TAMGEN: TIME-AWARE MOTION GENERATION FOR GOAL-CONDITIONED HUMANOID CONTROL

Anonymous authors
Paper under double-blind review

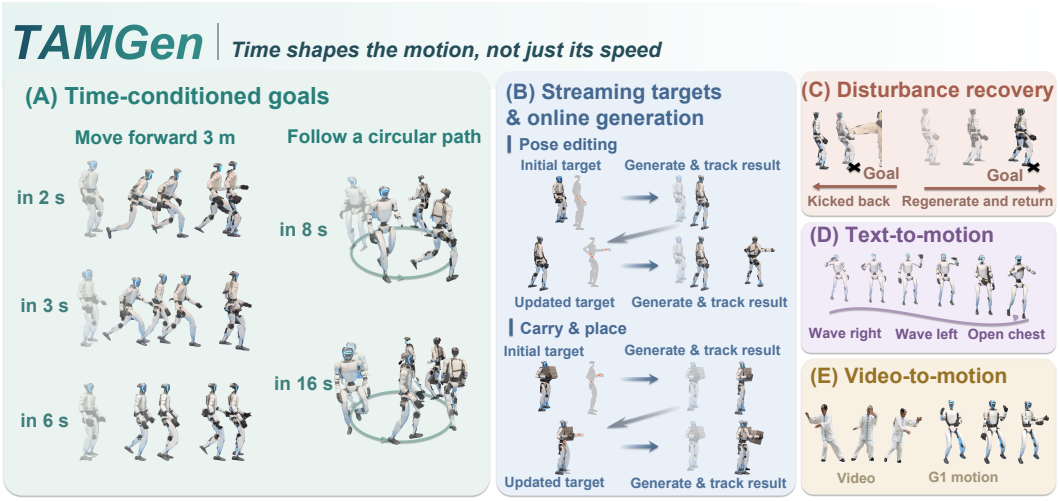

Figure 1: TAMGen capabilities in real-world humanoid control. **(A)** Forward and circular motions under different duration commands. **(B)** Real-time motion generation and tracking during pose editing and object carrying. Coral circles mark root and hand targets. **(C)** Motion regeneration for recovery from external disturbances and return to the goal. **(D–E)** Extensions to text- and video-conditioned motion generation.

ABSTRACT

Generative humanoid motion models can reach a spatial goal while still producing an inappropriate motion for the requested time. For the same spatial goal, simply speeding up or slowing down a motion to meet different time commands may compromise its physical quality, even if the endpoint remains accurate. This is because contact patterns and whole-body poses may also need to adapt to the time requirements. To address this challenge, we introduce **TAMGen**, a framework for time-aware motion generation and evaluation. Its generator uses flow matching with a root–pose architecture to capture the coupled effects of spatial goals and explicit time budgets, so that changing the time budget can change motion content rather than merely rescale playback speed. However, generated candidates may differ in target accuracy and physical quality. We therefore train a Motion Compatibility Evaluator (MCE) to assess complete candidates and select motions with better target accuracy and physical quality. When no duration is specified, MCE selects both a duration and a corresponding motion. Experiments show that TAMGen adjusts contact patterns and whole-body poses across different time commands. Compared with MotionBricks, TAMGen reduces root endpoint error by 81.6% on LAFAN and 84.9% on BoneSeed. Mean foot-sliding speed decreases by 48.6% on LAFAN and 31.6% on BoneSeed. Real-world experiments demonstrate real-time tracking of changing root and hand targets and recovery from external disturbances.

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.

1 INTRODUCTION

Humanoid whole-body control supports locomotion, manipulation, and tasks that combine both. These tasks require coordinated and reliable whole-body motion (Cao et al., 2026). Recent advances in motion tracking enable robots to execute diverse whole-body behaviors by following reference motions (Ji et al., 2024; He et al., 2025; Xie et al., 2025; Chen et al., 2025; Yin et al., 2025). Teleoperation systems allow users to provide such references in real time, guiding coordinated whole-body motions (Fu et al., 2024; He et al., 2024; Ze et al., 2025). However, accurate execution still requires a reference motion that suits the task. Recent systems therefore combine motion generation with tracking to produce task-dependent references (Luo et al., 2026; Tao et al., 2026; Bao et al., 2026).

Human motion generation provides flexible ways to specify such references. Methods can generate complete motions from partial keyframes, selected joint positions, or root trajectories (Cohan et al., 2024; Xie et al., 2024; Karunratanakul et al., 2023). These interfaces allow users to describe desired states without manually specifying a pose for every frame. Given the robot's current state and a target state, generators can produce multiple transition trajectories. Applying these capabilities to humanoid robots requires considering spatial goals and time budgets to generate coordinated contact patterns and whole-body poses.

Recent methods support motion generation directly on humanoid robot skeletons, bringing controllable motion generation closer to robot execution (Wang et al., 2026; Rempe et al., 2026; Zhao et al., 2026). Yet reaching the desired endpoint does not necessarily mean that the motion is appropriate for the requested time. For the same spatial goal, different time budgets may call for different stride lengths, contact patterns, and whole-body poses. Simply speeding up or slowing down a motion changes its timing, but does not adapt how the body moves and maintains ground support. Motion generation should therefore consider how spatial goals and time budgets together shape the motion.

To address this challenge, we introduce **TAMGen**, a framework comprising a motion generator and a learned Motion Compatibility Evaluator (MCE). The generator uses flow matching conditioned on spatial goals and explicit time budgets to capture their coupled effects on whole-body movement. MCE evaluates the generated motions for target accuracy and physical quality to select a final reference. To support this evaluation, we introduce time-indexed queries describing relations among body parts and between the body and the ground. During training, we vary goal matching and physical quality separately so that MCE responds to both rather than one alone.

Our contributions are:

1. **A time-aware generator that adapts motion content to the time budget.** Built on flow matching and a coupled root–pose architecture, it explicitly conditions on duration, physical time, and normalized progress, allowing changes in contact patterns and whole-body poses rather than merely rescaling playback speed.
2. **A motion compatibility evaluator with controlled pairwise supervision for goal matching and physical quality.** We construct training pairs that vary goal matching while holding the motion fixed, or physical quality while holding the query fixed. This supervision trains MCE to assess both requirements with a single score. MCE selects a motion at a specified duration, or selects the motion and duration jointly when timing is unspecified.
3. **A closed-loop pipeline for continuous tracking of streaming targets on a physical humanoid.** Built on the Generator and MCE, the pipeline accepts sparse or full targets, completing sparse targets into whole-body goals and using full targets directly. It regenerates motion from the current state to follow updated targets and return toward the goal after external disturbances.

2 RELATED WORK

2.1 GOAL-CONDITIONED HUMANOID CONTROL

Goal-conditioned humanoid control can use a policy that maps goals directly to control actions, or a motion generator followed by a tracker. BFM-Zero (Li et al., 2026) uses unsupervised reinforcement learning to learn a policy prompted by target poses, motions, or rewards, mapping observation histories and task representations directly to joint commands. Optimizing long-term goal-related

rewards does not by itself impose arrival at a specified time. Representing rewards as linear combinations of fixed features can limit goal-reaching precision (Cetin et al., 2024). BeyondMimic (Liao et al., 2025) instead models state–latent trajectories with guided diffusion and decodes them into actions using current observations. Another line of work connects motion generation to execution: general-purpose whole-body trackers execute reference trajectories (Chen et al., 2025; Luo et al., 2026), while Heracles (Tao et al., 2026) and PhyGile (Bao et al., 2026) generate or adapt references using robot states or physical information. Compared with policies that map goals directly to control actions, generation-and-tracking methods provide an explicit motion trajectory. This allows intermediate and final targets to be incorporated during generation and the complete motion to be evaluated before execution. TAMGen follows this generation-and-tracking approach, studying motion generation and selection under spatial goals and time requirements.

2.2 Controllable Motion Generation

Earlier work primarily focuses on controllable human motion generation. CondMDI (Cohan et al., 2024) uses full or partial keyframes, OmniControl (Xie et al., 2024) supports joint-level targets, and Guided Motion Diffusion (Karunratanakul et al., 2023) provides trajectory guidance. These interfaces specify selected aspects of a motion while leaving the remaining body movement to the generator. More recent frameworks also support humanoid robot skeletons, with capabilities such as duration prediction and online generation. MotionBricks (Wang et al., 2026) adopts a modular pipeline for timing prediction, root planning, and discrete pose-token generation. Kimodo (Rempe et al., 2026) uses a controllable motion diffusion model, while ARDY (Zhao et al., 2026) adopts autoregressive diffusion for streaming generation. For robot execution, such control also needs to account for how the requested timing affects body coordination and ground support.

