AUTOFLOW: AUTOREGRESSIVE NORMALIZING FLOWS FOR NEXT VECTOR PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Next-token prediction gives language models a simple interface for both understanding and generation. Extending this idea to continuous signals such as images and robot actions is challenging because these signals are continuous vectors rather than discrete tokens. We introduce AutoFlow, a framework for next-vector prediction in autoregressive foundation models. The key idea is to equip a language model with a conditional normalizing flow head that predicts the next continuous vector while leaving the autoregressive backbone unchanged. We find that this simple formulation better connects generation and understanding than diffusion-based alternatives: models trained for image generation transfer more effectively to visual understanding, while also supporting interleaved image–text reasoning and robot action prediction. AutoFlow further generates continuous outputs in a single forward evaluation and provides their likelihood directly. Our results suggest that next-vector prediction via AutoFlow is an elegant and effective formulation for joint autoregressive modeling of discrete and continuous data, which provides a new path towards omnimodal generative models.

1 INTRODUCTION

Large language models learn to both understand and generate text through a simple objective: predicting the next token (Brown et al., 2020; Touvron et al., 2023; Achiam et al., 2023). But much of our world that foundation models must reason about is not made of discrete tokens. Images, video, and robot actions are continuous signals, and models increasingly need to both understand and generate them (Team et al., 2023; Brohan et al., 2023; Kim et al., 2024; Team et al., 2024; Black et al., 2026). This raises a natural question: can the same predictive principle extend beyond language? In particular, can predicting the next continuous signal teach a model not only how to generate it, but also how to understand it? We study this problem as *next-vector prediction*.

A large body of work studied how to unify the generation of text and other continuous signals. One direction converts continuous signals into discrete tokens (Van Den Oord et al., 2017; Team, 2024; Wang et al., 2024; Wu et al., 2025), allowing them to reuse next-token prediction at the cost of compressing fine-grained information into a finite codebook. Another direction keeps the outputs continuous and adds diffusion-style prediction to an autoregressive model, either through a separate continuous head or by involving the shared backbone in denoising (Li et al., 2024; Zhou et al., 2025; Xie et al., 2025; Deng et al., 2025). These approaches make continuous generation possible, but they do not necessarily make generation useful for understanding. In many systems, generation and perception still rely on different objectives or even different visual representations (Chen et al., 2025; Deng et al., 2025). One central challenge is therefore not simply how to generate continuous signals, but how to make generation train representations that are also useful for understanding.

We begin by examining why diffusion training may not provide this connection. Diffusion learns generation by reconstructing targets from corrupted observations (Ho et al., 2020; Lipman et al., 2022; Chen et al., 2024). This creates a mismatch between generation and understanding: generation is learned from noisy inputs, while understanding operates on clean observations. The prediction problem also changes substantially with the noise level, ranging from highly ambiguous targets at high noise, to nearly revealed targets at low noise. We therefore seek a next-vector objective that learns directly from clean continuous targets while still modeling their full conditional distribution.

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

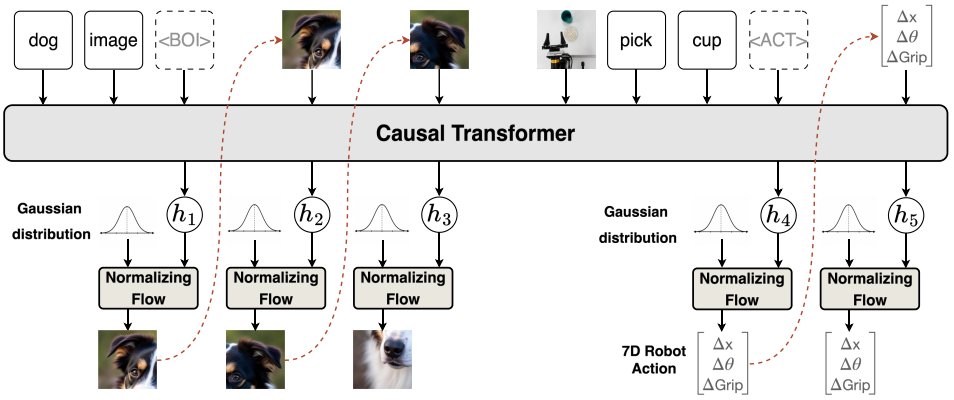


Figure 1: Overview of our proposed AutoFlow for next-vector prediction. A causal Transformer produces conditioning representations h_t from the current context. For image generation, the flow head maps Gaussian noise and h_t to continuous image latents. For VLA policy learning, the same head predicts continuous robot action chunks from instruction and state context. Generated continuous tokens are fed back into the autoregressive context for subsequent prediction.

We introduce AutoFlow, a simple alternative for next-vector prediction, see Figure 1. Instead of discretizing the target or training the model to denoise corrupted targets, AutoFlow attaches a conditional normalizing-flow head to an otherwise unchanged autoregressive language model. The key idea is to model the distribution of the next continuous vector directly, conditioned on the autoregressive representation. Unlike diffusion, this objective learns from the observed target without corrupting it, allowing generation to train the same backbone used for understanding. Because the mapping is invertible, AutoFlow also provides exact conditional likelihood and generates a continuous prediction in a single forward evaluation rather than through iterative denoising.

Empirical results show that AutoFlow more effectively connects continuous generation and understanding. After image-generation training, AutoFlow improves downstream visual understanding over diffusion-based alternatives by an average of 4.4 points across eight tasks. Our analysis provides evidence for this transfer: AutoFlow produces more stable gradients, relies more strongly on the autoregressive context, and better aligns the learning signals for generation and understanding. Beyond representation transfer, AutoFlow also improves interleaved image–text reasoning; for example, it reaches 38.8% accuracy versus 30.8% for the strongest diffusion baseline on spatial navigation, using a single sampling step. For image generation, AutoFlow reaches strong visual quality in one forward evaluation, while diffusion-based alternatives require multiple sampling steps to reach comparable quality. This efficiency matters when predictions must be made repeatedly or under tight latency constraints. On real-robot tasks, AutoFlow achieves 66.7% average success compared with 40.0% for the strongest baseline, while using a smaller prediction head and a single sampling step.

2 Related Work

Normalizing flows. Normalizing flows model continuous distributions through invertible transformations, enabling sampling and likelihood evaluation via the change-of-variables formula (Dinh et al., 2014; 2016; Papamakarios et al., 2021). Coupling and autoregressive constructions offer tractable Jacobians with different sampling–density evaluation trade-offs (Kingma et al., 2016; Papamakarios et al., 2017; Kingma & Dhariwal, 2018); subsequent work improves expressivity through dequantization, residual blocks, and splines (Ho et al., 2019; Chen et al., 2019; Durkan et al., 2019). Recent Transformer-based and latent-space flows scale this approach to image synthesis (Zhai et al., 2024; Gu et al., 2026a). JetFormer replaces the VAE image encoder/decoder with a normalizing flow jointly trained with an autoregressive Transformer, whose Gaussian-mixture prediction head models continuous image tokens (Tschannen et al., 2025). We build on conditional flows, which model output distributions given an external signal (Trippe & Turner, 2018; Ardizzone et al., 2019; Winkler et al., 2019). Rather than using a flow as the image encoder/decoder or main generative backbone, AutoFlow attaches it as a continuous-output head to an autoregressive foundation model. The backbone supplies the conditioning state, and the head learns from the multimodal continuous targets with a direct likelihood gradient.

Unified autoregressive multimodal models. One approach converts images into discrete tokens and trains an autoregressive model over mixed text–image sequences. VQ-VAE and VQGAN

provide codebook-based visual representations (Van Den Oord et al., 2017; Esser et al., 2021), while Chameleon, Emu3, and Janus extend this paradigm to unified visual understanding and generation (Team, 2024; Wang et al., 2024; Wu et al., 2025). However, finite codebooks constrain the precision of continuous visual signals. Continuous-output models avoid vector quantization: MAR predicts image latents with a diffusion loss (Li et al., 2024), and other unified architectures combine autoregressive language modeling with diffusion-style image generation, sometimes involving the shared backbone directly in denoising (Zhou et al., 2025; Xie et al., 2025; Deng et al., 2025). These models retain denoising objectives and iterative continuous sampling. AutoFlow instead uses a conditional normalizing-flow head for continuous images and actions, preserving the backbone's autoregressive role while enabling exact latent likelihood and direct flow sampling. We study the resulting training interface through downstream transfer and matched-batch optimization diagnostics.

3 METHOD

3.1 AUTOREGRESSIVE MODELING BEYOND DISCRETE TOKENS

Our goal is to extend autoregressive modeling from discrete tokens to native continuous signals, such as images or robotic actions. Let $\mathbf{s} = (s_1, \ldots, s_T)$ be a multimodal sequence, where each element is either a language token $x_t \in \mathcal{V}$ or a continuous signal $y_t \in \mathbb{R}^d$. We model the sequence left-to-right as $p(\mathbf{s}) = \prod_t p(s_t \mid s_{<t})$.

Given the previous context $s_{\leq t}$, a causal transformer produces an autoregressive state h_t. The next-element distribution is defined by the modality of the target:

$$p(s_{t+1} \mid s_{\leq t}) = \begin{cases} p_\theta(x_{t+1} \mid h_t), & x_{t+1} \in \mathcal{V}, \\ p_\psi(y_{t+1} \mid h_t), & y_{t+1} \in \mathbb{R}^d. \end{cases} \tag{1}$$

For language, p_θ is the standard categorical distribution over the vocabulary. The central challenge lies in parameterizing the continuous distribution $p_\psi(y_{t+1} \mid h_t)$ such that it is expressive, efficient, and seamlessly aligns with the training of the shared autoregressive backbone.

