HM^{2}O: Humanoid Motion Retargeting via Monolithic Optimization

Anonymous Authors

***Abstract*— Transferring human motion to humanoid robots remains difficult due to compounded *kinematics* and *dynamics* gaps, and common multi-stage pipelines can accumulate errors across stages. In this work, We propose HM^{2}O (Humanoid Motion Retargeting via Monolithic Optimization), a joint optimization framework that minimizes these gaps *simultaneously* by (i) optimizing robot joint-link scales together with base pose and joint angles *inside* the robot forward kinematics, (ii) decoupling position and rotation targets so they may use different human–humanoid joint mappings, and (iii) adding efficient dynamics regularizations including a ground-offset term, a stance (anti-slippage) term, and a biomechanics-inspired Froude-number prior to improve physical feasibility. HM^{2}O is efficient and runs in real time on a single CPU core. We evaluate our method on large-scale AMASS dataset, with off-the-shelf motion tracking policies including MaskedMimic, TWIST and GMT *without training*. Both the quantitative and qualitative results demonstrate that our HM^{2}O generates motion sequences that are more physically feasible, and boost the performance of a motion tracking policy.**

I. INTRODUCTION

Humanoid robots' locomotion and loco-manipulation have achieved significant progress in recent years, demonstrating agile and versatile motion skills [1, 2, 3]. One of the key elements that contributes to this progress is motion imitation learning on large-scale human motion datasets, including motion-capture datasets [4] that provide diverse and high-quality motion trajectories, internet-scale video datasets [5] that further expand coverage toward in-the-wild motions and long-tail behaviors, and synthetic datasets that provide smooth motion transitions [6] or precise human-object interactions [7].

However, transferring human motion to humanoid robots remains challenging due to kinematics and dynamics gaps. *The kinematics gaps* include the morphology mismatch between a human target and a humanoid embodiment, which arises from differences in body shapes and proportions, such as limb lengths and waist width, as well as the articulation mismatch—specifically, differences in joint degrees of freedom (DoF) and limits. These two kinematics gaps make direct joint-angle transfer unreliable, requiring inverse kinematics (IK) optimization to reduce end-effector and posture discrepancies [8]. Meanwhile, *the dynamics gaps* include contact mismatch: IK optimization objectives focus on joint matching and do not enforce the original contact relationships with the ground or objects, which can turn intended contacts into penetration or floating, leading to artifacts such as foot skating that are not physically feasible [5, 8, 9]. The second dynamics gap is actuation and inertia mismatch, because humanoids have different inertia distributions than humans and are driven by motors with limited torque. These two dynamics gaps can cause control policies to fail on motion sequences that are kinematically plausible but rely on human athletic power or rapid momentum changes, especially in highly dynamic sequences.

Prior motion retargeting methods address these gaps using a two-stage optimization, performed either offline on the entire sequence [1, 5, 10] or online frame-by-frame [11, 8]. *The first stage* is shape fitting that addresses the morphology mismatch between a human target and a humanoid. It optimizes shape parameters of the SMPL human model [12], and then updates the human joint positions using the fitted shape parameters and the original motion parameters. In contrast, some other methods [11, 8] directly rescale the offsets between joint positions and the human base position; these scales are pre-fixed and non-uniform across joints. *The second stage* is inverse kinematics optimization that tracks the updated human joint targets. This stage minimizes position and orientation errors of selected joints and/or end-effectors while enforcing joint limits, and optionally includes smoothness regularization when optimized over a sequence. To further reduce dynamics gaps, such as contact mismatch, some methods [10, 5, 8] also augment inverse kinematics objectives with soft or hard physical constraints, such as a grounding offset penalty that avoids floating and penetration, a slippage penalty that minimizes horizontal slippage of the stance foot, and an signed distance function (SDF) inequality constraint that avoids collisions at contact points.

Nevertheless, this two-stage strategy can accumulate errors across stages. *The first stage* performs shape fitting only on the canonical default pose, which makes the fitted shape parameters biased and sub-optimal for more diverse poses, especially for sequences with highly dynamic motion or rich contacts. Meanwhile, [5] observes that pre-fixed scales are unable to account for proportional differences between human and robot morphologies, leading to unnatural motion patterns. These discrepancies can accumulate and further propagate to *the second stage* of motion retargeting, where the IK optimization absorbs errors from the first stage through distorted joint angles or by trading physical feasibility for lower kinematics tracking errors. For instance, [5] observes that motion retargeting after shape fitting is often physically under-constrained, which introduces contact-related artifacts such as floating, penetration, and skating.

In this work, we propose **H**umanoid **M**otion Retargeting via **M**onolithic **O**ptimization (HM^{2}O), a joint optimization framework that efficiently minimizes kinematics and dynamics gaps all at once. Specifically, our HM^{2}O:

- parameterizes joint-link scales as optimizable variables jointly with the robot base pose and joint angles. These

scales are defined within the robot's forward kinematics, which adjusts the morphology of the robot itself to track the *original* human motion, thereby matching the standard definition of inverse kinematics. In contrast, shape fitting or pre-fixed scales in prior works modify the human joint positions before IK optimization, which can bake additional errors into target motion sequences.
- decouples position and rotation tracking targets for robot joints, and allows them to use different human–humanoid joint mappings, since they address different aspects of the kinematics gap. This is critical for serial single-DoF joint chains.
- incorporates dynamics regularizations, including (i) a ground offset term to enforce contacted foot keypoints on the same ground plane, (ii) a stance term that minimizes horizontal slippage of foot keypoints during contact, and (iii) a novel biomechanics-inspired Froude-number prior that associates the base position scale with the leg-link scale, to improve the dynamic feasibility of retargeted sequences.

