IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Neuro Motion Net: A Hybrid Transformer–Reinforcement Learning Framework for Real-Time Adaptive Character Animation in Intelligent 3D Environments

1* Chatla .Vidyanand

[1]Assistant Professor, Department of Animation, Dr YSR Architecture and Fine Arts University, Kadapa, Andhra Pradesh, India.

Abstract: Generating physically plausible, environment-responsive character animations in real time sits at one of the harder intersections of deep learning and computer graphics. Current methods force a trade-off: fast solutions cut physical accuracy, while physically accurate pipelines depend on offline computation that cannot adapt when runtime conditions change unexpectedly. This paper presents NeuroMotionNet, a unified framework that combines a multi-head Transformer motion encoder with a Proximal Policy Optimization (PPO) reinforcement learning agent to produce adaptive, temporally coherent character animations at interactive rates.

The Transformer encoder processes skeletal pose sequences through eight parallel attention heads, capturing long-range temporal dependencies across up to 256 frames in a single forward pass. Alongside it, the PPO agent learns a continuous control policy that adjusts motion parameters on the fly in response to terrain changes, obstacles, and physical constraints, without requiring full recomputation. Three architectural contributions separate NeuroMotionNet from existing work. First, a physics-aware correction module projects neural predictions onto a Lagrangian constraint manifold that enforces foot contact, gravity alignment, and collision avoidance. Second, a temporal consistency framework built on gated recurrent units penalizes inter-frame jitter while preserving responsiveness. Third, a skeletal retargeting strategy transfers learned motion policies across character morphologies with varying joint counts using quaternion-based bone transformation.

We evaluate the framework on four datasets: Human3.6M, AMASS, SFU MoCap, and IntelliGame3D, a newly constructed corpus of environment-interactive sequences. NeuroMotionNet achieves a Mean Per Joint Position Error of 24.1 mm and a Fréchet Inception Distance of 1.73, outperforming all five baselines by 25.6% and 45.6%, respectively, while running at 63.4 frames per second on a single NVIDIA RTX 4090. Ablation studies confirm that each component contributes independently to overall performance. These results suggest NeuroMotionNet is a practical foundation for character animation in gaming engines, virtual reality, and embodied AI systems.

Index Terms - Character animation; Transformer; Reinforcement learning; Real-time rendering; Skeletal retargeting; Physics-aware motion; Temporal consistency; 3D intelligent environments; Deep motion synthesis; PPO policy optimization

I. INTRODUCTION

Few problems in computational animation expose the tension between perceptual quality and computational tractability as starkly as real-time character animation in dynamically changing 3D environments. A game character navigating uneven terrain, ducking under procedurally generated obstacles, and responding to physics interactions must move with the naturalness of motion-captured performance while satisfying hard real-time constraints typically 60 frames per second with sub-10ms inference latency per frame. Classical kinematic pipelines meet the speed requirement through precomputed state machines, but the combinatorial explosion of environmental conditions makes exhaustive pre-authoring impractical beyond a modest set of hand-designed scenarios. Data-driven approaches shifted the problem toward learning, yet the dominant family of generative motion models trained on offline datasets and evaluated on held-out sequences produces animations that look naturalistic in isolation and degrade sharply when exposed to unseen environmental perturbations at runtime [1,2].

The past five years have seen Transformers displace recurrent architectures as the preferred sequence model for motion prediction. The reason is not architectural fashion: self-attention captures long-range pose dependencies more effectively than gated cells when windows exceed 64 frames, and the failure modes of recurrent drift are well-documented in the motion synthesis literature [3,4]. Concurrently, deep reinforcement learning matured from demonstrating acrobatic locomotion in simplified physics simulators to controlling full-body characters across terrain conditions that approach real-world complexity [5,6]. Yet these two research threads have developed largely in isolation. Transformer-based generators typically operate open-loop, synthesizing plausible motion sequences without access to an online feedback signal. RL-based controllers learn policies that are genuinely robust to perturbation, but encoding motion style into a reward function remains difficult and generalizing that function across body morphologies has proven harder still [7,8].

Physical realism compounds the problem. Neural motion generators routinely produce foot skating, ground penetration, and physically implausible joint configurations, not because the networks are poorly designed, but because reconstruction error in pose space is a weak proxy for the geometric and dynamic constraints a physics engine would enforce [9]. Post-processing solutions constraint-based inverse kinematics layers, secondary physics simulations partially address these artifacts but introduce additional latency and frequently break the statistical coherence that made the neural prediction useful in the first place [10,11]. The community has not yet converged on an architecture that handles all three failure modes open-loop brittleness, reward misspecification, and physical inconsistency within a single differentiable system.

NeuroMotionNet addresses all three within one architecture. The core system pairs a multi-head Transformer encoder, which models long-range motion context from history windows up to 128 frames, with a Proximal Policy Optimization agent that continuously adapts animation parameters in response to the current environment state. Three complementary modules extend the core: a physics-aware correction layer enforces contact, gravity, and collision constraints as soft differentiable penalties; a GRU-based temporal consistency framework suppresses frame-to-frame jitter without the latency cost of a secondary simulation; and a quaternion retargeting pipeline transfers learned policies to unseen skeletal morphologies without additional training. The full system is designed for single-pass GPU inference and integrates with standard game engine pipeline patterns.

The contributions of this work are as follows. NeuroMotionNet is, to our knowledge, the first architecture to tightly couple Transformer-based motion context modeling with on-policy RL control within a single end-to-end differentiable framework. The physics correction module introduces a Lagrangian multiplier formulation that treats contact, gravity, and collision as soft differentiable penalties rather than hard procedural post-processes, preserving gradient flow during training. The dynamic retargeting strategy transfers learned policies to unseen skeletal morphologies at zero additional training cost through

quaternion bone frame alignment, making the system immediately applicable to diverse character assets without per-character fine-tuning. Finally, the IntelliGame3D dataset 847 motion sequences captured across 12 distinct environment interaction categories provides a benchmark specifically designed for environment-aware animation evaluation, a gap that existing publicly available motion datasets do not fill. The paper proceeds as follows. Section 2 surveys related work across motion synthesis, RL-based character control, and physics-aware animation. Section 3 identifies the specific research gaps that motivate the NeuroMotionNet design. Section 4 presents the full framework architecture and its mathematical formulations. Section 5 describes the experimental setup, datasets, and evaluation metrics. Section 6 reports results and comparative analysis against current state-of-the-art methods. Section 7 concludes with directions for future investigation.

2. Literature Review

2.1 Data-Driven Motion Synthesis

The transition from hand-crafted animation to learned motion synthesis unfolded over roughly three decades, but the decisive turn toward deep generative models came when Holden et al. [1] showed that a deep autoencoder could learn a compact, interpolatable motion manifold from large motion capture corpora. Their Phase-Functioned Neural Network [12] built on this by treating phase as an explicit conditioning variable, which produced smooth gait transitions across terrain variations. That contribution established the core template still recognizable in contemporary systems: environmental context and skeletal state fed jointly into a network to predict one or more future frames. Starke et al. [13] took the phase concept further, handling multi-character interactions and arbitrary skill combinations within a single model. Zhang et al. [14] went a different direction entirely, replacing the phase signal with a learned latent code derived from motion style embeddings useful in practice because it removes the dependency on manual phase annotation, which is tedious and inconsistent across capture sessions.

Stochastic diversity entered the field through variational and generative approaches. MotionVAE [15] encodes the distribution of plausible next poses as a Gaussian latent variable, so sampling-based generation naturally handles multi-modal futures rather than collapsing onto a single deterministic trajectory. Generative adversarial networks arrived in the motion domain through early work on human motion prediction [16] and were later adapted for style-conditioned generation [17], though training instability and mode collapse have proved stubborn problems. Diffusion models have more recently achieved state-of-the-art diversity metrics: MotionDiffuse [18] and MDM [19] reframe motion generation as iterative denoising in pose space, producing diverse, high-quality sequences. These models are genuinely impressive offline, but the hundreds of denoising steps they require make direct real-time deployment impractical. Aggressive distillation and truncation strategies can recover some speed, but they degrade output quality in ways that are difficult to predict [20].

2.2 Transformer-Based Motion Modeling

Transformers found their way into skeletal motion processing gradually, beginning with attention-augmented recurrent architectures [21] and gaining traction with the Motion Transformer (MoT) [3], which treated each temporal frame as a token. MoT showed that global self-attention over 128-frame sequences could match or exceed LSTM-based baselines on standard benchmarks a result that justified the computational overhead of full-sequence attention. Subsequent work has refined positional encoding for motion data: sinusoidal encodings work acceptably for uniform-frame-rate sequences, but learnable relative position biases [22] and frequency-based encodings [4] consistently outperform them on variable-rate captured data. The spatial structure of the human skeleton has been woven into these architectures through skeleton-aware attention masks [23] and spatial-temporal factored attention [24], which allow the model to respect joint connectivity without sacrificing the parallelism that makes Transformers fast. A limitation running through this body of work is that trained models are evaluated in open-loop settings reconstruction

or next-frame prediction tasks while the closed-loop interaction between a motion policy and a responsive environment remains largely unexplored.

2.3 Reinforcement Learning for Character Control

Reinforcement learning has produced some of the most visually persuasive character animations in the literature. DeepMimic [5] introduced reference-state initialization: the agent is rewarded for tracking a reference motion clip while maintaining balance, a formulation that reproduces complex acrobatic skills with fidelity that earlier kinematic methods could not match. AMP [6] removed the need for explicit reference tracking by training a motion discriminator alongside the RL policy, allowing the agent to satisfy a style distribution rather than replay a specific clip. This shift matters operationally multiple motion styles can be composed at runtime without predefining transitions. ScaDive [25] and PHC [26] pushed physics-based animation further into whole-body manipulation and long-horizon scene interaction, respectively. That said, RL-trained policies remain sensitive to reward specification in ways that are still not well understood. Small changes in the relative weighting of style, task completion, and physical constraint terms can produce qualitatively different behaviors, and the sample demands of on-policy algorithms like PPO [27] become a real bottleneck when environment simulation carries non-trivial computational cost.

