PREDACTOR: PREDICTIVE ACTION DIFFUSION FOR STEERABLE ONBOARD HUMANOID CONTROL

Anonymous authors
Paper under double-blind review

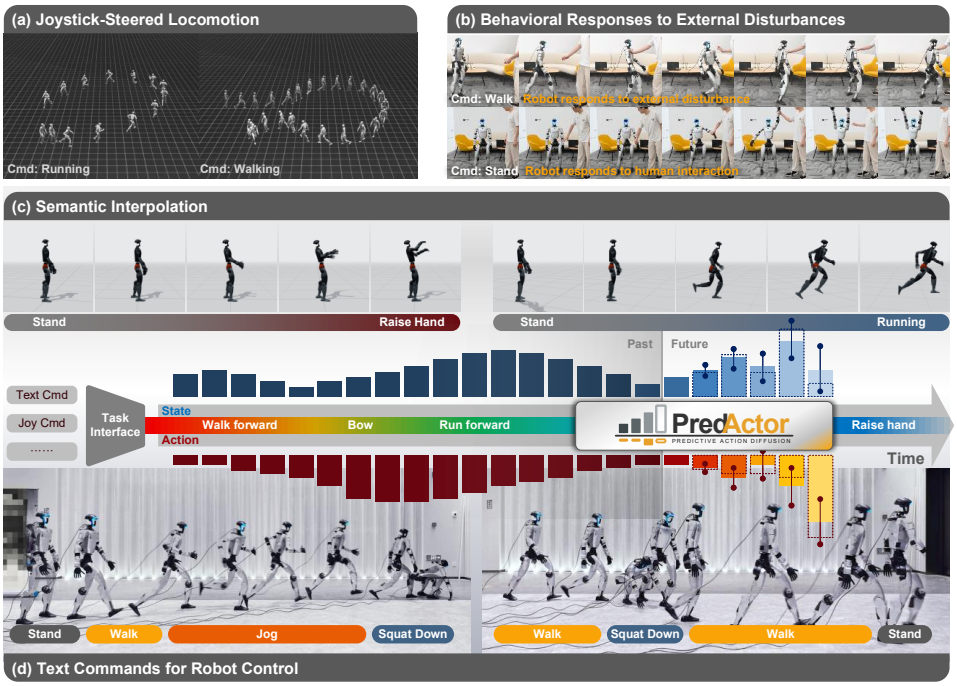

Figure 1: Qualitative demonstrations of PredActor. (a) Joystick steering redirects walking and running; (b) PredActor responds to external disturbances during walk and stand commands; (c) semantic interpolation connects standing with raising a hand and with running; and (d) text commands trigger stand, walk, jog, and squat sequences.

ABSTRACT

Diffusion models offer flexible motion generation, but translating this flexibility into feedback-responsive humanoid control remains challenging. Hierarchical systems steer motion through references that may exceed a separate tracker's capabilities, leaving recovery and physical execution largely to the tracker. Action-only diffusion generates actions directly but lacks an explicit future-state trajectory for test-time motion objectives. Joint state–action diffusion provides this representation, yet representative controllers often depend on privileged full-body states, and support for learned behavior selection and test-time motion steering remains fragmented. We present **PredActor**, a predictive action diffusion policy that brings these complementary steering capabilities into one directly executed policy using proprioceptive observations. Conditioned on proprioceptive history and optional task context, PredActor jointly generates executable actions and an internal future-state trajectory. Classifier-free guidance strengthens text-conditioned behavior, while classifier guidance steers predicted states toward test-time objectives. Only actions are executed, without a separate motion-reference tracker or externally estimated full-body states as policy inputs. In simulation, PredActor reaches 44 of 45 destination targets and achieves a text retrieval score of 0.539, compared with 0.424 for conditional action diffusion,

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.

with similar observed disturbance survival. To make this guided policy practical onboard, rolling denoising and computation-preserving runtime optimizations reduce the complete callback to 16.790 ms median and 19.383 ms p95 on a Jetson Orin NX, both below the 20 ms control period. We deploy PredActor on a Unitree G1; evaluations across simulation and physical hardware demonstrate text-conditioned motion, disturbance response, joystick control, and semantic interpolation. Project page: https://anonymous.4open.science/w/predactor_anonymous/.

1 INTRODUCTION

Humanoid policies need to follow task commands while adapting to physical feedback. Advances in motion tracking and behavior foundation models have expanded robot skills and task interfaces (Luo et al., 2023; RoboParty Lab Team, 2026; Luo et al., 2026; Chen et al., 2026a; Li et al., 2026; Zeng et al., 2025). Diffusion models offer flexible conditioning for diverse motion and action generation (Tevet et al., 2023; Chi et al., 2023). These capabilities raise a question: how can a policy guide future motion while directly generating executable actions?

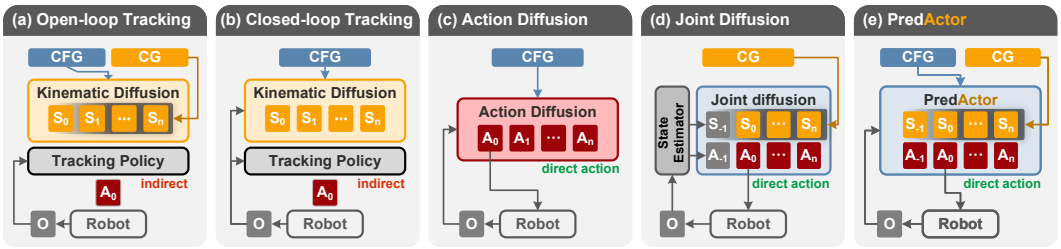

Figure 2: Paradigms of diffusion policies: (a,b) generator–tracker; (c) action-only; (d,e) joint state–action diffusion. O: observation; CG/CFG: method-specific guidance.

Diffusion control paradigms differ in how motion generation informs action execution (Figure 2). Kinematic generators supply references to trackers. Open-loop variants operate **without robot feedback** (Xie et al., 2026; Zhao et al., 2026; Wang et al., 2026), whereas closed-loop variants incorporate robot observations (Tevet et al., 2025; Chen et al., 2026b). Both may generate **dynamically infeasible motion outside tracker support**. Because these references are executed by a separate tracker, responses to environmental disturbances primarily reflect tracker-level recovery toward the reference trajectory rather than behavioral adaptation. **Separate planning and control clocks can also complicate synchronization and replanning** (Huang et al., 2025b; Gu et al., 2026). Action-only diffusion bypasses reference tracking by generating actions directly (Chi et al., 2023; Truong et al., 2024; Huang et al., 2025a; Høeg et al., 2025). Its predictions, however, contain no explicit future-state trajectory for classifier guidance (CG).

Joint state–action diffusion provides this predictive representation (Huang et al., 2025b; Carroll et al., 2026; Zhang et al., 2026a), but representative controllers often condition on privileged full-body states. That dependence makes sim-to-real transfer rely on a deployable state estimator, while the representation's complementary interfaces remain unevenly supported: classifier-free guidance (CFG) strengthens learned behavior conditions, whereas classifier guidance (CG) steers predicted motion toward test-time objectives, yet CFG is uncommon among representative joint controllers (Huang et al., 2025b; Liao et al., 2026). The open question is therefore functional: can a directly executed policy use deployable observations to combine learned behavior selection and test-time motion steering while retaining closed-loop disturbance robustness?

We introduce **PredActor**, a predictive action diffusion policy within this joint formulation (Figure 2(e)). It jointly denoises future states and actions from proprioceptive history and optional task context. Predicted states remain internal and provide the representation for CG; conditional and null predictions support CFG. Only the selected action is issued to the joint controller, without a separate motion-reference tracker. To make guided joint diffusion practical onboard, PredActor combines rolling denoising with computation-preserving runtime optimizations to reduce iterative-inference latency.

In simulation, PredActor reaches 44 of 45 destination targets and achieves 0.539 text retrieval versus 0.424 for conditional action diffusion, with similar observed push survival (0.511 versus 0.563). On Jetson Orin NX, the complete callback has a median latency of 16.790 ms and a p95 latency of 19.383 ms, both below the 20 ms control period. **To our knowledge, PredActor is the first joint state–action diffusion policy deployed entirely on a Unitree G1's onboard Jetson Orin NX for 50 Hz control**. Across simulation and physical G1 evaluation, demonstrations cover text commands, disturbance response, joystick steering, and semantic interpolation.

Our contributions are **(1) a predictive action diffusion policy for steerable humanoid control**, using only proprioceptive observations to simplify deployment and supporting both CG and CFG for task steering while retaining direct action execution; **(2) an end-to-end training and data-generation system** that turns a labeled motion library into asynchronous tracker rollouts, applies iterative data aggregation as a standard training recipe, and provides an editable task-control interface in an open-source implementation; and **(3) the first fully onboard deployment of a joint state–action diffusion policy on a Unitree G1's Jetson Orin NX for 50 Hz control**, supported by rolling inference, runtime optimization, and delay compensation.

2 Related Work

Kinematic Motion Generation and Tracking. Physics-trained trackers execute diverse reference motions (Luo et al., 2023; RoboParty Lab Team, 2026; Luo et al., 2026; Chen et al., 2026a), while diffusion generators provide multimodal, commandable kinematic synthesis (Tevet et al., 2023). Their complementarity motivates hierarchical systems that separate kinematic planning from physical execution (Xie et al., 2026; Zhao et al., 2026; Wang et al., 2026). Open-loop variants do not return robot feedback to the generator; CLoSD and ReactiveBFM instead return executed state to later planning windows (Tevet et al., 2025; Chen et al., 2026b). Both forms retain the same interface limitation: references may violate dynamics or tracker support, and separate planning/control clocks create synchronization and replanning delay. Feedback mitigates drift but does not intrinsically address dynamic-feasibility during generation. Generalist BFMs broaden skills and task interfaces but retain the separation between motion generation and physical execution (Li et al., 2026; Zeng et al., 2025; Shao et al., 2025).

Diffusion Policies for Humanoid Control. Diffusion control also spans policies that map observations and task conditions directly to actions (Chi et al., 2023; Truong et al., 2024; Huang et al., 2025a). Action-only diffusion, however, predicts no accompanying state trajectory for state objectives at test time. Joint state–action diffusion restores this structure and enables state-based shaping of executable actions (Janner et al., 2022; Huang et al., 2025b; Liao et al., 2026). Representative systems in Table 1 emphasize different capabilities and steering interfaces. For guidance, classifier-free guidance mixes conditional and null predictions, whereas classifier guidance adds gradients from a test-time objective (Ho & Salimans, 2021). Diffuse-CLoC and BeyondMimic emphasize classifier guidance (Huang et al., 2025b; Liao et al., 2026); SCDP addresses partial observation but reports neither classifier guidance, classifier-free guidance, nor rolling reuse; and SCRIPT adds language conditioning and post-training but is evaluated in simulation (Carroll et al., 2026; Zhang et al., 2026a). Beyond modeling and conditioning, deployment requires efficient closed-loop inference. Replanning with overlapping diffusion horizons can add latency and induce cross-tick disagreement (e.g., fixed action chunks delay feedback and ensembling can average incompatible modes) (Chi et al., 2023; Zhao et al., 2023). Fast samplers reduce denoising steps (Song et al., 2021; Lu et al., 2022; Song et al., 2023), and streaming methods reuse partially denoised horizons across ticks (Høeg et al., 2025; Zhang et al., 2024; Chen et al., 2024), but joint state–action prediction additionally requires aligned schedules and reuse at compatible noise levels. PredActor combines proprioceptive joint state–action diffusion, both guidance interfaces, and schedule-matched rolling inference in a fully onboard policy (Table 1). Appendix A records the evidence.

3 PredActor

PredActor augments action diffusion with an internal future-state trajectory for task guidance. It is a joint state–action policy: predicted states expose future motion to steering objectives, and actions are executed directly. We first define this representation and its CG/CFG interfaces (Figure 3), then

Table 1: Capabilities and deployment of generative control systems. Full hardware specifications, inference timings, and source evidence appear in Table 5 and Appendix A.1.

	Model settings		Task capability			Deployment		
Method	Paradigm	Proprio.	CFG	CG	Text	Robot	Inference GPU	Control rate
MotionBricks + SONIC	Generator + tracker	◇	–	–	–	✓	Orin (on)	50 Hz tracker[a]
ARDY + SONIC	Open-loop track	◇	◇	–	◇	–	RTX 4090[b]	–
TextOp	Open-loop track	◇	✓	–	✓	✓	RTX 4090 (off)[c]	50 Hz tracker
ReactiveBFM	Closed-loop track	◇	–	–	✓	✓	RTX 4090 (off)	50 Hz
CLoSD	Closed-loop track	–	✓	–	✓	–	–	N/A
DiffuseLoco	Action diffusion	✓	–	–	–	✓	RTX 4060M (on)	30 Hz target
Diffuse-CLoC	Joint diffusion	–	–	✓	–	–	RTX 4060 (sim)	N/A
BeyondMimic	Latent joint diffusion + dec.	◇	–	✓	–	✓	RTX 4060M (on)	25 Hz
SCDP	Joint diffusion	✓	–	–	–	✓	RTX 5090 (off)	50 Hz
SCRIPT	Joint diffusion	–	✓	–	✓	–	–	N/A
PredActor	Joint diffusion	✓	✓	✓	✓	✓	**Orin NX (on)**	**50 Hz**

Symbols: ✓: supported; ◇: partial/modular; ×: explicitly unsupported; –: unreported/unverified; N/A: not applicable.
Abbreviations: Proprio.: proprioceptive input; dec.: decoder; Robot: physical deployment; on/off: onboard/offboard; sim: simulation; M: Mobile. Rates refer to robot control, not generation throughput.
[a]MotionBricks uses a non-diffusion generator; 50 Hz is from SONIC, not separately reported for the composition. [b]ARDY generator-demo GPU; robot-stack rate unreported. [c]TextOp external generator GPU; onboard tracker.

explain how teacher trajectories supervise it (Figure 4), and finally describe rolling inference and onboard execution (Figure 5). Detailed tensor layouts, schedule derivations, and system interfaces appear in Appendix B.