In addition to the above contributions within a monolithic optimization, our HM^{2}O also runs in real time on a single CPU, which is important for handling large-scale human motion datasets and demonstrates potential for real-time applications such as teleoperation.

To demonstrate the advantages of our design, we evaluate our HM^{2}O along with prior methods using standard evaluation protocols [11, 13], where success rates and both global and local joint tracking errors are measured. In general, our HM^{2}O achieves state-of-the-art performance across these quantitative metrics when tracked with off-the-shelf motion tracking policies [14, 15, 2]. Notably, combining our retargeted motion references with GMT achieves the best overall performance *without* retraining GMT on our data, which is even superior to ProtoMotions+MaskedMimic, despite MaskedMimic being trained on and thus biased toward motion references generated by ProtoMotions. This advantage suggests that our retargeted motions are more physically feasible and generalize better to downstream motion tracking policies.

We also provide qualitative comparisons showing that our HM^{2}O generates motion sequences that better preserve human posture under morphology mismatch, where both shape fitting [5] and pre-fixed scaling [11] can introduce artifacts. Our method also produces more feasible velocities and more reliable contacts, which are critical for highly dynamic sequences where baseline retargeting methods can cause downstream motion tracking policies to fail. Together, the quantitative and qualitative results validate that our monolithic optimization framework addresses both the kinematics and the dynamics gaps more effectively than prior methods.

II. Related Work

A. Humanoid Motion Retargeting with IK Objectives

A common approach to human-to-humanoid retargeting formulates the problem as inverse kinematics (IK) fitting [16, 17, 18, 19], where the robot's base pose and joint angles are optimized to minimize position and/or orientation errors of selected key bodies or end-effectors, typically subject to joint limits and with optional smoothness regularization when solved over sequences. PHC [1] is a representative pipeline that first fits a human shape model (via SMPL [12]) to the robot skeleton and then optimizes the robot root pose and joint angles to track the resulting joint targets over time. ProtoMotions [10] includes an optimization-based retargeter that scales the source motion (e.g., using axis-wise scaling factors) and then minimizes joint position and orientation errors using an IK solver, with post-processing for height correction. GMR [11] highlights that scaling choices can be a dominant source of artifacts and proposes a pipeline with non-uniform local scaling and staged IK solving.

In this work, we upgrade forward kinematics with joint-link scales and then define scale-aware inverse kinematics objectives, which reflect how a robot should change its morphology natively to fit human motion targets. We further decouple the joint mappings in our position and rotation tracking errors, which further closes the joint DoF gaps between humanoid robots and humans.

B. Physics-Constrained Humanoid Motion Retargeting

Beyond pure IK fitting, a growing line of work augments retargeting with soft or hard physical constraints to reduce contact artifacts and improve physical plausibility. PHUMA [5] observes that shape-adaptive IK approaches can be physically under-constrained—leading to joint-limit violations and implausible ground interactions such as floating, penetration, and skating—and proposes PhySINK to directly incorporate physical terms (e.g., joint feasibility, grounding, and skating penalties) during retargeting. OmniRetarget [8] targets interaction-rich loco-manipulation and formulates retargeting as a constrained optimization that explicitly evaluates penetration, foot skating, and contact preservation; it notes that naive scaling of human keypoints to robot size can drive penetrations (e.g., hands into objects), and argues that hard-constraint formulations are less sensitive than soft-penalty tuning. SPIDER [9] approaches the missing-contact and missing-dynamics problem by transforming kinematic-only demonstrations into dynamically feasible trajectories via physics-based sampling, introducing curriculum-style virtual contact guidance to resolve contact ambiguity, and scaling across diverse embodiments and datasets with improved success and generation speed. Together, these methods emphasize that contact consistency (ground and object) is a key determinant of both retargeted motion quality and downstream policy performance.

In this work, we still incorporate soft physics constraints due to their efficiency. Specifically, we define ground-offset and stance regularizations for firm contact, and we introduce a Froude-number-inspired constraint [20] that adjusts a sequence of robot base positions and velocities proportionally to leg length. This significantly improves the physical feasibility of retargeted motions, whereas prior work often overlooks the relationship between desired robot base velocity

and leg length.

C. Learning-Based Humanoid Whole-Body Control

Learning-based humanoid whole-body control commonly follows a motion imitation paradigm: policies are trained (often with RL) to track retargeted reference trajectories while maintaining balance and robustness [21, 22, 23]. MaskedMimic [2] provides a unified policy-learning framework for motion tracking, including full-state tracking and teacher–student distillation for partial-state settings. Teleoperation systems further integrate perception/retargeting with real-time tracking control: TWIST [14] formulates teleoperation as real-time motion retargeting and tracking, deriving targets such as robot joint positions and root velocities and training a single robust controller using RL, while using a teacher–student design (privileged future frames for the teacher) to improve smoothness under low-latency constraints. BeyondMimic [3] further argues that, when reference motions are sufficiently clean, a minimal reward formulation can already yield high-quality tracking without extensive reward tuning, and it is developed independently from specific retargeting pipelines, making it a useful and fair evaluation tool for isolating the impact of retargeting quality on downstream policy performance. To address the challenge of tracking a highly diverse range of movements, GMT [24] introduces a general motion-tracking framework that trains a single unified policy by combining an Adaptive Sampling strategy to balance easy and difficult motions with a Motion Mixture-of-Experts (MoE) architecture to better specialize across different regions of the motion manifold.

In this work, we do not train any control policies on our retargeted motion sequences. Instead, we evaluate our method with off-the-shelf motion tracking policies from their official repositories, which demonstrate the superiority of our HM^{2}O by boosting the performance of existing control policies without training.