3 Method

3.1 Framework Overview

Given an observed motion history h_0 and a spatial goal, our task is to generate a whole-body trajectory X that satisfies the goal while maintaining good physical quality throughout the transition. The task may specify a physical duration τ or leave the completion time open. Each trajectory contains root and articulated body motion over that duration. We denote the target state used for generation by x_{goal} and the requested body relations used for evaluation by $\mathcal{Q}$.

TAMGen addresses this task through generation and selection (Figure 2). The Generator produces multiple candidates under a spatial target and an explicit time budget. MCE evaluates their goal matching and physical quality, selecting a reference for a whole-body Tracker. When time is unspecified, selection spans candidates generated under different durations. The two models are trained separately and cooperate at inference. The following sections describe the time-conditioned Generator, relational supervision, and MCE-based candidate selection.

3.2 Spatiotemporally Conditioned Flow Generator

Given the observed motion history h_0, target state x_{goal}, and physical duration τ, the Generator produces a whole-body trajectory $X_k = G_\phi(c_G, z_k)$, where $c_G = (h_0, x_{\text{goal}}, \tau)$ and $z_k \sim \mathcal{N}(0, I)$. Different noise samples yield alternative motions under the same conditions. The Generator first samples a root trajectory, then generates articulated motion conditioned on it. Each generated trajectory consists of states $x_n = (p_n, o_n, q_n)$, containing the root position $p_n \in \mathbb{R}^3$, root orientation $o_n \in \mathbb{R}^6$, and joint positions $q_n \in \mathbb{R}^{29}$.

Root-to-pose generation. The Root Flow model represents the root trajectory $R = (R_0, \ldots, R_N)$ in normalized coordinates as a boundary bridge plus a learned residual:

$$R = b(R_0, R_N; N) + B_N C, \qquad B_N[0] = B_N[N] = 0. \tag{1}$$

Here, $R_n = (p_n, o_n)$ contains the root position and orientation at step n, and N is the number of time intervals. The bridge b provides an initial path between the boundary states. A conditional-flow Transformer predicts coefficients C that adjust the path between the endpoints through 24 DCT-derived basis functions in B_N. These adjustments are zero at both endpoints, so the boundary

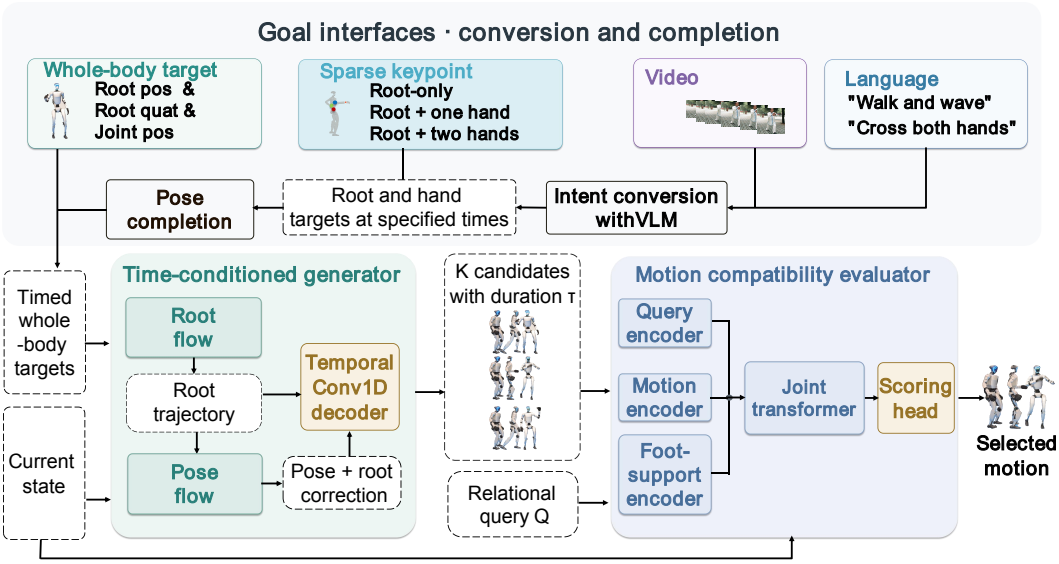

Figure 2: **TAMGen framework.** Top: language, video, and sparse targets are converted or completed into whole-body goals; full-state targets bypass completion. Bottom: the time-conditioned Generator produces motion candidates, and the Motion Compatibility Evaluator selects a reference based on target accuracy and physical quality. Dashed boxes denote variables; solid white boxes denote conversion and completion modules.

states remain unchanged. Restricting the basis to low-frequency functions limits high-frequency fluctuations while allowing different paths between the same endpoints.

The bidirectional Pose Flow Transformer takes the sampled root trajectory and its motion features, together with the initial and terminal poses. Its time inputs include duration τ, physical time t_n, and normalized progress t_n/τ. Progress indicates where a frame lies in the motion, while physical time distinguishes the same progress across different durations. Together, these inputs allow whole-body poses to adapt to the root motion and requested duration.

Flow matching. Root Flow and Pose Flow are trained using conditional flow matching (Lipman et al., 2023). For either model, let y denote the normalized training target and $\epsilon \sim \mathcal{N}(0, I)$ denote standard Gaussian noise. We interpolate between noise and data at flow time $s \in [0, 1]$ and train a velocity field v_ϕ:

$$y_s = (1-s)\epsilon + sy, \qquad \mathcal{L}_{\mathrm{FM}} = \mathbb{E}_{y,\epsilon,s}\left[\|v_\phi(y_s, s; c) - (y-\epsilon)\|_2^2\right], \tag{2}$$

The target y contains root coefficients for Root Flow and joint coordinates for Pose Flow. The path starts at noise ($s = 0$) and ends at data ($s = 1$). Each model predicts the target velocity $y - \epsilon$ from y_s, s, and its stage-specific conditions c. This is the flow velocity, not the robot's physical velocity; flow time s differs from physical motion time t_n.

Coupled Decoding. During Pose Flow training, the velocity prediction gives a one-step clean-state estimate $\widehat{y}$, which the shared temporal decoder D_ψ maps to joint positions and a bounded root correction:

$$\widehat{y} = y_s + (1-s)v_\phi(y_s, s; c), \qquad \widehat{X} = D_\psi(\widehat{y}, R), \qquad \mathcal{L}_G = \mathcal{L}_{\mathrm{FM}} + \lambda_{\mathrm{dec}}\mathcal{L}_{\mathrm{dec}}(\widehat{X}). \tag{3}$$

The shared decoder jointly adjusts pose and root. Its loss $\mathcal{L}_{\mathrm{dec}}$, weighted by λ_{dec}, combines reconstruction, FK, contact, support-related sliding, penetration, and temporal smoothness. These losses are computed on the decoded motion, not the noisy flow state. Pose Flow is trained on both ground-truth root trajectories and trajectories predicted by the Root Flow model. This prepares it to use predicted root trajectories at inference. See Appendices A.2 and C.1 for architecture/training details and the basis-count ablation, respectively.

3.3 Relational Queries and Joint Supervision

Relational Query Representation. To provide a common, task-oriented interface for MCE and extended tasks, we express goals through body relations rather than whole-body joint configurations. Describing where the hands and root should be is often more intuitive than specifying all 29 joint positions. We represent these goals as a relational query containing time-indexed relational constraints:

$$\mathcal{Q} = \{(e_i^{\text{src}}, e_i^{\text{dst}}, \rho_i, v_i, t_i)\}_{i=1}^{M}, \qquad t_i \in [0, \tau], \quad M \leq 18. \tag{4}$$

Here e_i^{src} and e_i^{dst} identify the reference and target entities; translation and orientation express the target relative to the reference frame. The requested physical time is t_i. Entities include the root, both hands, both feet, and the ground. The relation type ρ_i is translation, orientation, surface gap, or contact, with target value v_i. We use 18 entity–relation combinations. M is the number of constraints in a query, limited to 18 in our implementation. Each constraint applies at its specified time t_i, rather than throughout an interval. A query can combine constraints on different body parts at intermediate or terminal times. A candidate should satisfy every constraint at its specified time. We compute the corresponding relations from each candidate at the requested times and compare them with the query targets.

For sparse targets, our pose-prior-based completion module fills in unspecified body coordinates to form a whole-body goal before generation.

Training Data Construction. Training data combine generated trajectories, original motion recordings, and manually designed motions. Each training sample pairs a motion with a query. Motions with and without physical errors are paired with matching and mismatching queries, giving four combinations of physical quality and goal matching.

Relational negatives are constructed by keeping the motion unchanged and altering the requested relation or its timing. Physical negatives use motions with errors such as sliding, penetration, sustained hovering, position discontinuities, and root–limb motion mismatch. These errors may already exist in the motions or be introduced through manual edits.

Ranking Supervision. Let $s_\theta(X, \mathcal{Q}, \tau)$ denote the scalar score assigned by MCE, with network parameters θ. Lower scores indicate better motion compatibility. For the same motion, a matching query should receive a lower score than a mismatching query. Under the same query, a motion with good physical quality should receive a lower score than one with physical errors. See Appendices B.1 and B.3 for the relation registry and training-data construction, respectively.