The Bottleneck of Diffusion for Continuous Prediction. Recent approaches attempt to parameterize this continuous distribution using diffusion models. Methods like Transfusion (Zhou et al., 2025) or Bagel (Deng et al., 2025) interleave a bidirectional diffusion process directly into the autoregressive (AR) backbone. However, this fundamentally alters the architecture, requiring extensive retraining and specialized routing, like Mixture of Experts (Tong et al., 2026) and Mixture of Transformers (Deng et al., 2025; Agarwal et al., 2026). When adapting existing large language models (LLMs), this heavy modification can induce catastrophic forgetting.

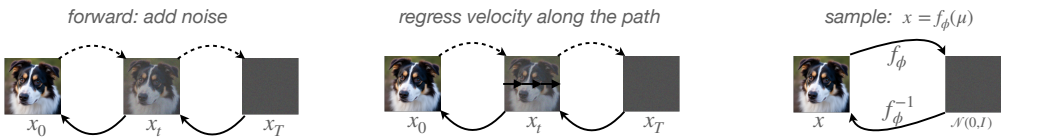
A more lightweight alternative attaches a conditional diffusion head to a frozen or minimally modified AR transformer (Li et al., 2024; Ma et al., 2025). Given a continuous target y, diffusion constructs a noisy variant $y_\tau = \alpha_\tau y + \sigma_\tau \epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$, and trains a network to denoise it conditioned on the AR state h_t:

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{y,\epsilon,\tau}\left[\|\epsilon - \epsilon_\psi(y_\tau, \tau, h_t)\|_2^2\right]. \tag{2}$$

While this preserves the autoregressive architecture, it creates a difficult learning problem for the shared representation. Diffusion conditions generation on a corrupted target y_τ, whereas understanding operates on clean observations. Moreover, the denoising objective creates a *lose-lose situation* for learning from generation. At high noise levels, much of the target is hidden, making the prediction highly uncertain. At low noise levels, much of the target is already revealed, allowing the denoiser to solve the task with limited dependence on the autoregressive context h_t. In either case, the signal passed back to the shared backbone can be weaker than the direct supervision we would like from next-vector prediction. Our goal is therefore to model the full conditional distribution $p(y_{t+1} \mid h_t)$ directly from the clean target, without introducing a noise-conditioned reconstruction problem.

3.2 CONDITIONAL NORMALIZING FLOWS FOR CONTINUOUS PREDICTION

This observation suggests a different interface for continuous prediction: rather than learning to recover the target from corruption, we directly model its conditional distribution from the autoregressive state. The key advantage of a normalizing flow is invertibility: although generation maps noise



(a) Diffusion: predict noise (b) Flow Matching: predict velocity (c) Normalizing Flow: invertible distribution map

Figure 2: Comparison of three continuous-generation objectives. Diffusion models learn to predict noise and generate samples through iterative denoising. Flow matching models learn a velocity field along a path between data and noise and generate samples by integrating an ODE. Normalizing flows learn an invertible distribution map between a base density and the data distribution, enabling both likelihood evaluation and one-step sampling.

to a clean continuous output, training can run the same mapping in reverse, from the clean target to a simple Gaussian distribution. We implement this with a conditional normalizing flow head module while leaving the autoregressive backbone unchanged.

Given an autoregressive state h_t, the flow learns an invertible mapping $y = T_\psi(u; h_t)$ from a simple base distribution $u \sim p_0(u)$ to the distribution of the next continuous vector. For simplicity, we use a standard channel-wise affine coupling (Dinh et al., 2016; Kingma & Dhariwal, 2018):

$$[u_a, u_b] = \mathrm{split}_c(u), \qquad y_a = u_a, \qquad y_b = u_b \odot \exp\big(s_\psi(u_a, h_t)\big) + t_\psi(u_a, h_t), \tag{3}$$

where split_c partitions the input vector and s_ψ, t_ψ are conditioned on h_t.

The key advantage of invertibility appears during training. Rather than corrupting the target and learning to recover it, we start from the clean target y and run the flow backward to obtain $u = T_\psi^{-1}(y; h_t)$. We can therefore evaluate its exact conditional likelihood using change of variables:

$$\log p_\psi(y \mid h_t) = \log p_0(T_\psi^{-1}(y; h_t)) + \log \left| \det \frac{\partial T_\psi^{-1}(y; h_t)}{\partial y} \right|. \tag{4}$$

The first term encourages the transformed targets to follow the Gaussian base distribution, while the Jacobian term accounts for the corresponding change in volume. With invertibility, this provides a tractable likelihood while preserving a one-to-one mapping between targets and latent codes.

This construction has three useful consequences. First, the model learns directly from clean targets, providing a direct likelihood signal to the autoregressive state without a noisy surrogate objective. Second, it models a full conditional distribution rather than a point estimate: unlike squared-error regression, which favors the conditional mean, the flow can represent multiple plausible outputs. Third, training and generation are simply two directions of the same learned mapping. Training maps clean targets to Gaussian latents; at inference, we reverse the direction, sample $u \sim p_0$, and evaluate $T_\psi(u; h_t)$ once to obtain a continuous prediction without iterative denoising.

Unified Likelihood Training. AutoFlow places discrete and continuous prediction under the same probabilistic objective. For text, we optimize the standard next-token cross-entropy, $-\log p_\theta(x_t \mid h_{t-1})$; for continuous signals, we optimize the negative flow likelihood, $-\log p_\psi(y_t \mid h_{t-1})$. The full objective is:

$$\mathcal{L} = \mathcal{L}_{\text{text}} + \lambda \mathcal{L}_{\text{cont}}. \tag{5}$$

Both modalities therefore train the shared autoregressive backbone by maximizing the likelihood of the next observation, without introducing a separate denoising objective for continuous signals. We hypothesize that this closer training interface makes generation more useful for the shared representation; Table 3 tests this through gradient variance, context dependence, and generation–understanding gradient alignment.

Autoregressive Generation. At inference, AutoFlow generates multimodal sequences left to right. Given a prefix $s_{\leq t}$, the transformer produces h_t; the next element is sampled either from the categorical language head or, for a continuous target, from the conditional flow.

Because each continuous prediction requires only a single flow evaluation, AutoFlow avoids the iterative sampling required by diffusion. This is particularly useful when prediction occurs repeatedly inside a larger computation loop: interleaved multimodal reasoning alternates between generation and perception, while robot policies must produce actions under tight latency constraints.

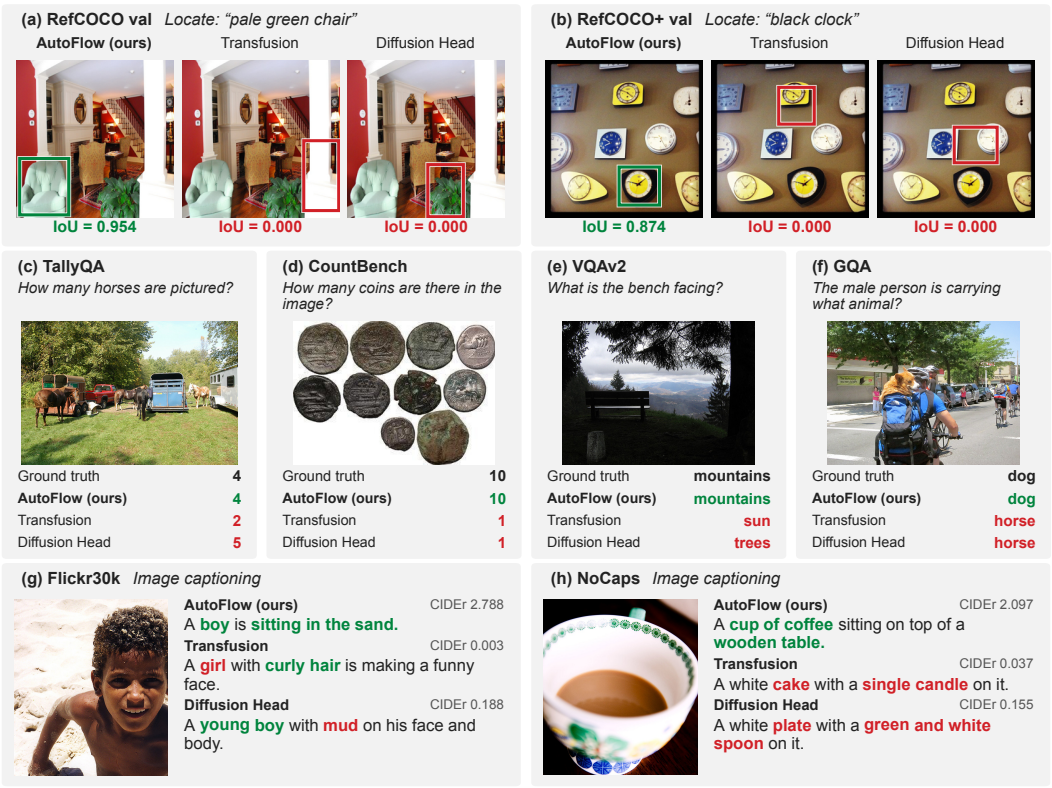


Figure 3: Downstream visual understanding after T2I-only pretraining and understanding fine-tuning: (a,b) visual grounding on RefCOCO and RefCOCO+; (c,d) counting on TallyQA and CountBench; (e,f) visual question answering on VQAv2 and GQA; and (g,h) image captioning on Flickr30k and NoCaps. Green/red marks correct/incorrect predictions or highlighted caption details.