3.1 Problem Formulation and Architecture

At control time t, let $\mathbf{o}_t \in \mathbb{R}^{d_o}$ denote the proprioceptive observation available on the robot, $\mathbf{s}(t) \in \mathbb{R}^{d_s^{\text{phys}}}$ its full-body motion state, and $\mathbf{a}(t) \in \mathbb{R}^{d_a}$ the action issued to the joint controller. Conditioned on an observation history $\mathbf{o}_{t-\ell+1:t}$ of length ℓ and optional task context $\mathbf{z} = [z_1, \ldots, z_n]$ of n task tokens, PredActor predicts clean trajectories $\mathbf{s}^0 \in \mathbb{R}^{h \times d_s}$ and $\mathbf{a}^0 \in \mathbb{R}^{h \times d_a}$ over h horizon positions in model coordinates. Superscripts denote diffusion noise levels (0 is clean), and subscripts $i \in \{1, \ldots, h\}$ index horizon positions. Given corrupted trajectories, the denoiser F_θ predicts:

$$(\hat{\mathbf{s}}^0, \hat{\mathbf{a}}^0) = F_\theta(\mathbf{s}^{\mathbf{k}_s}, \mathbf{a}^{\mathbf{k}_a}, \mathbf{k}_s, \mathbf{k}_a; \mathbf{o}_{t-\ell+1:t}, \mathbf{z}), \tag{1}$$

where $\mathbf{k}_s, \mathbf{k}_a \in \{0, \ldots, T\}^h$ specify per-position noise levels, T is the maximum diffusion level, and $\mathbf{s}^{\mathbf{k}_s}, \mathbf{a}^{\mathbf{k}_a}$ are the corresponding noisy inputs. Full-body states supply training targets, while deployment requires only proprioceptive observations and optional task context. Predicted states remain internal guidance variables; only the selected action is issued to the joint controller, without a separate motion-reference tracker. Feature definitions and the physical-to-model state representation are detailed in Appendix B.1.

Figure 3 follows conditioning, denoising, and sampling from top to bottom. The encoder maps the observation-history tokens (shown as $o_1, \ldots, o_\ell$) and task tokens $z_1, \ldots, z_n$ into condition memory $\mathbf{C} = [c_1, \ldots, c_m]$, with m encoded tokens. Separate projections embed noisy state and action tokens s_i, a_i with their noise levels and horizon positions. We adopt Diffuse-CLoC's interleaved state–action token pairs and asymmetric self-attention pattern (Huang et al., 2025b): state queries attend to all state tokens but not action tokens, while action queries attend causally to both streams. PredActor extends this backbone with cross-attention to $\mathbf{C}$, allowing the denoising tokens to query encoded proprioceptive history and task context. Separate heads produce clean predictions $\hat{\mathbf{s}}^0, \hat{\mathbf{a}}^0$ (shown as $\hat{s}_i, \hat{a}_i$). The noise scheduler selects per-position target levels $\mathbf{k}'_s, \mathbf{k}'_a$, and the denoising update returns both streams to the next sampling step (Section 3.3).

At application time, task conditioning and objective steering act as complementary interfaces. Conditional, null, and partial-condition training enables classifier-free guidance (CFG), which mixes conditional and null predictions to amplify learned task behavior (e.g., text-conditioned motion) (Ho & Salimans, 2021). Steering commands $\mathbf{g} = [g_1, \ldots, g_r]$ parameterize a differentiable objective $\mathcal{J}(\hat{\mathbf{s}}^0; \mathbf{g})$ with r inputs. Classifier guidance (CG) applies $\nabla_{\hat{\mathbf{s}}^0}\mathcal{J}$ before the reverse update (Huang et al., 2025b); it leaves the current action prediction unchanged, while the updated state can affect

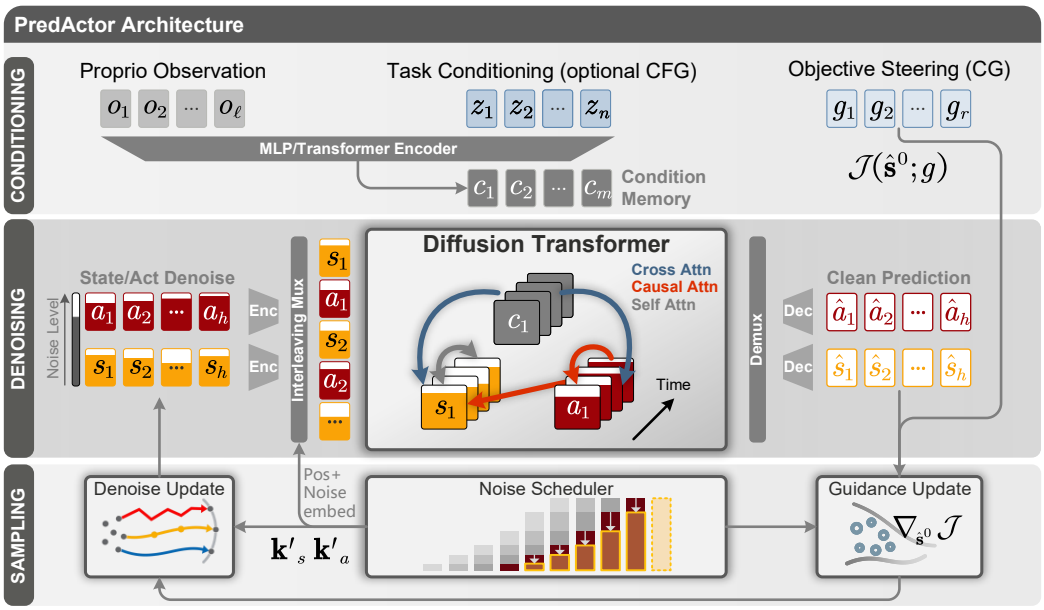

Figure 3: **Joint predictive architecture of PredActor.** Observation history and task commands condition a transformer that jointly denoises interleaved future state and action tokens. Steering changes the state update, not the current action prediction; the updated state can affect action at the next denoising step. Only actions are executed.

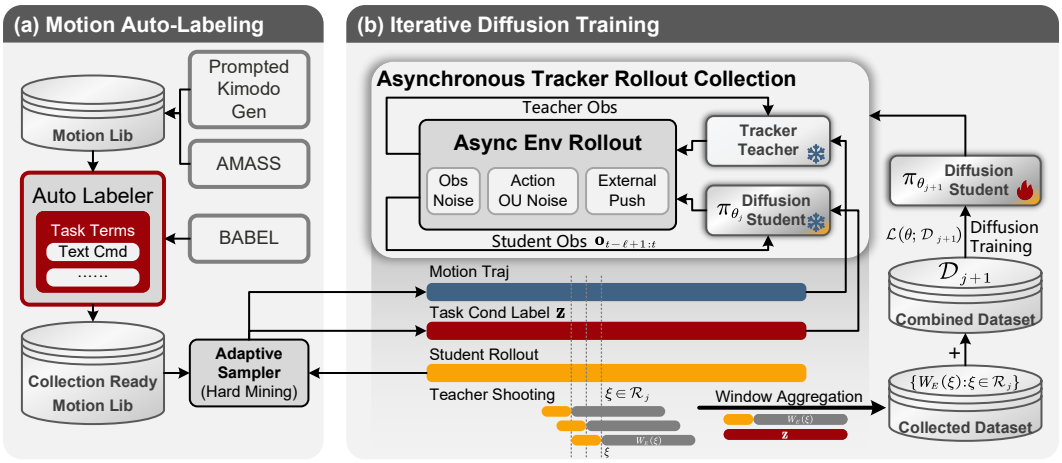

Figure 4: **Data and training pipeline.** Generated and annotated motion references drive asynchronous tracker rollouts. The resulting deployable observations and task labels condition joint state–action training. During iterative data aggregation, teacher trajectory shooting constructs target windows from learner rollout origins; the collected windows enter replay before the next reconstruction round.

the next action prediction through state-to-action attention. Thus CFG reinforces learned conditions, whereas CG steers future motion toward test-time objectives.

3.2 Data and Training

Figure 4(a) builds the motion library from generated motion and AMASS; BABEL supplies task labels, which MotionCLIP encodes as conditioning latents (Mahmood et al., 2019; Punnakkal et al., 2021; Tevet et al., 2022). The adaptive sampler feeds motion references to the asynchronous rollout

pipeline in (b). A frozen tracking teacher executes each reference under observation noise, action noise, and external pushes to seed $\mathcal{D}_0$ with observation histories, future states, teacher actions, and task labels. Training then uses a DAgger-style iterative data-aggregation recipe that alternates frozen-student rollout collection, isolated teacher trajectory shooting, and continued diffusion reconstruction (Ross et al., 2011). At round j, the student π_{θ_j} induces rollout origins $\mathcal{R}_j$; from each origin ξ, an isolated teacher branch follows the aligned reference to produce a target state–action window $W_E(\xi)$, paired with the student's history and task label. These branched windows are aggregated and the student is updated for the next round:

$$\begin{aligned} \mathcal{D}_{j+1} &= \mathcal{D}_j \cup \{W_E(\xi) : \xi \in \mathcal{R}_j\}, \\ \theta_{j+1} &\approx \arg\min_{\theta} \mathcal{L}(\theta; \mathcal{D}_{j+1}). \end{aligned} \tag{2}$$

Both initial training and subsequent replay reconstruct independently corrupted state and action windows using

$$\begin{aligned} \mathcal{L}(\theta; \mathcal{D}) = \mathbb{E}\Bigg[&\sum_{i=1}^{h} w_i^a \|\hat{\mathbf{a}}_i^0 - \mathbf{a}_i^0\|_2^2 + \rho \sum_{i=2}^{h} \|\Delta\hat{\mathbf{a}}_i^0 - \Delta\mathbf{a}_i^0\|_2^2 \\ &+ \gamma_s \sum_{i=1}^{h} w_i^s \|\hat{\mathbf{s}}_i^0 - \mathbf{s}_i^0\|_2^2 \Bigg], \end{aligned} \tag{3}$$

where the expectation covers training windows, noise, and diffusion levels; w_i^a, w_i^s weight horizon positions, $\Delta\mathbf{a}_i^0 = \mathbf{a}_i^0 - \mathbf{a}_{i-1}^0$, and ρ, γ_s weight action-difference regularization and state reconstruction. The three terms supervise actions, action differences, and future states. The teacher supplies training targets only and is absent at deployment; dataset partitioning, branch isolation, and window alignment are detailed in Appendix B.2.

3.3 Rolling Inference and Deployment

Figure 5(a) shows the receding-horizon loop that fits repeated denoising within each control period. Unlike training (independent noise-level sampling), rolling inference carries a partially denoised state–action horizon across ticks: after executing an action, buffers shift and the latest proprioceptive observation conditions the next denoise. For either stream $u \in \{s, a\}$, the DDIM update below converts a clean prediction into a reverse-sampling state at the target noise level:

$$\hat{\epsilon}_t = \frac{u^t - \sqrt{\bar{\alpha}_t}\hat{u}^0}{\sqrt{1-\bar{\alpha}_t}}, \qquad u^{t'} = \sqrt{\bar{\alpha}_{t'}}\hat{u}^0 + \sqrt{1-\bar{\alpha}_{t'}-\sigma_t^2}\hat{\epsilon}_t + \sigma_t\epsilon, \tag{4}$$

where $t' < t$, $\epsilon \sim \mathcal{N}(0, \mathbf{I})$, $\bar{\alpha}_t$ is the cumulative noise schedule, and the DDIM variance for this jump is

$$\sigma_t = \eta\sqrt{\frac{1-\bar{\alpha}_{t'}}{1-\bar{\alpha}_t}}\sqrt{1-\frac{\bar{\alpha}_t}{\bar{\alpha}_{t'}}}.$$

The evaluated deterministic profiles set $\eta = 0$, so $\sigma_t = 0$ and the random-noise term vanishes. The runtime can therefore carry intermediate samples across control steps instead of restarting the full horizon.

Each horizon position can take its own DDIM jump. PredActor represents the source and target levels by matrices $S, S' \in \mathbb{Z}^{K\times h}$, with each entry broadcast over the feature dimension at its horizon position; state and action streams may use different matrices. The runtime reuses the saved intermediate sample whose source level matches the next required position, adds fresh noise only to newly exposed tail positions, and denoises the shifted horizon under the latest observation. This schedule-matched reuse preserves every sample's diffusion coordinate rather than copying a clean prediction into an arbitrary noise level. In Figure 5(b), the delay coordinate $u = (n_{\text{obs}} - 1) + \delta/\Delta t$ selects adjacent clean action slots for interpolation, compensating actuation delay in the same horizon coordinates used for planning.

For onboard execution, we export the diffusion actor and control interface to C++. The evaluated Orin profile uses two-step DDIM; all-QKV packing, cached condition embeddings and DDIM schedules, stacked-axis WBG, and LibTorch inference mode reduce repeated work in the callback. These operation-level optimizations are evaluated end-to-end in Section 4.2. The same deployed interface supports text commands, semantic interpolation, and whole-body steering, while Appendix B.3 specifies the export, safety, and timing boundaries.

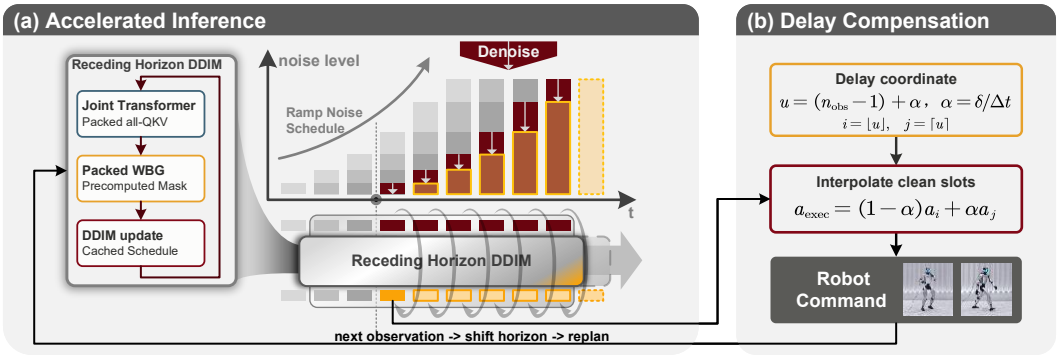

Figure 5: **Rolling inference and delay compensation.** Schedule-matched samples are shifted and replanned after each observation. A delay coordinate interpolates adjacent clean action slots before execution.