3.4 Motion Compatibility Evaluator (MCE)

MCE Architecture. MCE uses three groups of tokens. Motion tokens H_X encode the robot states $x_n = (p_n, o_n, q_n)$, candidate relation values, and their first- and second-order temporal differences. These relations are computed from the trajectory using forward kinematics. Foot-support tokens H_{support} encode each foot's ground-relative position and velocity, surface gap, and contact state. These features are derived from the candidate trajectory rather than supplied as additional task commands. Query tokens H_Q encode each task constraint $(e_i^{\text{src}}, e_i^{\text{dst}}, \rho_i, v_i, t_i)$, where v_i is the requested value, together with the corresponding actual value extracted from the candidate at t_i. A Transformer processes these three token groups together with a readout token $h_s(\tau)$. The updated readout token is mapped by a multilayer perceptron (MLP) to the scalar score $s_\theta(X, \mathcal{Q}, \tau)$.

Training objectives. We use a ranking loss and a binary classification loss. The ranking loss enforces these comparisons by encouraging the positive sample's score s_p to be lower than the negative sample's score s_n by a margin m. The classification loss predicts whether an individual sample satisfies both the goal and physical requirements. We assign label $u = 1$ when both requirements are satisfied, and $u = 0$ otherwise. The score s is converted into a probability $p = \text{sigmoid}(-s)$, which is compared with the label:

$$\ell_{\text{rank}} = \log(1 + \exp(m + s_p - s_n)), \qquad \ell_{\text{cls}} = -u \log p - (1 - u) \log(1 - p). \tag{5}$$

Candidate Selection. When a duration is specified, MCE selects the lowest-scoring motion generated for that duration. Otherwise, candidates are generated for several durations with the same spatial goal. MCE compares their scores to select a duration and its corresponding motion:

$$(b^*, k^*) = \arg\min_{b,k} s_\theta(X_{bk}, \mathcal{Q}_b, \tau_b). \tag{6}$$

Here, τ_b is the b-th candidate duration, X_{bk} is its k-th generated motion, and $\mathcal{Q}_b$ specifies the spatial goal at time τ_b.

4 EXPERIMENTS

Our experiments examine whether TAMGen generates motions that meet spatial goals and time requirements while maintaining physical quality. We ask three questions:

Q1. Can MCE assess both goal matching and physical quality?
Q2. Can TAMGen meet spatial and temporal requirements while maintaining physical quality?
Q3. Can TAMGen support real-time motion generation and execution on a physical humanoid?

4.1 Q1: CAN MCE ASSESS BOTH GOAL MATCHING AND PHYSICAL QUALITY?

To answer Q1, we test whether MCE detects physical errors across different query conditions and identifies goal-matching motions across different physical conditions.

Physical Error Detection. We compare MCE scores for physically qualified motions and motions with physical errors under the same query condition. The errors include ground penetration, sustained hovering, position discontinuities, root–limb temporal mismatch, and sliding, each tested at small, medium, and large levels. Query conditions include no query, a matching query $\mathcal{Q}^+$, and a mismatching query $\mathcal{Q}^-$. Across 45 combinations of error type, error level, and query condition, we evaluate 32 motion pairs per combination, yielding 1,440 pairwise comparisons per model. The evaluation is repeated with three random seeds. A ranking is correct when the qualified motion receives a lower score than the motion with errors. At the small level, ranking accuracy ranges from 92.5–100% without queries and 92.6–99.1% with matching or mismatching queries. All medium and large settings reach 100% mean accuracy. These results show that MCE identifies the tested physical errors across different query conditions. Detailed results are provided in Appendix C.3.

Query Matching Evaluation. We compare the scores of the same motion under matching and mismatching queries, using both physically qualified motions and motions with physical errors. The test contains 37 motion pairs and three query types: root translation, root yaw, and their combination. This gives 111 comparisons for each physical condition. A ranking is correct when the matching query receives a lower score. Accuracy reaches 99.10% (110/111) for qualified motions and 97.30% (108/111) for motions with physical errors. These results show that MCE distinguishes matching from mismatching queries across different physical conditions.

4.2 Q2: CAN TAMGEN MEET SPATIAL AND TEMPORAL REQUIREMENTS WHILE MAINTAINING PHYSICAL QUALITY?

To answer Q2, we compare the target accuracy and physical quality of TAMGen and baseline methods under the same whole-body targets, and examine how TAMGen adapts motion to different time requirements. To support additional tasks, we use pose completion to enable generation from sparse body targets and further evaluate target accuracy in this setting.

Full-State Targets. We evaluate on G1-retargeted motions derived from LAFAN1 (Harvey et al., 2020) and BONES-SEED (Bones Studio, 2026), denoted LAFAN and BoneSeed in the tables.[1] We compare target accuracy and physical quality using 36 conditions: 12 from LAFAN and 24 from BoneSeed. All motions are evaluated at 50 Hz, with generation repeated using three sampling seeds. Tables report the mean and standard deviation across these sampling seeds. Results are first averaged over test conditions from the same motion sequence. These averages are then combined within each dataset, with equal weight assigned to each sequence.

Three pretrained motion generators serve as external baselines. MB-greedy uses MotionBricks' official model (Wang et al., 2026) with greedy pose-token decoding and no reranking. Kimodo-G1 (Rempe et al., 2026) and ARDY-G1 (Zhao et al., 2026) use official pretrained G1 models. All

[1]For LAFAN1, we use the G1 motions from the LAFAN1 Retargeting Dataset on Hugging Face. For BONES-SEED, we use the official Unitree G1 CSV release.

Table 1: **Target accuracy and physical quality on LAFAN and BoneSeed.** Values are mean ± SD over three sampling seeds; bold and underline mark the best and second-best means for each metric within each dataset.

	Endpoint errors			Physical quality			
Method	Root (cm) ↓	Body (cm) ↓	Angle (°) ↓	Slide (m/s) ↓	Pen. (cm) ↓	Short contact (s^{-1}) ↓	Jerk (m/s^3) ↓
LAFAN							
MB-greedy	1.038 ± 0.004	<u>1.828</u> ± 0.051	<u>1.655</u> ± 0.001	0.222 ± 0.001	**0.360** ± 0.007	0.230 ± 0.016	**739.0** ± 0.5
Kimodo-G1	2.253 ± 0.102	2.409 ± 0.089	4.197 ± 0.044	0.160 ± 0.006	0.657 ± 0.050	0.369 ± 0.077	<u>797.4</u> ± 11.2
ARDY-G1	1.933 ± 0.114	1.991 ± 0.114	1.716 ± 0.043	<u>0.138</u> ± 0.002	0.573 ± 0.008	0.318 ± 0.070	855.8 ± 35.7
Ours Generator	<u>0.225</u> ± 0.001	1.850 ± 0.012	1.723 ± 0.021	0.1823 ± 0.0004	0.667 ± 0.047	<u>0.171</u> ± 0.010	1017.6 ± 30.1
Ours (+MCE)	**0.191** ± 0.024	**1.404** ± 0.002	**1.268** ± 0.012	**0.114** ± 0.007	<u>0.568</u> ± 0.125	**0.165** ± 0.007	814.5 ± 43.8
BoneSeed							
MB-greedy	1.545 ± 0.011	<u>1.927</u> ± 0.017	2.297 ± 0.023	0.421 ± 0.016	**1.072** ± 0.009	<u>1.146</u> ± 0.075	**1205.5** ± 0.5
Kimodo-G1	4.043 ± 0.113	4.168 ± 0.115	3.140 ± 0.052	<u>0.292</u> ± 0.021	1.259 ± 0.011	1.360 ± 0.008	1750.3 ± 33.9
ARDY-G1	1.635 ± 0.044	2.527 ± 0.036	2.305 ± 0.053	0.418 ± 0.015	1.500 ± 0.024	1.382 ± 0.079	1593.4 ± 22.9
Ours Generator	<u>0.268</u> ± 0.002	1.957 ± 0.024	<u>1.812</u> ± 0.007	0.391 ± 0.013	1.546 ± 0.009	1.258 ± 0.039	1633.0 ± 95.3
Ours (+MCE)	**0.234** ± 0.025	**1.714** ± 0.043	**1.504** ± 0.009	**0.288** ± 0.011	<u>1.163</u> ± 0.147	**0.795** ± 0.090	<u>1355.9</u> ± 97.2

methods receive the same observed start and terminal states and requested duration, without text or intermediate ground-truth frames.