III. Method

A. Problem setup and variables

We formulate human-to-humanoid retargeting as a monolithic joint optimization problem as follows:

$$\begin{aligned} \boldsymbol{x}, \boldsymbol{s} &= \arg\min_{\boldsymbol{x},\boldsymbol{s}} \sum E(\boldsymbol{x}_t, \boldsymbol{s}) + \sum R(\boldsymbol{x}, \boldsymbol{s}), \\ \text{s.t.} \quad & \boldsymbol{x}^- \le \boldsymbol{x} \le \boldsymbol{x}^+, \ \boldsymbol{s} > 0. \end{aligned} \tag{1}$$

where $\boldsymbol{x}$ denotes the generalized poses of a robot over an entire sequence, including the floating-base $SE(3)$ pose and all joint angles, t represents the index of a single frame, and $\boldsymbol{s}$ rescales base positions and joint-link lengths, aligning joint positions (after our scale-aware forward kinematics) to each motion capture subject.

B. Inverse Kinematics Costs

We first introduce our inverse kinematics (IK) costs that track the positions and orientations of human joints. First is a scale-aware position tracking cost:

$$E_{pos}(\boldsymbol{x}_{t,i\in\mathcal{C}_b}, \boldsymbol{s}_{i\in\mathcal{C}_b}) = \rho(\|\boldsymbol{p}_{t,b} - \prod_{i\in\mathcal{C}_b} \boldsymbol{L}_i(\boldsymbol{x}_{t,i}, \boldsymbol{s}_i)\|_2^2), \tag{2}$$

where b denotes a robot body or joint, $\mathcal{C}_b$ is the chain of DoFs that affect b, and $\boldsymbol{p}_{t,b}$ is the position of the corresponding human joint in a frame. Here, $\boldsymbol{L}_i(\boldsymbol{x}_{t,i}, \boldsymbol{s}_i)$ is the local $SE(3)$ transformation matrix of joint i in its parent's local coordinates, $\prod_{i\in\mathcal{C}_b}$ denotes the forward-kinematics composition of local transformations along the kinematic chain, and $\rho(\cdot)$ is a robust Huber kernel.

This scale-aware cost function distinguishes our work from prior methods in two ways:

- **First is how the scale is optimized:** We consider joint-link scales as optimizable variables $\boldsymbol{s}$ jointly with base pose and joint angles $\boldsymbol{x}$, enabling the optimizer to minimize position tracking errors by adjusting both joint angles and link scales, rather than absorbing morphology mismatch through joint configurations alone. In comparison, PHC [1] and PHUMA [5] minimize morphology mismatch via shape fitting of the human model [12], and fix the shape parameters that are fitted on a canonical default pose before generating per-frame joint targets for IK; while GMR [11] and OmniRetarget [8] use pre-fixed scales for all data, which cannot adapt to proportional differences across motion-capture subjects and body parts.
- **Second is how the scale is defined:** We directly define the scale on joint links and rescale a joint's offset within the forward kinematics process of a robot, *which is how morphology changes manifest natively on a robot*. While GMR [11] and OmniRetarget [8] define the scale as the scale ratio of the relative position between joints and the robot base. Although ProtoMotions [10] also optimizes scales concurrently with joint angles, it defines pairwise scales between all joint pairs, where scales are over-parameterized as an $N \times N$ matrix and applied *after* forward kinematics. These two scaling definitions do not consider the forward kinematics structure of the robot, and can introduce scaling-induced artifacts, such as foot sliding and ground/object penetration, forcing the IK optimization to compensate through distorted joint configurations or reduced contact feasibility.

Second is a rotation tracking cost, defined as the logarithmic map of the relative rotation between corresponding human and robot joints:

$$E_{rot}(\boldsymbol{x}_{t,i\in\mathcal{C}'_b}) = \rho(\|\log(\boldsymbol{R}_{t,b}^\top \prod_{i\in\mathcal{C}'_b} \boldsymbol{R}_i(\boldsymbol{x}_{t,i}))\|_2^2), \tag{3}$$

where $\boldsymbol{R}_{t,b}$ is the target rotation, $\log(\cdot)$ maps the relative rotation on $SO(3)$ to a rotation vector in $\mathfrak{so}(3)$, and $\mathcal{C}'_b = \mathcal{C}_b \cup \{b\}$ is the forward kinematics chain for rotation, since a joint also contributes to its own global rotation $\prod_{i\in\mathcal{C}'_b} \boldsymbol{R}_i(\boldsymbol{x}_{t,i})$ in addition to its parent joints.

Furthermore, we decouple the human joint target mapping of Eq. (3) from Eq. (2), and do not require them to share the same human joint mapping. For instance, given that the shoulder of Unitree G1 is implemented as a serial chain of single-DoF joints (pitch → roll → yaw):

- For position, we map the left or right human shoulder joints to G1's corresponding `shoulder_roll` joints,

because it is located at the center of the pitch → roll → yaw joint chain, provides a stable and consistent position anchor, and further reduces the morphology mismatch introduced in Sec. I.

- For rotation, we map the human shoulder joints to the `shoulder_yaw` joints, because yaw is the terminal joint of the full shoulder pitch → roll → yaw joint chain, better representing the overall 3-DoF shoulder rotation and reducing the articulation mismatch introduced in Sec. I.

C. Dynamics Regularizations

In addition to the IK costs in Sec. III-B, we incorporate dynamics regularization terms to make a retargeted motion sequence more physically feasible for existing policies to track.