Table 1: Generation-to-understanding transfer after T2I-only pretraining and downstream fine-tuning (↑). †: CIDEr; other task scores are accuracy (%). RefCOCO and RefCOCO+ report validation Acc@0.5. Bold marks the best scores; Avg. is the unweighted mean of eight task scores.

Method	Flickr†	NoCaps†	RefCOCO	RefCOCO+	TallyQA	CountBench	VQAv2	GQA	Avg.
Transfusion	23.9	25.5	32.4	17.8	48.3	9.8	53.2	44.7	32.0
Diffusion Head	33.7	36.1	46.9	27.6	51.6	18.5	**56.0**	46.4	39.6
AutoFlow (Ours)	**34.2**	**36.9**	**61.0**	**43.8**	**53.4**	**19.7**	**56.0**	**47.0**	**44.0**

4 Experiments

We evaluate generation-to-understanding transfer and interleaved visual reasoning through fine-tuning, then assess pretrained generation and understanding directly. Gradient diagnostics probe the underlying optimization behavior, and robot control tests the approach on continuous action prediction.

4.1 Experimental Setup

Models and pretraining settings. We compare AutoFlow with a Transfusion-style model (Zhou et al., 2025) and a MAR-inspired Diffusion Head (Li et al., 2024), all using the same autoregressive LLM backbone. AutoFlow and the Diffusion Head differ only in their continuous-output heads, whereas Transfusion uses the backbone itself for diffusion denoising. We pretrain on LLaVA-ReCap-CC12M (Liu et al., 2024) and COCO captioning data under two settings: *T2I-only*, which learns text-to-image generation, and *T2I–I2T co-training*, which jointly learns image generation and image-to-text captioning. Figure 8 summarizes the checkpoint usage; full hyperparameters are in Appendix A.

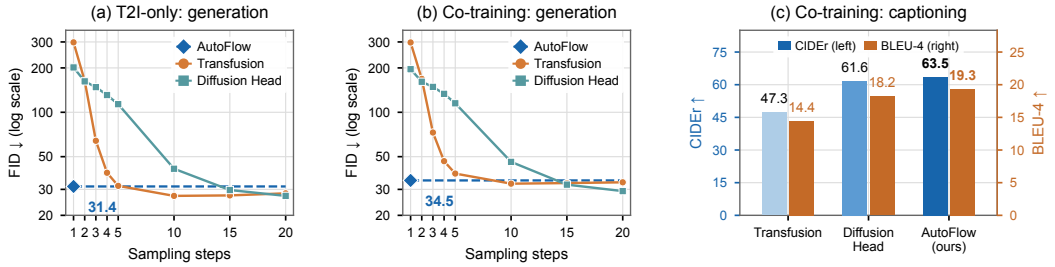


Figure 4: Direct evaluation of pretrained generation and understanding, before downstream fine-tuning. (a,b) FID versus sampling steps under T2I-only pretraining and T2I–I2T co-training (log scale). AutoFlow uses one step; dashed lines mark its FID for reference. (c) COCO Karpathy test (5k) captioning after 200k co-training steps: CIDEr (blue, left axis) and BLEU-4 (orange, right axis).

Generation-to-understanding transfer. Following the evaluation-through-fine-tuning paradigm of PaliGemma 2 (Steiner et al., 2024), we initialize understanding fine-tuning from *T2I-only checkpoints*. This isolates generation-to-understanding transfer from direct I2T pretraining supervision and the choice of generation–understanding loss weight λ. We evaluate eight benchmarks: Flickr and NoCaps (Agrawal et al., 2019) captioning; RefCOCO and RefCOCO+ validation grounding (Yu et al., 2016); TallyQA (Acharya et al., 2019) and CountBench (Paiss et al., 2023) counting; and VQAv2 (Goyal et al., 2017) and GQA (Hudson & Manning, 2019) question answering. Captioning uses CIDEr (Vedantam et al., 2015); other tasks use accuracy, with IoU $\geq$ 0.5 for grounding. Fine-tuning budgets and cross-dataset evaluations are specified in Appendix A.2.

Interleaved reasoning fine-tuning. We initialize from *T2I–I2T co-trained checkpoints*, since interleaved reasoning requires both generating and interpreting images. We fine-tune separately on ThinkMorph (Gu et al., 2026b) navigation and Jigsaw for 4,000 steps, using 6,000 examples per task and 256-pixel images. Navigation evaluates action sequences on 400 FrozenLake grids (3×3–6×6); Jigsaw evaluates piece-order choices on 400 problems. We report answer accuracy with 1 image-sampling step for AutoFlow, 1 or 10 for Transfusion, and 1 or 20 for the Diffusion Head. Traces follow the Zebra-CoT serialization format (Li et al., 2025).

Direct pretrained-model evaluation. Before downstream fine-tuning, we evaluate 200K-step checkpoints from *both* pretraining settings with FID (Heusel et al., 2017), using 1 base sampling step for AutoFlow and 1–20 for the diffusion-based baselines. We additionally evaluate *co-trained checkpoints* on COCO Karpathy test (Karpathy & Fei-Fei, 2015) using CIDEr and BLEU-4. These measurements assess pretrained generation and captioning directly (Figure 4), complementing the fine-tuning evaluations above.

Optimization diagnostics. Using T2I–I2T co-trained checkpoints and matched training batches, we measure intrinsic gradient variance, the generation gradient on the conditioning representation, and generation–understanding gradient cosine to assess optimization stability, conditioning strength, and objective compatibility, respectively.

Real-robot control. Robot policies share Qwen3-VL-2B-Instruct with Diffusion, Flow Matching (Black et al., 2026; Intelligence et al., 2025), or normalizing-flow action heads. We evaluate a 6-DoF Piper arm (AgileX Robotics) with a 1-DoF gripper on cup placement onto a coaster (PC), stacking a blue cube on a wooden cube (SC), and phone placement onto a stand (PP). Policies predict action chunks from the backbone state and robot history. Each method runs 30 rollouts per task; we report success counts, mean success rate, and Wilson 95% intervals over 90 pooled rollouts. Action-head sampling budgets are 16 for Diffusion, 5 for Flow Matching, and 1 for AutoFlow.

4.2 Transfer from Image Generation to Understanding

Table 1 compares downstream understanding after T2I-only pretraining and understanding fine-tuning. AutoFlow leads on seven tasks and ties the Diffusion Head on VQAv2, with an average score of 44.0 versus 39.6 for the Diffusion Head and 32.0 for Transfusion. The largest gains over the Diffusion Head are in grounding: 14.1 points on RefCOCO and 16.2 on RefCOCO+. Smaller gains in captioning, counting, and GQA show that the benefit extends across understanding tasks. Figure 3 presents qualitative comparisons of downstream grounding, counting, question answering, and captioning under the same T2I-only transfer protocol. In these examples, AutoFlow predicts the correct regions and answers and captures key caption details that the baselines miss.

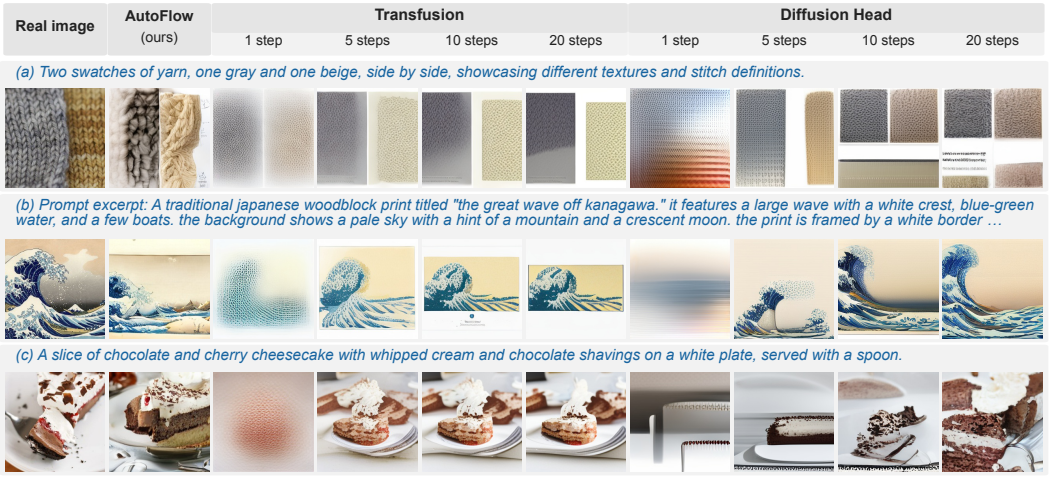


Figure 5: Selected image-generation examples from 200K-step T2I-only checkpoints. Each row shares a training-set prompt and random seed; the associated real image is shown on the left. Transfusion and the Diffusion Head use 1, 5, 10, and 20 sampling steps.

4.3 Interleaved Visual Chain-of-Thought Reasoning

Visual chain-of-thought can support multimodal reasoning by generating intermediate visual states that complement textual analysis, as demonstrated by ThinkMorph (Gu et al., 2026b). Starting from T2I–I2T co-trained checkpoints, we fine-tune on interleaved traces and compare final-answer accuracy on navigation and Jigsaw across image-sampling budgets (Table 2).