MCE-guided selection improves all seven metrics on both datasets (Table 1), including sliding reductions of 37.5% on LAFAN and 26.3% on BoneSeed. These gains isolate the benefit of selection from additional sampling or Generator updates. Compared with the external baselines, TAMGen achieves the lowest endpoint position and joint-angle errors and short-contact frequency on both datasets, while MotionBricks retains lower penetration and jerk. The results demonstrate improved target accuracy alongside competitive physical quality.

Duration Response. To examine how generated motions respond to different duration commands, the test uses the same start state and whole-body target 3 m away, with requested durations of 3 s and 5 s. Both the 3 s and 5 s motions are generated in multiple segments. We compare foot-contact patterns to measure segment similarity within and across the two durations. Time within each segment is normalized to $[0, 1]$. Let i and j index generation segments, and $A_f^{(i)}$ denote the airborne intervals of foot $f \in \{L, R\}$ in segment i. We measure airborne timing similarity using intersection over union (IoU) (Jaccard, 1901), averaged over the two feet:

$$\mathrm{IoU}_{\mathrm{air}}(i, j) = \frac{1}{2} \sum_{f \in \{L,R\}} \frac{\left| A_f^{(i)} \cap A_f^{(j)} \right|}{\left| A_f^{(i)} \cup A_f^{(j)} \right|}, \tag{7}$$

Here, $|\cdot|$ denotes interval duration; higher IoU indicates more similar airborne timing. There are four within-duration and six cross-duration segment pairs. TAMGen has lower airborne IoU than all three baselines in both comparisons (Figure 3a), indicating less repetition of segment-level contact patterns in these examples. A more detailed diversity analysis is provided in Appendix C.2.

To examine whether adapting contact patterns can improve physical quality while meeting spatial and temporal requirements, we further shorten the requested duration to 1.5 s while keeping the same 3 m displacement (Figure 3b). MotionBricks retains a stepping pattern visually similar to those in Figure 3a, with one step per foot. To cover the requested displacement, it generates larger strides and pronounced airborne motion, increasing execution risk while deviating from the requested root height. In contrast, the TAMGen Generator produces a running motion to meet the requested displacement and duration. This example illustrates how adapting motion patterns to the time budget can maintain physical quality while meeting the spatial goal.

Sparse Targets. Humanoid control tasks often require specifying only where the hands and root should be, rather than a complete whole-body pose. We use our pose-prior-based completion module to convert these partial targets into whole-body goals for generation. To test how well TAMGen follows partial targets through this pipeline, six initial states are each tested with root-plus-one-hand and root-plus-two-hands targets, giving 12 tasks. Root position and orientation, together with hand positions, are specified at an intermediate time of 1.2 s and a final time of 2.4 s. Generation is repeated using three sampling seeds. TAMGen, MotionBricks, Kimodo, and ARDY share the same completion module for a controlled downstream comparison. We additionally evaluate OmniControl

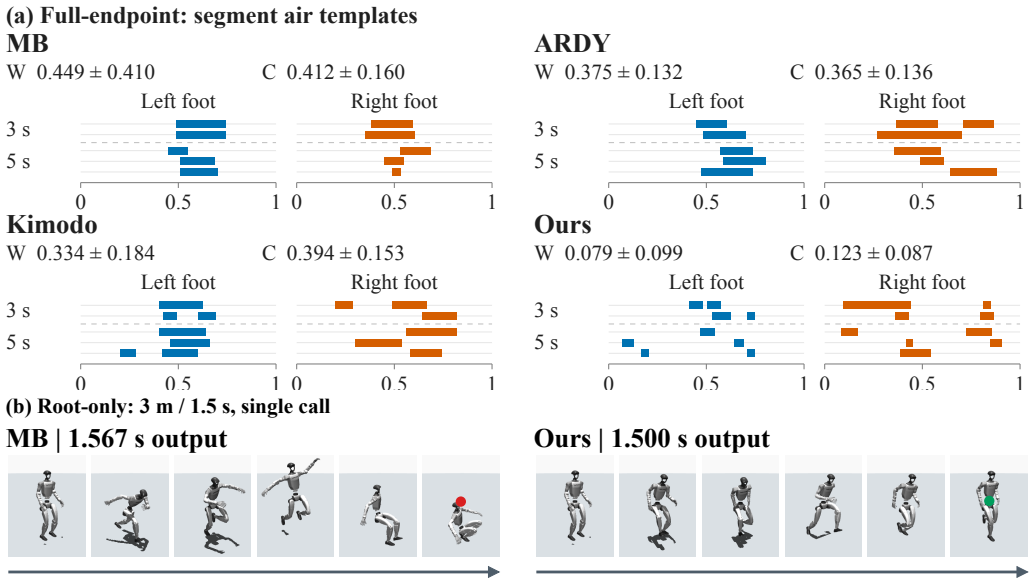

Figure 3: **Duration response: foot-air timing and short-horizon poses. (a)** Blue/orange: left/right foot-air intervals on normalized phase; rows follow segment order within each duration. W/C: within-/cross-duration IoU (mean ± SD). **(b)** Short-horizon poses; red indicates failure to reach the Root goal (MB), while green indicates success (Ours).

(Xie et al., 2024) and CondMDI (Cohan et al., 2024). Human motions generated by these methods are retargeted to G1 using GMR (Araújo et al., 2025). These methods receive the same world-position targets and target times, but not the full root orientation constraints, and are reported in separate position-only groups.

root-position, root-orientation, and hand forward-kinematics (FK) errors are measured at both times (Table 2). TAMGen achieves the lowest root-position and hand-position errors among the G1-interface methods through the same pipeline in both single-hand and two-hand tasks: intermediate Root errors are 0.504/0.672 cm for the one-/two-hand tasks, respectively, and terminal hand errors are 2.237/2.453 cm.

4.3 Q3: Can TAMGen support real-time motion generation and execution on a physical humanoid?

The hardware experiments examine time-conditioned execution, online target updates, and disturbance recovery, followed by text and video extensions. TAMGen is deployed on a Unitree G1 humanoid, with generation and selection running on a single NVIDIA RTX 4090 and GRIT (mrzuang, 2026) tracking the whole-body references. Distances are measured using aligned onboard odometry.

Time-conditioned execution. We test whether motions generated for different time requirements can be executed on the robot. Figure 1A shows forward motion over 3 m and circular motion under different duration commands. The executed motions exhibit different stepping patterns and whole-body coordination across durations.

Online target updates and disturbance recovery. We measure inference latency to assess whether TAMGen supports online motion regeneration. On a single RTX 4090, motion generation takes approximately 143–145 ms. Even a complete search over 124 candidates across 31 durations takes 0.993 s on average, including candidate generation and selection, while TAMGen supports generating motions up to 3 s in a single call, equivalent to 150 tracking steps at 50 Hz. TAMGen can therefore generate updated trajectories while the tracker executes the current motion, supporting online target changes. Figure 1B shows successive Root and hand target updates and their execution in pose editing and carrying tasks. When an external disturbance moves the robot away from an unchanged goal, TAMGen regenerates a motion from the current state to return toward that goal

Table 2: **Sparse-target errors.** Values are mean ± SD over three sampling seeds. Bold and underline mark the best and second-best means among the four G1-interface methods for each metric and task; position-only baselines are reported separately.

	Intermediate target errors			Final target errors		
Method	Root pos. (cm) ↓	Root ori. (°) ↓	Hand FK (cm) ↓	Root pos. (cm) ↓	Root ori. (°) ↓	Hand FK (cm) ↓
Root + one hand						
Ours	**0.504** ± 0.100	1.672 ± 0.447	**1.827** ± 0.554	**0.889** ± 0.464	1.758 ± 0.428	**2.237** ± 0.636
MotionBricks	1.278 ± 0.367	**1.534** ± 0.429	2.147 ± 0.497	1.686 ± 0.539	**1.711** ± 0.561	2.375 ± 0.630
Kimodo	1.683 ± 0.456	2.164 ± 0.437	2.867 ± 0.305	1.983 ± 0.660	2.927 ± 0.402	2.973 ± 0.618
ARDY	2.207 ± 0.620	3.067 ± 0.504	2.929 ± 0.335	2.080 ± 0.679	2.976 ± 0.595	3.242 ± 0.509
Root + one hand: position-only						
OmniControl+GMR	3.938 ± 1.248	—	9.695 ± 1.506	4.958 ± 1.340	—	7.939 ± 1.329
CondMDI+GMR†	14.497 ± 1.727	—	21.438 ± 2.373	17.773 ± 3.939	—	21.844 ± 7.398
Root + two hands						
Ours	**0.672** ± 0.359	1.785 ± 0.512	**1.968** ± 0.453	**0.980** ± 0.345	1.646 ± 0.570	**2.453** ± 0.554
MotionBricks	1.098 ± 0.505	**1.581** ± 0.434	2.746 ± 0.672	1.710 ± 0.411	**1.572** ± 0.396	2.698 ± 0.435
Kimodo	2.060 ± 0.295	4.370 ± 0.525	3.591 ± 0.197	2.457 ± 0.274	3.152 ± 1.004	3.946 ± 0.585
ARDY	2.321 ± 0.576	3.426 ± 0.529	3.004 ± 0.381	1.824 ± 0.500	3.085 ± 0.392	3.564 ± 0.373
Root + two hands: position-only						
OmniControl+GMR	5.657 ± 1.586	—	13.683 ± 0.201	10.584 ± 1.969	—	11.985 ± 1.727
CondMDI+GMR†	15.946 ± 4.850	—	24.171 ± 2.044	16.641 ± 1.312	—	23.687 ± 4.254

—: full root-orientation targets are unsupported. †: CondMDI uses a world-to-local coordinate adapter.