4 Experiments

We ask three questions: **Q1**, can PredActor combine CG destination steering and CFG text control while retaining disturbance recovery? **Q2**, can exact guided inference meet the 20 ms onboard budget? **Q3**, can the same policy support simulated interfaces and physical G1 execution? The policy controls 29 actuated joints at 50 Hz from proprioceptive history and predicts a 20-step state–action horizon. Full protocols, metrics, coverage, and uncertainty treatment are in Appendix C.

4.1 Steering and Closed-Loop Control

Setup. We compare seven policy rows in one calibrated MuJoCo plant using matched push, destination, and text protocols (Table 2). Six learned policies are represented by three training seeds: DiffuseLoco w TextCFG, CLOC, CLOC w TextCFG, PredActor w/o TextCFG, PredActor, and PredActor w/o state. ARDY+SONIC is reported as a single-model reference. Appendix C.2 separates these training and policy contracts. **Results.** Across three independently trained checkpoints, PredActor records 0.511 ± 0.030 push survival versus 0.563 ± 0.009 for DiffuseLoco w TextCFG, and reaches 44/45 destination targets with 0.605 ± 0.029 m last-valid error. Text survival is 0.888 ± 0.068 over 135 trials and retrieval is 0.539 ± 0.075 over 119/135 eligible windows, versus 0.947 ± 0.047 and 0.424 ± 0.159 over 126/135 for DiffuseLoco w TextCFG. Actor latency is 1.321 ± 0.011 ms versus 12.141 ± 0.145 ms and 8.788 ± 0.141 ms for DiffuseLoco w TextCFG and CLOC w TextCFG, respectively; jerk RMS is 16.810 ± 1.409, close to 15.899 ± 6.870 for DiffuseLoco w TextCFG

Table 2: **Seven-policy comparison of robustness, navigation, and text control.** For the six learned-policy rows, entries are means ± sample SD across three independently trained checkpoint means; brackets show evaluated/requested rollout cells. ARDY + SONIC is a single-model reference. Within each numeric column, cell color runs from the worst value through a white midpoint to the best value, respecting the direction of the metric.

Method	Basic		Perturbation	Point Navigation				Text Command		
	Jerk RMS↓ (10^3 s^{-3})	Latency↓ (ms)	Survival↑	Survival↑	Arrival↑	Last valid err.↓ (m)	Min valid err.↓ (m)	Survival↑	Retrieval↑	Admit.↑
DiffuseLoco w TextCFG	15.899 +/-6.870 [60/60]	12.141 +/-0.145 [15/15]	0.563 +/-0.009 [300/300]	N/A [0/45]	N/A [0/45]	N/A [0/45]	N/A [0/45]	0.947 +/-0.047 [135/135]	0.424 +/-0.159 [126/135*]	N/E [0/135]
CLOC	266.605 +/-32.412 [60/60]	6.102 +/-0.029 [15/15]	0.303 +/-0.031 [300/300]	0.064 +/-0.005 [45/45]	0.000 +/-0.000 [45/45]	3.194 +/-0.112 [45/45]	3.151 +/-0.053 [45/45]	N/A [0/135]	N/A [0/135]	N/E [0/135]
CLOC w TextCFG	101.682 +/-2.110 [60/60]	8.788 +/-0.141 [15/15]	0.296 +/-0.020 [300/300]	0.450 +/-0.059 [45/45]	0.333 +/-0.067 [45/45]	2.217 +/-0.450 [45/45]	2.076 +/-0.373 [45/45]	0.782 +/-0.033 [135/135]	0.416 +/-0.106 [103/135*]	N/E [0/135]
PredActor w/o TextCFG	**11.849** +/-0.899 [60/60]	1.310 +/-0.006 [15/15]	**0.579** +/-0.012 [300/300]	0.845 +/-0.120 [45/45]	0.822 +/-0.139 [45/45]	0.983 +/-0.301 [45/45]	0.906 +/-0.199 [45/45]	N/A [0/135]	N/A [0/135]	N/E [0/135]
PredActor	16.810 +/-1.409 [60/60]	1.321 +/-0.011 [15/15]	0.511 +/-0.030 [300/300]	0.983 +/-0.030 [45/45]	**0.978** +/-0.038 [45/45]	**0.605** +/-0.029 [45/45]	**0.604** +/-0.028 [45/45]	0.888 +/-0.068 [135/135]	**0.539** +/-0.075 [119/135*]	N/E [0/135]
PredActor w/o state	12.448 +/-0.675 [75/75]	**1.050** +/-0.004 [15/15]	0.487 +/-0.033 [300/300]	N/A [0/45]	N/A [0/45]	N/A [0/45]	N/A [0/45]	0.778 +/-0.145 [135/135]	0.397 +/-0.023 [98/135*]	N/E [0/135]
ARDY + SONIC	19.221 [20/20]	N/E [0/5]	0.534 [100/100]	**1.000** [2/15*]	0.500 [2/15*]	0.656 [2/15*]	0.656 [2/15*]	**1.000** [23/45*]	0.237 [23/45*]	**0.511** [45/45]

The six learned policies use training seeds 92025/92026/92027. Bold marks the numeric optimum at the displayed precision, including conditional cells and ties. N/A marks an unavailable task interface; N/E marks a metric not evaluated under the common protocol, including reference admission where no admission test was performed. Neither is zero-imputed. An asterisk marks a conditional subset excluded from full-coverage comparisons. The ARDY + SONIC destination entries use the strict raw-admission subset (2/15); the separate limit-projected arm is not substituted. Latency is synchronized, post-warmup RTX 4090 actor time amortized per action at $B = 5$; text retrieval uses complete windows beginning at least 3 s after command activation.

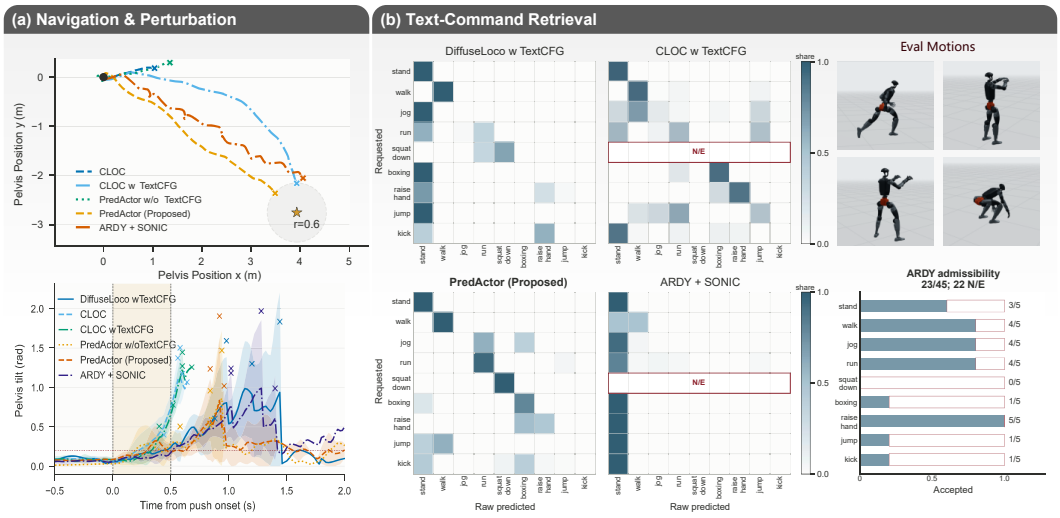

Figure 6: **Representative MuJoCo traces for six of the seven policy rows. (a)** Representative destination trajectories in the common protocol frame (top; the star is the target and the circle marks the 0.6 m arrival threshold) and direction-matched pelvis tilt under a 300 N push (bottom; lines are means, bands show sample SD, and crosses mark early terminals) across 20 paired direction/seed cells per policy. **(b)** Per-class raw MotionCLIP top-1 retrieval for each evaluable policy, example evaluation motions, and ARDY+SONIC reference admissibility by requested class. Missing cells remain distinct from zero; 22/45 ARDY references are rejected before SONIC rollout. The seven-row aggregate and its training-replica denominators are reported in Table 2.

and far below 101.682 ± 2.110 for CLOC w TextCFG. PredActor w/o state records 0.487 ± 0.033 push survival, 0.778 ± 0.145 text survival, and 0.397 ± 0.023 retrieval over 98 eligible windows; PredActor's predicted-state stream additionally provides the interface for destination CG. Figure 6 and Table 2 report the task-level panels, aggregate metrics, and coverage denominators. Together, these results show that PredActor combines destination steering and text-conditioned behavior with low actor latency, competitive smoothness, and reported push survival.

4.2 Inference Acceleration on Jetson Orin

Setup. We benchmark batch-one FP32 guided inference and the complete callback on one Jetson Orin NX. The sampler sweep and staged graph, structural, and runtime measurements are defined in Appendix D.1; the timing benchmark produces no actuation commands. **Results.** Figure 7(a,d) shows that two-step NS-DDIM gives the best tested speed–robustness trade-off. The selected implementation reduces actor p50 from 21.461 to 15.930 ms along the structural path, and inference mode brings the remeasured actor to 13.175 ms (Table 4). The complete callback is 16.790/19.383 ms at p50/p95; 593/600 calls meet 20 ms, with output drift below 1.073×10^{-6}. Thus the optimized export is feasible for 50 Hz onboard control at the measured p95, while the seven misses leave no hard-real-time guarantee.

4.3 Onboard and Simulation Demonstrations

Physical behavior. Figure 1(b,d) and Figure 9 in Appendix C.3 show disturbance responses and staged text-command execution; the latter adds four command-conditioned response sequences. **Simulation control.** Figure 1(a,c) demonstrates joystick steering and semantic interpolation. **Onboard execution.** Figure 7(g) confirms that the same conditioned policy runs on the Unitree G1–Jetson Orin NX stack. **Physical trials.** Table 3 reports five physical trials for each of six transitions or disturbance settings: success ranges from 4/5 for squat-to-stand and both push settings to 5/5 for stand-to-walk, walk-to-jog, and stand-to-squat. Appendix D.4 shows every recorded success and failure under the same scoring rule, and Appendix C.3 specifies the evaluation scope.

Table 3: **Real-robot trial outcomes** (five trials per setting).

	Success
Stand→Walk	5/5
Walk→Jog	5/5
Stand→Squat	5/5
Squat→Stand	4/5
Stand push	4/5
Walk push	4/5

Success: complete the requested transition or remain upright after the push. Falls, incomplete behavior, and safety stops count as failures. Five trials per setting; counts are descriptive. Per-trial sequences appear in Appendix D.4.

Table 4: **Cumulative acceleration on Orin.** Frozen actor CUDA timing except the final live callback; NS-DDIM2, B=1 FP32, one forward/step (two total), WBG on, eager finite checks.

Composition / boundary	p50	p95	Saved	$>$ 20 ms
Structural baseline; seq. WBG	21.461	22.036	–	600/600
+ stacked WBG	18.315	19.677	3.147	8/600
+ DDIM precomputation	17.442	18.347	0.872	0/600
+ all-QKV packing	16.331	17.372	1.111	0/600
+ condition cache	15.930	16.728	0.401	0/600
Same structure, runtime remeasurement	16.234	16.912	–	1/600
+ inference mode (final actor)	**13.175**	**13.400**	**3.058**	**0/600**
Final live callback-to-host	**16.790**	**19.383**	**–**	**7/600**

Times in ms: medians of three block quantiles (200 calls/block). Saved: preceding-row p50 decrease within one suite, not additive factor effects. Rules mark suite/boundary changes; Appendix D.1 distinguishes the original-export P_0 from the matched structural baseline and reports effect ranges. The last column counts calls exceeding the 20 ms budget.

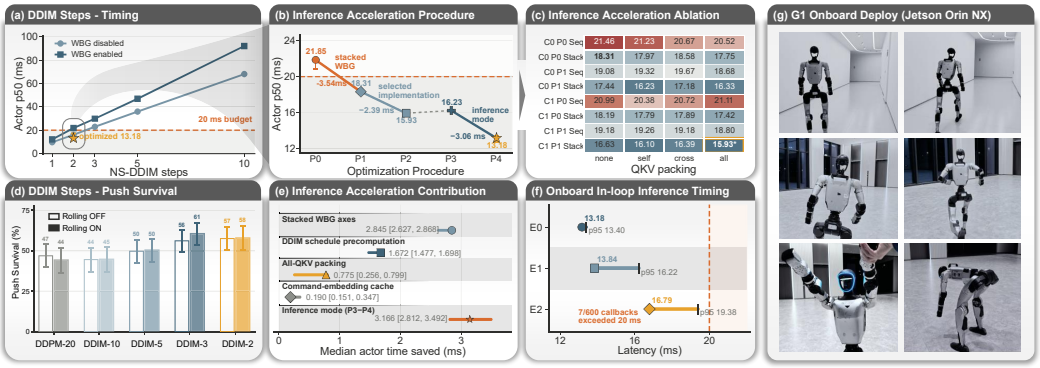

Figure 7: **Orin acceleration and onboard G1 deployment. (a,d)** Sampler timing and push survival. **(b)** Path P_0–P_4 from the original actor through cache-capable re-export, the selected configuration, runtime remeasurement, and LibTorch inference mode. **(c)** Ablation matrix (C: condition cache; P: schedule precomputation; Seq/Stack: WBG axes). **(e)** Implementation savings; markers show the median reduction across three paired blocks, and brackets give the corresponding min–max range. **(f)** p50–p95 for E_0 (fixed-input actor), E_1 (new-state receipt/conversion plus actor), and E_2 (complete callback to host action). Each endpoint uses 3×200 retained calls; 593/600 E_2 calls finish within 20 ms and p95 is 19.383 ms. **(g)** Qualitative frames from onboard G1 deployment with Jetson Orin NX.