Ground Offset: First is a contact-ground offset regularization that enforces all contacted foot keypoints to lie on the same horizontal ground plane:

$$R_{ground}(\boldsymbol{x}, \boldsymbol{s}) = \sum_{t,k} \left\| c_{t,k} \left(z_k(\boldsymbol{x}_t, \boldsymbol{s}) - \bar{z}(\boldsymbol{x}, \boldsymbol{s}) \right) \right\|_2^2, \quad (4)$$

where we omit the notation of forward kinematics for simplicity and let $z_k(\boldsymbol{x}_t, \boldsymbol{s})$ be the z-coordinate of foot contact keypoint $k \in \{left, right\} \times \{heel, toe\}$ at frame t. The contact confidence $c_{t,k} \in [0, 1]$ is precomputed in the same way as [5], and

$$\bar{z}(\boldsymbol{x}, \boldsymbol{s}) = \frac{\sum_{t,k} c_{t,k}\, z_k(\boldsymbol{x}_t, \boldsymbol{s})}{\sum_{t,k} c_{t,k}} \quad (5)$$

is the ground offset that is calculated as the weighted average of all z-coordinates of foot contact keypoints.

The Eq. (4) minimizes a weighted variance. However, it is computationally expensive, because the error of each contacted heel or toe keypoint depends on $\bar{z}(\boldsymbol{x}, \boldsymbol{s})$ and therefore couples all contacted keypoints together. This dense cross-frame dependency leads to a large dense Hessian matrix for sequences with long contact phases. So we parameterize the ground offset z_g as an auxiliary variable to optimize:

$$R_{ground}(\boldsymbol{x}, \boldsymbol{s}, z_g) = \sum_{t,k \in \mathcal{F}} \left\| c_{t,k} \left(z_k(\boldsymbol{x}_t, \boldsymbol{s}) - z_g \right) \right\|_2^2, \quad (6)$$

which maintains the sparsity of the problem and improves computational efficiency. In addition, for the same $\boldsymbol{x}$ and $\boldsymbol{s}$, the optimal $z_g = \bar{z}(\boldsymbol{x}, \boldsymbol{s})$ in the sense of least squares.

Foot Stance: Second is a foot stance regularization that penalizes horizontal foot slippage during contact:

$$R_{stance}(\boldsymbol{x}, \boldsymbol{s}) = \sum_{t \in \mathcal{P}} \left\| c_{t,k} \left(\boldsymbol{u}_k(\boldsymbol{x}_t, \boldsymbol{s}) - \bar{\boldsymbol{u}}_k(\boldsymbol{x}, \boldsymbol{s}) \right) \right\|_2^2, \quad (7)$$

where $\mathcal{P}$ represents a continuous contact phase, and $\boldsymbol{u}_k(\boldsymbol{x}_t, \boldsymbol{s}) \in \mathbb{R}^2$ is the horizontal (x, y) position of a contacting foot keypoint. Similar to Eq. (4), this variance-based regularization is also computationally expensive for sequences with long contact phases. Nevertheless, the average position $\bar{\boldsymbol{u}}_k(\boldsymbol{x}, \boldsymbol{s})$ is contact-phase specific, and we cannot parameterize it as a global auxiliary variable. Instead, we decompose a continuous contact phase $\mathcal{P}$ into a sequence of sliding windows. Each window contains at most 10 consecutive frames, and when contact persists we advance the window forward by 5 frames every step to maintain overlap between neighboring windows.

Froude Number: Last is the biomechanics-inspired Froude Number regularization. It is a dynamics prior that requires similar locomotion to have similar values of $\mathrm{Fr} = v^2/(g\,\ell)$ [20]. In the case of our motion retargeting, we rescale robot base positions with a scale factor s_g, and require consistency between s_g and the average of the hip–knee and knee–ankle link scales s_{knee} and s_{ankle}:

$$R_{\text{froude}} = \left\| s_g^2 - \frac{1}{2} \sum_{i \in \mathcal{L}} s_i \right\|_2^2 \quad (8)$$

where $\mathcal{L} = \{knee, ankle\}$. The Eq. (8) enforces the Froude Number of a robot to be consistent with the human subject by minimizing the difference between squared base-position scale and the scale of leg links. This Froude Number regularization not only maintains the kinematic similarity of the motion but also adjusts the robot base velocities proportionally to its leg scale, making dynamic motion sequences more feasible to track. It is also a convex quartic function that does not introduce additional non-convexity to the optimization.

D. Initialization

Since the objective function defined above is non-convex, good initialization is critical for faster convergence of the monolithic optimization.

Scale: We initialize each joint-link scale as the ratio between the corresponding human joint-link length and the humanoid joint-link length, both measured in a canonical default pose. We initialize the scale s_j of the link that connects joint j with its parent i as:

$$s_j = \frac{\|\boldsymbol{p}_j - \boldsymbol{p}_i\|}{\|\boldsymbol{q}_j - \boldsymbol{q}_i\|}, \quad (9)$$

where $\boldsymbol{p}_j$ and $\boldsymbol{p}_i$ are human joint positions in the default pose, and $\boldsymbol{q}_j$ and $\boldsymbol{q}_i$ are the corresponding humanoid joint positions in the same default pose. To maintain the left–right symmetry of the humanoid robot, we average the scales of paired left and right links and assign the averaged scales to joints on both sides.

Robot Base Pose: We initialize the humanoid base rotation *exactly* as the human root orientation. For base position, we centralize and rescale the sequence of human root positions by the square root of the average scales of the hip–knee and knee–ankle links:

$$s_g = \sqrt{\frac{1}{2}\left(s_{\text{knee}} + s_{\text{ankle}}\right)}, \quad (10)$$

and set the humanoid base position at each frame as $\boldsymbol{q}_{base} = s_g \boldsymbol{p}_{base}$. This provides good initial base positions and velocities that are consistent with the Froude Number prior.