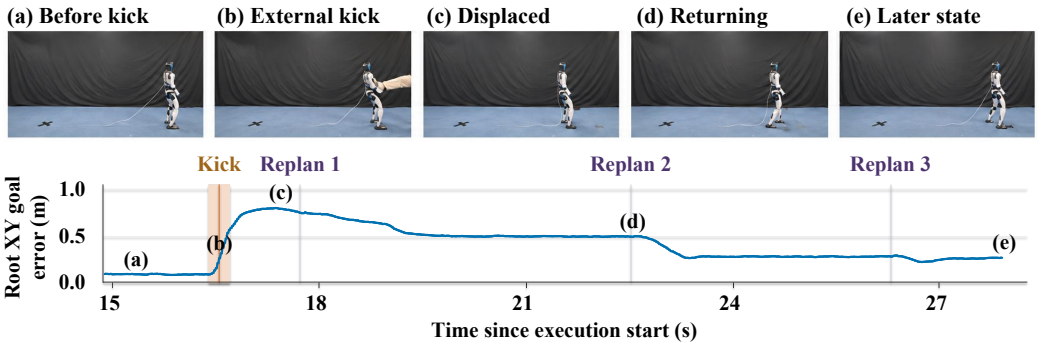

Figure 4: **Hardware replanning after an external kick.** Top: execution snapshots. Bottom: measured Root XY distance to the original goal. The orange band marks the approximate kick time; Replan labels indicate new-reference activation.

(Figure 1C). In the 3 m walking trial (Figure 4), a kick increases horizontal Root-to-goal distance from approximately 0.08 m to 0.80 m. Successive reference updates redirect the robot toward the original goal, rather than merely maintaining balance.

Task Extensions. The query interface allows TAMGen to extend to other tasks. By using a VLM to convert text and video inputs into time-indexed relational queries, TAMGen supports zero-shot text- and video-conditioned motion generation without modality-specific training (Figure 1D–E).

5 Discussion

TAMGen uses a flow-matching generator to capture how spatial goals and time budgets shape whole-body motion. MCE selects generated motions for target accuracy and physical quality. Together, they produce motions that adapt to different time requirements while maintaining target accuracy and physical quality. However, several limitations remain. Further evaluation is needed for unseen body–ground relations, new combinations of goals, and longer motions. MCE does not yet directly account for the tracker's execution limits. Incorporating execution feedback and safety checks is an important direction for real-world deployment.

AI USE STATEMENT

We used generative AI tools to assist with literature retrieval, research ideation and experimental execution, and manuscript drafting and language refinement. These tools also assisted with code implementation, debugging, result analysis, and figure preparation. The authors take responsibility for the final manuscript, experimental results, and conclusions.

REFERENCES

João Pedro Araújo, Yanjie Ze, Pei Xu, Jiajun Wu, and C. Karen Liu. Retargeting matters: General motion retargeting for humanoid motion tracking. *arXiv preprint arXiv:2510.02252*, 2025. URL https://arxiv.org/abs/2510.02252.

Jiacheng Bao, Haoran Yang, Yucheng Xin, Junhong Liu, Yuecheng Xu, Han Liang, Pengfei Han, Xiaoguang Ma, Dong Wang, and Bin Zhao. PhyGile: Physics-prefix guided motion generation for agile general humanoid motion tracking. *arXiv preprint arXiv:2603.19305*, 2026. URL https://arxiv.org/abs/2603.19305.

Bones Studio. BONES-SEED: Skeletal everyday embodiment dataset. Dataset, 2026. URL https://huggingface.co/datasets/bones-studio/seed. Accessed September 25, 2026.

Zhanxiang Cao, Liyun Yan, Yang Zhang, Sirui Chen, Jianming Ma, Tianyue Zhan, Shengcheng Fu, Yufei Jia, Cewu Lu, and Yue Gao. HiWET: Hierarchical world-frame end-effector tracking for long-horizon humanoid loco-manipulation. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026. doi: 10.15607/RSS.2026.XXII.030. URL https://www.roboticsproceedings.org/rss22/p030.html.

Edoardo Cetin, Ahmed Touati, and Yann Ollivier. Finer behavioral foundation models via auto-regressive features and advantage weighting. *arXiv preprint arXiv:2412.04368*, 2024. URL https://arxiv.org/abs/2412.04368.

Zixuan Chen, Mazeyu Ji, Xuxin Cheng, Xuanbin Peng, Xue Bin Peng, and Xiaolong Wang. GMT: General motion tracking for humanoid whole-body control. *arXiv preprint arXiv:2506.14770*, 2025. URL https://arxiv.org/abs/2506.14770.

Setareh Cohan, Guy Tevet, Daniele Reda, Xue Bin Peng, and Michiel van de Panne. Flexible motion in-betweening with diffusion models. In *ACM SIGGRAPH 2024 Conference Papers*, 2024. URL https://setarehc.github.io/CondMDI/.

Zipeng Fu, Qingqing Zhao, Qi Wu, Gordon Wetzstein, and Chelsea Finn. HumanPlus: Humanoid shadowing and imitation from humans. *arXiv preprint arXiv:2406.10454*, 2024. URL https://arxiv.org/abs/2406.10454.

Félix G. Harvey, Mike Yurick, Derek Nowrouzezahrai, and Christopher Pal. Robust motion in-betweening. *ACM Transactions on Graphics*, 39(4), 2020. URL https://github.com/ubisoft/ubisoft-laforge-animation-dataset.

Tairan He, Zhengyi Luo, Xialin He, Wenli Xiao, Chong Zhang, Weinan Zhang, Kris Kitani, Changliu Liu, and Guanya Shi. OmniH2O: Universal and dexterous human-to-humanoid whole-body teleoperation and learning. *arXiv preprint arXiv:2406.08858*, 2024. URL https://arxiv.org/abs/2406.08858.

Tairan He, Jiawei Gao, Wenli Xiao, Yuanhang Zhang, Zi Wang, Jiashun Wang, Zhengyi Luo, Guanqi He, Nikhil Sobanbab, Chaoyi Pan, Zeji Yi, Guannan Qu, Kris Kitani, Jessica Hodgins, Linxi Fan, Yuke Zhu, Changliu Liu, and Guanya Shi. ASAP: Aligning simulation and real-world physics for learning agile humanoid whole-body skills. In *Robotics: Science and Systems*, 2025. URL https://arxiv.org/abs/2502.01143.

Paul Jaccard. Étude comparative de la distribution florale dans une portion des alpes et du jura. *Bulletin de la Société Vaudoise des Sciences Naturelles*, 37:547–579, 1901. doi: 10.5169/seals-266450. URL https://doi.org/10.5169/seals-266450.

Mazeyu Ji, Xuanbin Peng, Fangchen Liu, Jialong Li, Ge Yang, Xuxin Cheng, and Xiaolong Wang. ExBody2: Advanced expressive humanoid whole-body control. *arXiv preprint arXiv:2412.13196*, 2024. URL https://arxiv.org/abs/2412.13196.

Korrawe Karunratanakul, Konpat Preechakul, Supasorn Suwajanakorn, and Siyu Tang. Guided motion diffusion for controllable human motion synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2151–2162, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Karunratanakul_Guided_Motion_Diffusion_for_Controllable_Human_Motion_Synthesis_ICCV_2023_paper.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 *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=jkhl2oI0g5.

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. *arXiv preprint arXiv:2508.08241*, 2025. URL https://arxiv.org/abs/2508.08241.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2023. URL https://arxiv.org/abs/2210.02747.

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):eaed4592, 2026. doi: 10.1126/scirobotics.aed4592. URL https://arxiv.org/abs/2511.07820.

mrzuang. GRIT_teleop_deploy. GitHub software repository, 2026. URL https://github.com/mrzuang/GRIT_teleop_deploy. Accessed September 25, 2026.

Davis Rempe, Mathis Petrovich, Ye Yuan, Haotian Zhang, Xue Bin Peng, Yifeng Jiang, Tingwu Wang, Umar Iqbal, David Minor, Michael de Ruyter, Jiefeng Li, Chen Tessler, Edy Lim, Eugene Jeong, Sam Wu, Ehsan Hassani, Michael Huang, Jin-Bey Yu, Chaeyeon Chung, Lina Song, Olivier Dionne, Jan Kautz, Simon Yuen, and Sanja Fidler. Kimodo: Scaling controllable human motion generation. *arXiv preprint arXiv:2603.15546*, 2026. URL https://arxiv.org/abs/2603.15546.