5 Conclusion

PredActor unifies predictive motion generation and direct action execution in a joint state–action diffusion policy for steerable humanoid control. Predicted future states form a differentiable interface for classifier guidance, classifier-free guidance selects task-conditioned behaviors, and paired actions close the loop from proprioceptive history. This eliminates a separate motion tracker while retaining test-time steering. Across matched simulations, PredActor reaches 44 of 45 destination targets, achieves 0.539 text retrieval, and maintains competitive disturbance survival and smoothness. Runtime optimization yields 16.790 ms median and 19.383 ms p95 complete-callback latency on Jetson Orin NX, while onboard G1 execution demonstrates the deployable interface across six physical settings. These results show that internal predictive trajectories can connect high-level steering with responsive low-level execution. Future work will strengthen guidance, expand language–motion coverage, and support richer multi-contact behaviors.

Reproducibility statement

The anonymous project page linked in the abstract provides the open-source code and reproduction materials for PredActor. Appendix B documents the model, training, and deployment details; Appendix C specifies the evaluation protocols, metrics, trial coverage, timing boundaries, and failure handling; and Appendix D reports complementary ablations and limitations.

AI use statement

Generative AI tools assisted with conceptual framing, experiment-design review, literature discovery and summarization, manuscript editing, and code-based figures and scripts. We verified literature claims against primary sources, inspected generated code and artifacts, rebuilt the manuscript, and reviewed all AI-assisted content. Generative AI output was not used as experimental evidence. The authors take responsibility for the final text, claims, and artifacts.

Ethics statement

This work does not collect or analyze human-subject data. Real-robot experiments were conducted by trained operators using appropriate safety procedures, and identifiable faces in released visual materials are blurred.

References

Anurag Ajay, Yilun Du, Abhi Gupta, Joshua B. Tenenbaum, Tommi S. Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision making? In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=sP1fo2K9DFG.

Milo Carroll, Tianhu Peng, Lingfan Bao, Chengxu Zhou, and Zhibin Li. SCDP: Learning humanoid locomotion from partial observations via mixed-observation distillation, 2026. URL https://arxiv.org/abs/2603.09574.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In *Advances in Neural Information Processing Systems 37*, NeurIPS 2024, pp. 24081–24125. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2024. doi: 10.52202/079017-0759. URL https://doi.org/10.52202/079017-0759.

Maiyue Chen, Kaihui Wang, Bo Zhang, Xihan Ma, Zhiyuan Yang, Yi Ren, Qijun Huang, Zihao Zhu, Yucheng Wang, and Zhizhong Su. HoloMotion-1 technical report, 2026a. URL https://arxiv.org/abs/2605.15336.

Xiao Chen, Weishuai Zeng, Xiaojie Niu, Zirui Wang, Jianan Li, Huayi Wang, Furui Xu, Jiahe Chen, Weixiang Zhong, Lihe Ding, Kailin Li, Jiangmiao Pang, Tai Wang, Tianfan Xue, and Jingbo Wang. ReactiveBFM: Reactive closed-loop motion planning towards universal humanoid whole-body control, 2026b. URL https://arxiv.org/abs/2606.30362.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin C. M. Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems XIX*, RSS2023. Robotics: Science and Systems Foundation, July 2023. doi: 10.15607/rss.2023.xix.026. URL https://doi.org/10.15607/rss.2023.xix.026.

Zhaoyuan Gu, Yipu Chen, Zimeng Chai, Alfred Cueva, Thong Nguyen, Yifan Wu, Huishu Xue, Minji Kim, Isaac Legene, Fukang Liu, KyoungMok Kim, Ayan Barula, Yongxin Chen, and Ye Zhao. REFINE-DP: Diffusion policy fine-tuning for humanoid loco-manipulation via reinforcement learning. *IEEE Robotics and Automation Letters*, 11(10):11617–11624, Oct 2026. doi: 10.1109/lra.2026.3723742. URL https://doi.org/10.1109/lra.2026.3723742.

Tairan He, Zhengyi Luo, Wenli Xiao, Chong Zhang, Kris Kitani, Changliu Liu, and Guanya Shi. Learning human-to-humanoid real-time whole-body teleoperation. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8944–8951. IEEE,

Oct 2024. doi: 10.1109/iros58592.2024.10801984. URL https://doi.org/10.1109/iros58592.2024.10801984.

Tairan He, Zhengyi Luo, Xialin He, Wenli Xiao, Chong Zhang, Weinan Zhang, Kris M. Kitani, Changliu Liu, and Guanya Shi. OmniH2O: Universal and dexterous human-to-humanoid whole-body teleoperation and learning. In *Proceedings of the 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 1516–1540. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/he25b.html.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. URL https://openreview.net/forum?id=qw8AKxfYbI.

Sigmund H. Høeg, Yilun Du, and Olav Egeland. Fast policy synthesis with variable noise diffusion models. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4821–4828. IEEE, 2025. doi: 10.1109/ICRA55743.2025.11127858. URL https://doi.org/10.1109/ICRA55743.2025.11127858.

Xiaoyu Huang, Yufeng Chi, Ruofeng Wang, Zhongyu Li, Xue Bin Peng, Sophia Shao, Borivoje Nikolic, and Koushil Sreenath. DiffuseLoco: Real-time legged locomotion control with diffusion from offline datasets. In *Proceedings of the 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 1567–1589. PMLR, 06–09 Nov 2025a. URL https://proceedings.mlr.press/v270/huang25a.html.

Xiaoyu Huang, Takara Truong, Yunbo Zhang, Fangzhou Yu, Jean Pierre Sleiman, Jessica Hodgins, Koushil Sreenath, and Farbod Farshidian. Diffuse-CLoC: Guided diffusion for physics-based character look-ahead control. *ACM Transactions on Graphics*, 44(4):132:1–132:12, July 2025b. doi: 10.1145/3731206. URL https://doi.org/10.1145/3731206.

Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9902–9915. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/janner22a.html.

Zhengyao Jiang, Yingchen Xu, Nolan Wagener, Yicheng Luo, Michael Janner, Edward Grefenstette, Tim Rocktäschel, and Yuandong Tian. H-GAP: Humanoid control with a generalist planner. In *International Conference on Learning Representations*, volume 2024, pp. 17774–17791, 2024. URL https://openreview.net/forum?id=LYG6tBlEX0.

Michael Kelly, Chelsea Sidrane, Katherine Driggs-Campbell, and Mykel J. Kochenderfer. HG-DAgger: Interactive imitation learning with human experts. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8077–8083. IEEE, May 2019. doi: 10.1109/icra.2019.8793698. URL https://doi.org/10.1109/icra.2019.8793698.

Michael Laskey, Jonathan Lee, Roy Fox, Anca Dragan, and Ken Goldberg. DART: Noise injection for robust imitation learning. In *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pp. 143–156. PMLR, 13–15 Nov 2017. URL https://proceedings.mlr.press/v78/laskey17a.html.

Yitang Li, Zhengyi Luo, Tonghe Zhang, Cunxi Dai, Anssi Kanervisto, Andrea Tirinzoni, Haoyang Weng, Kris Kitani, Mateusz Guzek, Ahmed Touati, Alessandro Lazaric, Matteo Pirotta, and Guanya Shi. BFM-Zero: A promptable behavioral foundation model for humanoid control using unsupervised reinforcement learning. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 79697–79725, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/80b677bb4d17f77220be88e1d1fdebde-Paper-Conference.pdf.

Yixuan Li, Yutang Lin, Jieming Cui, Tengyu Liu, Wei Liang, Yixin Zhu, and Siyuan Huang. CLONE: Closed-loop whole-body humanoid teleoperation for long-horizon tasks. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 4493–4505. PMLR, 27–30 Sep 2025. URL https://proceedings.mlr.press/v305/li25h.html.

Qiayuan Liao, Takara E. Truong, Xiaoyu Huang, Yuman Gao, Guy Tevet, Koushil Sreenath, and C. Karen Liu. BeyondMimic: From motion tracking to versatile humanoid control via guided diffusion. *Science Robotics*, 11(117), Aug 2026. doi: 10.1126/scirobotics.adx8924. URL https://doi.org/10.1126/scirobotics.adx8924.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In *Advances in Neural Information Processing Systems 35*, NeurIPS 2022, pp. 5775–5787. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2022. doi: 10.52202/068431-0418. URL https://doi.org/10.52202/068431-0418.

Zhengyi Luo, Jinkun Cao, Alexander Winkler, Kris Kitani, and Weipeng Xu. Perpetual humanoid control for real-time simulated avatars. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10895–10904. IEEE, Oct 2023. doi: 10.1109/iccv51070.2023.01000. URL https://doi.org/10.1109/iccv51070.2023.01000.

Zhengyi Luo, Ye Yuan, Tingwu Wang, Chenran Li, Fernando Castañeda, Sirui Chen, Zi-Ang Cao, Jiefeng Li, David Minor, Qingwei Ben, Jinhyung Park, David Sami, Zi Wang, Xingye Da, Runyu Ding, Cyrus Hogg, Lina Song, Edy Lim, Eugene Jeong, Tairan He, Haoru Xue, Wenli Xiao, Simon Yuen, Jan Kautz, Yan Chang, Umar Iqbal, Linxi "Jim" Fan, and Yuke Zhu. SONIC: Supersizing motion tracking for natural humanoid whole-body control. *Science Robotics*, 11 (117), Aug 2026. doi: 10.1126/scirobotics.aed4592. URL https://doi.org/10.1126/scirobotics.aed4592.

Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5442–5451, 2019. doi: 10.1109/ICCV.2019.00554. URL https://doi.org/10.1109/ICCV.2019.00554.

Abhinanda R. Punnakkal, Arjun Chandrasekaran, Nikos Athanasiou, Alejandra Quiros-Ramirez, and Michael J. Black. BABEL: Bodies, action and behavior with english labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 722–731, 2021. doi: 10.1109/CVPR46437.2021.00078. URL https://doi.org/10.1109/CVPR46437.2021.00078.

Allen Z. Ren, Justin Lidard, Lars L. Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. In *International Conference on Learning Representations*, volume 2025, pp. 77288–77329, 2025. URL https://openreview.net/forum?id=mEpqHvbD2h.

RoboParty Lab Team. MimicLite: Efficient and effective general humanoid motion tracking. https://github.com/EGalahad/mimic-lite, 2026. Technical report: https://github.com/Roboparty/MimicLite/blob/main/mimic-lite.pdf.

Ralf Römer, Alexander von Rohr, and Angela Schoellig. Diffusion predictive control with constraints. In *Proceedings of the 7th Annual Learning for Dynamics & Control Conference*, volume 283 of *Proceedings of Machine Learning Research*, pp. 791–803. PMLR, 04–06 Jun 2025. URL https://proceedings.mlr.press/v283/romer25a.html.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL https://proceedings.mlr.press/v15/ross11a.html.

Carmelo Sferrazza, Dun-Ming Huang, Xingyu Lin, Youngwoon Lee, and Pieter Abbeel. HumanoidBench: Simulated humanoid benchmark for whole-body locomotion and manipulation. In *Robotics: Science and Systems XX*, RSS2024. Robotics: Science and Systems Foundation, July 2024. doi: 10.15607/rss.2024.xx.061. URL https://doi.org/10.15607/rss.2024.xx.061.

Yiyang Shao, Bike Zhang, Qiayuan Liao, Xiaoyu Huang, Yuman Gao, Yufeng Chi, Zhongyu Li, Sophia Shao, and Koushil Sreenath. LangWBC: Language-directed humanoid whole-body control via end-to-end learning. In *Robotics: Science and Systems XXI*, RSS2025. Robotics: Science and Systems Foundation, June 2025. doi: 10.15607/rss.2025.xxi.065. URL https://doi.org/10.15607/rss.2025.xxi.065.

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.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32211–32252. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/song23a.html.

Chen Tessler, Yunrong Guo, Ofir Nabati, Gal Chechik, and Xue Bin Peng. MaskedMimic: Unified physics-based character control through masked motion inpainting. *ACM Transactions on Graphics*, 43(6):1–21, Nov 2024. doi: 10.1145/3687951. URL https://doi.org/10.1145/3687951.

Chen Tessler, Yifeng Jiang, Xue Bin Peng, Erwin Coumans, Yi Shi, Haotian Zhang, Davis Rempe, Gal Chechik, and Sanja Fidler. ProtoMotions3: An open-source framework for humanoid simulation and control. https://github.com/NVLabs/ProtoMotions/, 2025. URL https://github.com/NVLabs/ProtoMotions.

Guy Tevet, Brian Gordon, Amir Hertz, Amit H Bermano, and Daniel Cohen-Or. MotionCLIP: Exposing human motion generation to CLIP space. In *European Conference on Computer Vision (ECCV)*, pp. 358–374. Springer, Springer, 2022. doi: 10.1007/978-3-031-20047-2_21. URL https://doi.org/10.1007/978-3-031-20047-2_21.

Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H. Bermano. Human motion diffusion model. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=SJ1kSyO2jwu.

Guy Tevet, Sigal Raab, Setareh Cohan, Daniele Reda, Zhengyi Luo, Xue Bin Peng, Amit H. Bermano, and Michiel van de Panne. CLoSD: Closing the loop between simulation and diffusion for multi-task character control. In *International Conference on Learning Representations*, volume 2025, pp. 46506–46520, 2025. URL https://openreview.net/forum?id=pZISppZSTv.

Takara Everest Truong, Michael Piseno, Zhaoming Xie, and C. Karen Liu. PDP: Physics-based character animation via diffusion policy. In *SIGGRAPH Asia 2024 Conference Papers*, SA '24, pp. 1–10. ACM, Dec 2024. doi: 10.1145/3680528.3687683. URL https://doi.org/10.1145/3680528.3687683.

Tingwu Wang, Olivier Dionne, Michael De Ruyter, David Minor, Davis Rempe, Kaifeng Zhao, Mathis Petrovich, Ye Yuan, Chenran Li, Zhengyi Luo, Brian Robison, Xavier Blackwell, Bernardo Antoniazzi, Xue Bin Peng, Yuke Zhu, and Simon Yuen. MotionBricks: Scalable real-time motions with modular latent generative model and smart primitives. *ACM Transactions on Graphics*, 45(4):1–22, July 2026. doi: 10.1145/3811334. URL https://doi.org/10.1145/3811334.