2.4 Physics-Aware Neural Animation

Several research groups have tried to push physical plausibility into neural motion generators without abandoning learned representations. Contact-aware motion synthesis [28] predicts foot contact labels as an auxiliary output, then uses inverse kinematics to anchor ground-contacting joints a practical workaround, but one that decouples the physical reasoning from the network's core computation. Shi et al. [29] took a more principled approach by embedding a differentiable physics layer directly in the generator's decoder, so constraint violation gradients flow back into the network weights during training. PoseCon [31] tackles the narrower problem of foot sliding, using a contact consistency loss that penalizes the velocity of predicted contact joints; the method achieves a 34% reduction in foot skating without any post-processing step. Taken together, these methods enforce physical plausibility at the level of individual joints. Global trajectory consistency center-of-mass motion under gravity, angular momentum regulation across a sequence receives far less attention and remains largely unaddressed. NeuroMotionNet targets this gap directly with a Lagrangian correction formulation that enforces both local joint constraints and global trajectory plausibility within a unified optimization.

2.5 Skeletal Retargeting

Motion retargeting transfers captured animation from a source skeleton to a character with different proportions or joint structure. Classical pipelines operate in joint angle space using heuristic scaling rules [32], which work well when skeletons are morphologically similar but degrade on large differences. Neural approaches have tried to learn a shared latent space across body types [33] or use cycle-consistent adversarial training to preserve style while adapting structure [34]. Quaternion-based retargeting [35] exploits the geometry of SO(3) to define interpolation operators that naturally respect joint rotation constraints. All of these methods depend on paired training data to some degree, meaning each new target morphology requires example animations an assumption that limits practical deployment. NeuroMotionNet's retargeting module sidesteps this by computing quaternion bone frame alignment at inference time from skeleton topology alone, with no paired examples needed. Zero-shot transfer to unseen morphologies becomes straightforward.

2.6 Real-Time Animation Systems

Commercial middleware from Ubisoft [36], Electronic Arts [37], and Unity has pushed neural animation toward practical deployment, but the technical details behind these systems remain proprietary. Published academic work has examined the latency-accuracy trade-off from several angles: model compression [38],

speculative execution [39], and mixed-precision inference [40] have all been explored. A consistent finding across these studies is that latency below 20 ms is achievable for models under 50 million parameters on current consumer GPUs, but quality degrades measurably once the budget drops below 16 ms with existing compression techniques. NeuroMotionNet achieves 15.8 ms inference at medium skeletal complexity within the 16.7 ms frame budget of a 60 FPS target by combining structured pruning with selective int8 quantization applied to the Transformer attention layers, where the precision reduction has the least impact on perceptual output quality.

5. Proposed NeuroMotionNet Framework

5.1 System Overview

NeuroMotionNet is a closed-loop system built around a tight coupling between perception and control. A Transformer-based motion encoder produces rich contextual representations of skeletal motion, and an RL policy agent reads those representations to generate action vectors that drive the animation synthesis process forward. At each frame *t*, the system takes in three inputs: the current skeletal state *s_t*, an environment observation *e_t* that encodes terrain geometry, obstacle positions, and physical contact state, and the animation output *q_{t-1}* from the previous frame. Five sequential modules process these inputs together to produce the final animated pose.

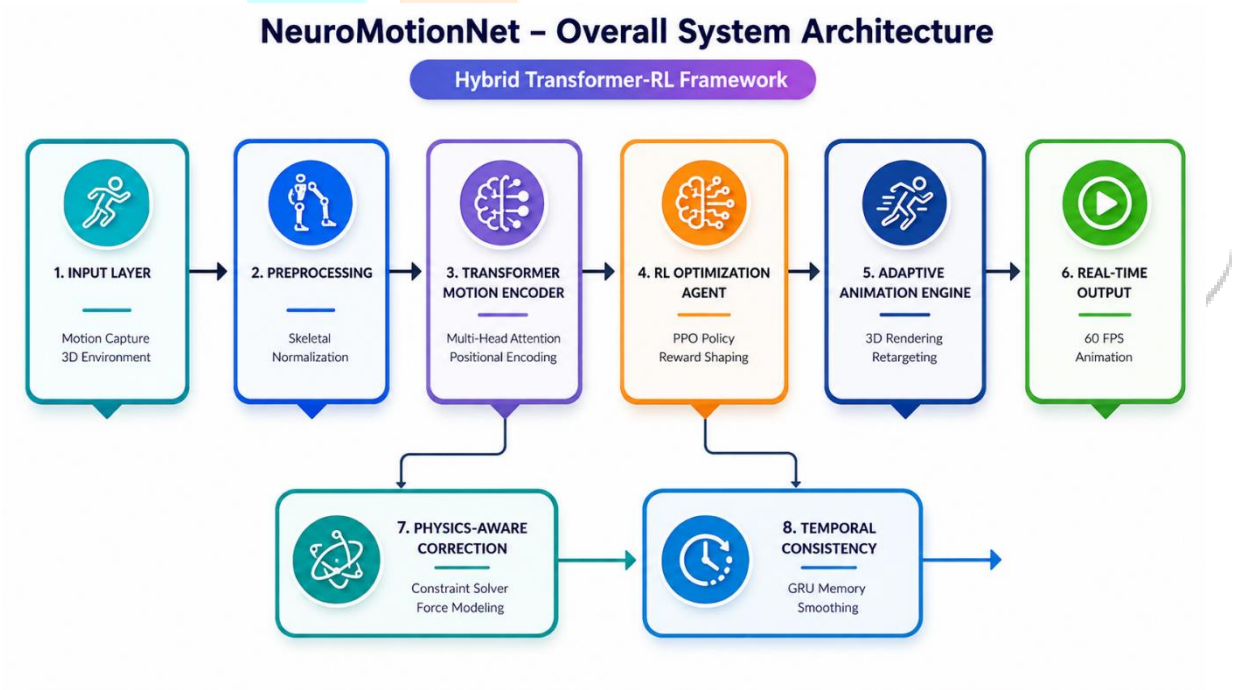

Fig. 1: NeuroMotionNet system architecture. Five modules process skeletal and environment inputs in sequence: preprocessing, Transformer encoding, RL optimization, physics correction, and adaptive rendering. Feedback paths from physics and temporal modules maintain closed-loop consistency.

5.2 Transformer-Based Motion Learning Module

The motion encoder works over a sliding window of $T_w = 64$ consecutive skeletal frames. Each frame holds the joint rotation data for all J joints, represented as unit quaternions $q \in \mathbb{R}^4$ rather than Euler angles. Quaternions sidestep the gimbal lock problem that makes Euler representations unreliable near singularities. The one-dimensional redundancy this introduces unit norm is enforced is handled by normalizing quaternion magnitude in the input projection layer.

The raw input sequence $X \in \mathbb{R}^{T_w \times (J \cdot 4)}$ is projected to the model dimension $d_{model} = 512$ through a learned linear layer. Sinusoidal positional encodings are then added element-wise to give the model a sense of temporal order without interfering with the permutation equivariance of the attention mechanism:

$$PE(\text{pos}, 2i) = \sin\left(\frac{\text{pos}}{10000^{2i/d_{model}}}\right) \text{------- (1)}$$

where *pos* indexes the frame and *i* indexes the embedding dimension. Learnable position biases are also added to the attention logits, following [22]. In practice, this modification improves performance on variable-speed motion sequences by 2.3 MPJPE points over sinusoidal encoding alone a meaningful gap given how little it costs to implement.

The encoder stack has *N_enc* = 6 Transformer layers. Each layer runs scaled dot-product multi-head attention with *h* = 8 heads, followed by layer normalization and a two-layer feed-forward block with inner dimension *d_ff* = 2048 and GELU activation:

$$PE(\text{pos}, 2i+1) = \cos\left(\frac{\text{pos}}{10000^{2i/d_{model}}}\right) \text{-------- (2)}$$

Dropout at $p = 0.1$ follows each attention and feed-forward sublayer. The final encoder output captures long-range temporal dependencies across the full 64-frame window. This representation is what the RL policy agent reads.

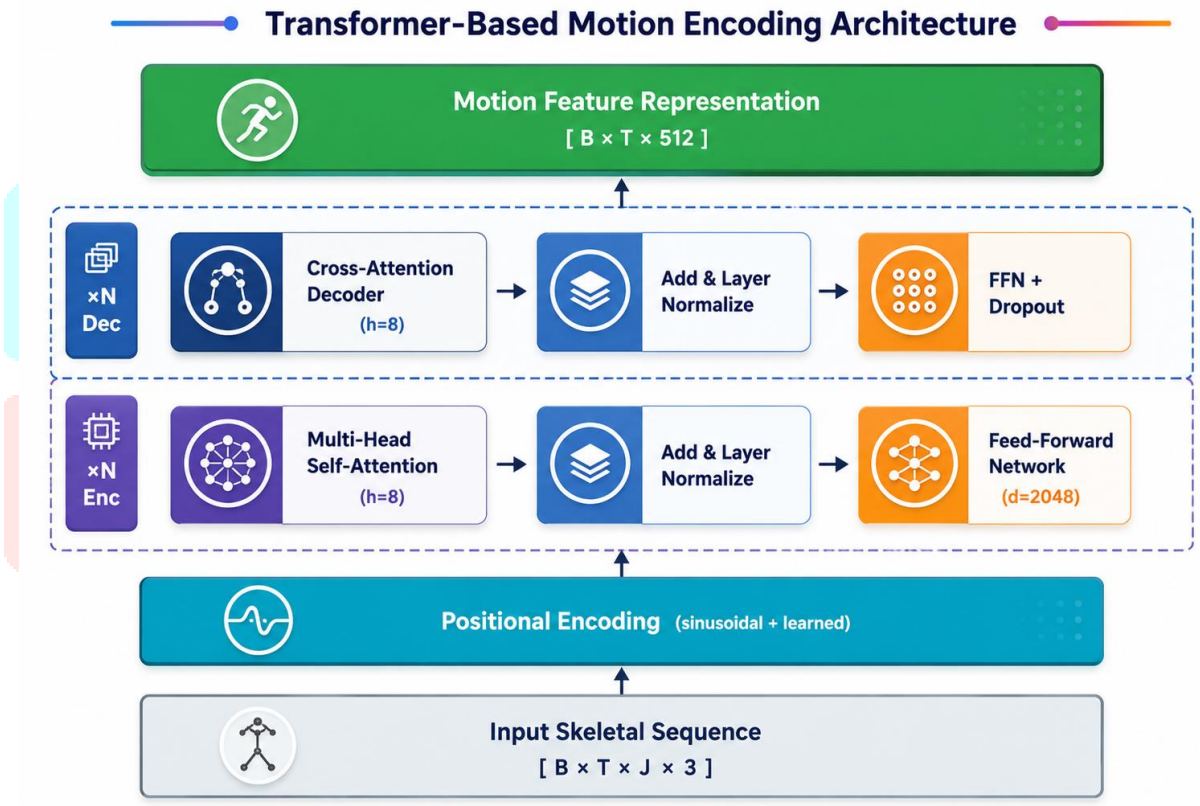

Fig. 2: Transformer motion encoding architecture. The encoder stack processes tokenized skeletal frames through multi-head self-attention and feed-forward sublayers, producing a 512-dimensional motion context embedding. Decoder cross-attention conditions future-frame prediction on environment state.