Joint Positions: We first compute the local 3-DoF rotation of a human joint in its parent's coordinate frame,

and then convert this local rotation into Euler angles as initial joint angles. The rotation-axis order aligns with the robot joint chain. For instance, for the Unitree G1 shoulder, we convert the local rotation of the human shoulder in the coordinate frame of the spine joint into Euler angles in the order pitch→roll→yaw, and initialize the humanoid `shoulder_pitch`, `shoulder_roll`, and `shoulder_yaw` joints accordingly. If the DoF of a joint chain is less than 3, we project the 3-DoF rotation onto the corresponding lower-DoF subspace. This approach provides an initial solution that is articulatorily close to human motion sequences before optimization.

Ground offset: After initializing the base pose and joint angles, we perform a single forward kinematics pass, obtain the vertical positions of all contacted foot toe and heel keypoints, and initialize the ground height as the weighted average of contact keypoints as in Eq. (5). In practice, these initializations significantly reduce the number of iterations required by the monolithic optimization and improve robustness against local minima, especially for long sequences.

E. Implementation Details

We implement our HM^{2}O pipeline in Python with the following details:

MuJoCo: We use MuJoCo to load the robot model and access standard robot utilities such as the kinematic tree structure and body/site definitions. However, MuJoCo's native forward kinematics assumes fixed link geometry. To enable joint optimization with our joint-link scales, we reimplement forward kinematics and Jacobian evaluations in Python.

PyCeres: We solve our monolithic optimization problem using `pyceres`, a Python wrapper for Ceres-Solver. Ceres-Solver is a widely used C++ library for large-scale nonlinear least-squares problems, offering robust loss functions, mature trust-region solvers (e.g., Levenberg–Marquardt and DogLeg), and strong support for sparse, block-structured Jacobians and manifold parameterizations (e.g., rotations). These features match our setting with per-frame variables and a small set of sequence-shared parameters. Since our normal equations are highly sparse, we use `SPARSE_NORMAL_CHOLESKY` as the linear solver option.

Residuals and Jacobians: We implement each cost and regularization term in PyCeres as a residual block. We manually derive and implement analytic Jacobians for all residual blocks, including derivatives with respect to both robot state variables and link-scale parameters. Since PyCeres does not currently support auto-differentiation, and analytic Jacobians are more efficient and stable, especially for the $SO(3)$ rotation manifold.

IV. Experiments

A. Experimental Setup

Datasets: In this work, we focus on humanoid locomotion retargeting from human motion datasets, specifically AMASS [4] and LAFAN1 [6]. AMASS is a large-scale mocap dataset that standardizes 15 public datasets with disparate capture settings, while LAFAN1 is smaller in scale and curated from high-quality professional animation data. Due to the limited space, we only demonstrate results on AMASS in the paper. *Please refer to the supplementary video for results on LAFAN1.*

Motion Tracking Policy: To maintain the reproducibility of our comparison, we leverage off-the-shelf, general motion tracking policies from their official repositories, including MaskedMimic [2], TWIST [14], and GMT [15]. This strategy provides a comprehensive assessment of how diverse policies track motion trajectories retargeted by different methods, and reduces potential bias from a specific policy and training recipe.

Retargeting Methods: We compare our HM^{2}O with recent representative retargeting methods. These baseline methods can be categorized based on whether their optimizer is first-order [1, 5] or second-order [10], and whether the optimization is frame-by-frame [11, 8] or at the sequence level [1, 5, 10]. They also address the human-to-humanoid gaps differently, as introduced in Sec. I and Sec. II.

B. Qualitative Results

We show the retargeted reference motions and the corresponding rollouts of off-the-shelf tracking policies side-by-side to qualitatively compare (i) kinematic alignment and (ii) dynamic feasibility on highly dynamic motions. *Please refer to the supplementary video for more comparisons.*

Kinematics Alignment: As introduced in Sec. I and Sec. II,

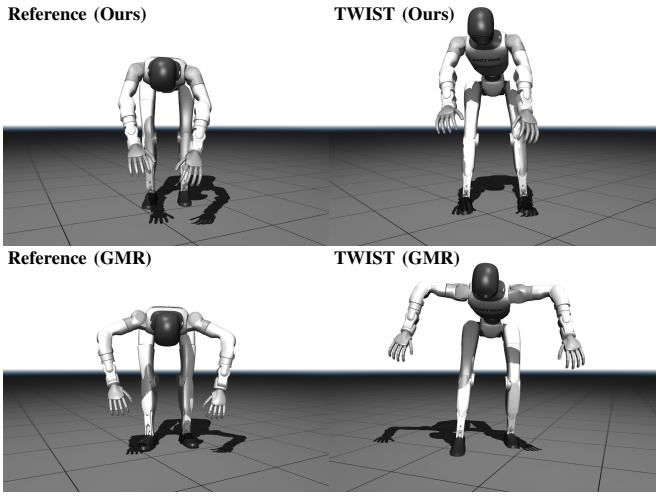

Fig. 1. GMR (manually defined scales) v.s. Ours

methods with pre-fixed scaling [11, 8] can produce unnatural postures because a single set of pre-fixed scales cannot account for proportional differences across subjects and motions, while fitting the shape parameters of the SMPL model [12, 1, 5] can introduce artifacts including joint-limit violations and implausible ground interactions. As shown in Fig. 1, GMR with pre-fixed scales produces distorted whole-body postures. Fig. 2 shows that PHUMA fails to track the motion of lifting a box from the ground, since shape fitting can distort the human joint positions. In contrast, our HM^{2}O improves morphology alignment with human

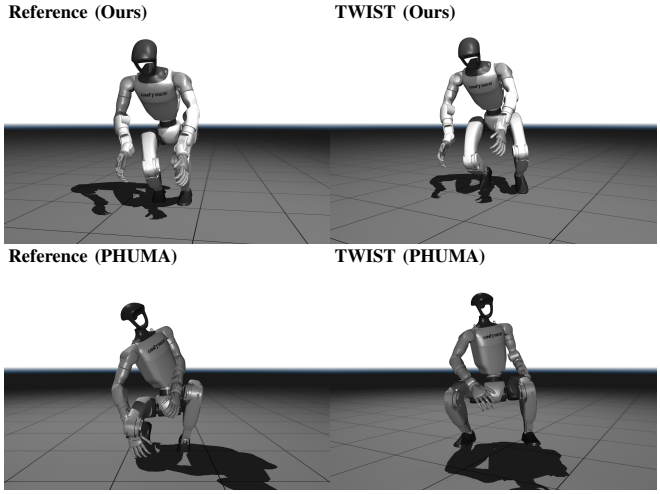

Fig. 2. PHUMA (Shape Fitting) v.s. Ours

targets by optimizing joint-link scales within the robot's forward kinematics, as outlined in Sec. III-B.