Zhendong Wang, Max Li, Ajay Mandlekar, Zhenjia Xu, Jiaojiao Fan, Yashraj Narang, Linxi Fan, Yuke Zhu, Yogesh Balaji, Mingyuan Zhou, Ming-Yu Liu, and Yu Zeng. One-step diffusion policy: Fast visuomotor policies via diffusion distillation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 63399–63416. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/wang25ba.html.

Weiji Xie, Jiakun Zheng, Jinrui Han, Jiyuan Shi, Weinan Zhang, Chenjia Bai, and Xuelong Li. TextOp: Real-time interactive text-driven humanoid robot motion generation and control, 2026. URL https://arxiv.org/abs/2602.07439.

Yanjie Ze, Zixuan Chen, João Pedro Araújo, Zi-ang Cao, Xue Bin Peng, Jiajun Wu, and C. Karen Liu. TWIST: Teleoperated whole-body imitation system. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 2143–2154. PMLR, 27–30 Sep 2025. URL https://proceedings.mlr.press/v305/ze25a.html.

Weishuai Zeng, Shunlin Lu, Kangning Yin, Xiaojie Niu, Minyue Dai, Jingbo Wang, and Jiangmiao Pang. Behavior foundation model for humanoid robots, 2025. URL https://arxiv.org/abs/2509.13780.

Jingyan Zhang, Han Liang, Ruichi Zhang, Bin Li, Juze Zhang, Xin Chen, Jingya Wang, Lan Xu, and Jingyi Yu. SCRIPT: Scalable diffusion policy with multi-stage training for language-driven physics-based humanoid control, 2026a. URL https://arxiv.org/abs/2605.22894.

Zhikai Zhang, Jun Guo, Chao Chen, Jilong Wang, Chenghuai Lin, Yunrui Lian, Han Xue, Zhenrong Wang, Maoqi Liu, Jiangran Lyu, Huaping Liu, He Wang, and Li Yi. Track any motions under any disturbances. In *2026 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2026b. URL https://ras.papercept.net/conferences/conferences/ICRA26/program/ICRA26_ContentListWeb_5.html.

Zihan Zhang, Richard Liu, Kfir Aberman, and Rana Hanocka. TEDi: Temporally-entangled diffusion for long-term motion synthesis. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, SIGGRAPH '24, pp. 1–11. ACM, July 2024. doi: 10.1145/3641519.3657515. URL https://doi.org/10.1145/3641519.3657515.

Kaifeng Zhao, Mathis Petrovich, Haotian Zhang, Tingwu Wang, Siyu Tang, and Davis Rempe. ARDY: Autoregressive diffusion with hybrid representation for interactive human motion generation. *ACM Transactions on Graphics (TOG)*, 45(4), 2026. doi: 10.1145/3811284. URL https://doi.org/10.1145/3811284.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems XIX*, RSS2023. Robotics: Science and Systems Foundation, July 2023. doi: 10.15607/rss.2023.xix.016. URL https://doi.org/10.15607/rss.2023.xix.016.

A Extended Related-Work Coverage

A.1 Deployment Hardware and Timing Sources

Table 5: Policy capabilities and deployment configurations of generative control systems. Top: representation and task interfaces. Bottom: inference hardware, location, and reported execution rates; component timings are not complete-loop measurements.

	Model settings			Task capability				Deployment
Method	Paradigm	Proprio.	Rolling reuse	CFG	CG	Text	Joy/Goal	Robot
MotionBricks + SONIC	Generator + tracker	◇	–	–	–	–	✓	✓
ARDY + SONIC	Open-loop track	◇	–	◇	–	◇	✓	–
TextOp	Open-loop track	◇	–	✓	–	✓	✓	✓
ReactiveBFM	Closed-loop track	◇	–	–	–	✓	✓	✓
CLoSD	Closed-loop track	–	–	✓	–	✓	✓	–
DiffuseLoco	Action diffusion	✓	–	–	–	–	✓	✓
Diffuse-CLoC	Joint diffusion	–	✓	–	✓	–	✓	–
BeyondMimic	Latent joint diffusion + dec.	◇	–	–	✓	–	✓	✓
SCDP	Joint diffusion	✓	–	–	–	–	✓	✓
SCRIPT	Joint diffusion	–	–	✓	–	✓	✓	–
PredActor	Joint diffusion	✓	✓	✓	✓	✓	✓	✓

Method	Inference CPU	Inference GPU	Compute location	Rate / timing (reported boundary)
MotionBricks + SONIC	Orin SoC[a]	Orin integrated[a]	G1 / Jetson Orin	50 Hz tracker[*]; 10 Hz replanning; 5 ms generator
ARDY + SONIC	–	RTX 4090[b]	Workstation[b]	33 / 63 ms generator (4 / 10 steps)[b]
TextOp	i7-13700[c]	RTX 4090[c]	External generator; onboard tracker	6.25 Hz generator; 50 Hz tracker
ReactiveBFM	i9-13900K	RTX 4090	Offboard workstation	50 Hz controller; async planner (19.3 ms)
CLoSD	–	–	Simulation only	11.4 ms for a 2 s motion plan
DiffuseLoco	i7-13700H	RTX 4060 Mobile	Onboard Go1 mini-PC	30 Hz control target; 116.5 Hz inference
Diffuse-CLoC	–	RTX 4060	Simulation only	77.41 Hz inference (selected profile)
BeyondMimic	– (CPU decoder)	RTX 4060 Mobile	Onboard mini-PC (v4)	25 Hz control; async diffusion about 20 ms
SCDP	–	RTX 5090	Remote workstation	50 Hz control; 105 Hz inference
SCRIPT	–	–	Simulation only	Physical control: N/A
PredActor	**Orin NX SoC**	**Orin NX integrated**	**Fully onboard G1**	**50 Hz control; callback p95 19.383 ms**

Symbols: ✓: supported; ◇: partial/modular; ×: explicitly unsupported; –: unreported/unverified; N/A: not applicable.
Capability definitions: Diffusion paradigms follow Figure 2; MotionBricks uses a non-diffusion generator + tracker. Proprio. means proprioceptive input; dec. means decoder; Robot denotes physical deployment. Proprioceptive input permits task commands but excludes privileged physical-state input; ◇ marks tracker-level input in generator–tracker systems. Rolling reuse requires explicit reuse of overlapping state/action predictions or their partially denoised representations, not just replanning. CFG requires conditional–null prediction mixing; ◇ marks ARDY generator-level CFG/text support without separate validation in the SONIC composition.
Hardware notes: [*]50 Hz is sourced from MotionBricks' adopted SONIC tracker, not independently reported for the composition. [a]Orin variant, CPU model, and full-stack timing are unreported. [b]ARDY generator-demo benchmark; robot-stack hardware and rate are unreported. [c]TextOp external workstation; tracker CPU is unreported. Inference hardware is not inferred from training hardware.
Timing boundary: PredActor's callback covers received robot-state messages through host-action production; 593/600 calls meet 20 ms. The timing benchmark does not actuate the robot. Component timings are not full-loop measurements. Sources: Appendix A.1.

Table 5 expands the main-text comparison with the full capability matrix, inference hardware, and timing boundaries. It separates inference hardware from training hardware and robot control rates from generator throughput. MotionBricks reports Jetson Orin deployment on G1 with 10 Hz replanning and 5 ms generator latency (Section 7.4), without specifying the Orin variant or a complete-loop latency (Wang et al., 2026). Its resource-sharing discussion suggests that the generator and tracker share onboard compute, although their individual hardware assignments and the tracker rate are not explicitly specified. The table's 50 Hz tracker entry is sourced from its adopted SONIC controller (Section 3.5), not an independently reported MotionBricks measurement (Luo et al., 2026). ARDY's Section 4.2 reports an RTX 4090 interactive generator with 33/63 ms latency for 4/10 diffusion steps; these measurements do not characterize an ARDY–SONIC robot control loop (Zhao et al., 2026).

The two generator–tracker compositions have separate capability rows. MotionBricks steers motion through keyframes and root-trajectory constraints supplied by smart primitives (Sections 5–6). Its masked-token generator does not report conditional–null prediction mixing (CFG) or sampling-time objective gradients (CG); the CFG discussion in Section 7.2 concerns comparison baselines.

ARDY explicitly supports text and spatial conditioning with CFG (Section 3.5); these are marked as generator-level capabilities in the SONIC composition. The ballet and locomotion videos under Humanoid Robot Control on its official project page show simulated G1 tracking, not physical deployment. We therefore leave its physical-deployment entry unmarked. Both systems support spatial steering, which does not by itself establish CG. Their proprioceptive-input marks refer to the tracker, not robot feedback to the generator.

TextOp runs its generator on an external i7-13700/RTX 4090 workstation at 6.25 Hz and its tracker onboard at 50 Hz (Section III-E and Appendix B-F) (Xie et al., 2026). ReactiveBFM uses an offboard i9-13900K/RTX 4090 workstation, a 50 Hz controller, and an asynchronous planner; the reported 19.3 ms planner and 5.9 ms controller timings are component measurements (Deployment and Appendix B.5) (Chen et al., 2026b). SCDP uses a remote RTX 5090 workstation for 50 Hz G1 control and reports 105 Hz inference throughput (Section II-D) (Carroll et al., 2026).

For BeyondMimic, we use arXiv v4, Supplement S4 (Diffusion): the diffusion-stage setup uses 25 Hz rather than the standalone tracker's 50 Hz, with onboard RTX 4060 Mobile inference and a synchronous CPU VAE decoder (Liao et al., 2026). Its approximately 20 ms asynchronous diffusion latency is not a complete-loop measurement. Earlier versions (v1, v2, and v3, Section VI-B) instead describe offboard inference without stating this 25 Hz setting. The table follows v4, not the earlier deployment description. We classify its latent state–action model as joint diffusion, rather than a kinematic generator followed by a reference tracker.

DiffuseLoco reports onboard Go1 deployment on an i7-13700H/RTX 4060 Mobile mini-PC; its 30 Hz control requirement differs from the 116.5 Hz TensorRT inference benchmark (Appendix E and Figure 16) (Huang et al., 2025a). Diffuse-CLoC evaluates simulated characters and reports RTX 4060 inference, with 77.41 Hz for the selected profile (Section V-B and Table II) (Huang et al., 2025b). CLoSD's 11.4 ms measurement generates a two-second motion plan, not a physical-robot callback (Section 4.1) (Tevet et al., 2025); SCRIPT evaluates simulated humanoids rather than onboard robot execution (Section 5.1) (Zhang et al., 2026a). Unreported hardware and rates are marked "–", even when training hardware is stated; N/A denotes physical control rates for simulation-only systems.

PredActor combines proprioceptive conditioning, CG/CFG, direct joint-action execution, and fully onboard G1 inference on Jetson Orin NX. Its measured callback runs from received robot-state messages to host-action production, with p50/p95 of 16.790/19.383 ms and 593/600 calls within 20 ms. Physical execution is demonstrated separately; the timing benchmark issues no actuation commands.

A.2 Motion-Control Landscape

The literature spans several control paradigms that are easy to conflate. A *motion tracker* receives a reference trajectory or motion representation and produces physically executed actions. PHC established scalable perpetual tracking and recovery for simulated avatars Luo et al. (2023). Recent robot trackers emphasize different bottlenecks: MimicLite reduces the reported training cost of a deployable G1 tracker RoboParty Lab Team (2026); SONIC, released as GEAR-SONIC, scales model capacity, motion data, and compute while exposing several downstream motion interfaces Luo et al. (2026); HoloMotion-1 uses a hybrid motion corpus and temporal mixture-of-experts model for zero-shot whole-body tracking Chen et al. (2026a); and Any2Track couples a general tracker with a history-conditioned dynamics adapter for disturbances Zhang et al. (2026b). ProtoMotions3 is instead an open simulation, learning, and deployment framework that can train tracking policies; it is not one additional controller architecture Tessler et al. (2025). These trackers and tools strengthen the execution side of generator–tracker stacks, but do not themselves select a long-horizon task behavior without a reference or higher-level input.

A.3 Reactive Planner–Tracker Comparisons

ReactiveBFM distinguishes open-loop cascades from closed-loop compositions with physical-state feedback; those labels describe the evaluated composition and do not imply that every component replans from physical state (Chen et al., 2026b).

A second family broadens the policy interface. BFM-Zero learns a shared latent space for motion, goal, and reward prompts using unsupervised reinforcement learning Li et al. (2026). BFM4Humanoid combines masked online distillation with a conditional variational autoencoder to support multiple whole-body control modes Zeng et al. (2025). LangWBC is a non-diffusion comparison: reinforcement learning, policy distillation, and a conditional variational autoencoder yield one language-conditioned policy that outputs actions directly Shao et al. (2025). These are unified, promptable, or direct controllers rather than a kinematic planner followed by an independently trained tracker.

Teleoperation systems occupy another role. H2O, OmniH2O, TWIST, and CLONE map live human inputs to whole-body robot control, with CLONE specifically adding closed-loop global-position correction for long-horizon teleoperation He et al. (2024; 2025); Ze et al. (2025); Li et al. (2025). They are important sources of control interfaces and demonstrations, but are not autonomous motion generators in the sense used here. HumanoidBench, introduced by Sferrazza et al. (2024), is likewise an evaluation suite for locomotion and whole-body manipulation rather than a controller. This taxonomy motivates comparisons along reference dependence, task conditioning, generated variables, deploy-time observations, guidance, and closed-loop compute rather than a single overloaded "end-to-end" label.

A.4 Trajectory Generative Models for Control

Diffusion control methods differ in what they generate. Action-only policies denoise action chunks conditioned on observations Chi et al. (2023); this formulation extends to offline legged locomotion Huang et al. (2025a) and physics-based character control Truong et al. (2024). Joint trajectory models also represent future states: Diffuser performs return- or constraint-guided planning Janner et al. (2022), while Decision Diffuser uses conditional state-sequence generation with inverse dynamics to recover actions Ajay et al. (2023). Outside diffusion, H-GAP learns a discrete latent state–action trajectory model and uses model-predictive planning for generalist simulated humanoid control Jiang et al. (2024). PredActor follows the joint state–action diffusion route: future states support guidance within the model that generates executable actions.