Zelin Tao, Zeran Su, Peiran Liu, Jingkai Sun, Wenqiang Que, Jiahao Ma, Jialin Yu, Jiahang Cao, Pihai Sun, Hao Liang, Gang Han, Wen Zhao, Zhiyuan Xu, Jian Tang, Qiang Zhang, and Yijie Guo. Heracles: Bridging precise tracking and generative synthesis for general humanoid control. *arXiv preprint arXiv:2603.27756*, 2026. URL https://arxiv.org/abs/2603.27756.

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. *arXiv preprint arXiv:2604.24833*, 2026. URL https://arxiv.org/abs/2604.24833.

Weiji Xie, Jinrui Han, Jiakun Zheng, Huanyu Li, Xinzhe Liu, Jiyuan Shi, Weinan Zhang, Chenjia Bai, and Xuelong Li. KungfuBot: Physics-based humanoid whole-body control for learning highly-dynamic skills. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2506.12851.

Yiming Xie, Varun Jampani, Lei Zhong, Deqing Sun, and Huaizu Jiang. OmniControl: Control any joint at any time for human motion generation. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=gd0lAEtWso.

Kangning Yin, Weishuai Zeng, Ke Fan, Minyue Dai, Zirui Wang, Qiang Zhang, Zheng Tian, Jingbo Wang, Jiangmiao Pang, and Weinan Zhang. UniTracker: Learning universal whole-body motion tracker for humanoid robots. *arXiv preprint arXiv:2507.07356*, 2025. URL https://arxiv.org/abs/2507.07356.

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 *Conference on Robot Learning*, 2025. URL https://arxiv.org/abs/2505.02833.

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*, 45(4), 2026. doi: 10.1145/3811284. URL https://arxiv.org/abs/2607.08741.

Appendix Contents

A **Implementation and Training Details** . . . 13
A.1 Network Architectures . . . 13
A.2 Generator Training and Inference . . . 13
A.3 MCE Encoding and Optimization . . . 15
B **Relational Representation and Training Data** . . . 16
B.1 Relation Types and Supported Entity Pairs . . . 16
B.2 Input Tensors and Validity Masks . . . 17
B.3 Data Construction and Partitions . . . 17
C **Additional Experiments** . . . 17
C.1 DCT Capacity Ablation . . . 17
C.2 Generator Diversity under Fixed Conditions . . . 18
C.3 Physical Error Detection . . . 18
C.4 Cross-Duration Physical Ranking . . . 20

A Implementation and Training Details

A.1 Network Architectures

Figure 5 expands the system overview in Figure 2 into layer-level architectures for the Generator and MCE.

A.2 Generator Training and Inference

Generator architecture. Table 3 lists the configuration of the two flow models and their shared temporal decoder. Parameter counts include each module's projections and embeddings; the shared decoder is counted once.

Table 3: Generator architecture and parameter counts.

Configuration	Root Flow	Pose Flow	Temporal decoder
Condition inputs	Root history/goal, duration	Root path, boundary poses, duration	Clean joint pose, same-frame root features
Flow inputs	Noisy root state, flow time	Noisy joint state, flow time	—
Output	Endpoint and DCT coefficient velocities	Joint-state flow velocities	Joint pose and root update
Output dimensions	$9 + 24 \times 9$	29 per frame	$29 + 6$ per frame
Hidden width	384	384	384
Transformer blocks	4	6	—
Attention heads	8	8	—
Transformer FFN width	1536	1536	—
Temporal residual blocks	—	—	6
Temporal kernel size	—	—	5
Temporal dilations	—	—	1, 2, 4, 8, 16, 32
Block activation	GELU	GELU	SiLU
Block normalization	Post-LN	Post-LN	Pre-LN
Dropout	0.10	0.10	0
Parameters (M)	7.939	11.047	5.386
Total parameters (M)		24.373	

Generator integration and bridge. Root Flow and Pose Flow each use 20 explicit Euler steps, $y^{j+1} = y^j + v_\phi(y^j, j/20; c)/20$, followed by one pass through the shared temporal decoder. These are 20 velocity evaluations per stage, not a variable-tolerance solver. The root bridge connects the boundary states and preserves the observed initial translation increment. Here R_{-1} is the root state

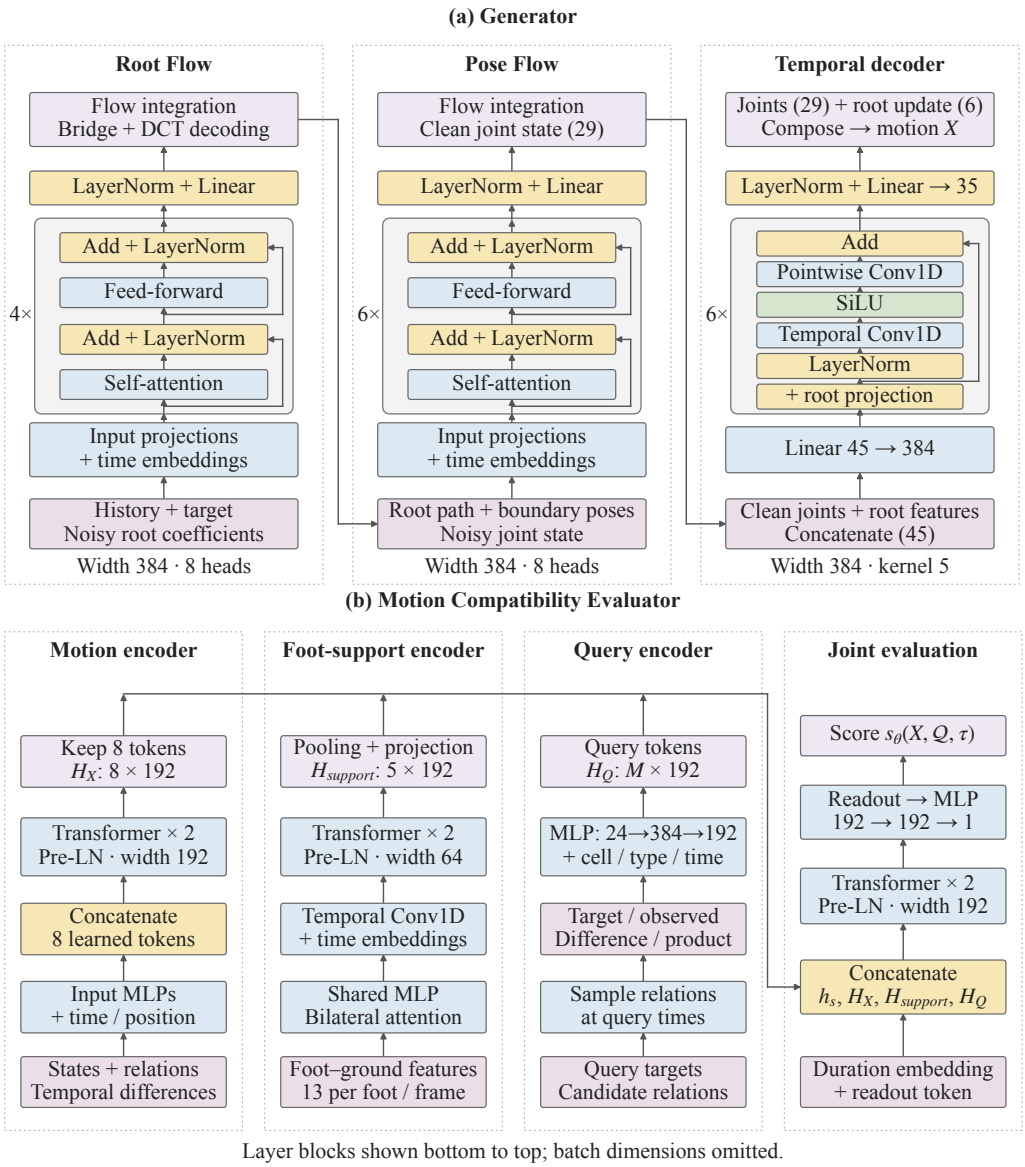

Figure 5: TAMGen network architectures. (a) Root Flow, Pose Flow, and the shared temporal decoder. (b) MCE encoders, Transformer, and scalar readout. Data flow proceeds from bottom to top, with arrows showing connections between modules and residual paths. Repetition counts and feature dimensions are indicated; batch dimensions are omitted.

immediately preceding R_0; history and generated states use the same time step. For normalized root states, let $\alpha_n = n/N$, $b_n^{\text{lin}} = (1-\alpha_n)R_0 + \alpha_n R_N$ and $g(\alpha) = \alpha(1-\alpha)^2$. Its translation is

$$b_{n,\text{xyz}} = b^{\text{lin}}_{n,\text{xyz}} + \frac{g(\alpha_n)}{g(1/N)}\left[(R_0 - R_{-1})_{\text{xyz}} - (R_N - R_0)_{\text{xyz}}/N\right].$$

This correction preserves both endpoints and matches the first translation increment to the observed history. The six orientation coordinates use the linear bridge; they are continuous rotation coordinates rather than interpolated Euler angles.