Dynamics Feasibility: In addition to better kinematic alignment, our HM^{2}O framework also generates motion trajectories that are more physically feasible for a locomotion policy to track, especially on highly dynamic sequences. We show two dynamic sequences where Fig. 3 demonstrates a running motion that GMT can successfully track when using the reference motion retargeted by our HM^{2}O, but fails when tracking the motion retargeted by PHUMA [5]. This is because most prior methods do not adjust robot base velocities proportionally to leg scale; as a result, the base velocities of retargeted motions can exceed the maximum speed that a robot can track. In contrast, our HM^{2}O rescales the robot base positions and velocities and satisfies the Froude-number prior introduced in Sec. III-C.

As shown in Fig. 4, Our method can even enable some sequences that fails and are filtered by SMPL robot policy [2], which is generally considered an upper bound for policy performance, since there is no kinematics gap. The reason for this advantage is because the motion can still be dynamically infeasible for tracking policies due to inconsistent contacts and overly aggressive base velocities/accelerations. As outlined in Sec. III-C, our HM^{2}O explicitly stabilizes contacts and further rescales base motion/velocity consistently with leg scale to satisfies Froude Number prior, making these high-dynamic references substantially more trackable.

C. Quantitative Results

Metrics: We follow standard evaluation protocol [11] to evaluate our method and other baselines, where we only evaluate AMASS motion sequences after adopting the filtering process in [14], with the following metrics:

- $\mathbf{E_{g\text{-}mpbpe}}$ is the average joint position error in the world coordinate frame and it captures both root base drift and pose errors.
- $\mathbf{E_{mpbpe}}$ is the average joint position error in the local coordinate frame of robot base and it is less sensitive to global transformation errors than $\mathbf{E_{g\text{-}mpbpe}}$.

Policy	Method	Success Rate ↑	$\mathbf{E_{g\text{-}mpbpe}}$ (cm) ↓	$\mathbf{E_{mpbpe}}$ (cm) ↓	$\mathbf{E_{mpjpe}}$ (°) ↓
MaskedMimic	PHC	71.5	17.51	3.58	14.59
	PHUMA	82.47	15.72	3.08	11.32
	GMR	73.18	14.29	3.65	13.80
	ProtoMotions*	**89.75**	14.04	**2.69**	**8.57**
	HM^{2}O (Ours)	86.73	**11.74**	3.05	11.50
TWIST	PHC*	82.23	17.86	2.10	8.00
	PHUMA	77.66	23.04	2.16	6.15
	GMR	59.25	31.09	2.84	7.70
	ProtoMotions	76.88	23.16	2.25	5.96
	HM^{2}O (Ours)	**88.73**	**17.30**	**1.64**	**5.19**
GMT	PHC	80.24	14.68	1.85	6.96
	PHUMA	83.03	15.59	1.84	5.54
	GMR	76.16	17.70	2.70	7.23
	ProtoMotions	76.94	18.57	2.18	6.17
	HM^{2}O (Ours)	**90.45**	**12.67**	**1.49**	**5.21**

TABLE I

TRACKING ERRORS OF ALL POLICY × RETARGETING COMBINATIONS. * DENOTES THAT A POLICY **was trained** ON DATA RETARGETED BY A SPECIFIC METHOD.

Method	OmniRetarget	PHUMA	PHC	HM^{2}O (Ours)	ProtoMotions	GMR
FPS	1.10	6.17	12.02	27.25	28.23	42.77

TABLE II

FRAME RATE PER SECOND OF EACH RETARGETING METHOD.

- $\mathbf{E_{mpjpe}}$ is the average angular error of joint rotations, which is as important as the position metrics especially for the end effectors of humanoids.
- **Success rate** is the ratio between the number of successful policy rollouts and the total number of rollouts. In addition to the termination condition, similar to [13], we also consider the motion tracking of a sequence as a failure,if its $E_{\text{g-mpbpe}}$ is larger than 50cm or E_{mpjpe} is larger than 20°.

Performance: Tab. I summarizes the quantitative evaluation metrics. Our proposed HM^{2}O achieves state-of-the-art performance across all metrics when evaluated with TWIST [14] and GMT [15]. For instance, HM^{2}O improves the success rate of GMT to 90.45% while reducing $E_{g\text{-}mpbpe}$ to 12.67cm compared to other baselines. This superior tracking performance demonstrates that the proposed method outlined in Sec. III effectively minimizes both kinematics and dynamics gaps simultaneously.

The only exception is observed with MaskedMimic [2], where ProtoMotions achieves a higher success rate (89.75% vs. 86.73%). *This discrepancy is expected*, because MaskedMimic was trained on and thus biased toward motion references generated by ProtoMotions [10]. Even with this bias, HM^{2}O remains highly competitive, maintaining a comparable success rate and outperforming ProtoMotions on $E_{g\text{-}mpbpe}$. This suggests that even when a downstream policy is tuned to a different retargeting framework, HM^{2}O generates more physically feasible motion references that are easier to track.