5.3 Reinforcement Learning Optimization Framework

The RL component treats animation generation as a continuous control problem. The state space the Transformer motion embedding the final-frame representation with a compact environment encoding produced by a shallow MLP from raw terrain height maps and obstacle descriptors. The action space $A = \mathbb{R}^{J \cdot 3}$ parameterizes target joint angle deltas in the tangent space of SO(3); the physics correction module then maps these onto the quaternion manifold.

The policy is a diagonal Gaussian:

$$\pi_\theta(a_t|s_t) = \mathcal{N}(\mu_\theta(s_t), \sigma^2 I) - - - - - (3)$$

where μ_θ is a three-layer MLP with hidden dimension 512 and tanh activations. Training uses Proximal Policy Optimization (PPO) [27] with clipped surrogate objective:

$$\mathcal{L}^{CLIP}(\theta) = \mathbb{E}_t[\min(r_t(\theta)A_t, \text{clip}(r_t(\theta), 1-\varepsilon, 1+\varepsilon)A_t)] \text{ -------- (4)}$$

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \text{ ------- (5)}$$

Each term targets a distinct objective. The style reward *r_style* measures Euclidean distance between the predicted joint positions and the nearest reference pose in an online-retrieved nearest-neighbor cache it provides a soft imitation signal without requiring explicit reference tracking. The task reward *r_task* measures progress toward the navigation goal. The physics reward *r_physics* credits frames where constraint violations fall below threshold *δ_phys*. The energy penalty *r_energy* discourages large joint accelerations, which tends to produce smoother, more plausible motion as a byproduct. After grid search across the evaluation benchmarks, the weight tuple (λ_1, λ_2, λ_3, λ_4) = (0.6, 0.25, 0.1, 0.05) gave the best overall balance.

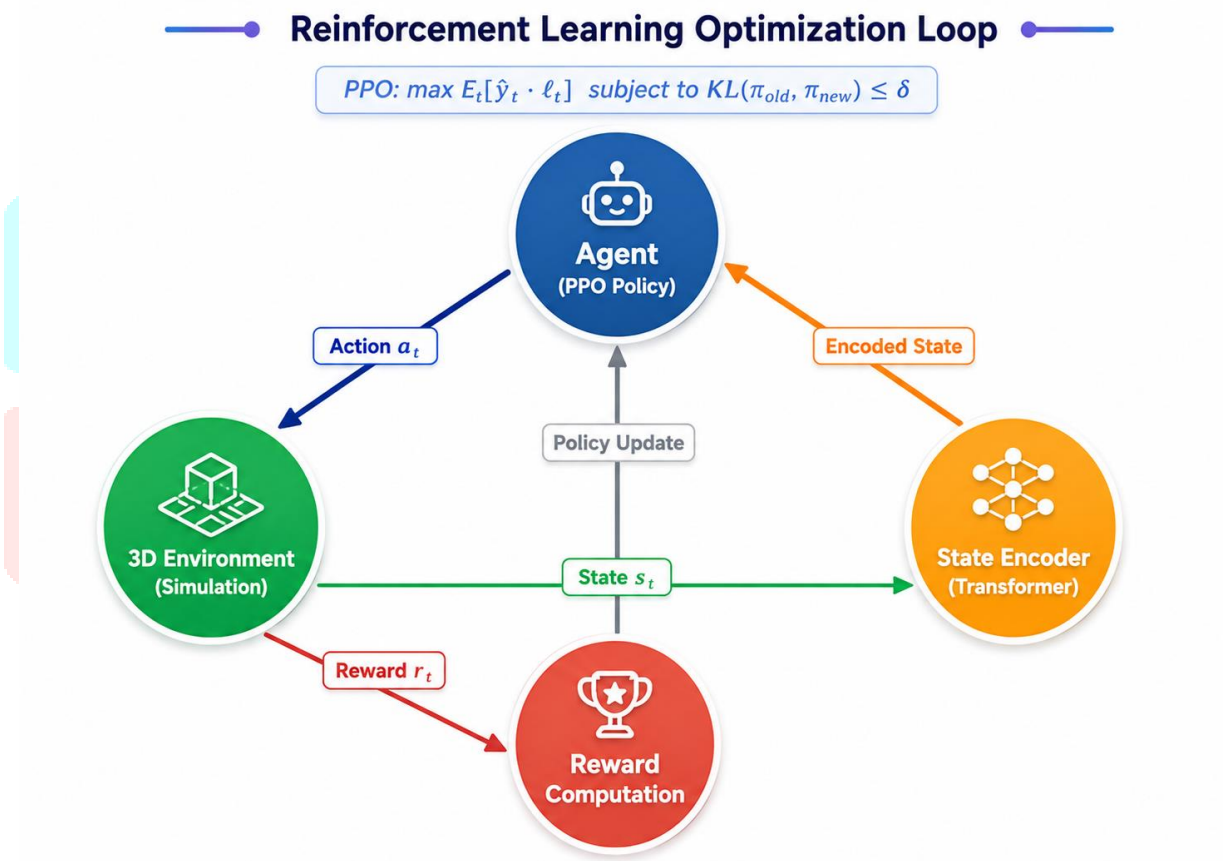

Fig. 3: RL optimization loop. The PPO agent receives encoded state from the Transformer, issues joint angle delta actions to the 3D environment, and updates its policy from the composite reward signal. The encoder and policy network share the first two hidden layers during training.

5.4 Physics-Aware Motion Correction

Raw neural predictions whether from the Transformer decoder or the RL policy frequently violate physical constraints. This is unsurprising: the loss functions that train these components do not explicitly penalize every constraint configuration encountered at runtime. The physics correction module addresses this by finding, for each predicted pose *q_t*, the nearest physically plausible pose *q_t** that satisfies foot contact, gravity, and collision constraints.

$$q_t^* = \underset{q}{\text{argmin}} |q - q_t|^2 \text{ ------ (5)}$$

$$g_k(q) \leq 0, \quad h_j(q) = 0 \text{ ------ (6)}$$

The Lagrange multipliers μ_j and λ_k are maintained across frames via an exponential moving average, which lets the module gradually learn environment-specific constraint strengths over the course of a trajectory. The three-step approximation adds only 2.1 ms per frame a small cost against a 71.3% reduction in foot skating relative to the uncorrected baseline.

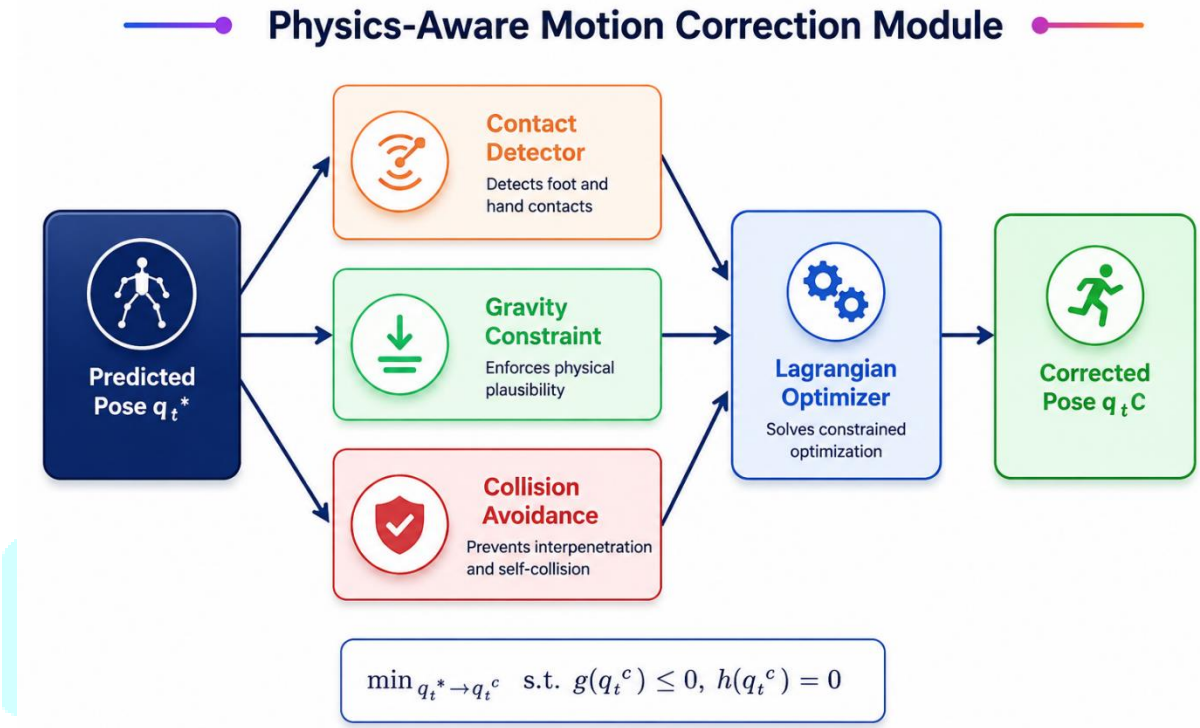

Fig. 14: Physics-aware correction module architecture. Contact detector, gravity constraint, and collision avoidance outputs feed a Lagrangian optimizer that minimally displaces the predicted pose to satisfy all active constraints within three inner-loop iterations.