Training losses. The flow-matching loss in Equation 2 is implemented as a mean squared error over valid coordinates, excluding padding and inactive joints. The shared decoder is trained with reconstruction and physical losses computed on its decoded joint poses and root trajectory. Table 4

lists the individual loss weights before applying the overall decoder weight; the combined decoder loss has an overall weight of $\lambda_{\text{dec}} = 0.02$.

Table 4: Decoder loss weights.

Loss	Weight
Joint reconstruction	1.00
World-space FK	0.25
Contact	0.10
Temporal jerk	0.05
Support-foot sliding	0.05
Ground penetration	0.05
Root endpoint translation	0.10
Root endpoint orientation	0.05
Root-residual regularization	0.01
Root translation jerk	0.05

Root reconstruction losses are applied only when ground-truth root trajectories are used; physical losses are applied to both ground-truth and generated root trajectories.

A.3 MCE Encoding and Optimization

MCE architecture. Table 5 summarizes the three encoders, the joint Transformer, and the scalar readout. Here M is the number of relational constraints. Parameter counts include projections and embeddings but exclude normalization buffers. Shared token-type embeddings are counted in the joint network.

Table 5: MCE architecture and parameter counts.

Configuration	Motion encoder	Foot-support encoder	Query encoder	Joint Transformer & readout
Inputs	States, relations, temporal differences	Foot–ground features	Query targets, candidate relations	$H_X, H_{\text{support}}, H_Q$, duration
Output dimensions	8×192	5×192	$M \times 192$	1
Hidden width	192	64	192	192
Input MLP widths	$144 \to 384 \to 192$ $342 \to 384 \to 192$	$13 \to 64 \to 64$	$24 \to 384 \to 192$	—
Transformer blocks	2	1 bilateral + 2 temporal	—	2
Attention heads	6	4	—	6
Transformer FFN width	768	256	—	768
Temporal conv. kernels	—	3, 5	—	—
Learned pooling tokens	8	4	—	1 readout
Readout MLP widths	—	—	—	$192 \to 192 \to 1$
Activation	GELU	GELU	GELU	GELU
Transformer norm.	Pre-LN	Pre-LN	—	Pre-LN
Transformer dropout	0.08	0.08	—	0.08
Parameters (M)	1.287	0.359	0.133	0.966
Total parameters (M)	2.745			

MCE optimization. Table 6 lists the training settings and pair weights. Weighted losses are normalized by their weight sum.

Table 6: MCE optimization settings and pair weights.

Setting	Value
Initialization	Random
Physical-only updates	1,000
Joint updates	4,000
Optimizer	AdamW
Learning rate	3×10^{-4}
Weight decay	10^{-4}
Gradient-norm clipping	2
Configured batch size	72
Additional crossed physical units	128 per update, sampled separately
Crossed-unit sampling	Source-balanced, without replacement
Ranking margin	0.20
Relational pair weight	1
Reciprocal pair weight	2
Physical pair weight (no query or root query)	4
Replay loss	Smooth-L1, transition parameter 0.10
Replay target	Saved score after physical pretraining
Frozen encoders during joint training	None

Ranking supervision does not impose a fixed preference between a physically qualified motion paired with a mismatching query and a motion with physical errors paired with a matching query. Here $\mathcal{L}_{\text{pair}}$ and $\mathcal{L}_{\text{cls}}$ aggregate the per-pair ranking loss ℓ_{rank} and per-sample classification loss ℓ_{cls} from Equation 5, respectively, using normalized weighted averages. During joint training the objective is

$$\mathcal{L}_{\text{eval}} = \mathcal{L}_{\text{pair}} + \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{slide-pair}} + 0.5\mathcal{L}_{\text{crossed-phys}} + 3\mathcal{L}_{\text{replay}}.$$

Here $\mathcal{L}_{\text{slide-pair}}$ ranks an original motion below its online foot-sliding augmentation under the same query, using ℓ_{rank}. $\mathcal{L}_{\text{crossed-phys}}$ averages the two physical edges of the four-way construction, while relational edges are supplied by relational pairs. The replay loss compares the current score with the same network's saved score after physical pretraining.

B Relational Representation and Training Data

B.1 Relation Types and Supported Entity Pairs

All relation values are stored as six-dimensional vectors. Translation uses the first three entries, orientation uses all six entries, and gap and contact each use only the first entry. Unused entries are set to zero. Gap stores the signed distance to the ground, while contact stores a binary value. Entity-pair and relation-type embeddings tell the network what each vector represents.

The fixed registry contains:

Table 7: Supported entity pairs and relation types.

Ordered entity pair	Operators	Cells
Ground–FootL	translation, orientation, gap, contact	4
Ground–FootR	translation, orientation, gap, contact	4
Ground–Root	translation, orientation	2
Root–FootL	translation, orientation	2
Root–FootR	translation, orientation	2
Root–HandL	translation, orientation	2
Root–HandR	translation, orientation	2
Total		**18**

The entities and relation types are predefined. MCE learns to evaluate combinations of these relations.

B.2 Input Tensors and Validity Masks

Let F denote the number of frames in an MCE input window and M the number of relational constraints. MCE receives the following tensors:

Table 8: MCE input tensors, shapes, and meanings.

Tensor	Shape	Meaning
`raw_motion`	$[B, F, 48]$	Canonical serialized motion state at each frame
`relation`	$[B, F, 18, 6]$	Measured value of every registered relation cell
`relation_valid`	$[B, F, 18]$	Validity of every frame–cell measurement
`timestamps`	$[B, F]$	Physical frame times in seconds
`frame_valid`	$[B, F]$	Variable-length padding mask
`tau`	$[B]$	Requested physical horizon
`atom_slot`	$[B, M]$	Queried cell ID
`atom_target`	$[B, M, 6]$	Typed target v_i
`atom_time`	$[B, M]$	Requested time t_i
`atom_valid`	$[B, M]$	Query-set padding mask

Each robot state is stored in a 48-dimensional vector containing root and joint information. Unused dimensions are set to constant values. The `relation` tensor stores the actual relation values in the candidate motion. The `atom_target` tensor stores the values requested by the query. MCE compares these values to assess whether the motion meets the requested goals.

B.3 Data Construction and Partitions

The dataset contains 10,307 trajectories: 8,128 generated motions, 182 original motion clips, and 1,997 modified motions with physical errors. To construct physical errors, we keep the first and last two states unchanged and modify the motion between them. Vertical shifts create ground penetration or hovering, sudden translations create position jumps, and shifting the pose sequence in time while keeping the root motion unchanged creates root–limb mismatch. Each modified motion is checked to confirm the intended error. During training, we also introduce foot sliding without changing the endpoints. The training, validation, and test sets contain 7,613, 1,584, and 1,110 trajectories, respectively, from 47, 8, and 12 original motion segments. Modified motions remain in the same set as their original motions.

C Additional Experiments

C.1 DCT Capacity Ablation

DCT capacity comparison. We vary the number of DCT-derived basis functions used to generate root paths, testing 8, 12, 16, 24, 32, and 48 functions. More basis functions allow a wider range of path shapes. Each function is zero at both endpoints, so it changes only the path between them. We test whether increasing this number improves generation accuracy and motion diversity.

Evaluation with different numbers of DCT basis functions. Using an earlier generator architecture, we train six models, each using a different number of basis functions, for 60,000 updates. All models use the same network width and depth, optimizer, loss weights, and training-data sampling. We then evaluate them on the same 24 cases covering six durations: 0.30, 0.62, 1.20, 1.50, 2.00, and 3.00 s. We report root-path reconstruction error and query translation error. Query translation error

is averaged equally over the six durations. To measure diversity, we generate four motions for one case at each duration and compute the average pairwise distance (APD) between their root paths and whole-body poses. No candidates are removed based on physical quality. Each model is trained with one random seed.

Table 9: Ablation results for different numbers of DCT basis functions.