Table 2: Interleaved reasoning accuracy after fine-tuning T2I–I2T co-trained checkpoints on spatial navigation and Jigsaw. Steps denotes the sampling steps per generated image; each column uses the same budget on both tasks.

Method	Transfusion		Diffusion Head		AutoFlow (Ours)
Steps ↓	1	10	1	20	1
Spatial navigation ↑	26.2	26.8	27.8	30.8	**38.8**
Jigsaw ↑	41.5	41.8	43.6	45.5	**46.8**

With one image-sampling step, AutoFlow reaches 38.8% accuracy on navigation and 46.8% on Jigsaw, exceeding the 20-step Diffusion Head (30.8%/45.5%) and 10-step Transfusion (26.8%/41.8%). It also leads both baselines at an equal one-step budget. The gains are larger on navigation, but both tasks show higher answer accuracy with fewer sampling evaluations when generation and understanding operate within the same reasoning chain. Figure 6 highlights AutoFlow's consistency across the displayed reasoning steps, generated images, and correct final answers. The baselines can return correct answers despite intermediate errors: Transfusion gives the correct navigation answer but generates a conflicting path, while the Diffusion Head selects the correct Jigsaw order but reasons about a nonexistent coffee machine.

4.4 Pretrained Generation and Understanding

Figure 4 evaluates the 200K-step pretrained checkpoints before downstream fine-tuning. Panels (a,b) compare FID across sampling budgets for T2I-only pretraining and T2I–I2T co-training, respectively. With one base sampling step, AutoFlow obtains 31.4 FID after T2I-only pretraining and 34.5 after T2I–I2T co-training, versus 298.1/297.6 for Transfusion and 201.5/196.0 for the Diffusion Head at one step. Both baselines improve with additional steps: 10-step Transfusion reaches 27.1/32.8, and the 20-step Diffusion Head reaches 27.1/29.2. Thus, AutoFlow offers much better single-step quality, while the multi-step baselines attain lower FID. Inference FLOPs and the corresponding FID comparisons are provided in Appendix C.

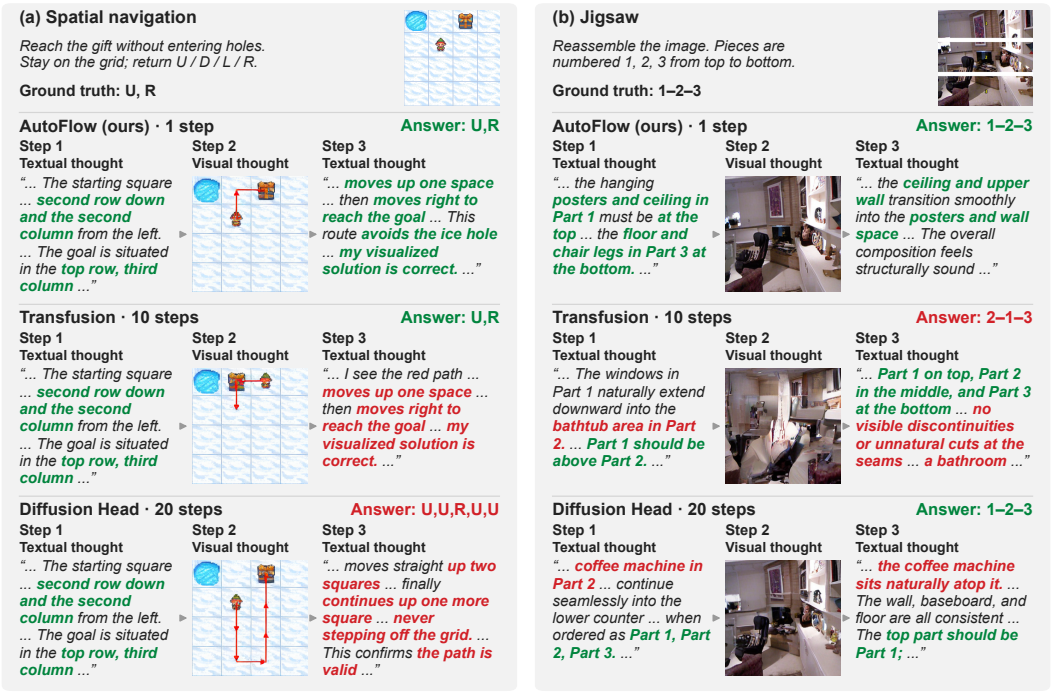


Figure 6: Interleaved reasoning on (a) navigation and (b) Jigsaw. Read each row left to right: text → generated image → text; final answers appear beside method names. Green/red marks correct/incorrect claims or answers. Excerpts are verbatim (ellipses mark omissions); images are unmodified. See Appendix B for annotation and selection details.

Table 3: Matched-batch optimization diagnostics on T2I–I2T co-trained checkpoints: gradient variance, conditioning-signal magnitude, and generation–understanding alignment. ↓ indicates lower is better. ↑ indicates higher is better.

Measurement	AutoFlow (Ours)	Diffusion Head	Transfusion
Intrinsic gradient variance ↓	**0.0039**	0.0366	1.1444
Generation gradient on conditioning ↑	$\mathbf{4.58\times10^{-4}}$	2.27×10^{-5}	—
Generation–understanding gradient cosine ↑	**0.276**	0.132	0.104

Understanding under co-training. At 200k co-training steps, AutoFlow achieves 63.5 CIDEr and 19.3 BLEU-4 on COCO Karpathy test, exceeding the Diffusion Head (61.6/18.2) and Transfusion (47.3/14.4) in Figure 4c. Co-training thus combines stronger captioning with one-step base sampling. These direct captioning scores complement the downstream transfer results in Section 4.2. Figure 5 visualizes selected image-generation outputs from 200K-step T2I-only checkpoints. Each row shares a prompt and random seed and includes the associated real training image.

4.5 Mechanistic Evidence for the Transfer Advantage

Using T2I–I2T co-trained checkpoints for all three diagnostics, we test whether the continuous-generation objective provides a stable training signal that engages the autoregressive condition and aligns with understanding. Diffusion's noise-dependent reconstruction task motivates this hypothesis (Section 3.1); Table 3 reports the matched-batch diagnostics.

Optimization stability and conditioning. AutoFlow's intrinsic gradient variance is 0.0039, approximately $293\times$ lower than Transfusion's and $9.4\times$ lower than the Diffusion Head's. Its generation gradient on the conditioning representation is approximately $20\times$ larger than the Diffusion Head's. Timestep-wise measurements further show that the Diffusion Head has weaker conditioning gradients near both the clean and noisy ends of its diffusion trajectory (Appendix F.1).

Generation–understanding alignment. AutoFlow achieves the highest gradient cosine, 0.276 versus 0.132 for the Diffusion Head and 0.104 for Transfusion. All are positive, with AutoFlow showing

Table 4: Real-robot success over 30 rollouts per task. All methods use Qwen3-VL-2B-Instruct as the transformer backbone. We evaluate three tasks: cup placement (PC); cube stacking (SC); and place phone on the charger (PP). Average success equals number of pooled successes divided by 90. Wilson intervals use that pooled count, where we report the 95% confidence interval.

Head	Params	Steps ↓	PC	SC	PP	Avg. (%) ↑	Wilson 95% CI
Diffusion Head	73.9M	16	17/30	11/30	5/30	36.7	[27.4, 47.0]
Flow Matching	73.8M	5	16/30	13/30	7/30	40.0	[30.5, 50.3]
AutoFlow (Ours)	**52.5M**	**1**	**26/30**	**23/30**	**11/30**	**66.7**	**[56.4, 75.5]**

the strongest alignment. Together, these diagnostics support the hypothesis that exact-likelihood generation provides training signals more compatible with understanding, consistent with the transfer and interleaved-reasoning results.

4.6 Vision-Language-Action Policy Learning

Table 4 evaluates the same continuous-prediction interface on three robot tasks, using the protocol in Section 4.1. AutoFlow succeeds in 60 of 90 rollouts and leads on every task. Its 66.7% average exceeds the Diffusion Head by 30.0 points and Flow Matching by 26.7 points, while its 52.5M-parameter head is approximately 29% smaller and uses one sampling step rather than 16 or 5. This combines higher control success with a smaller head and fewer sampling evaluations. Figure 7 shows selected successful AutoFlow executions alongside failed Diffusion Head rollouts.