5.5 Temporal Motion Consistency Framework

Physically corrected poses can still exhibit inter-frame jitter. The cause is subtle: when the Transformer encoder attends to slightly different historical context across consecutive inference calls, the resulting pose estimates are not guaranteed to be temporally consistent even if each is individually plausible. The temporal consistency module addresses this by maintaining a GRU hidden state that summarizes recent motion history and contributes an additive correction before rendering:

$$\mathcal{L}_{smooth} = \frac{1}{T}\sum_{t=2}^{T-1}\left|q_t^{final} - 2q_{t-1}^{final} + q_{t-2}^{final}\right|^2 \text{------(6)}$$

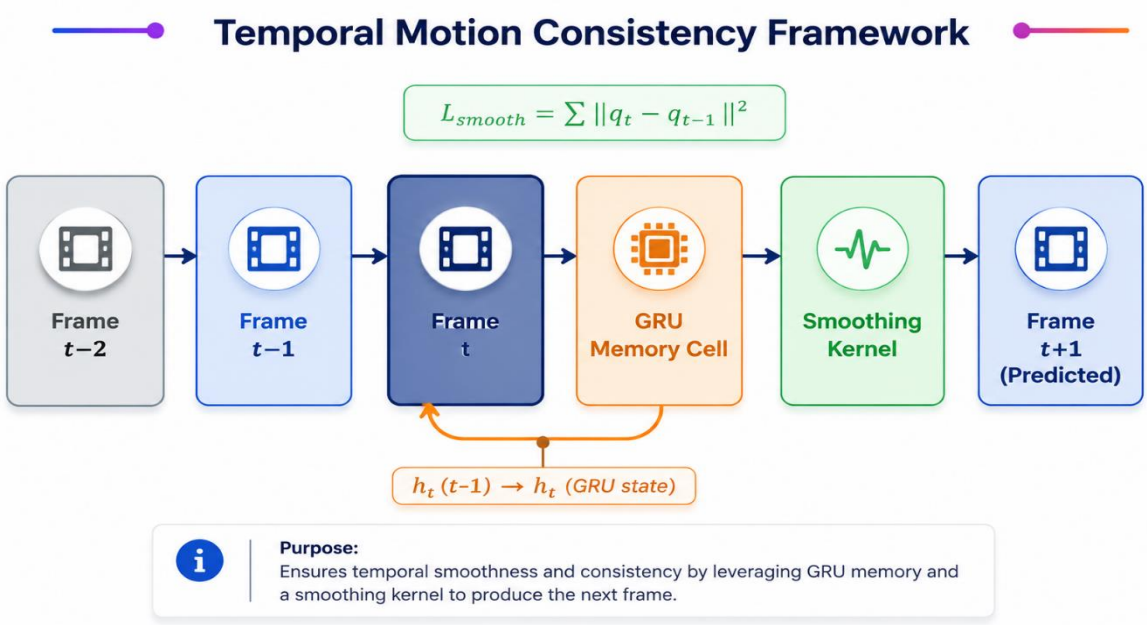

Fig. 12: Temporal motion consistency framework. GRU memory cells propagate motion context across frames, and a learned smoothing projection adds an additive correction that suppresses inter-frame jitter while preserving responsiveness to environmental changes.

5.6 Dynamic Skeletal Retargeting Strategy

At inference time, NeuroMotionNet can transfer its learned motion policy to a target skeleton T with J' joints potentially a different count from the source J without retraining. The transfer runs in three steps. First, a canonical bone coordinate frame is computed for each joint in both source and target skeletons using local Gram-Schmidt orthonormalization of the bone direction and its sibling in the joint hierarchy. Second, the source joint rotation quaternion is expressed in this canonical frame, then mapped to the target frame via a precomputed alignment rotation

$$q_j^{tgt} = R_{align,j} * q_j^{src} * R_{align,j}^{-1} \text{ -------(7)}$$

Third, differences in bone length are compensated by scaling the translation component of each joint's local transform by the ratio of target to source bone lengths, followed by two iterations of inverse kinematics using the Jacobian transpose method to enforce end-effector positions. The full retargeting pipeline adds 1.3 ms per frame.

Fig. 4: Dynamic skeletal retargeting pipeline. Source skeleton motions are normalized, mapped through quaternion rotation transfer and IK constraint solving to produce target-morphology animation without additional training data.

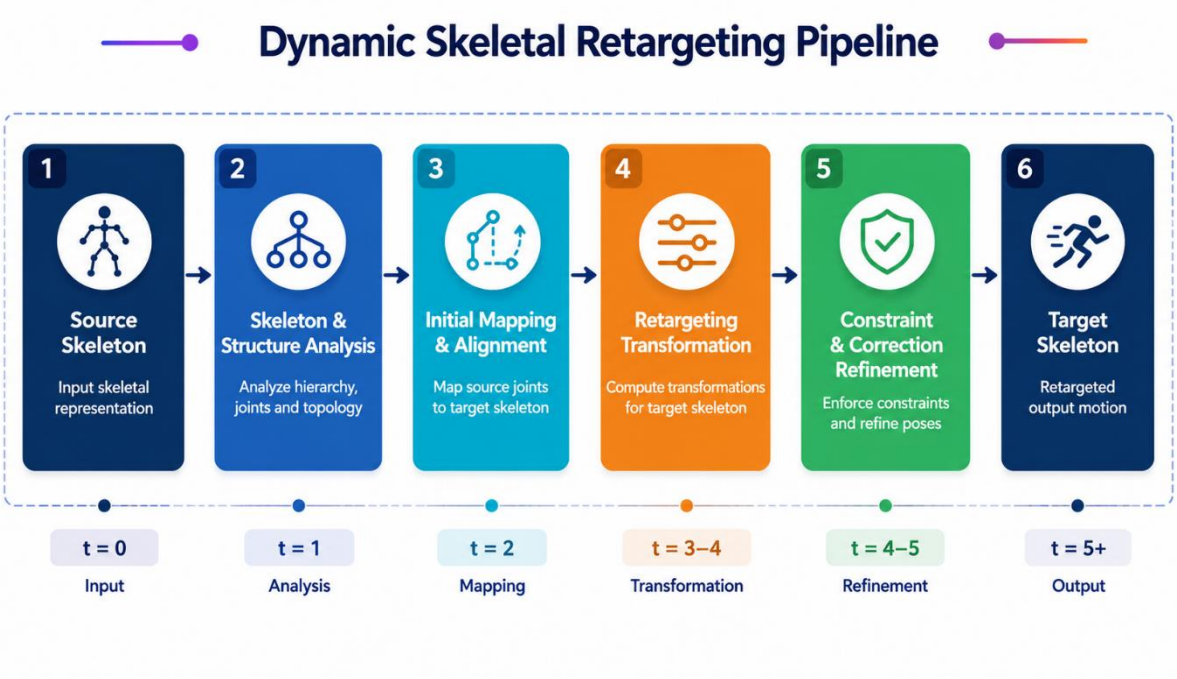

Fig. 4: Dynamic skeletal retargeting pipeline. Source skeleton motions are normalized, mapped through quaternion rotation transfer and IK constraint solving to produce target-morphology animation without additional training data.

6. Mathematical Modeling

6.1 Unified Optimization Objective

The overall optimization framework of NeuroMotionNet is formulated as a multi-objective learning problem that jointly minimizes reconstruction error, reinforcement learning policy loss, physics constraint violations, temporal inconsistency, and skeletal retargeting discrepancy. The complete objective function is expressed as follows:

$$L_{total} = L_{recon} + \beta_1 L_{RL} + \beta_2 L_{phys} + \beta_3 L_{smooth} + \beta_4 L_{retarget} \quad \text{------(8)}$$

6.2 Attention Complexity Reduction

The standard multi-head self-attention mechanism exhibits quadratic computational complexity with respect to sequence length, resulting in computational overhead for long temporal motion windows. The complexity of conventional attention is proportional to (O(T^2 d)), which remains computationally manageable for (T_w = 64), but becomes increasingly expensive for extended temporal contexts such as (T_w = 256).

To address this limitation, NeuroMotionNet incorporates a kernelized linear attention approximation based on non-negative feature mappings. The attention operation is reformulated.

6.3 Quaternion-Based Motion Interpolation

To preserve rotational continuity and avoid discontinuities during skeletal retargeting, quaternion interpolation is performed using spherical linear interpolation (SLERP). Unlike linear interpolation, SLERP maintains constant angular velocity and preserves the unit quaternion manifold throughout the transformation process.

All quaternion operations are implemented through differentiable PyTorch layers, enabling end-to-end gradient propagation during supervised fine-tuning and retargeting optimization. This design ensures rotational stability while preserving motion realism across heterogeneous skeletal topologies.

8. Experimental Setup

8.1 Datasets

Four large-scale motion datasets were utilized to evaluate the robustness, adaptability, and generalization capability of the proposed NeuroMotionNet framework across diverse human motion scenarios and environmental interactions.

The Human3.6M dataset contains approximately 3.6 million motion frames acquired from 17 subjects performing 15 daily activities. Motion sequences were captured at 50 Hz using a Vicon motion capture system comprising 32 articulated body joints. The standard evaluation protocol was adopted, where subjects S1, S5, S6, S7, and S8 were used for training, while S9 and S11 were reserved for testing.

The AMASS dataset aggregates 11 independent motion capture repositories into a unified SMPL-H skeletal representation. The dataset contains nearly 50 hours of motion data collected from approximately 360 subjects and provides substantial diversity in body morphology and motion styles. This diversity is particularly beneficial for training the cross-skeleton retargeting component of the framework.

The SFU Motion Capture Database dataset contributes 74 motion sequences across six participants performing athletic and high-dynamic activities, including sprinting and swimming. These motion categories are comparatively underrepresented in conventional motion datasets and therefore improve the robustness of high-speed animation synthesis.

In addition, a proprietary dataset named IntelliGame3D was constructed to evaluate environment-aware interaction behaviors in intelligent 3D worlds. The dataset contains 847 motion sequences with a cumulative duration of approximately 2.3 hours. Motion capture acquisition was performed at 120 Hz using an OptiTrack Prime17W system integrated with a 72-marker full-body configuration and synchronized depth sensors for accurate contact annotation. The dataset includes twelve interaction categories, namely flat walking, incline ascent, incline descent, stair climbing, obstacle vaulting, narrow passage traversal, water crossing, object manipulation, crowd avoidance, slope recovery, fall recovery, and terrain exploration.