Modes	Projection error ↓		Query translation error ↓	Diversity (APD) ↑	
	Mean (cm)	P95 (cm)	(cm)	Root (cm)	FK (cm)
8	0.704	2.662	4.496	15.29	22.42
12	0.322	1.360	4.162	15.21	21.83
16	0.209	0.788	**3.390**	15.29	21.11
24	0.156	0.525	3.913	**25.86**	**32.37**
32	0.063	0.212	3.950	20.12	25.40
48	**0.038**	**0.123**	3.651	14.35	21.50

Results. More basis functions reduce root-path reconstruction error, but do not consistently reduce query translation error. The 16-mode model has the lowest query translation error, while 24 modes produce the greatest variation in both root paths and whole-body poses. Beyond 24 modes, reconstruction accuracy improves only slightly in absolute terms, while diversity decreases. We therefore choose 24 modes to balance accuracy and motion diversity (Table 9).

C.2 Generator Diversity under Fixed Conditions

To evaluate the Generator's ability to produce different motions under the same conditions, we compare variation in root paths and body poses with fixed start states, end states, and duration. Four generators are evaluated on six LAFAN and twelve BoneSeed conditions, each with a duration of 1.2 s and 16 motions generated per method and condition. All generated motions are included, without MCE selection, quality filtering, or endpoint correction. Average pairwise distance (APD) measures how much the generated motions differ from one another. Root APD compares 3D root positions over the 59 intermediate frames at 50 Hz. Local-pose APD compares 13 shared G1 body points in each frame's root coordinate system, removing differences in global position and orientation. Results are first averaged within each motion sequence, then equally across sequences.

Path variation and endpoint fidelity. Figure 6 shows all candidates for two conditions selected in advance. Our Generator produces distinct paths while remaining close to the requested endpoint. Across LAFAN/BoneSeed, its Root endpoint error is 0.251/0.279 cm, compared with 1.037/2.491 cm for MotionBricks, 2.043/2.890 cm for Kimodo, and 1.619/1.458 cm for ARDY. MotionBricks produces more concentrated paths: Root APD is 0.526/0.954 cm versus our 1.697/2.505 cm. Kimodo has greater Root APD (3.304/3.287 cm), but larger endpoint errors in this test. Our Generator therefore produces varied paths while keeping the endpoint accurate. APD measures variation between candidates, not target error.

Local motion variation across datasets. Figure 7 combines body-region statistics with all-candidate skeleton overlays at 0.6 s. Our Generator's local-pose APD is 2.988/4.676 cm on LAFAN/BoneSeed, compared with 1.517/2.803 cm for MotionBricks, 3.025/3.558 cm for Kimodo, and 1.852/2.792 cm for ARDY. The variation therefore persists after removing global path and orientation differences: our mean is comparable to Kimodo on LAFAN and highest on BoneSeed. These results show variation in body poses as well as root paths.

C.3 Physical Error Detection

Table 10 presents the detailed physical-error detection results from Section 4.1. Each comparison pairs a physically qualified motion with a motion containing physical errors under the same query condition. A ranking is correct when the qualified motion receives a lower MCE score. Values are mean accuracy (%) ± standard deviation (percentage points) across three random seeds.

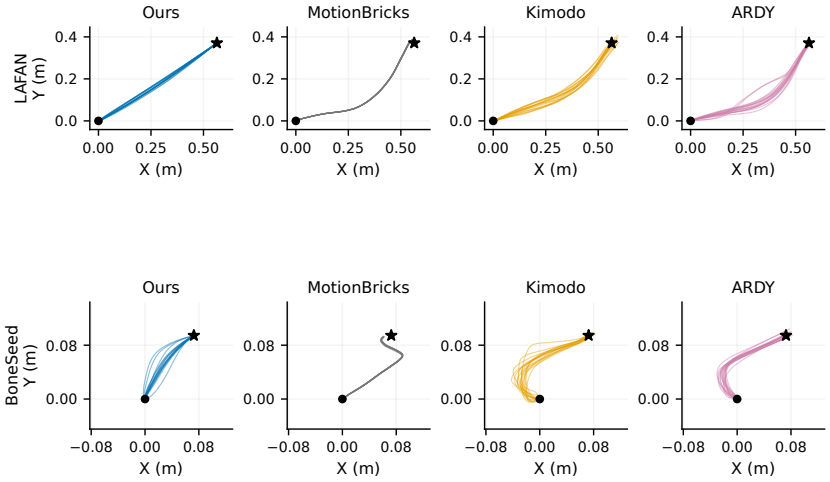

Figure 6: Root paths with fixed start/end states and a 1.2 s duration. Each panel shows all 16 generated paths for one LAFAN or BoneSeed example. The dot marks the start; the star marks the target. Paths are shown from above, with the same axes within each row.

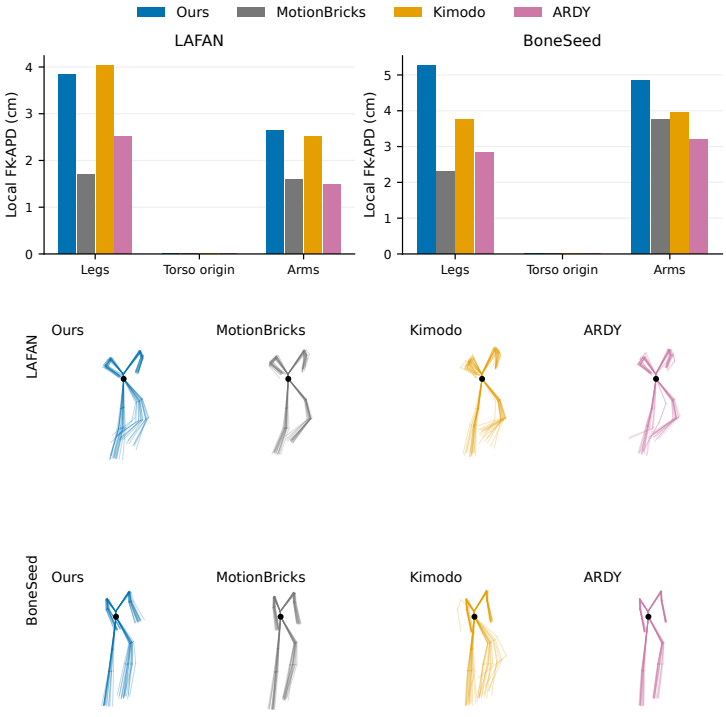

Figure 7: Body-pose variation under the same conditions. Top: average pairwise distances for the legs, torso position, and arms in each dataset. Below: all 16 poses at 0.6 s for the two examples above, aligned by root position and orientation. Black dots mark the root.

Table 10: Physical-error ranking accuracy under different query conditions (%, ↑).

	Pairwise ranking accuracy (%) ↑								
	No query			Matched query			Mismatched query		
Physical error	S	M	L	S	M	L	S	M	L
Ground penetration	96.9 ±1.4	100.0 ±0.0	100.0 ±0.0	99.1 ±0.7	100.0 ±0.0	100.0 ±0.0	98.9 ±0.8	100.0 ±0.0	100.0 ±0.0
Sustained hovering	95.8 ±1.8	100.0 ±0.0	100.0 ±0.0	92.6 ±2.5	100.0 ±0.0	100.0 ±0.0	93.3 ±2.3	100.0 ±0.0	100.0 ±0.0
Position discontinuity	92.5 ±2.3	100.0 ±0.0	100.0 ±0.0	96.2 ±1.5	100.0 ±0.0	100.0 ±0.0	92.7 ±1.9	100.0 ±0.0	100.0 ±0.0
Root–limb mismatch	100.0 ±0.0	100.0 ±0.0	100.0 ±0.0	95.6 ±1.0	100.0 ±0.0	100.0 ±0.0	98.0 ±1.3	100.0 ±0.0	100.0 ±0.0
Sliding	95.8 ±1.9	100.0 ±0.0	100.0 ±0.0	92.7 ±2.3	100.0 ±0.0	100.0 ±0.0	93.5 ±1.5	100.0 ±0.0	100.0 ±0.0

S/M/L denote small/medium/large perturbations.

C.4 Cross-Duration Physical Ranking

We compare a physically qualified motion with a motion containing physical errors when their durations differ. Both motions start from the same state and share the same spatial goal. A ranking is correct when the qualified motion receives a lower MCE score.

The candidate pool contains 1,536 generated motions covering eight initial states, six spatial goals, and two durations, 0.78 s and 1.48 s, with 16 candidates per condition. Pairs are evaluated without a query and with a matching terminal query at each motion's own end time. Equal weight is assigned to pairs where the qualified motion is shorter and pairs where it is longer. The final evaluation uses 1,985 motion pairs and is repeated with three fixed MCE models. We report the mean and sample standard deviation across these three evaluations.

Table 11: Physical pair-ranking accuracy across durations (%, ↑).

Query condition	Same duration	Cross duration
No query	97.43 ± 0.87	96.75 ± 2.17
Matched terminal query	97.42 ± 0.62	96.17 ± 2.51

Cross-duration ranking accuracy reaches 96.75% without queries and 96.17% with matching queries, compared with 97.43% and 97.42% for same-duration pairs (Table 11). These results show that MCE maintains high physical-ranking accuracy across the tested durations.