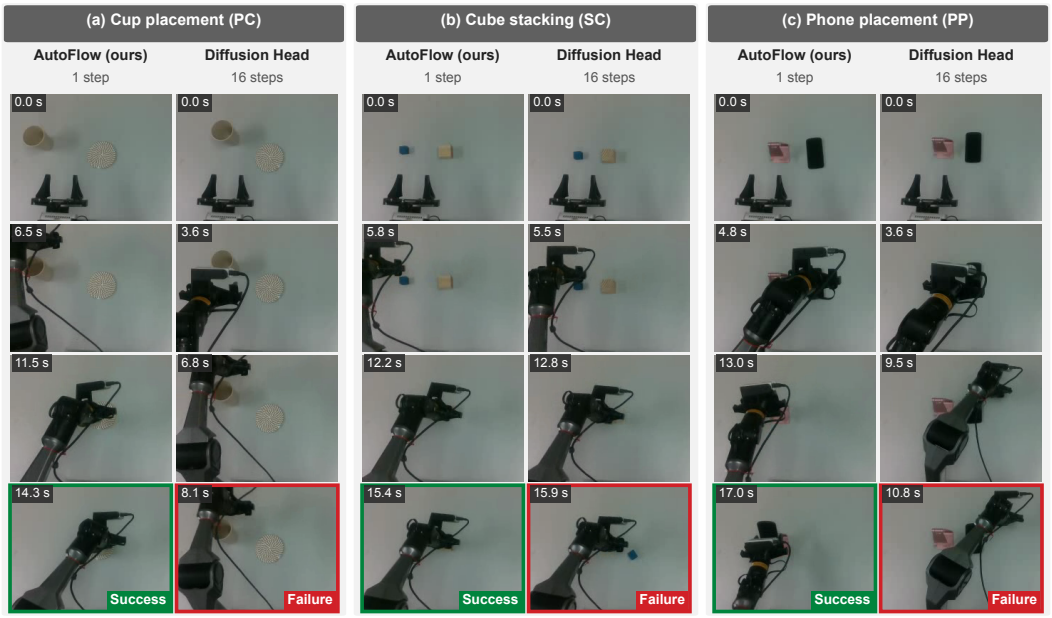


Figure 7: Selected robot rollouts for (a) cup placement, (b) cube stacking, and (c) phone placement. Each method column shows four state-based keyframes from top to bottom in a fixed overhead view; AutoFlow ends at the first visually verified successful release. Green/red borders mark successful/failed rollouts. AutoFlow uses one action-head sampling step and the Diffusion Head uses 16. AutoFlow outperforms diffusion with lower action latency and higher success rate on real robots.

5 Conclusion

We introduced AutoFlow, a framework for extending next-token prediction to continuous signals through conditional normalizing flows. Our main result is that directly modeling the likelihood of the next continuous vector better connects generation and understanding: training for image generation produces representations that transfer more effectively to visual understanding than diffusion-based alternatives. The same interface supports efficient single-pass generation, interleaved visual reasoning, and real-time robot control. These results suggest that next-vector prediction can extend the simplicity and generality of next-token prediction beyond language.

REFERENCES

Manoj Acharya, Kushal Kafle, and Christopher Kanan. Tallyqa: Answering complex counting questions. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 8076–8084, 2019.

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.

Niket Agarwal, Arslan Ali, Jon Allen, Martin Antolini, Adeline Aubame, Alisson Azzolini, Junjie Bai, Maciej Bala, Yogesh Balaji, Josh Bapst, et al. Cosmos 3: Omnimodal world models for physical ai. *arXiv preprint arXiv:2606.02800*, 2026.

AgileX Robotics. PiPER robotic arm. `https://global.agilex.ai/products/piper`. Accessed: 2026-09-25.

Harsh Agrawal, Karan Desai, Yufei Wang, Xinlei Chen, Rishabh Jain, Mark Johnson, Dhruv Batra, Devi Parikh, Stefan Lee, and Peter Anderson. Nocaps: Novel object captioning at scale. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8947–8956. IEEE, 2019.

Lynton Ardizzone, Carsten Lüth, Jakob Kruse, Carsten Rother, and Ullrich Köthe. Guided image generation with conditional invertible neural networks. *arXiv preprint arXiv:1907.02392*, 2019.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2026. URL `https://arxiv.org/abs/2410.24164`.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2024.

Ricky TQ Chen, Jens Behrmann, David K Duvenaud, and Jörn-Henrik Jacobsen. Residual flows for invertible generative modeling. *Advances in neural information processing systems*, 32, 2019.

Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. Janus-pro: Unified multimodal understanding and generation with data and model scaling. *arXiv preprint arXiv:2501.17811*, 2025.

Chaorui Deng, Deyao Zhu, Kunchang Li, Chenhui Gou, Feng Li, Zeyu Wang, Shu Zhong, Weihao Yu, Xiaonan Nie, Ziang Song, et al. Emerging properties in unified multimodal pretraining. *arXiv preprint arXiv:2505.14683*, 2025.

Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural spline flows. *Advances in neural information processing systems*, 32, 2019.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12873–12883, 2021.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6904–6913, 2017.

Jiatao Gu, Tianrong Chen, David Berthelot, Huangjie Zheng, Yuyang Wang, Ruixiang Zhang, Laurent Dinh, Miguel Angel Bautista, Joshua Susskind, and Shuangfei Zhai. Starflow: Scaling latent normalizing flows for high-resolution image synthesis. *Advances in Neural Information Processing Systems*, 38:120986–121022, 2026a.

Jiawei Gu, Yunzhuo Hao, Huichen Wang, Linjie Li, Michael Qizhe Shieh, Yejin Choi, Ranjay Krishna, and Yu Cheng. Thinkmorph: Emergent properties in multimodal interleaved chain-of-thought reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 141405–141447, 2026b.

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. *Advances in neural information processing systems*, 30, 2017.

Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *International conference on machine learning*, pp. 2722–2730. PMLR, 2019.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6693–6702. IEEE, 2019.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL `https://arxiv.org/abs/2504.16054`.

Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3128–3137, 2015.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018.

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29, 2016.

Ang Li, Charles Wang, Deqing Fu, Kaiyu Yue, Zikui Cai, Wang Bill Zhu, Ollie Liu, Peng Guo, Willie Neiswanger, Furong Huang, et al. Zebra-cot: A dataset for interleaved vision language reasoning. *arXiv preprint arXiv:2507.16746*, 2025.

Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *Advances in Neural Information Processing Systems*, 37: 56424–56445, 2024.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llavanext: Improved reasoning, ocr, and world knowledge, 2024.

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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7739–7751. IEEE, 2025.

Roni Paiss, Ariel Ephrat, Omer Tov, Shiran Zada, Inbar Mosseri, Michal Irani, and Tali Dekel. Teaching clip to count to ten. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3147–3157. IEEE, 2023.

George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. *Advances in neural information processing systems*, 30, 2017.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

Andreas Steiner, André Susano Pinto, Michael Tschannen, Daniel Keysers, Xiao Wang, Yonatan Bitton, Alexey Gritsenko, Matthias Minderer, Anthony Sherbondy, Shangbang Long, et al. Paligemma 2: A family of versatile vlms for transfer. *arXiv preprint arXiv:2412.03555*, 2024.

Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*, 2024.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

Shengbang Tong, David Fan, John Nguyen, Ellis Brown, Gaoyue Zhou, Shengyi Qian, Boyang Zheng, Théophane Vallaeys, Junlin Han, Rob Fergus, et al. Beyond language modeling: An exploration of multimodal pretraining. *arXiv preprint arXiv:2603.03276*, 2026.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Brian L Trippe and Richard E Turner. Conditional density estimation with bayesian normalising flows. *arXiv preprint arXiv:1802.04908*, 2018.

Michael Tschannen, André Susano Pinto, and Alexander Kolesnikov. Jetformer: An autoregressive generative model of raw images and text. In *International Conference on Learning Representations*, volume 2025, pp. 85995–86016, 2025.

Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4566–4575, 2015.

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.

Christina Winkler, Daniel Worrall, Emiel Hoogeboom, and Max Welling. Learning likelihoods with conditional normalizing flows. *arXiv preprint arXiv:1912.00042*, 2019.

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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12966–12977. IEEE, 2025.

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. In *International Conference on Learning Representations*, volume 2025, pp. 28240–28264, 2025.

Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In *European conference on computer vision*, pp. 69–85. Springer, 2016.

Shuangfei Zhai, Ruixiang Zhang, Preetum Nakkiran, David Berthelot, Jiatao Gu, Huangjie Zheng, Tianrong Chen, Miguel Angel Bautista, Navdeep Jaitly, and Josh Susskind. Normalizing flows are capable generative models. *arXiv preprint arXiv:2412.06329*, 2024.

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. In *International Conference on Learning Representations*, volume 2025, pp. 6446–6469, 2025.

AI USE STATEMENT

Generative AI tools were used to polish the writing of this paper, assist in checking the correctness of experimental implementations, and help search for relevant literature. They were not used to develop the research ideas or methods, or draw conclusions from the results.

A EXPERIMENTAL PROTOCOL AND IMPLEMENTATION DETAILS

Training settings and evaluation scope. We use two image–text pretraining settings on LLaVA-ReCap-CC12M (Liu et al., 2024) and COCO captioning data: T2I-only and T2I–I2T co-training. Figure 8 summarizes their downstream uses. Understanding transfer starts from T2I-only checkpoints (Table 1 and Figure 3), excluding direct I2T supervision during image–text pretraining. Interleaved reasoning starts from co-trained checkpoints, which support both image generation and interpretation. Separately, direct evaluation at 200K pretraining steps measures FID for both settings and COCO Karpathy test (5K) CIDEr/BLEU-4 for co-training, before any downstream fine-tuning (Figure 4). All three gradient diagnostics use T2I–I2T co-trained checkpoints. Robot policies use the separate training protocol in Appendix A.4.