Efficiency: Tab. II summarizes the frame rate of each retargeting method, which measures efficiency and indicates whether a method can process human motion datasets at scale, and potentially be extended to real-time teleoperation.

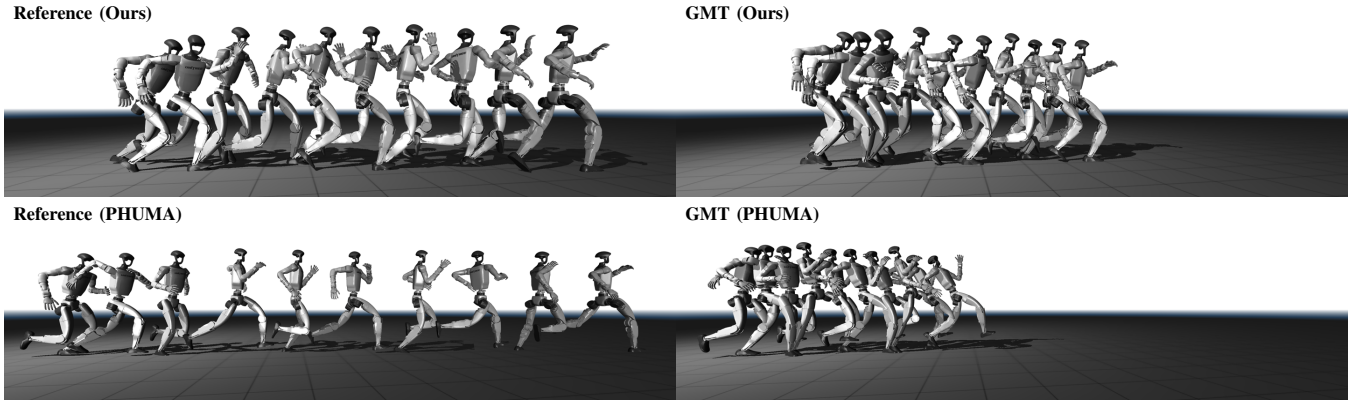

Fig. 3. PHUMA v.s. Ours on high dynamic sequence.

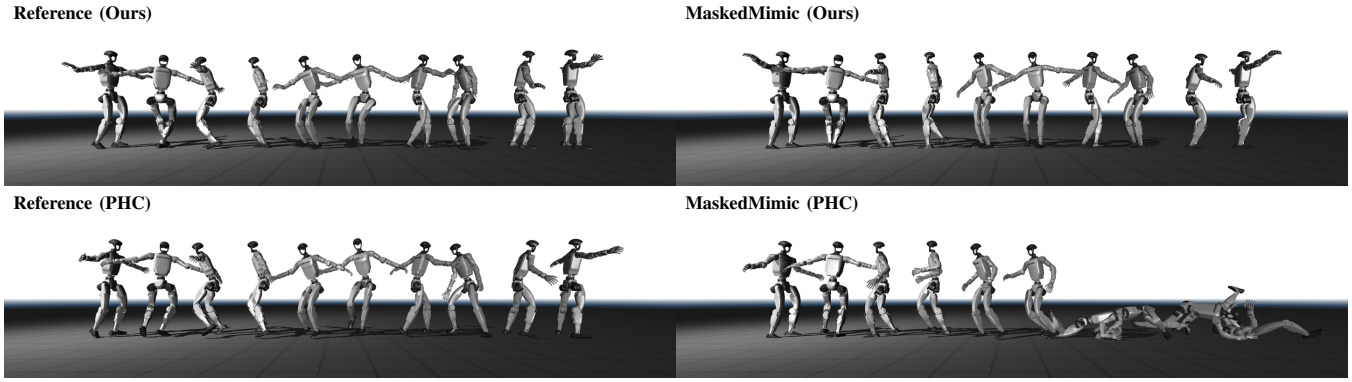

Fig. 4. PHC v.s. Ours on contact switching sequence.

Specifically,

- PHC [1] and PHUMA [5] use the first-order Adam(W) optimizer [25, 26] for both shape fitting and motion retargeting, which is inefficient because first-order optimizers typically require many more iterations to converge than second-order optimizers. ProtoMotions [10] has near real-time efficiency since it uses a second-order optimizer implemented in PyRoki [27] along with JIT compilation of JAX on GPU.
- GMR has the highest FPS since it only considers inverse kinematics objectives in a frame-by-frame sequential optimization: the solution at each frame is initialized from the previous frame and then refined for up to 20 iterations. Although OmniRetarget also adopts a sequential, quadratic-programming, frame-by-frame formulation, it has the lowest FPS among the baselines. The reason is two-fold: first, it solves the problem with inequality constraints, which is more complicated than the unconstrained problems in the other methods; second, it uses a general-purpose optimizer from CVXPY [28], which is a less optimized implementation and does not leverage GPU acceleration. This inefficiency makes us to exclude OmniRetarget in our quantitative comparisons on large-scale AMASS dataset, but *only qualitatively in the supplementary video.*

In comparison, our method maintains real-time efficiency using only a single CPU core. This performance is mainly enabled by the initialization strategy detailed in Sec. III-D and the efficient second-order optimizer described in Sec. III-E. Furthermore, this computational efficiency demonstrates the potential of HM^{2}O for online applications, such as teleoperation. Future work could transition this approach to an online setting without performance degradation by augmenting the optimization with sliding-window and marginalization techniques that are commonly used in modern Simultaneous Localization and Mapping (SLAM) systems [29, 30].