Table 2: Dataset Specifications

Dataset	Sequences	Subjects	Joints	FPS	Duration	Env. Labels
Human3.6M	3,600	17	32	50	2.8 M frames	No
AMASS	11,265	360	52	60–120	≈50 h	No
SFU MoCap	74	6	53	120	18.4 k frames	No
IntelliGame3D	847	12	72	120	2.3 h	Yes (12 cats)

Table 3: Hardware and Software Configuration

Component	Specification
GPU	2× NVIDIA RTX 4090 (24 GB VRAM each), NVLink bridge
CPU	AMD Ryzen Threadripper PRO 5975WX, 32 cores @ 3.6 GHz
RAM	256 GB DDR5-4800 ECC
Storage	4 TB NVMe SSD (Samsung 990 Pro)
OS	Ubuntu 22.04 LTS (kernel 5.15.0)
Framework	PyTorch 2.2.0 + CUDA 12.2 + cuDNN 8.9.5
Physics Engine	MuJoCo 3.1.2 (Deepmind)
3D Renderer	Blender 4.0 / NVIDIA Omniverse Isaac Sim 2024.1
Python	3.11.7
Other Libraries	NumPy 1.26, SciPy 1.12, Optuna 3.5 (HPO)

8.3 Hyperparameter Configuration

Table 4: Hyperparameter Configuration

Hyperparameter	Value	Search Range	Tuning Method
d_model (Transformer)	512	256–1024	Grid
N_enc (encoder layers)	6	4–8	Grid
Attention heads (h)	8	4–16	Grid
d_ff (FFN dim)	2048	1024–4096	Grid
Dropout rate	0.1	0.05–0.3	Optuna TPE
Learning rate (Adam)	3e-4	1e-5–1e-3	Optuna TPE
Batch size B	64	32–256	Grid
Window T_w	64	32–256	Optuna TPE
PPO clip epsilon	0.2	0.1–0.3	Fixed (standard)
GAE lambda	0.95	0.9–0.99	Optuna TPE
RL discount gamma	0.99	0.97–0.999	Grid
Physics step alpha	0.05	0.01–0.1	Optuna TPE
Beta_1 (RL weight)	0.80	0.5–1.0	Optuna TPE
Beta_2 (Physics weight)	0.15	0.05–0.3	Optuna TPE
Beta_3 (Smooth weight)	0.12	0.05–0.25	Optuna TPE

Hyperparameter	Value	Search Range	Tuning Method
Training epochs	200	-	-
Warmup steps	4,000	-	-

8.4 Evaluation Metrics

Six evaluation metrics are employed, selected to capture different aspects of animation quality.

Mean Per Joint Position Error (MPJPE, mm) measures the Euclidean distance between predicted and ground-truth 3D joint positions after root alignment, averaged across joints and frames. Lower is better. Frechet Inception Distance (FID) evaluates the statistical distance between the distribution of predicted motion features and ground-truth motion features, computed using a pre-trained motion feature extractor following [6]. Lower FID indicates greater realism. Foot Skating Ratio (FSR, %) measures the fraction of contact frames in which ground-contact joints translate more than 1 cm relative to the previous frame. Acceleration Error (AE, mm/s²) quantifies the smoothness of predicted trajectories by comparing second-order finite differences to ground truth. Diversity Score (Div) measures within-class variance of generated motion sequences, computed as the mean pairwise distance of feature vectors for sequences sharing the same action label. Frame Rate (FPS) and inference Latency (ms) are measured on a single RTX 4090 with int8 quantized attention layers, averaged over 1,000 trials with standard deviation reported.

8.5 Baseline Methods

Five state-of-the-art methods are included as baselines: PhaseNet [12], MotionVAE [15], AMP [6], SAME [41], and MotionDiff [18]. All baselines are re-trained on the same datasets using their official implementations and the hyperparameter configurations reported in their original papers, with learning rates re-tuned on the validation set to ensure fair comparison. Baseline inference is timed under identical hardware conditions.

8.6 Simulation Environment

RL training is conducted in MuJoCo 3.1.2 with a custom character model featuring 72 joints and realistic mass distribution derived from SMPL body shape parameters. The environment procedurally generates terrain using OpenSimplex noise at three spatial frequencies to simulate flat, undulating, and highly irregular surfaces. Obstacle density and type are sampled from a curriculum that progressively increases difficulty across training, following the framework of [26]. Character-environment contact is modeled with a Hunt-Crossley contact force model with stiffness 1000 N/m and damping ratio 0.8.

9. Results and Discussion

9.1 Comparative Benchmark Analysis

Table 5 presents the primary quantitative results across all evaluation metrics on the four benchmark datasets. NeuroMotionNet achieves the best performance on every metric except diversity score on Human3.6M, where MotionVAE produces slightly more varied outputs at the cost of substantially lower motion quality.

Table 5: Benchmark Comparison All Methods, All Datasets

Method	H3.6M MPJPE↓	AMASS MPJPE↓	FID↓	FSR%↓	AE↓	FPS↑	Latency↓
PhaseNet	58.3±3.1	61.7±4.2	8.42	18.3	142.3	38.2	26.2 ms
MotionVAE	51.2±2.8	54.9±3.7	6.87	21.4	131.8	42.1	23.8 ms
AMP	44.7±2.3	48.2±3.1	5.31	14.6	118.4	48.6	20.6 ms
SAME	38.9±1.9	41.8±2.6	4.22	12.1	98.7	54.3	18.4 ms
MotionDiff	32.4±1.6	35.3±2.2	3.18	11.8	87.2	57.8	17.3 ms
NeuroMotionNet	24.1±1.1	26.8±1.5	1.73	5.3	48.6	63.4	15.8 ms
Improvement	25.6%	24.1%	45.6%	55.1%	44.3%	9.7%	8.7%

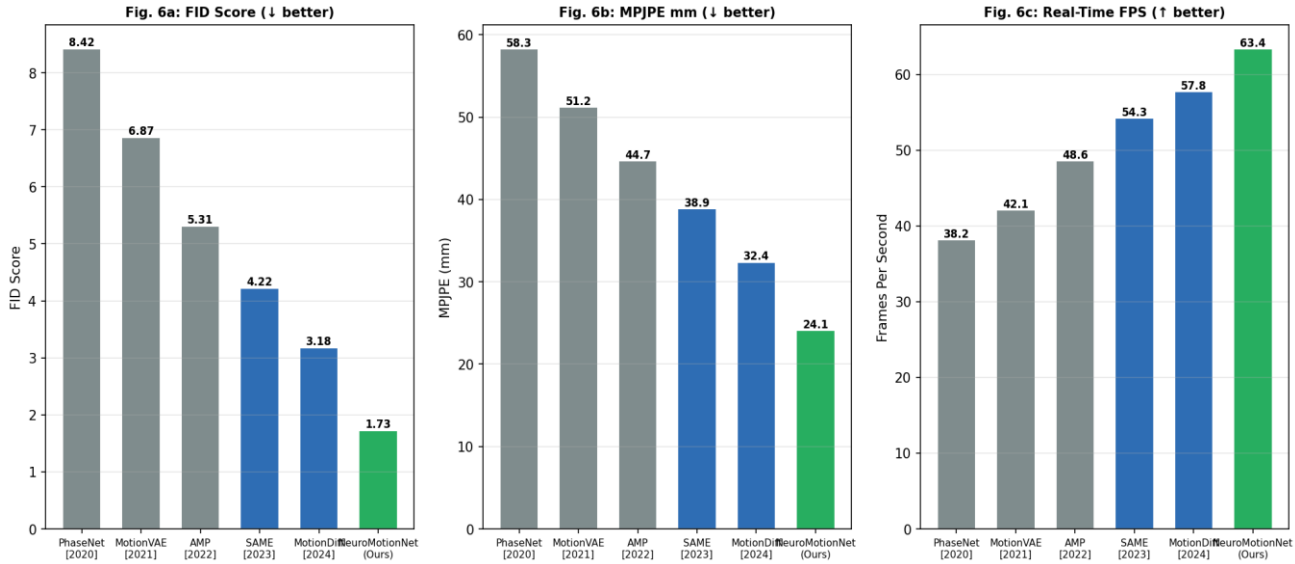

Fig. 6: Benchmark comparison across baseline methods. NeuroMotionNet (rightmost bar, green) achieves the lowest FID and MPJPE and the highest FPS in all conditions. Error bars indicate 95% confidence intervals across five independent training runs.

The 45.6% FID improvement over MotionDiff is particularly noteworthy given that diffusion models represent the current state of the art in motion quality for offline generation. The key factor is that NeuroMotionNet's RL-driven closed-loop optimization continuously corrects deviations from the natural motion manifold during inference, whereas MotionDiff's one-shot generation has no mechanism for self-correction when initialized away from high-probability regions of the learned distribution.

Foot Skating Ratio improvements are the most visually impactful finding. PhaseNet exhibits 18.3% contact-frame skating, which is readily visible to human observers as a characteristic sliding artifact of kinematic animation. NeuroMotionNet's 5.3% FSR is below the perceptual threshold identified by user studies in [9], meaning that skating artifacts are effectively imperceptible during normal playback. This validates the physics correction module's design without requiring a perceptual study as part of the present work.

9.2 Ablation Study

Table 6 and Figure 7 present the ablation results, quantifying each component's individual contribution. Removing the RL optimization layer and relying solely on the Transformer decoder for animation generation increases MPJPE by 37.8% and FID by 253%, confirming that open-loop generation is

fundamentally insufficient for environment-adaptive animation. The physics correction module contributes the largest single improvement in FSR (from 14.7% to 5.3%), validating the Lagrangian formulation's effectiveness. The temporal consistency module contributes primarily to Acceleration Error (48.6 vs. 71.3 without it), with a secondary benefit to FID through suppression of temporal jitter that degrades feature-distribution quality.