Recent humanoid systems provide closer comparisons. BeyondMimic guides a latent diffusion model of human motion primitives toward test-time objectives and transfers the resulting controller to hardware Liao et al. (2026). SCDP generates interleaved future states and actions from onboard observation histories, using privileged future trajectories for supervision and partial observations at deployment Carroll et al. (2026). SCRIPT jointly represents action, state, and text streams in a diffusion transformer and adds reinforcement-learning post-training for language-driven simulated humanoids Zhang et al. (2026a). Within this landscape, PredActor combines proprioceptive-history conditioning with learned-command CFG and future-state CG, then implements these interfaces within an onboard control loop. Generated variables, policy inputs, guidance mechanisms, and hardware evidence therefore form separate comparison axes in Table 1.

A.5 Efficient and Temporally Consistent Diffusion Inference

Real-time control constrains the cost of iterative sampling. DDIM introduces a non-Markovian implicit process that permits deterministic subsampling without retraining Song et al. (2021); DPM-Solver uses dedicated high-order integration for diffusion ODEs Lu et al. (2022); and consistency models learn direct mappings from noise to data Song et al. (2023). Robot-specific distillation similarly trades extra training for fewer evaluations: OneDP distills a pretrained diffusion policy into a one-step generator Wang et al. (2025). These approaches accelerate or replace the sampler, but do not specify how successive control horizons should share computation.

Temporal reuse addresses that second issue. Streaming Diffusion Policy carries forward a partially denoised action trajectory whose near-term entries are cleaner than its tail Høeg et al. (2025). TEDi entangles diffusion time with motion time in a recursively advanced buffer for long-horizon motion synthesis Zhang et al. (2024), while Diffusion Forcing assigns per-token noise levels to combine next-token prediction with full-sequence diffusion Chen et al. (2024). Action chunking with temporal ensembling smooths overlapping predictions Zhao et al. (2023), and constrained diffusion adds feasibility projections at sampling time Römer et al. (2025). PredActor's (K, h) noise matrix provides one implementation of per-position schedules, DDIM jumps, and rolling-buffer reuse.

A.6 Imitation Learning and Iterative Data Aggregation

Behavior cloning suffers from covariate shift because learner actions change the future state distribution. DAgger queries the expert on learner-induced states and aggregates the resulting labels Ross et al. (2011). DART instead perturbs demonstrations to collect recovery behavior without executing the novice during collection Laskey et al. (2017), whereas HG-DAgger gates human intervention using a learned uncertainty threshold Kelly et al. (2019). These methods separate three design choices that are sometimes conflated: who visits the state, who supplies the label, and whether the collected sample is appended to replay.

Diffusion policies can also be trained with reward optimization rather than label aggregation. DPPO supplies a policy-gradient recipe for fine-tuning diffusion policies Ren et al. (2025), and REFINE-DP jointly fine-tunes a diffusion planner and low-level controller for humanoid loco-manipulation Gu et al. (2026). PredActor instead uses a standard iterative aggregation recipe: learner-rollout origins seed isolated teacher continuations, whose trajectory labels enter combined replay for continued diffusion reconstruction. This training procedure is distinct from offline noise injection and reinforcement-learning fine-tuning.

A.7 Conditioning and Humanoid Control Interfaces

Humanoid objectives may be expressed as velocity commands, sparse body targets, motion clips, language, or teleoperation. MotionCLIP aligns text and human motion embeddings Tevet et al. (2022); CLoSD couples text-conditioned motion diffusion to physics-based control Tevet et al. (2025); and TextOp streams text changes through a motion generator and a low-level tracking policy Xie et al. (2026). MaskedMimic treats diverse partial motion descriptions—including text, keyframes, objects, and paths—as masks for one physics-based controller Tessler et al. (2024). Whole-body teleoperation systems such as H2O, OmniH2O, and TWIST instead map live human motion to humanoid control He et al. (2024; 2025); Ze et al. (2025).

PredActor exposes each conditioning source through a shared runtime interface. Text embeddings, joystick commands, semantic interpolation, and whole-body guidance can therefore reuse the same exported policy interface. The experiments evaluate behavioral execution and control-period timing separately from interface availability.

B Extended Method and System Details

The following details separate PredActor's predictive representation from its training and execution machinery. Future states are internal variables for guidance, proprioceptive histories supply policy inputs, and selected actions are issued to the joint controller. Teacher supervision, rolling schedules, and runtime optimizations implement this policy without introducing a separate motion-generation and tracking hierarchy at deployment.

B.1 Predictive State–Action Representation

B.1.1 Proprioceptive inputs and internal future states

Table 6 separates the deployable observation from the physical state predicted by PredActor. The evaluated PredActor policies use a 96-dimensional observation that includes base linear velocity. The 192-dimensional physical state is mapped by the fixed emphasis projection to a 384-dimensional model representation. The action $\mathbf{a}(t)$ contains 29 joint-controller commands, converted to PD targets by the runtime. Optional task context $\mathbf{z}$ encodes the requested task; the text-conditioned setting uses a 512-dimensional semantic embedding. Normalization statistics are fitted on the training split and fixed at test time.

The model directly predicts clean trajectories and is optimized with reconstruction losses on both streams. PredActor does not contain a separate deterministic observation-to-state module. In the evaluated setting, deployment initializes the entire state stream from Gaussian noise and reconstructs it jointly with actions through conditional co-diffusion. The reconstructed state trajectory remains internal; only an action is sent to the robot. Onboard signal processing remains part of observation construction.

Table 6: Observation and physical-state components for the selected G1 profile. The evaluated 96-dimensional observation includes base linear velocity. The emphasis projection changes representation dimension, not the underlying physical state.

Component	Dimension	Description
Deployable observation $\mathbf{o}_t$		
Joint positions	29	Robot joint coordinates
Joint velocities	29	Robot joint velocities
Projected gravity	3	Gravity in the base frame
Base angular velocity	3	IMU angular velocity
Previous action	29	Last joint command
Base linear velocity	3	Base-frame linear velocity
Selected observation total	**96**	Used by the evaluated profile
Physical state $\mathbf{s}(t)$		
Local body positions	90	30 bodies $\times$ 3 coordinates
Local body linear velocities	90	30 bodies $\times$ 3 velocities
Root position	3	Local character frame
Root orientation	3	Local rotation representation
Root linear velocity	3	Local character frame
Root angular velocity	3	Local character frame
Physical-state total	**192**	$d_{\mathrm{s}}^{\mathrm{phys}}$
Emphasis-projected state	384	Model representation d_s

B.1.2 Interleaved conditional transformer

State and action inputs are projected separately, concatenated with sinusoidal embeddings of their respective noise levels, and interleaved as $[s_1^{\mathrm{tok}}, a_1^{\mathrm{tok}}, \ldots, s_h^{\mathrm{tok}}, a_h^{\mathrm{tok}}]$. Learned positional embeddings retain horizon order. A transformer decoder applies self-attention over this sequence and cross-attention to encoded condition tokens. In the reference attention mask, state queries attend to all state tokens but not action tokens; action queries attend causally to state and action tokens. Separate output heads predict clean states and clean actions. The current reference backbone uses two decoder layers, four attention heads, and a 256-dimensional embedding; these are configuration choices rather than architectural requirements.

A fixed, profile-driven state emphasis transform E can be applied before the backbone and its pseudoinverse $E^{\dagger}$ after prediction. The evaluated transform concatenates a weighted symmetric random projection with an identity component. It is specified before training and is not learned by gradient descent.

B.2 Dataset Construction and Iterative Data Aggregation

B.2.1 Kinematic library and initial rollouts

Figure 4(a) combines prompted motion generation and annotated motion sources into a common kinematic library with task labels. A tracking teacher executes these references under observation noise, action noise, and external perturbations. Asynchronous rollouts store synchronized observation histories, future body states, teacher actions, and task context to form the initial offline dataset $\mathcal{D}_0$. Complete rollout trajectories are split into training and evaluation sets before extracting fixed-length windows; normalization statistics are fitted only on the training split. Thus, no rollout contributes windows to both partitions. The teacher tracker is a training-time data source, not a component of PredActor's inference path.

B.2.2 Teacher trajectory shooting and aggregation

The aggregation update is given in Equation 2. The teacher continuation, not the student's future actions, supplies the target trajectory. The update adds supervised windows whose origins were visited during learner rollout; it does not change the diffusion objective or introduce reward optimization.

The horizon-teacher procedure in Figure 4(b) implements this training recipe and provides joint trajectory supervision for Equation 2. At round j, a frozen student advances the live simulator. A rollout origin ξ contains the simulator state, student observation/action history, and motion reference phase needed to start an aligned teacher continuation. The tracking teacher is queried in a separate simulation branch with the same origin and reference. State and random-number-generator checks ensure that the query does not advance or alter the live student trajectory. Thus teacher actions label the branch; they are not interventions in the live rollout.

The window constructor W_E retains the observed student-history prefix and appends the teacher continuation. Past actions are the student's executed actions paired with their pre-transition states; the action at the branch origin and subsequent actions come from the teacher. State targets follow the same temporal alignment. Task labels are aligned with the reference, and incomplete continuations are marked so that only complete training windows enter replay. Stored-window prefix lengths are configuration details and do not change the horizon-position convention $i = 1, \ldots, h$. Each set of collected windows is used for continued diffusion reconstruction before the next collection round.

B.2.3 Joint reconstruction objective

Equation 3 is used for both initial training and subsequent replay. Its expectation covers windows from $\mathcal{D}$, sampled noise, and diffusion levels. The action-difference term compares consecutive predicted and target actions, while the state reconstruction term supervises the internal future-state trajectory used for guidance. Iterative collection changes which windows enter replay without replacing this reconstruction objective.

B.3 Task and Deployment Interface

Task-specific providers inject raw condition keys on every step, and a composite provider distributes updates to its children. A condition composer then assembles the history consumed by the policy. This division allows observation-only and semantic/command-conditioned actors to share the same policy interface without hard-coding a particular input source.

The exported artifact contains a TorchScript backbone, diffusion schedule, normalizer statistics, actor configuration, and deployment configuration. The C++ runtime normalizes observations and conditions, runs the actor, unnormalizes the action trajectory, and forms the delay-compensated PD target by linearly interpolating the two clean action slots surrounding $u = (n_{\text{obs}} - 1) + \delta/\Delta t$. An asynchronous UDP receiver latches the latest external condition under a mutex, so condition transport is decoupled from the control loop. A Passive–Ready–Running state machine gates policy actions and falls back to Passive on fall or torque faults.

B.4 Evaluated Deployment Profile

The evaluated G1 Orin profile is a batch-one FP32 conditioned PredActor policy with two-step deterministic DDIM, packed self- and cross-attention projections, cached command embeddings, a precomputed schedule, stacked-axis WBG, eager finite checks, and LibTorch inference mode.

For each level t_i, the reduced backbone predicts clean action and state trajectories once,

$$(\hat{a}_i^0, \hat{s}_i^0) = F_\theta(a_i, s_i, t_i, c_{\text{enc}}). \tag{5}$$

When guidance is active, the same state prediction is adjusted by an analytic whole-body cost and reused in the DDIM update,

$$\tilde{s}_i^0 = \hat{s}_i^0 - G(\hat{s}_i^0; u_{v_x}, u_{v_y}, u_{v_z}), \qquad (a_{i-1}, s_{i-1}) = \mathcal{D}(a_i, s_i, \hat{a}_i^0, \tilde{s}_i^0; t_i, t_{i-1}). \tag{6}$$

The same clean state prediction is reused by guidance, and $\hat{a}_i^0$ is not recomputed at level t_i. The modified state enters s_{i-1}, so the next backbone evaluation can propagate the steering effect to its action prediction through state-to-action attention. The evaluated path therefore performs one backbone evaluation per diffusion step, or two in total. A duplicate-backbone variant executes four forwards and changes the guidance equation, so it is excluded from the exact acceleration ablation. The actor also performs encoding, guidance, scheduler updates, and boundary checks, so operation counts do not determine complete-loop speedups. Actor and callback timings, their acceptance thresholds, and the nonadditive component diagnostics are reported in Section 4.2 and Appendix D.1.

C Detailed Experimental Protocol

C.1 Outcome Definitions and Missingness

Survival and recovery. Architecture-study push survival is the observed fraction of the 650-step horizon, so an early terminal contributes its observed duration rather than a binary zero; it is not the fraction of cells that finish. A terminal is triggered by root height below 0.55 m, absolute pelvis tilt at least 0.3 rad, or base angular speed at least 1 rad/s. Strict recovery requires 25 sustained post-force steps inside all three gates. In the architecture study, early failures remain in every denominator.

Destination and text outcomes. Destination arrival is first entry within 0.6 m of the target. Final and minimum error use horizontal root-to-target distance. Text retrieval is raw top-1 MotionCLIP prompt retrieval over complete windows beginning at least 3 s after command activation. A trial may therefore contribute to survival but lack an eligible retrieval window. Generated references outside the tracker converter's accepted joint range are retained as non-evaluable rather than assigned failure or zero.

Timing. The timing endpoints in Figure 7 are E_0 (actor execution with a fixed input tensor), E_1 (receipt and conversion of a new robot state followed by actor execution), and E_2 (the complete callback through production of the host action tensor). These boundaries are measured directly; their percentiles are not added. The 20 ms line is the 50 Hz target used for percentile and deadline-attainment reporting.

C.2 Architecture Protocol and Coverage

The six learned policies use the same 7,600-episode dataset, comprising 50 teacher rollouts for each of 152 G1-retargeted ACCAD motions. Training uses 50 epochs, a base learning rate of 10^{-4}, cosine decay with 10,000 warm-up steps, batch size 256, data sampling rate 0.01, and seeds 92025, 92026, and 92027. Reported means and sample SDs are computed across the three checkpoint means. ARDY+SONIC is a single-model reference.

Table 7: Key Q1 policy settings. S+A denotes joint future-state and action prediction; A denotes action-only prediction. Text/CFG records the learned text-conditioning and classifier-free-guidance interface, whereas state CG records the predicted-state classifier-guidance interface used for destination steering.