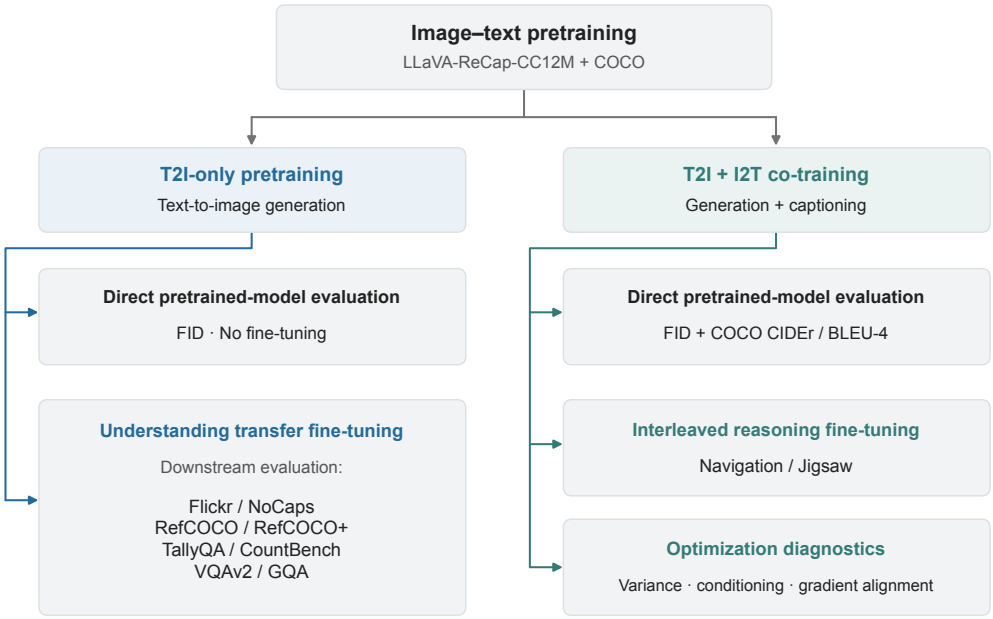


Figure 8: Image–text pretraining and evaluation protocols. T2I-only checkpoints initialize understanding transfer (Table 1; Figure 3); T2I–I2T co-trained checkpoints initialize interleaved reasoning (Table 2). Direct evaluation uses pretrained checkpoints without fine-tuning (Figure 4). All three optimization diagnostics use co-trained checkpoints. Evaluation and fine-tuning are separate uses of each checkpoint.

A.1 IMAGE PRETRAINING

Table 5 summarizes T2I-only pretraining and T2I–I2T co-training. The training corpus combines LLaVA-ReCap-CC12M with COCO captioning data. Long and short captions are sampled with equal probability.

Table 5: Image-pretraining configuration. Counts refer to image–text training samples.

Hyperparameter	Value
Backbone	Qwen2.5-1.5B-Instruct
Attention and positions	FlashAttention-2 for the autoregressive backbone; 1D RoPE. Transfusion uses SDPA with bidirectional attention within image-token blocks.
Image representation	256-pixel images; KL-16 VAE (`kl16.ckpt`); 16×16 latent grid with 16 channels (256 visual tokens); latent scale 0.201779
Training corpus	LLaVA-ReCap-CC12M: 9.79M samples; COCO: 567K samples; approximately 10.35M combined
Caption sampling	Long/short captions: 50%/50%; caption dropout: 0.1; maximum length: 256 tokens
Effective batch size	28 images/GPU $\times$ 8 GPUs $= 224$; gradient accumulation: 1
Training budget	200K steps (44.8M sample presentations, approximately 4.3 epochs)
Co-training task mixture	Text-to-image / image-to-text: 0.8 / 0.2
Optimizer	AdamW; $\beta = (0.9, 0.95)$; $\epsilon = 10^{-8}$
Weight decay / gradient clip	0.1 / 3.0
EMA decay	0.9999
Warmup phase	First 5,000 steps: head/projector LR rises from 0 to 5×10^{-4}, then decays with a cosine schedule to 1.2×10^{-5}
Subsequent LR schedule	Cosine decay from 1.2×10^{-5} to 2.5×10^{-6} through step 200K

A.2 Understanding Fine-Tuning

All compared methods use the common optimization settings in Table 6, initialized from the EMA weights of their 200K-step T2I-only pretraining checkpoints. Table 7 specifies the task-specific training budgets and cross-dataset evaluations. NoCaps, RefCOCO+, and CountBench receive no additional task-specific fine-tuning.

Table 6: Shared hyperparameters for downstream understanding fine-tuning.

Hyperparameter	Value
Initialization	EMA weights from the 200K-step T2I-only pretraining checkpoint
Trainable components	Full model except the VAE
Optimizer	AdamW; $\beta = (0.9, 0.95)$; weight decay: 0
Learning rate	2×10^{-5}; 200-step warmup; cosine decay to 2×10^{-6}
Effective batch size	28 samples/GPU $\times$ 8 GPUs $= 224$
Maximum text length	96 tokens
Gradient clip	1.0
Precision / seed	Bfloat16 autocast / 0

Table 7: Fine-tuning budgets and evaluation datasets. Cross-dataset evaluations use the indicated fine-tuned checkpoint without further adaptation.

Fine-tuning dataset	Steps	Evaluation
Flickr30k captioning	2,000	Flickr30k
COCO captioning	2,000	NoCaps (cross-dataset)
RefCOCO	2,000	RefCOCO; RefCOCO+ (cross-dataset)
TallyQA	4,000	TallyQA; CountBench (cross-dataset)
VQAv2	6,000	VQAv2
GQA	6,000	GQA

A.3 Interleaved Reasoning Fine-Tuning

Navigation and Jigsaw initialize from the 200K-step T2I–I2T co-trained checkpoints and each use 4,000 fine-tuning steps with the settings in Table 8.

Table 8: Shared training hyperparameters for interleaved navigation and Jigsaw reasoning.

Hyperparameter	Value
Initialization	200K-step T2I–I2T co-trained checkpoint
Optimizer	AdamW; $\beta = (0.9, 0.95)$; weight decay: 0
Learning rate	2×10^{-5}; 200-step warmup; cosine decay to 2×10^{-6}
Effective batch size	128; 4 samples/GPU, using 8 GPUs with accumulation 4 or 4 GPUs with accumulation 8
Training steps	4,000 per task
Gradient clip	1.0
Precision / seed	Bfloat16 / 0

A.4 VLA Policy Learning

Table 9 gives the robot-policy architecture and training settings. The three methods share the same backbone, observation history, action horizon, and optimization configuration; their action heads differ. Sampling steps match the operating points reported in the main experiments.

Table 9: Hyperparameters for VLA policy learning. Head parameters are additional to the shared Qwen3-VL-2B-Instruct backbone.

Hyperparameter	Diffusion Head	Flow Matching	AutoFlow
Architecture			
Backbone		Qwen3-VL-2B-Instruct	
Vision tower		Frozen	
State projection		Linear ($7 \to 2048$)	
Head type	MLP	MLP	Affine coupling
Head depth / width		8 / 1280	
Head parameters	73.9M	73.8M	52.5M
Data			
Observation history		2	
Action horizon / dimension		16 / 7	
Action normalization		$[-1, 1]$	
Training / validation split		95% / 5%	
Training			
Optimizer		AdamW	
Learning rate		8×10^{-5}	
(β_1, β_2)		$(0.9, 0.999)$	
Weight decay		10^{-2}	
Batch size		64	
LR warmup		Linear, 500 steps	
Gradient clip		3.0	
Training epochs		200	
EMA		Not used	
Diffusion and sampling			
Noise schedule	Cosine	—	—
Training diffusion timesteps	100	—	—
Inference sampling steps	16	5	1

B Qualitative Analysis for Interleaved Reasoning

Figure 6 uses navigation test index 135 (4×4) and Jigsaw test index 257 (three pieces). Examples were selected for clear AutoFlow visual predictions, correct final answers with consistent task-level reasoning, and recognizable baseline images. This is a purposive qualitative comparison, not a random sample. Both displayed AutoFlow outputs use the same variant, with exact variant and checkpoint metadata retained in the source records. Source images are reproduced without retouching. Text excerpts are verbatim spans from before and after the generated-image marker; ellipses mark omitted text. Green/red highlights annotate individual claims against the task input and reference, including correct claims made by baselines; claims about a generated image are also checked against that

image. Unhighlighted text is not assigned a correctness label. In the navigation example, Transfusion moves the player from $(2, 2)$ to $(1, 3)$ and the goal from $(1, 3)$ to $(1, 2)$ (one-based row, column), with arrows inconsistent with its stated up-then-right path. Its Step 3 assertion that the visualized solution is correct is therefore highlighted in red, while its final answer remains green because it solves the original grid. Complete responses and phrase-level annotations are retained with the figure assets. In Jigsaw, the Diffusion Head returns the correct piece order but bases its explanation on a nonexistent coffee machine, while Transfusion describes bathroom fixtures absent from the input. In contrast, AutoFlow's displayed reasoning correctly uses the ceiling and posters above and the floor and chair legs below to support its coherent reconstruction and correct piece order. The displayed ground-truth navigation path is independently verified on the input grid.

C Inference FLOPs and Image-Generation Quality

Figure 9 replots the FID sweeps from Figure 4(a,b) against inference FLOPs per image, using the same step-to-compute mapping for generation-only pretraining and co-training. AutoFlow uses 1.6 TFLOPs for its single forward pass. Across the reported sampling budgets, Transfusion increases from 1.4 to 28.1 TFLOPs, while the Diffusion Head increases from 1.6 to 4.9 TFLOPs. The reported configuration uses a 170M-parameter generation head. FLOPs include the backbone, generation head, and additional sampling computation, excluding VAE decoding. The curves show the generation-quality trade-off with computation beyond sampling-step counts.