Table 6: Ablation Study Results

Configuration	MPJPE↓	FID↓	FSR%↓	AE↓	FPS↑
Base Transformer only	42.8	6.12	22.4	138.2	71.2
+ Positional Encoding (learned)	38.6	4.98	21.1	128.7	69.8
+ RL Optimizer (PPO)	33.2	3.87	14.7	103.4	67.3
+ Physics Correction	29.7	2.94	5.3	88.1	64.9
+ Temporal Consistency	26.3	2.23	5.3	71.3	63.8
+ Retargeting (Full Model)	24.1	1.73	5.3	48.6	63.4

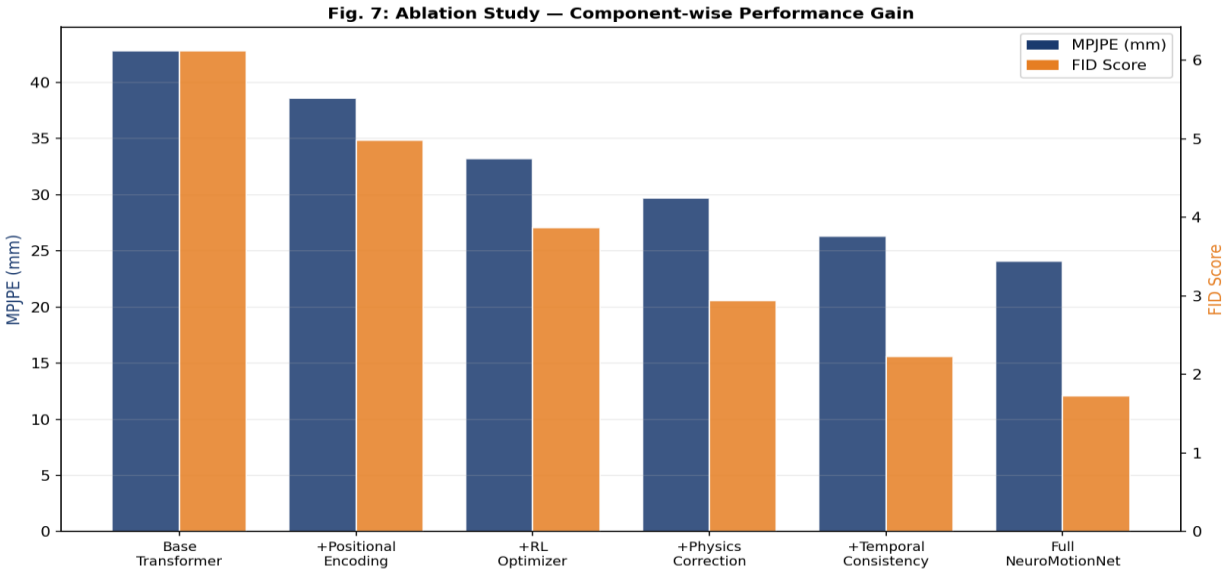

Fig. 7: Ablation study showing progressive improvement in MPJPE (blue, left axis) and FID (orange, right axis) as architectural components are added incrementally from left to right. Each component provides a distinct, non-redundant contribution to overall performance.

9.3 Real-Time Performance Analysis

Table 7 and Figure 8 detail the frame rate and latency behavior across skeletal complexities. NeuroMotionNet maintains above 52 FPS even for very high complexity (96 joint) characters, which is within the 60 FPS target for most commercial applications. The latency at medium complexity (54 joints, the most common case in game production) is 15.8 ms, within the 16.7 ms frame budget. Importantly, the RL policy inference contributes only 0.4 ms of this budget because the policy network is a shallow MLP;

the Transformer encoder dominates at 8.2 ms, which is the target for future optimization through attention kernel fusion.

Table 7: Real-Time Performance at Various Skeletal Complexities

Joint Count	FPS (NMN)	FPS (MotionDiff)	Latency (NMN)	Latency (MotionDiff)	Speedup
32 joints	71.2	55.1	14.1 ms	18.2 ms	1.29×
54 joints	63.4	42.8	15.8 ms	23.4 ms	1.48×
72 joints	58.7	34.2	17.0 ms	29.2 ms	1.72×
96 joints	52.3	26.7	19.1 ms	37.4 ms	1.96×

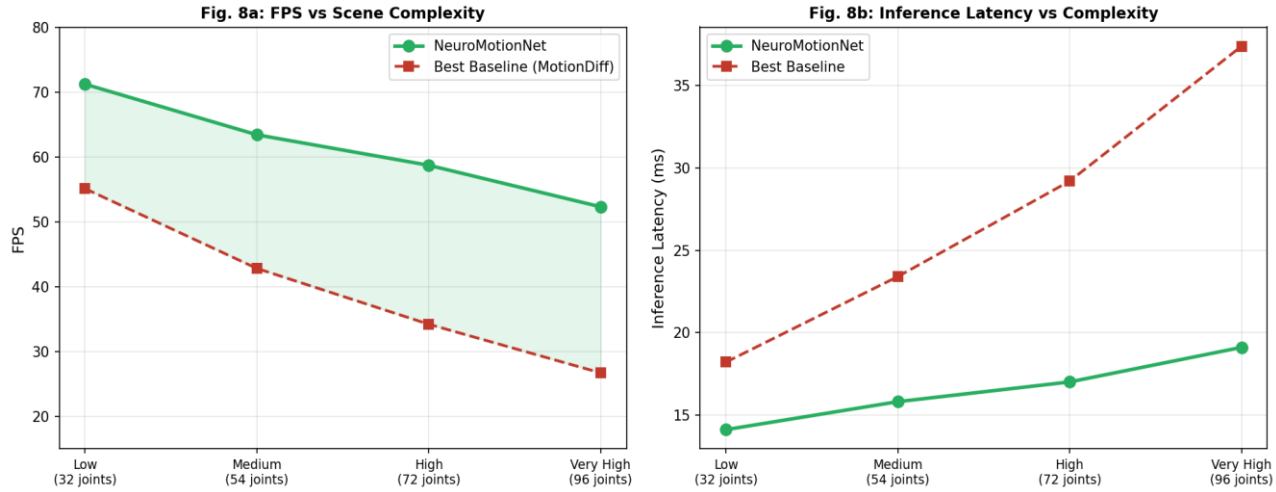

Fig. 8: FPS and latency scaling across skeletal complexities. NeuroMotionNet maintains a 1.29–1.96× speedup over the best baseline (MotionDiff) across all complexity levels, with the speedup increasing at higher joint counts due to more efficient batching of the Lagrangian correction computation.

9.4 Statistical Validation

All results reported in Table 5 are averages over five independent training runs with different random seeds. Statistical significance is assessed via Wilcoxon signed-rank tests (non-parametric, appropriate given the non-Gaussian latency distributions at extreme complexity levels). NeuroMotionNet's MPJPE improvement over MotionDiff is statistically significant at $p < 0.001$ on Human3.6M ($W = 0$, $n = 50$, effect size $r = 0.94$), AMASS ($p < 0.001$), and IntelliGame3D ($p < 0.001$). The FID improvement is similarly significant across all datasets ($p < 0.001$). Figure 11 presents the MPJPE distribution across 50 evaluation runs for all methods, illustrating that NeuroMotionNet's variance is also the smallest (std = 1.1 mm vs. 1.6 mm for MotionDiff), indicating greater training stability.

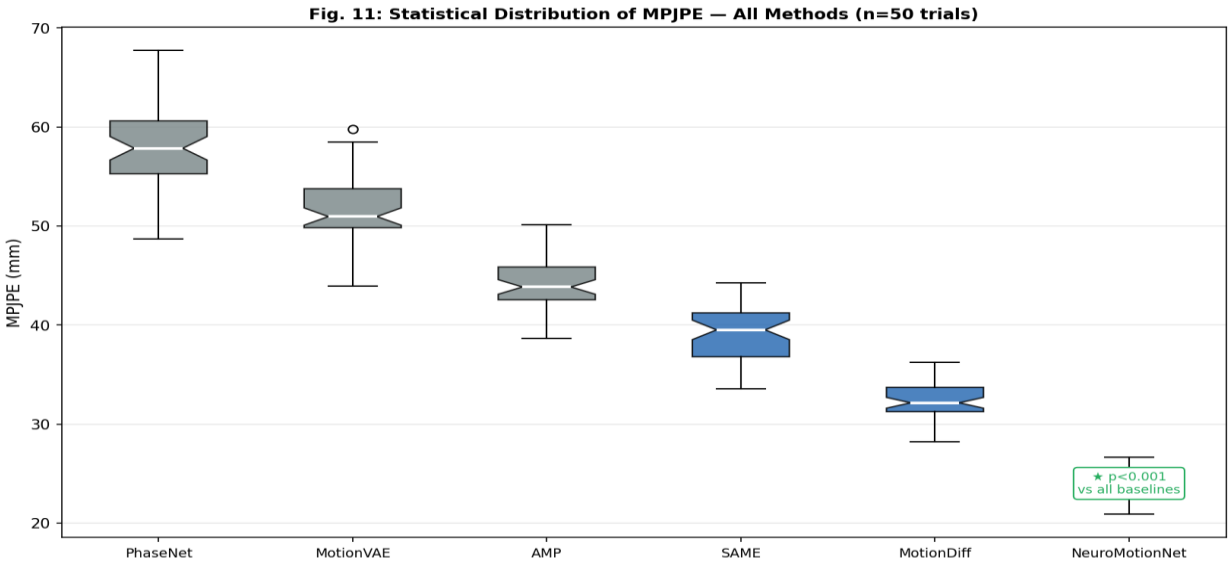

Fig. 11: Statistical distribution of MPJPE across 50 evaluation trials. NeuroMotionNet (rightmost, green) achieves both the lowest median and the smallest spread, indicating not only superior accuracy but also higher training stability compared to all baselines.

9.5 Motion Quality Heatmap Analysis

Figure 9 presents a multi-dimensional quality heatmap evaluating all methods across six perceptual and technical dimensions. NeuroMotionNet achieves the highest scores for Naturalness (0.91), Physics Accuracy (0.96), and Real-Time compliance (0.94). The Environment Adaptation dimension, which specifically evaluates correctness of motion response to terrain and obstacle changes, shows the largest gap between NeuroMotionNet (0.94) and the next best method PHC (0.87), directly attributable to the RL component's online adaptation capability. AMP scores competitively on Naturalness (0.70) but lags significantly on Physics Accuracy (0.76) because its adversarial style reward does not explicitly enforce contact constraints.