Policy	Output	Text/CFG	State CG	Observation/condition profile	Sampler	n_{train}
DiffuseLoco w TextCFG	A	yes	no	`g1_rlobs`	DDPM-20	3
CLOC	S+A	no	yes	`g1_standard`	DDPM-20	3
CLOC w TextCFG	S+A	yes	yes	`g1_standard`	DDPM-20	3
PredActor w/o TextCFG	S+A	no	yes	`g1_prdp_standard`	NS-DDIM-2	3
PredActor	S+A	yes	yes	`g1_prdp_standard_cond`	NS-DDIM-2	3
PredActor w/o state	A	yes	no	`g1_prdp_standard_cond`	NS-DDIM-2	3
ARDY+SONIC	reference $\rightarrow$ A	reference	no	`sonic_release`	ARDY-H8	1

PredActor w/o TextCFG and PredActor both predict a 20-step state–action trajectory with full cross-attention and two deterministic NS-DDIM steps. PredActor w/o TextCFG receives a 96-D proprioceptive condition through `g1_prdp_standard`; PredActor adds a 512-D text latent through `g1_prdp_standard_cond`, giving a 608-D condition. Both use data sampling rate 0.01 during training. Their destination controllers use v_x ranges of $[0, 5]$ and $[0, 8]$ m/s, respectively.

The matched state-prediction ablation pairs separately retrained joint state–action and action-only checkpoints by seed under a primary no-CG contract. The matched joint state–action arm is distinct from the configured PredActor row in Table 2; the action-only row in that table is the other arm. Table 8 reports the paired comparison.

Table 8: Matched state-prediction ablation under the primary no-CG contract. Both arms are separately retrained for this comparison. Arm entries are mean ± sample SD across three training checkpoints. Effects are matched joint state–action minus action-only, with two-sided 95% t intervals across the three paired seed differences.

Metric	Matched PredActor	PredActor w/o state	Paired effect [95% CI]
Push survival ↑	0.535 ± 0.009	0.487 ± 0.033	$+0.048$ $[-0.024, 0.120]$
Push recovery ↑	0.043 ± 0.040	0.193 ± 0.110	-0.150 $[-0.494, 0.194]$
Text survival ↑	0.866 ± 0.011	0.778 ± 0.145	$+0.088$ $[-0.246, 0.421]$
Text retrieval ↑	0.538 ± 0.087	0.397 ± 0.023	$+0.142$ $[-0.019, 0.302]$
Jerk RMS ↓	15.684 ± 2.368	12.448 ± 0.675	$+3.236$ $[-2.200, 8.671]$
Latency at $B = 5$ (ms) ↓	1.335 ± 0.039	1.050 ± 0.004	$+0.285$ $[0.199, 0.372]$

Both arms use the same data, observation and condition tensors, 20-step action horizon, optimizer and schedule, two-step sampler, runtime, and seeds. Trainable parameter counts differ by 0.0161%. Destination metrics are unavailable because online state CG is disabled. Retrieval uses 113/135 eligible windows for matched PredActor and 98/135 for the action-only arm.

The architecture study uses one calibrated MuJoCo plant with whole-body guidance disabled outside the destination task. Push evaluation applies 100, 200, 300, 400, or 500 N horizontally for 0.5 s during walking. Each checkpoint uses the same 20 paired direction/seed draws at every force, giving 100 requested cells; the table therefore contains 300 cells for a three-checkpoint row and 100 for a single-model row. Destination following uses three target draws for each of five evaluation seeds and runs for at most 20 s. Text evaluation crosses nine prompts with the five evaluation seeds. Actor latency is synchronized on one exclusive RTX 4090, measured after warm-up, and reported per action from a fixed policy input repeated to batch five. Basic walking jerk uses the paired pre-push nominal segment: 20 cells per checkpoint for the original architecture-suite rows and 25 cells per checkpoint for PredActor w/o state from its matched state-ablation suite, yielding 60/60 and 75/75, respectively. Table 9 records evaluated and requested cells after combining training replicas without changing the outer uncertainty unit.

Table 9: Architecture-study coverage. Counts are evaluated/requested cells.

Policy	Push	Destination	Text survival	Text retrieval
DiffuseLoco w TextCFG	300/300	0/45	135/135	126/135
CLOC	300/300	45/45	0/135	0/135
CLOC w TextCFG	300/300	45/45	135/135	103/135
PredActor w/o TextCFG	300/300	45/45	0/135	0/135
PredActor	300/300	45/45	135/135	119/135
PredActor w/o state	300/300	0/45	135/135	98/135
ARDY+SONIC	100/100	2/15	23/45	23/45

Figure 8 resolves the aggregate push results by force magnitude.

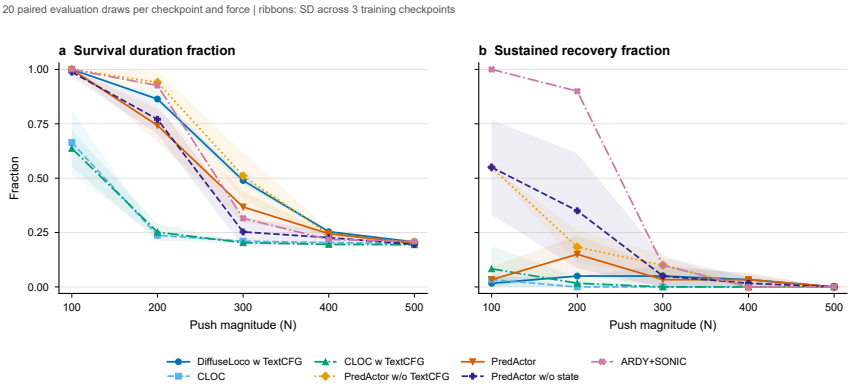

Figure 8: **Architecture survival and recovery over all tested push magnitudes.** Each checkpoint contributes 20 paired evaluation draws at each force. Points are means across checkpoint means and ribbons show sample SD across the three training checkpoints. ARDY+SONIC has one checkpoint and no ribbon. Table 2 averages all five forces.

C.3 Deployment Demonstration Scope

The demonstration suite spans simulation and hardware. Joystick steering and semantic interpolation are evaluated in simulation (Figure 1(a,c)); external interference and staged text commands are demonstrated on the physical Unitree G1 (Figure 1(b,d)). Figure 7(g) provides additional onboard frames from the G1–Jetson Orin NX stack. The deployed actor uses the observation, condition, export, and safety interfaces described in Appendix B.3.

The selected physical sequences show successful integrated execution, while Table 3 and Appendix D.4 report all five outcomes per setting.

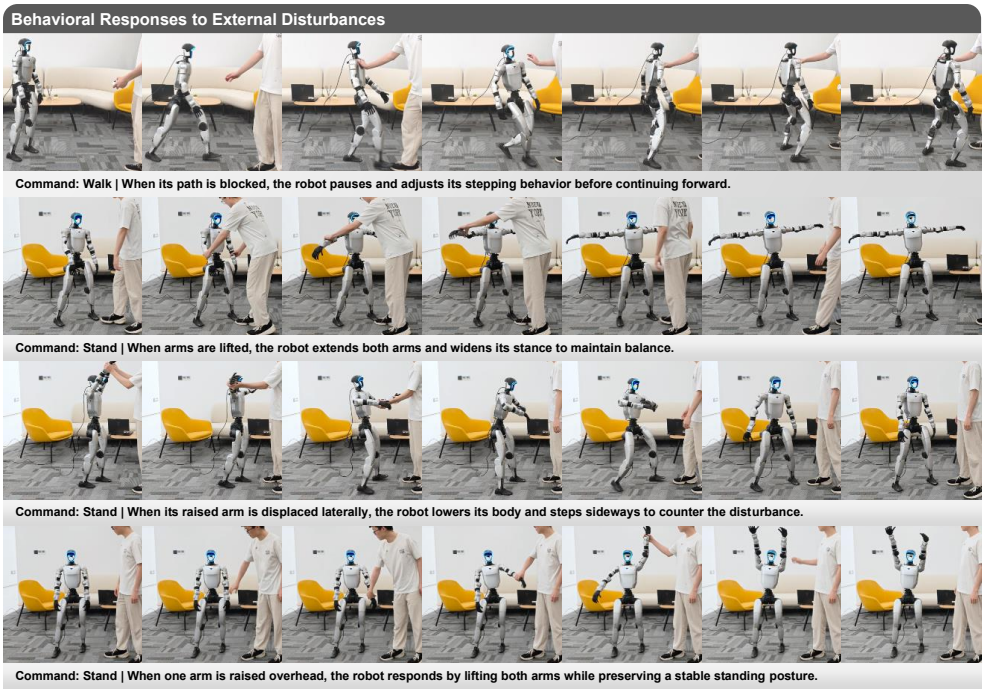

Figure 9: Qualitative behavioral responses of the physical Unitree G1 under external interaction. Each sequence identifies the active command and the observed response: the walking policy adjusts its stepping when obstructed, while the standing policy coordinates arm extension, stance widening, body lowering, and lateral stepping in response to upper-body or arm displacement. These selected successful sequences illustrate the observed response modes.

D Complementary Results and Limitations

D.1 Orin Sampler and Implementation Ablations

The sampler sweep in Table 10 uses P_0, the original TorchScript actor before the acceleration changes evaluated here. Rolling DDPM has 20 schedule levels but executes 15 updates in the deployed rolling window. With legacy WBG it evaluates the backbone twice per update. The two-step WBG path executes two backbone forwards in total. Figure 7(a) reports the timing sweep, and Figure 7(d) separately measures push survival across the tested sampler budgets; together they identify DDIM-2 as the best tested trade-off between inference speed and disturbance survival.

Table 10: P_0 actor timing on Jetson Orin. Values are actor CUDA p50 in milliseconds, each the median of three block medians.

Sampler	Steps	WBG off	WBG on
NS-DDIM	1	9.725	12.221
NS-DDIM	2	16.148	21.846
NS-DDIM	3	22.915	29.946
NS-DDIM	5	36.001	46.948
NS-DDIM	10	68.091	92.011
Rolling DDPM	15	121.229	235.216

The exact factorial study crosses four attention-projection layouts with condition-embedding reuse, DDIM schedule precomputation, and sequential versus stacked WBG axes. All 32 valid combinations use the production one-backbone-per-denoising-step guidance update (two backbone forwards

total for NS-DDIM2). Across matched valid cells, all-QKV packing saves 0.775 ms with a range of [0.256, 0.799] ms across the three block-level effects; condition reuse saves 0.190 ms, with a range of [0.151, 0.347] ms; precomputation saves 1.672 [1.477, 1.698] ms, and stacked WBG saves 2.845 [2.627, 2.868] ms. These conditional effects depend on the other settings and must not be summed. The cumulative path in Table 4 reports individual matched composition differences rather than these matrix-wide conditional effects. Cross-attention-only packing is not consistently faster. An implementation that duplicated backbone execution changes the guidance equation and is excluded from the exact ablation.

The optimization path also crosses two controlled boundaries. The original-export sampler endpoint $P_0 = 21.846$ ms and the matched structural-ablation baseline of 21.461 ms belong to separate suites. Within the structural-ablation suite, stacked WBG reduces this matched baseline to $P_1 = 18.315$ ms; P_0 is not differenced against P_1. The factorially selected exact configuration P_2 measures 15.930 ms. The same configuration measures 16.234 ms as P_3 in the runtime suite; the difference reflects the measurement suite rather than an implementation change. Enabling LibTorch inference mode at P_4 reduces the median by 3.058 ms to 13.175 ms.

Table 11 collects all 32 valid structural compositions as 16 sequential/stacked WBG pairs, all eight runtime controls, and matched component effects. Frozen actor and live callback timings remain separate; detailed per-boundary records are retained in the source data.

Table 11: **Comprehensive exact Orin implementation ablation.** NS-DDIM2, B=1 FP32, WBG on, two backbone forwards. Times are milliseconds.

A. All 32 structural compositions (16 paired rows).

QKV	C	P	Seq. p50	Seq. p95	Stk. p50	Stk. p95	$\text{Saved}_{\text{WBG}}$
none	0	0	21.461	22.036	18.315	19.677	3.147
self	0	0	21.233	22.260	17.972	18.376	3.261
cross	0	0	20.669	22.016	18.583	18.885	2.086
all	0	0	20.522	20.812	17.751	18.873	2.771
none	0	1	19.082	20.631	17.442	18.347	1.640
self	0	1	19.315	20.292	16.232	17.116	3.083
cross	0	1	19.674	20.656	17.180	17.580	2.494
all	0	1	18.685	19.374	16.331	17.372	2.354
none	1	0	20.985	21.948	18.188	19.359	2.798
self	1	0	20.384	21.196	17.789	19.430	2.595
cross	1	0	20.717	21.823	17.888	18.414	2.829
all	1	0	21.112	22.008	17.423	18.262	3.688
none	1	1	19.184	19.492	16.631	17.572	2.553
self	1	1	19.258	20.695	16.104	16.825	3.154
cross	1	1	19.177	19.492	16.387	16.720	2.790
all	1	1	18.803	19.467	**15.930**	**16.728**	2.873

C: condition cache; P: DDIM precomputation (0/1 = off/on). Seq./Stk.: sequential/stacked WBG. $\text{Saved}_{\text{WBG}}$ = Seq. p50 − Stk. p50 in the same cell; inference off, eager checks.

B. All eight runtime configurations (separate suite).

Structure	I	Checks	Actor p50	p95	Saved	Host p50	$>$ 20 ms
base	0	eager	18.266	19.104	0.000	21.042	600/600
base	0	defer	18.301	19.286	-0.035	20.943	600/600
base	1	eager	15.401	15.651	2.865	19.451	49/600
base	1	defer	14.924	15.201	3.342	18.873	61/600
all	0	eager	16.234	16.912	2.032	18.855	60/600
all	0	defer	15.374	15.932	2.892	18.446	90/600
all	**1**	**eager**	**13.175**	**13.400**	**5.090**	**16.790**	**7/600**
all	1	defer	13.183	13.609	5.083	16.836	13/600

Base/all: QKV packing, cache and precompute all off/on; WBG stacked. I: inference mode. Saved compares frozen actor p50 to the first row (18.266 ms). Host = live callback-to-host; last column counts host deadline misses. Deferred checks change failure-detection semantics.