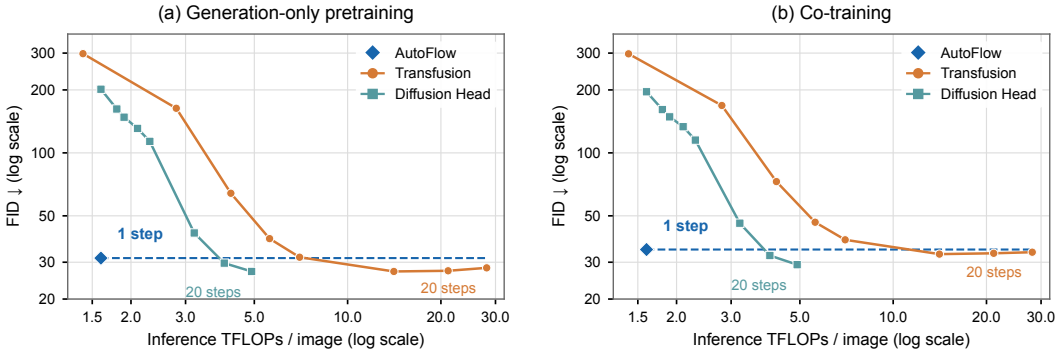


Figure 9: FID versus inference TFLOPs per image for (a) generation-only pretraining and (b) co-training. Both axes use logarithmic scales. Baseline markers correspond to 1, 2, 3, 4, 5, 10, 15, and 20 sampling steps. AutoFlow has one operating point; its dashed line marks the same FID for reference.

D Robot Success versus Sampling Steps

Figure 10 examines how action-head sampling steps affect robot success. This sweep uses 10 rollouts per task; Table 4 reports the subsequent evaluation with 30 rollouts per task. The Diffusion Head generally improves with larger sampling budgets, while Flow Matching achieves its highest average success at five steps and then declines. AutoFlow achieves the highest success on each task and on average with a single sampling step.

E Robot Success Rates and Confidence Intervals

Policies predict continuous action chunks from the backbone state at an <ACT> token, with robot state history injected through <STATE> embeddings. The task order throughout the results is PC (cup), SC (cube stacking), and PP (phone). Each task has 30 evaluation rollouts. Because task sample sizes are equal, the mean of the three task success rates equals the pooled success rate $\hat{p} = k/n$, with $n = 90$. The Diffusion Head succeeds in $17 + 11 + 5 = 33$ rollouts, Flow Matching in $16 + 13 + 7 = 36$, and AutoFlow in $26 + 23 + 11 = 60$.

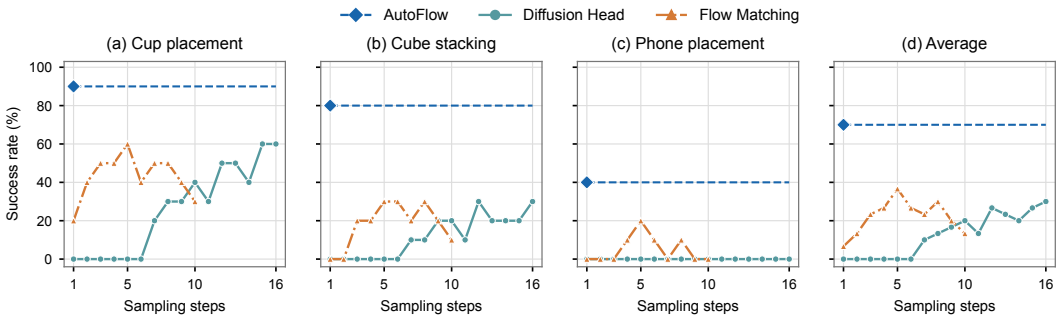


Figure 10: Robot success versus sampling steps: (a) cup placement, (b) cube stacking, (c) phone placement, and (d) the mean across tasks. AutoFlow is evaluated at one step; dashed horizontal lines show its success rates for reference.

We compute the Wilson interval using $z = 1.95996398454$:

$$\frac{\hat{p} + z^2/(2n)}{1 + z^2/n} \pm \frac{z}{1 + z^2/n}\sqrt{\frac{\hat{p}(1-\hat{p})}{n} + \frac{z^2}{4n^2}}. \tag{6}$$

This yields [27.4, 47.0]% for the Diffusion Head, [30.5, 50.3]% for Flow Matching, and [56.4, 75.5]% for AutoFlow. The intervals summarize the pooled rollout outcomes for these three tasks.

The head-size reductions are $(73.9 - 52.5)/73.9 = 28.96\%$ relative to the Diffusion Head and $(73.8 - 52.5)/73.8 = 28.86\%$ relative to Flow Matching. Both round to the 29% reduction reported in the main text.

F Additional Conditioning and Input-Noise Analyses

F.1 Conditioning Gradients across Diffusion Timesteps

Table 10 expands the conditioning-gradient comparison in Table 3 with measurements at individual diffusion timesteps. Here, $\tau = 0$ denotes the noisy end and $\tau = 1$ the clean end of the diffusion trajectory. The Diffusion Head's conditioning gradient is smaller near both ends than at the interior timesteps shown, and remains below AutoFlow at every sampled timestep.

Table 10: Generation-gradient magnitude on the conditioning representation: the reported overall average and selected diffusion timesteps. AutoFlow does not use a diffusion timestep.

Method / timestep	Conditioning-gradient magnitude
AutoFlow	$\mathbf{4.58 \times 10^{-4}}$
Diffusion Head (average)	2.27×10^{-5}
Diffusion Head ($\tau = 0.95$)	1.33×10^{-5}
Diffusion Head ($\tau = 0.75$)	3.00×10^{-5}
Diffusion Head ($\tau = 0.50$)	2.90×10^{-5}
Diffusion Head ($\tau = 0.25$)	2.20×10^{-5}
Diffusion Head ($\tau = 0.00$)	1.40×10^{-5}

This pattern is consistent with the noise-dependent conditioning trade-off discussed in Section 3.1. Near the clean end, the nearly intact input can support reconstruction with limited reliance on the autoregressive condition. Near the noisy end, little target structure remains in the input. The measurements support weaker conditioning supervision at both ends of the trajectory.

F.2 Inference-Time Input-Noise Perturbation

We test the hypothesis that diffusion-style backbone training makes Transfusion more tolerant of noisy visual inputs. Without changing model weights, we perturb the input image latents during captioning

as $x + \sigma\epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$ and the latents are normalized to unit variance. We evaluate the EMA weights of the 200K-step co-training checkpoints on COCO Karpathy test (5K images), using greedy decoding and `pycocoevalcap` CIDEr. All other captioning settings remain unchanged.

Noise-level embeddings stay fixed at their clean-input settings: $\tau = 1$ for Transfusion and $s = 0$ for AutoFlow. Thus, the intervention changes the input latents without signaling the perturbation through the noise embedding. During image-generation pretraining, Transfusion's backbone receives inputs across the diffusion trajectory, with $\tau \sim \mathcal{U}(0, 1)$, whereas AutoFlow uses only lightly perturbed context with $s \leq 0.25$. Image-to-text training uses clean inputs for both models.

Table 11: COCO captioning under additive input-latent noise. Parentheses show the percentage of each model's clean-input CIDEr retained, rounded to the nearest integer. Noise-level embeddings remain at their clean-input settings.

Noise scale σ	0	0.05	0.10	0.20	0.30	0.50
AutoFlow	63.5 (100%)	63.0 (99%)	61.6 (97%)	53.6 (84%)	27.2 (43%)	0.7 (1%)
Transfusion	47.3 (100%)	47.7 (101%)	47.0 (99%)	44.5 (94%)	39.7 (84%)	18.8 (40%)

AutoFlow leads by 16.2 CIDEr on clean inputs and retains higher absolute scores through $\sigma = 0.20$. Transfusion is less sensitive to the added noise and overtakes AutoFlow at $\sigma = 0.30$; at $\sigma = 0.50$, it retains 40% of its clean-input score, compared with 1% for AutoFlow. These results show a trade-off between clean-input captioning performance and robustness to latent corruption, consistent with the models' different exposure to noisy inputs during generation pretraining.

G Additional Qualitative Results

G.1 Downstream Visual Understanding

Figure 11 supplements Figure 3 with sixteen additional examples, two per dataset. We select cases in which AutoFlow correctly identifies the target region or answer, or captures caption details missed by the baselines. Images and model outputs are preserved; caption highlights are checked against the images and reference captions.

G.2 Interleaved Visual Reasoning

Figures 12 and 13 add four examples selected for correct AutoFlow answers and clear intermediate images. The displays retain the original text–image–text order. Text excerpts are verbatim, with ellipses marking omissions; highlights assess individual displayed claims against the task input and generated image.