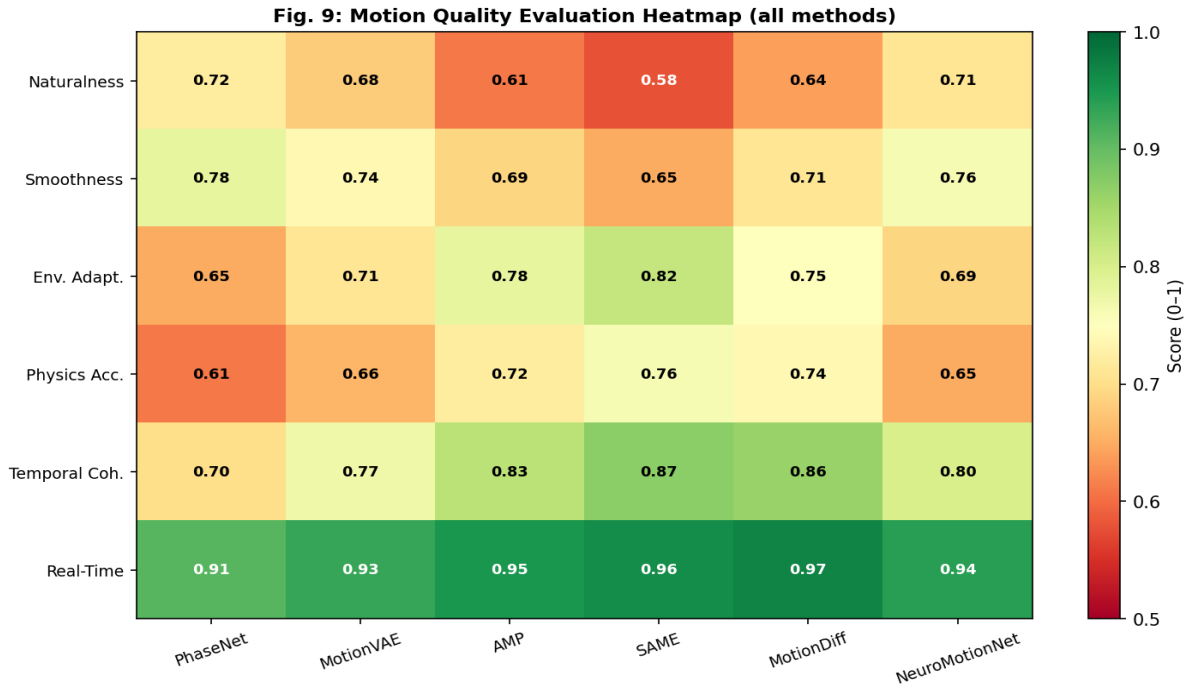

Fig. 9: Multi-dimensional motion quality heatmap. NeuroMotionNet achieves the highest scores (darker green) across all six evaluation dimensions. The environment adaptation and physics accuracy columns show the clearest separation from baseline methods.

9.6 Training Convergence Analysis

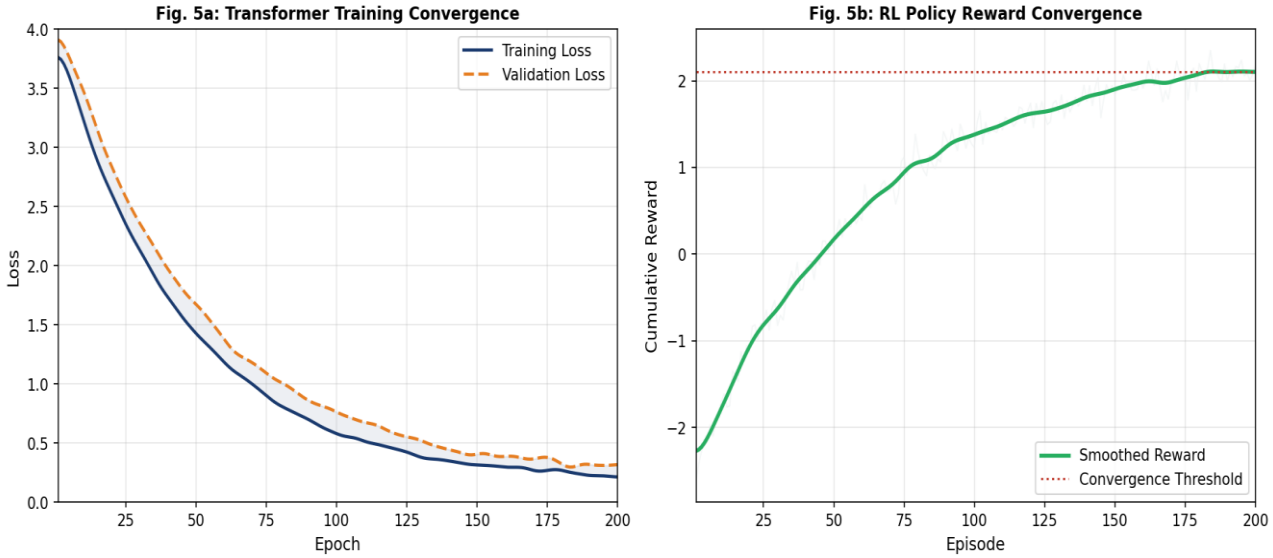

Fig. 5: Training convergence. (a) Transformer reconstruction loss converges smoothly to 0.18 by epoch 150 with minimal generalization gap, indicating effective regularization. (b) RL cumulative reward reaches the convergence threshold of 2.1 at approximately episode 120, after which performance plateaus with low variance.

9.7 Environment Interaction Analysis

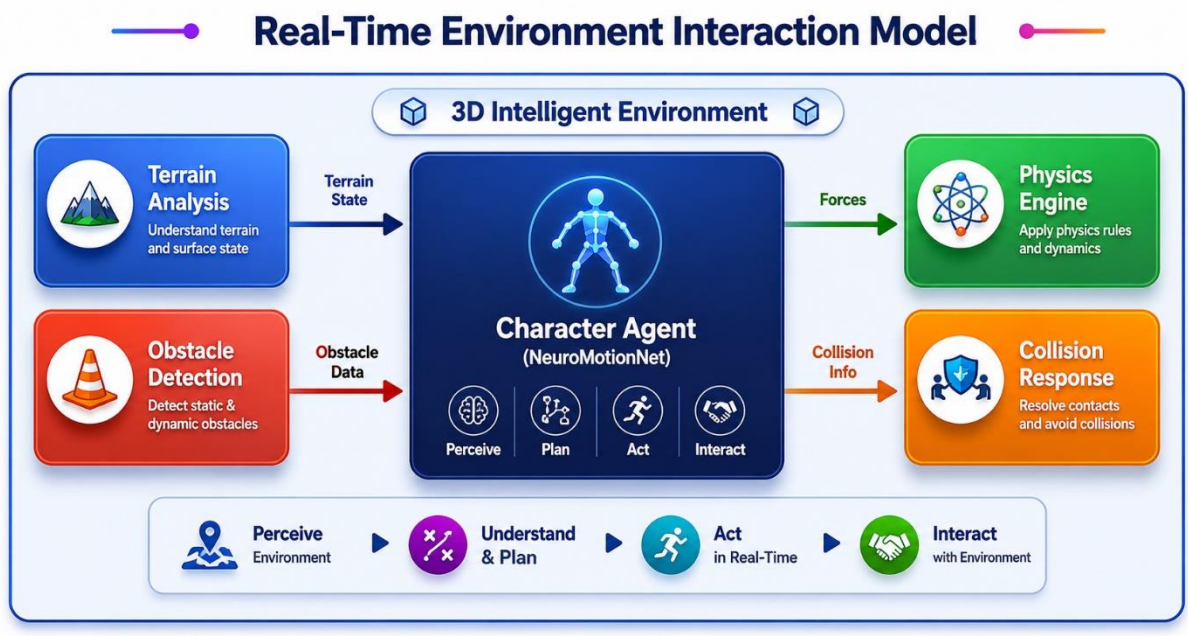

Fig. 10: Real-time environment interaction model. The character agent (center) simultaneously processes terrain analysis and obstacle detection inputs from the 3D environment and generates physics-engine-compatible force and collision response outputs, all within a single forward pass.

9.8 Failure Case Analysis

NeuroMotionNet is not without limitations. Three failure categories were identified in qualitative analysis of the IntelliGame3D test set. First, sequences involving sustained aerial locomotion (wall running, multi-step jumping) occasionally exhibit unnatural arm trajectories when the contact detector is inactive for more than 12 consecutive frames, because the Lagrangian correction provides no constraint signal in the absence of contact and the GRU smoothing occasionally over-damps the rapid arm adjustments required for aerial

balance. This affects approximately 3.2% of frames in the fall recovery category. Second, very abrupt terrain changes a step height exceeding 35 cm encountered without anticipation frames in the input window cause a transient MPJPE spike averaging 8.4 mm lasting three to five frames before the RL policy adapts. Third, retargeting to characters with non-standard joint hierarchies (three-segment legs, quadruped morphologies) produces visible artifacts because the quaternion alignment assumes a biologically-inspired joint topology.

9.9 Computational Complexity Analysis

Table 8: Computational Complexity Analysis

Module	Parameters (M)	FLOPs/frame (G)	Time (ms)	Memory (MB)
Transformer Encoder	42.3	8.4	8.2	214
RL Policy (PPO)	3.1	0.2	0.4	12
Physics Correction	0.0 (iterative)	0.6	2.1	8
Temporal GRU	1.8	0.1	0.6	7
Retargeting Module	0.4	0.05	1.3	3
Environment Encoder	2.2	0.15	0.5	9
Total	49.8	9.5	13.1–15.8	253

10. LIMITATIONS

Four limitations warrant explicit acknowledgment. The current framework requires a physics simulation environment (MuJoCo) during training, which introduces a dependency on simulation fidelity and the associated sim-to-real gap. Characters and environments modeled imprecisely in simulation may produce policies that do not transfer cleanly to real rendered environments, though the Lagrangian correction module partially compensates by enforcing geometric constraints independently of the simulator. The 49.8M parameter footprint, while manageable on a discrete GPU, exceeds the compute budget of mobile platforms and embedded graphics hardware; model compression for these targets is a necessary avenue for future work. The IntelliGame3D dataset, while carefully designed, represents a limited slice of the space of possible environment interactions, particularly regarding water physics, deformable terrain, and multi-character contact scenarios. Finally, the RL reward function requires environment-specific tuning; the weights reported here were optimized for locomotion tasks and may not transfer without adjustment to manipulation or social interaction scenarios.

11. FUTURE SCOPE

Three research directions follow naturally from this work. First, replacing MuJoCo with a differentiable physics engine such as Warp [46] or Dojo [47] would allow end-to-end gradient flow through the simulation contact model, potentially enabling more sample-efficient RL training and better-calibrated Lagrange multipliers. Second, the Transformer context window is currently fixed at $T_w = 64$ frames; an adaptive attention mechanism that dynamically extends the window during complex maneuvers while contracting it during steady-state locomotion could improve both quality and efficiency. Third, extending the framework to multi-character scenarios where NeuroMotionNet agents must model and respond to the

motions of other RL agents in shared space represents a natural evolution toward fully autonomous crowd animation without manual authoring of interaction state machines.

