Unic: Unified in-context video editing. *arXiv preprint arXiv:2506.04216*, 2025.

[188] Hidir Yesiltepe, Tuna Han Salih Meral, Adil Kaan Akan, Kaan Oktay, and Pinar Yanardag. Infinity-RoPE: Action-controllable infinite video generation emerges from autoregressive self-rollout. *arXiv preprint arXiv:2511.20649*, 2025.

[189] Fukun Yin, Shiyu Liu, Yucheng Han, Zhibo Wang, Peng Xing, Rui Wang, Wei Cheng, Yingming Wang, Aojie Li, Zixin Yin, et al. Reasonedit: Towards reasoning-enhanced image editing models. *arXiv preprint arXiv:2511.22625*, 2025.

[190] Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and Bill Freeman. Improved distribution matching distillation for fast image synthesis. *Advances in Neural Information Processing Systems*, pages 47455–47487, 2024.

[191] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6613–6623, 2024.

[192] Tianwei Yin, Qiang Zhang, Richard Zhang, William T Freeman, Fredo Durand, Eli Shechtman, and Xun Huang. From slow bidirectional to fast causal video generators. *arXiv preprint arXiv:2412.07772*, 2024.

[193] Weihao Yu, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Xinchao Wang, and Lijuan Wang. Mm-vet: Evaluating large multimodal models for integrated capabilities. *arXiv preprint arXiv:2308.02490*, 2023.

[194] Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason E Weston. Self-rewarding language models. In *Forty-first International Conference on Machine Learning*, 2024.

[195] Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9556–9567, 2024.

[196] Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35: 15476–15488, 2022.

[197] Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. MagicBrush: A manually annotated dataset for instruction-guided image editing. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/64008fa30cba9b4d1ab1bd3bd3d57d61-Abstract-Datasets_and_Benchmarks.html.

[198] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.

[199] Lvmin Zhang, Shengqu Cai, Muyang Li, Gordon Wetzstein, and Maneesh Agrawala. Frame context packing and drift prevention in next-frame-prediction video diffusion models. *Advances in Neural Information Processing Systems*, 38:34436–34456, 2025.

[200] Lvmin Zhang, Shengqu Cai, Muyang Li, Chong Zeng, Beijia Lu, Anyi Rao, Song Han, Gordon Wetzstein, and Maneesh Agrawala. Pretraining frame preservation in autoregressive video memory compression, 2025. URL https://arxiv.org/abs/2512.23851.

[201] Shiwei Zhang, Jiayu Wang, Yingya Zhang, Kang Zhao, Hangjie Yuan, Zhiwu Qin, Xiang Wang, Deli Zhao, and Jingren Zhou. I2vgen-xl: High-quality image-to-video synthesis via cascaded diffusion models. *arXiv preprint arXiv:2311.04145*, 2023.

[202] Shu Zhang, Xinyi Yang, Yihao Feng, Can Qin, Chia-Chih Chen, Ning Yu, Zeyuan Chen, Huan Wang, Silvio Savarese, Stefano Ermon, et al. Hive: Harnessing human feedback for instructional visual editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9026–9036, 2024.

[203] Wenxuan Zhang, Xiaodong Cun, Xuan Wang, Yong Zhang, Xi Shen, Yu Guo, Ying Shan, and Fei Wang. Sadtalker: Learning realistic 3d motion coefficients for stylized

audio-driven single image talking face animation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8652–8661, 2023.

[204] Xinchen Zhang, Ling Yang, Guohao Li, Yaqi Cai, Jiake Xie, Yong Tang, Yujiu Yang, Mengdi Wang, and Bin Cui. IterComp: Iterative composition-aware feedback learning from model gallery for text-to-image generation. *arXiv preprint arXiv:2410.07171*, 2024.

[205] Yabo Zhang, Yuxiang Wei, Dongsheng Jiang, Xiaopeng Zhang, Wangmeng Zuo, and Qi Tian. Controlvideo: Training-free controllable text-to-video generation. *arXiv preprint arXiv:2305.13077*, 2023.

[206] Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19632–19642, 2024.

[207] Haozhe Zhao, Xiaojian Ma, Liang Chen, Shuzheng Si, Rujie Wu, Kaikai An, Peiyu Yu, Minjia Zhang, Qing Li, and Baobao Chang. Ultraedit: Instruction-based fine-grained image editing at scale. *arXiv preprint arXiv:2407.05282*, 2024.

[208] Min Zhao, Guande He, Yixiao Chen, Hongzhou Zhu, Chongxuan Li, and Jun Zhu. Riflex: A free lunch for length extrapolation in video diffusion transformers. *arXiv preprint arXiv:2502.15894*, 2025.

[209] Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. Diffusionnft: Online diffusion reinforcement with forward process. *arXiv preprint arXiv:2509.16117*, 2025.

[210] Chunting Zhou, Lili Yu, Arun Babu, Kushal Tirumala, Michihiro Yasunaga, Leonid Shamis, Jacob Kahn, Xuezhe Ma, Luke Zettlemoyer, and Omer Levy. Transfusion: Predict the next token and diffuse images with one multi-modal model. *arXiv preprint arXiv:2408.11039*, 2024.

[211] Tinghui Zhu, Sheng Zhang, James Y. Huang, Selena Song, Xiaofei Wen, Yuankai Li, Hoifung Poon, and Muhao Chen. Video models can reason with verifiable rewards, 2026. URL https://arxiv.org/abs/2605.15458.