C. Conditional frozen-actor savings, not additive.

Component	Matched change	Pairs/block	Saved	Block-effect range
QKV packing	none → self	8	0.221	[0.003, 0.555]
QKV packing	none → cross	8	0.172	[-0.100, 0.363]
QKV packing	none → all	8	0.775	[0.256, 0.799]
Condition cache	off → on	16	0.190	[0.151, 0.347]
DDIM precompute	off → on	16	1.672	[1.477, 1.698]
WBG axes	sequential → stacked	16	2.845	[2.627, 2.868]
Inference mode	off → on (selected structure)	1	3.166	[2.812, 3.492]

Each primary cell uses three rotated blocks of 200 retained calls; p50/p95 are medians of block quantiles. Panel C gives median matched block effects and their min–max range, not confidence intervals. Its inference-mode paired effect (3.166 ms) differs from the 3.058 ms difference of endpoint medians shown in Figure 7(e) and Table 4. Bold marks the selected structure/runtime. The E1 new-state-plus-actor endpoint has p50/p95 of 13.843/16.221 ms; callback p95 is 19.383 ms, with 7/600 calls over 20 ms. Sampler controls remain in Table 10; the complete per-boundary source data retain block ranges, actor misses and live structural timings omitted here. The 32 duplicate-backbone cells are excluded as an execution bug; no cross-suite savings are inferred.

D.2 Timing Decomposition

Table 12 records the median component diagnostics most relevant to the deployment boundary. The two backbone and WBG spans are inclusive traced CUDA regions. Their medians cannot be added to each other or to host spans. Tracing inflates the final actor span to 40.090 ms, whereas an independent unprofiled fixed-input pass measures 12.198 ms; neither replaces the primary 13.175 ms E_0 endpoint.

Table 12: Diagnostic component p50 timing in milliseconds.

Component	P_0	P_4	Boundary
State fetch/copy	0.002	0.001	host
Pre-FK and FK	0.757	0.779	host
Observation construction	0.919	0.997	host
Condition packing	0.083	0.082	host
Action selection/copy	0.234	0.238	host
Backbone forward 1	18.053	13.056	traced CUDA
Backbone forward 2	18.253	13.852	traced CUDA
WBG gradient 1	3.095	2.562	traced CUDA
WBG gradient 2	3.030	2.648	traced CUDA
Actor-loop residual	8.054	2.474	traced CUDA

D.3 Capability, Coverage, and Interpretation Limits

Architecture study. Six policies each use three independently trained checkpoints with seeds 92025/92026/92027. Means and sample SDs are taken across checkpoint means. Each checkpoint contributes 20 paired evaluation draws at each of five push forces, 15 destination trials, 45 text trials, and five timing runs where the policy supports the corresponding task. ARDY+SONIC uses one checkpoint; 2/15 destination references and 23/45 text references pass its joint-range converter. MotionCLIP retrieval measures agreement between the executed motion and the requested text.

Timing study. Timing applies to one batch-one FP32 artifact on one Jetson Orin. The benchmark measures the offline callback-to-host-action boundary from received robot-state messages; all actuation counters remain zero. The complete-callback p95 is below 20 ms, and 593/600 calls meet the target; the seven observed overruns define the remaining tail-latency margin. Physical closed-loop behavior is documented by the deployment demonstrations. Graph bytes, thermal state, runtime version, and implementation settings are part of the measured system; a different export or device requires requalification.

Real-robot demonstrations and scope. Figures 1(b,d), 7(g), and 9 document selected successful hardware executions of the text-command and disturbance-response interfaces. Table 3 gives descriptive success counts from five trials per setting, and Figures 10–15 show all recorded outcomes. Success requires completing the requested transition or remaining upright after the push; falls, incomplete behaviors, and safety stops are failures.

Future scope. The current guidance interface is calibrated over the tested objective range; stronger guidance can move denoising outside the policy's training distribution. Text control is limited to learned semantic categories because the conditioning model and the coverage of the training data and motion library do not yet support reliable interpretation of complex or compositional commands. The training library also does not cover sustained multi-contact behaviors such as crawling or climbing. Future work will investigate constrained or adaptive guidance, broader language–motion data, and training objectives for richer contact transitions.

D.4 Real-Robot Trial Sequences

Figures 10–15 show six temporally ordered frames from each recorded trial for all six settings reported in Table 3. The left rail reports the trial index and outcome under the same success rule.

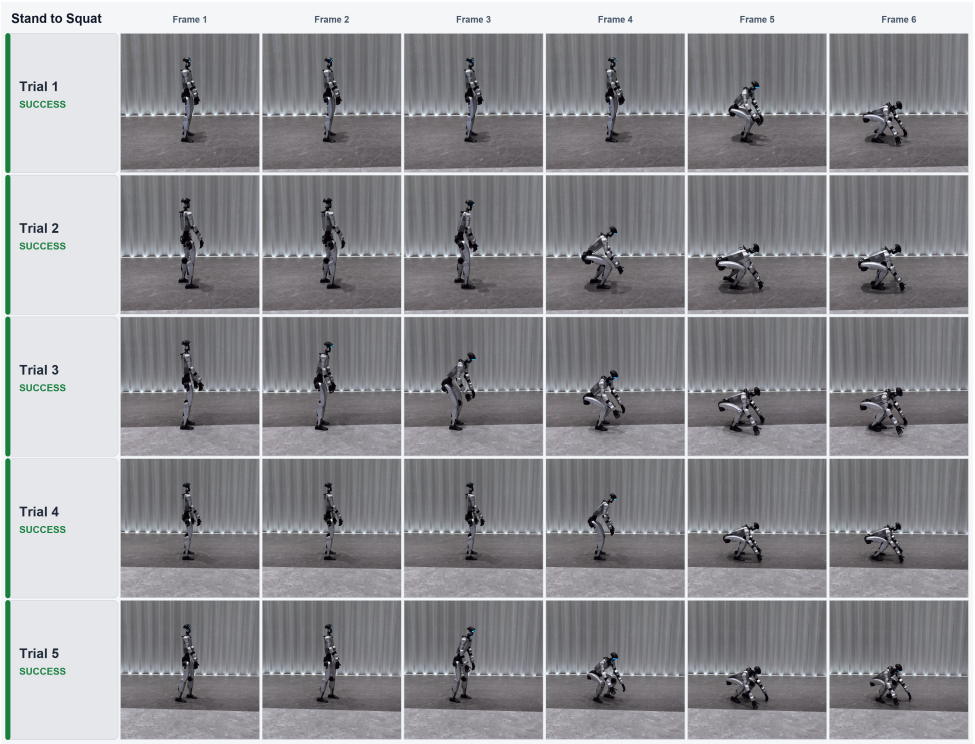

Figure 10: Stand-to-squat trials. Each row is one trial; columns progress from left to right.

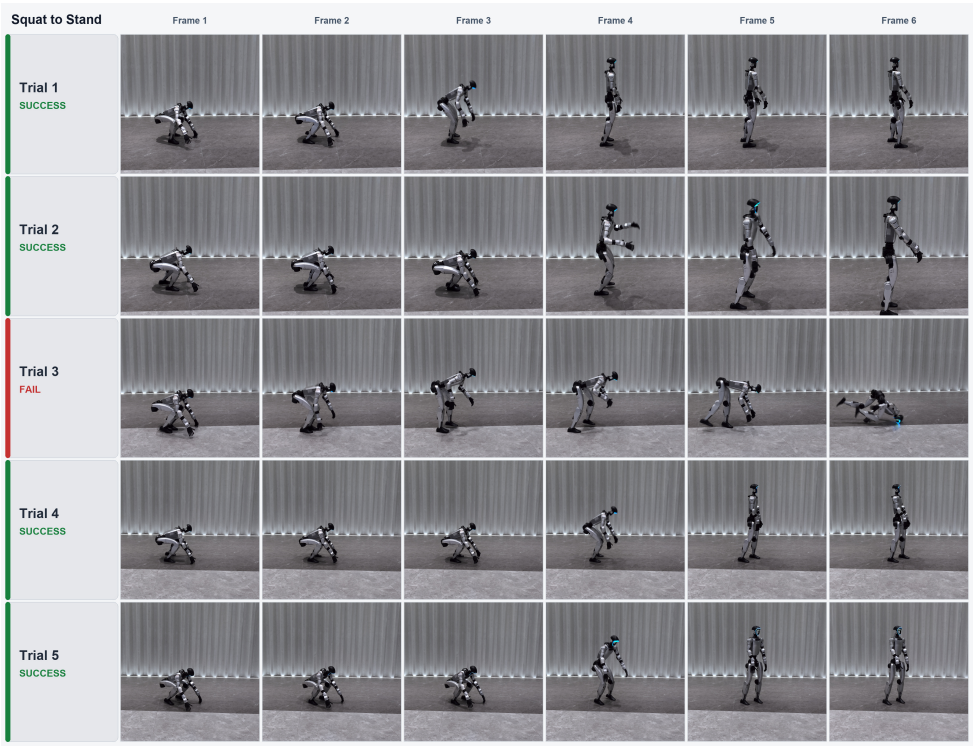

Figure 11: Squat-to-stand trials, including the failed third trial.

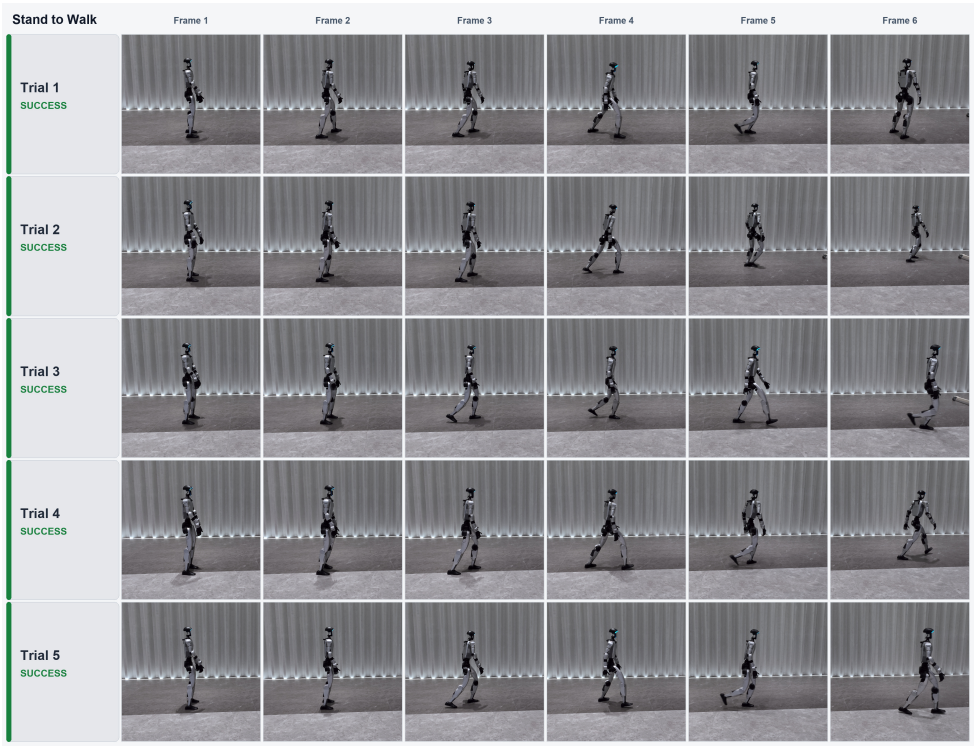

Figure 12: Stand-to-walk trials. The first column shows the stationary initial pose; the remaining columns show walking.

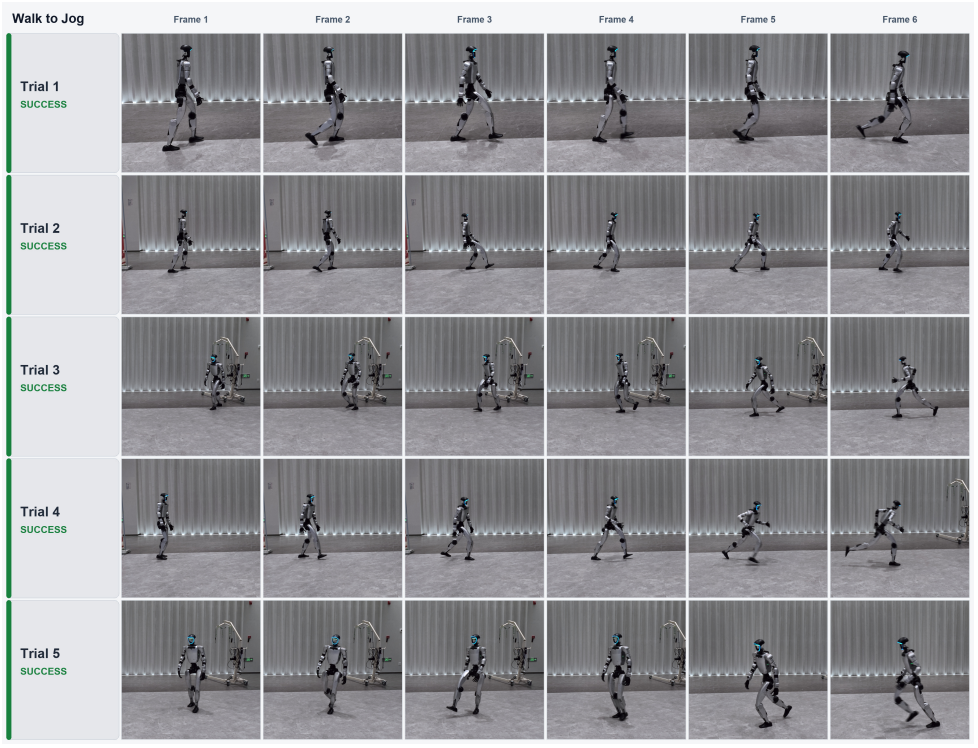

Figure 13: Walk-to-jog trials. Columns 1–3 show walking and the transition; columns 4–6 show the brief jogging phase.



Figure 14: Standing push-response trials, including the failed fifth trial. Successful rows end after stable recovery; the failed row retains the fall outcome.



Figure 15: Walking push-response trials, including the failed first trial. Successful rows show walking, perturbation, and a final frame after stable walking resumes; the failed row retains the fall outcome.