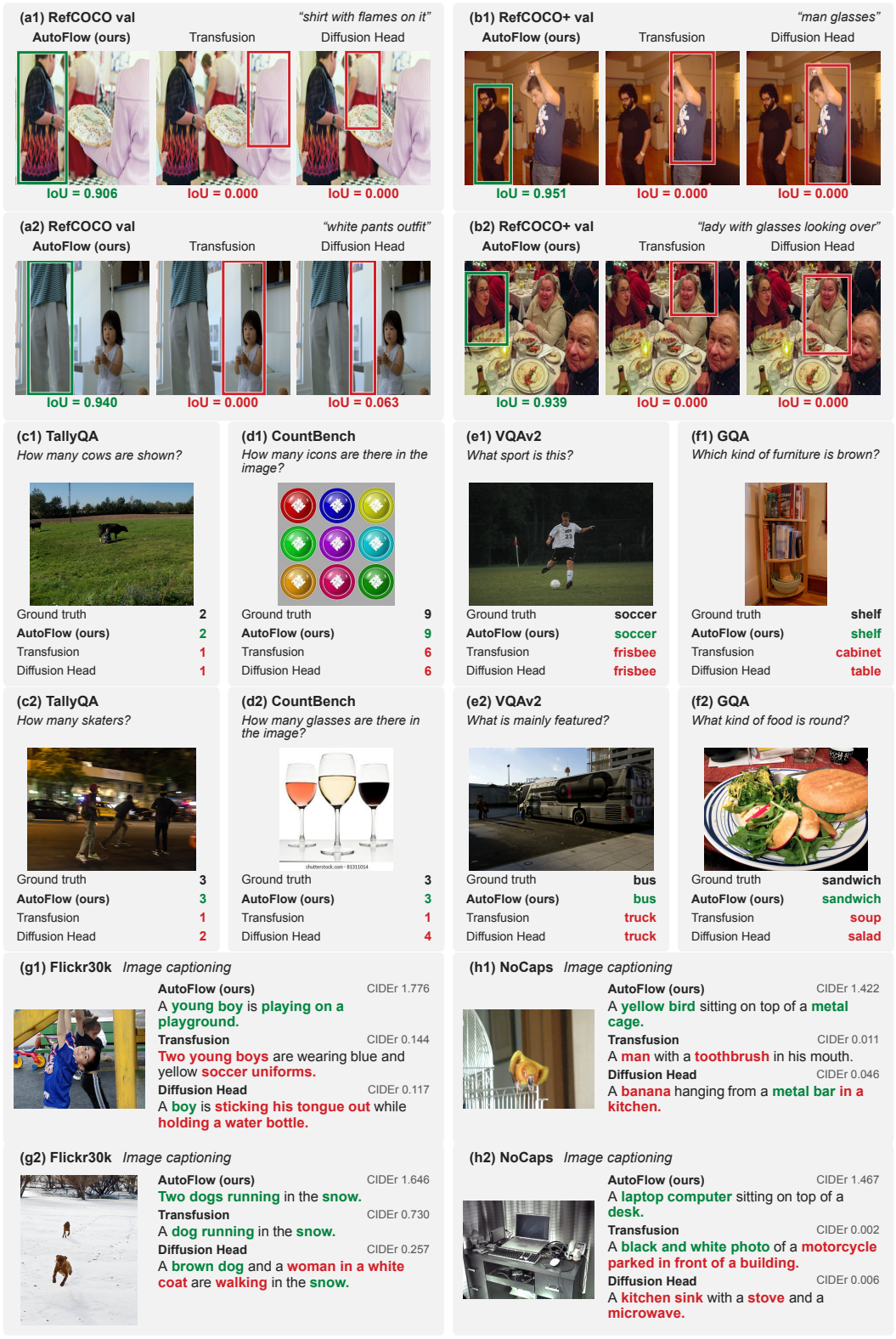


Figure 11: Additional downstream understanding examples, with two cases per dataset: (a) RefCOCO, (b) RefCOCO+, (c) TallyQA, (d) CountBench, (e) VQAv2, (f) GQA, (g) Flickr30k, and (h) NoCaps. Green/red marks correct/incorrect predictions or highlighted caption details. Caption scores are per-image CIDEr values from the evaluation export.

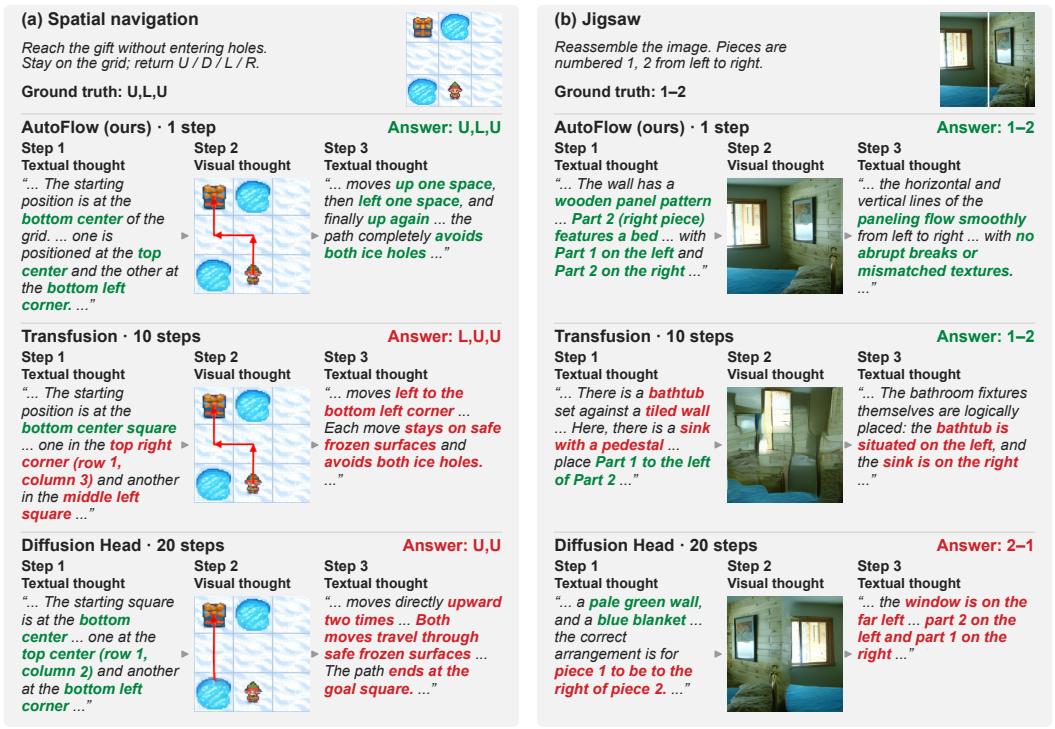


Figure 12: Additional interleaved reasoning on (a) navigation and (b) Jigsaw. AutoFlow produces the correct path and piece order with consistent displayed reasoning. Transfusion answers Jigsaw correctly but describes nonexistent bathroom fixtures. Green/red marks supported/incorrect textual claims and final answers.

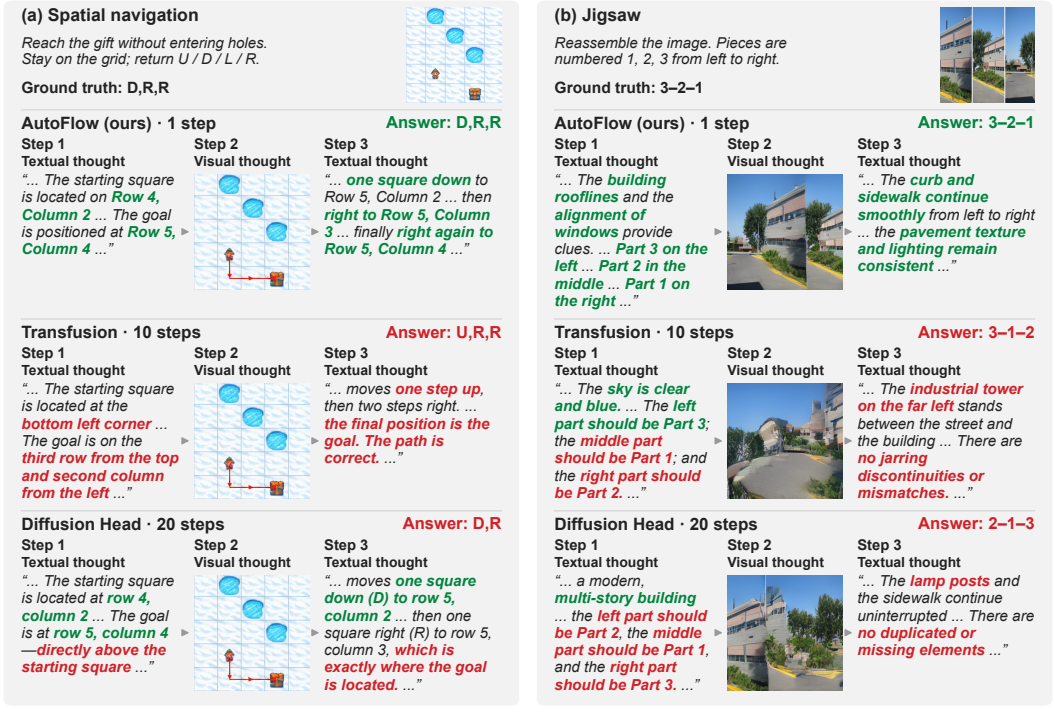


Figure 13: Further navigation and Jigsaw examples. AutoFlow returns the correct move sequence and reconstructs the piece order, while the baseline answers are incorrect. The visualizations and highlighted excerpts follow the conventions in Figure 12.

H BROADER IMPACT

AutoFlow aims to improve the efficiency and usefulness of continuous generation in autoregressive multimodal models. Potential benefits include lower sampling cost for image generation, visual reasoning, and action prediction. More capable generators can also be used to create misleading synthetic content, and improved robot policies can create risks when used outside their evaluated conditions. The real-robot experiments concern controlled tabletop manipulation under human supervision. Deployment in broader settings would require appropriate safety evaluation and safeguards.