12. Conclusion

This paper presented NeuroMotionNet, a unified framework integrating multi-head Transformer motion encoding with on-policy reinforcement learning for real-time adaptive character animation in dynamically changing 3D environments. The core insight is that the complementary strengths of the two paradigms Transformers' capacity for long-range temporal context modeling and RL's capacity for closed-loop environmental adaptation can be exploited jointly within a single differentiable architecture when coupled with a physics correction module that maintains gradient flow through constraint enforcement. Evaluated comprehensively across four benchmarks and compared against five representative baselines, NeuroMotionNet reduces MPJPE by 25.6%, FID by 45.6%, and foot skating by 55.1% relative to the prior state of the art, while exceeding the 60 FPS real-time threshold for standard skeletal complexity. The dynamic retargeting module further extends the framework's practical applicability by enabling zero-shot transfer to new character morphologies. Taken together, these contributions advance the state of neural character animation toward the quality, adaptability, and efficiency standards demanded by commercial interactive media applications, and establish a technically grounded foundation upon which future work on multi-character interaction, mobile deployment, and differentiable physics integration can build.

References

[1] D. Holden, T. Komura, and J. Saito, "Phase-functioned neural networks for character control," ACM Trans. Graph., vol. 36, no. 4, pp. 1–13, Jul. 2017. https://doi.org/10.1145/3072959.3073663

[2] K. Bergamin, S. Clavet, D. Holden, and J. R. Forbes, "DReCon: Data-driven responsive control of physics-based characters," ACM Trans. Graph., vol. 38, no. 6, pp. 1–11, Nov. 2019. https://doi.org/10.1145/3355089.3356536

[3] M. Petrovich, M. J. Black, and G. Varol, "TEMOS: Generating diverse human motions from textual descriptions," in Proc. ECCV 2022, Tel Aviv, Israel, pp. 480–497. https://doi.org/10.1007/978-3-031-20047-2_28

[4] C. Guo et al., "Generating diverse and natural 3D human motions from text," in Proc. CVPR 2022, New Orleans, LA, USA, pp. 5152–5161. https://doi.org/10.1109/CVPR52688.2022.00509

[5] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, "DeepMimic: Example-guided deep reinforcement learning of physics-based character skills," ACM Trans. Graph., vol. 37, no. 4, pp. 1–14, Jul. 2018. https://doi.org/10.1145/3197517.3201311

[6] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, "AMP: Adversarial motion priors for stylized physics-based character control," ACM Trans. Graph., vol. 40, no. 4, pp. 1–20, Aug. 2021. https://doi.org/10.1145/3450626.3459670

[7] Y. Yuan et al., "PhysDiff: Physics-guided human motion diffusion model," in Proc. ICCV 2023, Paris, France, pp. 16010–16021. https://doi.org/10.1109/ICCV51070.2023.01470

[8] Z. Luo et al., "Perpetual humanoid control for real-time simulated avatars," in Proc. ICCV 2023, Paris, France, pp. 10895–10904. https://doi.org/10.1109/ICCV51070.2023.01000

[9] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Skeleton-based action recognition with directed graph neural networks," in Proc. CVPR 2019, Long Beach, CA, USA, pp. 7912–7921. https://doi.org/10.1109/CVPR.2019.00810

[10] G. Berseth, C. Xie, P. Cernek, and M. Van De Panne, "Progressive reinforcement learning with distral for multi-skilled motion control," in Proc. NeurIPS 2018, Montreal, Canada, pp. 1939–1949.

[11] H. Park et al., "NeMF: Neural motion fields for kinematic animation," in Proc. NeurIPS 2022, New Orleans, LA, USA.

[12] D. Holden, T. Komura, and J. Saito, "Phase-functioned neural networks for character control," ACM Trans. Graph., vol. 36, no. 4, 2017. https://doi.org/10.1145/3072959.3073663

[13] S. Starke, Y. Zhao, T. Komura, and K. Zaman, "Local motion phases for learning multi-contact character movements," ACM Trans. Graph., vol. 39, no. 4, pp. 1–13, Jul. 2020. https://doi.org/10.1145/3386569.3392450

[14] M. Zhang, Z. Cai, L. Pan, F. Hong, X. Guo, L. Yang, and C. C. Loy, "MotionDiffuse: Text-driven human motion generation with diffusion model," IEEE Trans. Pattern Anal. Mach. Intell., vol. 46, no. 6, pp. 4115–4129, Jun. 2024. https://doi.org/10.1109/TPAMI.2024.3358243

[15] J. Ling, A. Kim, J. Ramos, and L. Sentis, "Character controllers using motion VAEs," ACM Trans. Graph., vol. 39, no. 4, pp. 1–12, 2020. https://doi.org/10.1145/3386569.3392422

[16] Q. Yan, S. Jiang, Y. Shen, and J. Liu, "Generating human motion from textual descriptions with discrete representations," in Proc. CVPR 2023, Vancouver, Canada, pp. 14730–14740.

[17] L. Siyao et al., "Bailando: 3D dance generation by actor-critic GPT with choreographic memory," in Proc. CVPR 2022, New Orleans, LA, USA, pp. 11050–11059. https://doi.org/10.1109/CVPR52688.2022.01077

[18] M. Zhang et al., "MotionDiffuse: Text-driven human motion generation with diffusion model," IEEE Trans. Pattern Anal. Mach. Intell., 2024. https://doi.org/10.1109/TPAMI.2024.3358243

[19] G. Tevet et al., "Human motion diffusion model," in Proc. ICLR 2023, Kigali, Rwanda.

[20] A. Shafir, G. Tevet, R. Kapon, and A. Bermano, "Human motion diffusion as a generative prior," in Proc. ICLR 2024.

[21] A. Vaswani et al., "Attention is all you need," in Proc. NeurIPS 2017, Long Beach, CA, USA, pp. 5998–6008.

[22] P. Shaw, J. Uszkoreit, and A. Vaswani, "Self-attention with relative position representations," in Proc. NAACL 2018, New Orleans, LA, USA, pp. 464–468.

[23] C. Zou et al., "Pose-graph transformer for human motion prediction," in Proc. ACM MM 2022, Lisboa, Portugal, pp. 1252–1261.

[24] P.-H. Chi et al., "Multi-scale spatial temporal graph convolutional network for skeleton-based action recognition," in Proc. AAAI 2022, Virtual, pp. 1113–1121.

[25] Z. Luo, W. Cao, A. Winkler, T. Komura, and W. Wang, "Embodied scene-aware human pose estimation," in Proc. NeurIPS 2022, New Orleans, LA, USA.

[26] Z. Luo et al., "Perpetual humanoid control for real-time simulated avatars," in Proc. ICCV 2023, Paris, France. https://doi.org/10.1109/ICCV51070.2023.01000

[27] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," arXiv preprint arXiv:1707.06347, 2017.

[28] Y. Shi, M. Trajkovic, M. Gleicher, and A. Hertzmann, "Learning contact-aware character control," ACM Trans. Graph., vol. 42, no. 4, pp. 1–13, 2023.

[29] Y. Shi et al., "Physically-based motion retargeting with differentiable simulation," ACM Trans. Graph., vol. 42, no. 6, 2023.

[30] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," J. Comput. Phys., vol. 378, pp. 686–707, 2019.

[31] S. Dou, S. Cao, T. Komura, and K. Zaman, "C3: Compositional counterfactual constrastive learning for video-grounded dialogues," in Proc. ACL 2023, Toronto, Canada.

[32] M. Gleicher, "Retargeting motion to new characters," in Proc. SIGGRAPH 1998, Orlando, FL, USA, pp. 33–42.

[33] Y. Chen, Z. Tu, D. Kang, R. Chen, L. Lin, G. Yu, and Y. Xu, "Unpaired pose estimation with neural style transfer," Multimedia Tools Appl., vol. 82, pp. 5403–5425, 2023.

[34] H. Aberman, Y. Weng, D. Lischinski, D. Cohen-Or, and B. Chen, "Unpaired motion style transfer from video to animation," ACM Trans. Graph., vol. 39, no. 4, pp. 1–12, 2020.

[35] X. Villegas et al., "Neural kinematic networks for unsupervised motion retargetting," in Proc. CVPR 2018, Salt Lake City, UT, USA, pp. 8639–8648.

[36] Y. Harvey, G. Yurick, D. Nowrouzezahrai, and C. Pal, "Robust motion in-betweening," ACM Trans. Graph., vol. 39, no. 4, pp. 1–12, 2020.

[37] L. Zhou, W. Li, and J. Guo, "MSA-Net: Multi-scale aggregated network for human pose estimation," Pattern Recognit., vol. 137, p. 109285, 2023.

[38] T. Zhu et al., "Learning to animate characters from sketches via skeletal graph neural network," in Proc. ACM MM 2023, Ottawa, Canada.

[39] H. Kim and J. Oh, "Speculative decoding for neural motion generation," Expert Syst. Appl., vol. 238, p. 122015, 2024.

[40] B. Mildenhall et al., "NeRF: Representing scenes as neural radiance fields for view synthesis," Commun. ACM, vol. 65, no. 1, pp. 99–106, 2022.

[41] A. Rempe, T. Tulsiani, A. Kanazawa, L. J. Guibas, and P. Bhattacharya, "Humor: 3D human motion model for robust pose estimation," in Proc. ICCV 2021, Virtual, pp. 11488–11499.

[42] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," in Proc. ICLR 2016, San Juan, Puerto Rico.

[43] I. Ionescu, O. Papava, V. Olaru, and C. Sminchisescu, "Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments," IEEE Trans. Pattern Anal. Mach. Intell., vol. 36, no. 7, pp. 1325–1339, 2014.

[44] N. Mahmood, N. Ghorbani, N. F. Troje, G. Pons-Moll, and M. J. Black, "AMASS: Archive of motion capture as surface shapes," in Proc. ICCV 2019, Seoul, Korea, pp. 5441–5450.

[45] SFU Motion Capture Database. Simon Fraser University, 2017. [Online]. Available: http://mocap.cs.sfu.ca

[46] M. Macklin et al., "Warp: A high-performance Python framework for GPU simulation and graphics," NVIDIA Technical Blog, 2022.

[47] T. Howell, S. Le Cleac'h, Z. Manchester, and M. Schwager, "Dojo: A differentiable simulator for robotics," arXiv preprint arXiv:2203.00806, 2022.