V. Conclusion

In this work, we address a key limitation in humanoid motion retargeting: multi-stage pipelines (shape fitting/scaling $\rightarrow$ IK $\rightarrow$ contact correction) can accumulate errors across stages and produce motion references that are infeasible for downstream control policies to track. To this end, we introduce HM^{2}O , a monolithic joint optimization framework that minimizes kinematics and dynamics gaps simultaneously by optimizing robot joint-link scales together with base pose and joint angles inside robot forward kinematics, decoupling position and rotation targets with flexible joint mappings, and incorporating efficient dynamics regularizations for stable contacts and dynamically consistent base motion.

Our experiments show that HM^{2}O consistently improves downstream tracking performance under off-the-shelf motion tracking policies, achieving state-of-the-art results with di-

verse motion tracking policies (e.g., improving the GMT success rate to 90.45% while reducing $E_{g\text{-}mpbpe}$ to 12.67 cm). Moreover, even when evaluated with MaskedMimic—whose training data is biased toward ProtoMotions references—HM^{2}O remains highly competitive and improves global tracking errors, suggesting that it generates more physically feasible motion references that are easier to track. Finally, HM^{2}O maintains real-time efficiency using only a single CPU core, indicating its potential for scalable dataset processing and online applications such as teleoperation.

Looking forward, future work includes transitioning the optimization to an online setting without performance degradation by adopting sliding-window optimization and marginalization techniques commonly used in modern SLAM systems, and further exploiting hard physics constraints to extend HM^{2}O from locomotion to loco-manipulation.

REFERENCES

[1] Z. Luo, J. Cao, K. Kitani, W. Xu *et al.*, "Perpetual humanoid control for real-time simulated avatars," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 10 895–10 904.

[2] C. Tessler, Y. Guo, O. Nabati, G. Chechik, and X. B. Peng, "Maskedmimic: Unified physics-based character control through masked motion inpainting," *ACM Transactions on Graphics (TOG)*, vol. 43, no. 6, pp. 1–21, 2024.

[3] T. E. Truong, Q. Liao, X. Huang, G. Tevet, C. K. Liu, and K. Sreenath, "Beyondmimic: From motion tracking to versatile humanoid control via guided diffusion," *arXiv preprint arXiv:2508.08241*, 2025.

[4] N. Mahmood, N. Ghorbani, N. F. Troje, G. Pons-Moll, and M. J. Black, "Amass: Archive of motion capture as surface shapes," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 5442–5451.

[5] K. Lee, S. Kim, M. Park, H. Kim, D. Hwang, H. Lee, and J. Choo, "Phuma: Physically-grounded humanoid locomotion dataset," *arXiv preprint arXiv:2510.26236*, 2025.

[6] F. G. Harvey, M. Yurick, D. Nowrouzezahrai, and C. Pal, "Robust motion in-betweening," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, 2020.

[7] J. Li, J. Wu, and C. K. Liu, "Object motion guided human motion synthesis," *ACM Trans. Graph.*, vol. 42, no. 6, 2023.

[8] L. Yang, X. Huang, Z. Wu, A. Kanazawa, P. Abbeel, C. Sferrazza, C. K. Liu, R. Duan, and G. Shi, "Omniretarget: Interaction-preserving data generation for humanoid whole-body loco-manipulation and scene interaction," 2025. [Online]. Available: https://arxiv.org/abs/2509.26633

[9] C. Pan, C. Wang, H. Qi, Z. Liu, H. Bharadhwaj, A. Sharma, T. Wu, G. Shi, J. Malik, and F. Hogan, "Spider: Scalable physics-informed dexterous retargeting," 2025. [Online]. Available: https://arxiv.org/abs/2511.09484

[10] C. Tessler*, Y. Jiang*, X. B. Peng, E. Coumans, Y. Shi, H. Zhang, D. Rempe, G. Chechik†, and S. Fidler†, "Protomotions3: An open-source framework for humanoid simulation and control," https://github.com/NVLabs/ProtoMotions/, 2025.

[11] J. P. Araujo, Y. Ze, P. Xu, J. Wu, and C. K. Liu, "Retargeting matters: General motion retargeting for humanoid motion tracking," *arXiv preprint arXiv:2510.02252*, 2025.

[12] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "SMPL: A skinned multi-person linear model," *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, vol. 34, no. 6, pp. 248:1–248:16, 2015.

[13] T. He, Z. Luo, W. Xiao, C. Zhang, K. Kitani, C. Liu, and G. Shi, "Learning human-to-humanoid real-time whole-body teleoperation," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 8944–8951.

[14] Y. Ze, Z. Chen, J. P. Araújo, Z.-a. Cao, X. B. Peng, J. Wu, and C. K. Liu, "Twist: Teleoperated whole-body imitation system," *CoRL*, 2025.

[15] Z. Chen, M. Ji, X. Cheng, X. Peng, X. B. Peng, and X. Wang, "Gmt: General motion tracking for humanoid whole-body control," *arXiv:2506.14770*, 2025.

[16] J. Koenemann, F. Burget, and M. Bennewitz, "Real-time imitation of human whole-body motions by humanoids," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 2806–2812.

[17] A. Di Fava, K. Bouyarmane, K. Chappellet, E. Ruffaldi, and A. Kheddar, "Multi-contact motion retargeting from human to humanoid robot," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2016, pp. 1081–1086.

[18] L. Penco, B. Clément, V. Modugno, E. M. Hoffman, G. Nava, D. Pucci, N. G. Tsagarakis, J.-B. Mouret, and S. Ivaldi, "Robust real-time whole-body motion retargeting from human to humanoid," in *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2018, pp. 425–432.

[19] K. Darvish, Y. Tirupachuri, G. Romualdi, L. Rapetti, D. Ferigo, F. J. A. Chavez, and D. Pucci, "Whole-body geometric retargeting for humanoid robots," in *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2019, pp. 679–686.

[20] R. M. Alexander, "Optimization and gaits in the locomotion of vertebrates," *Physiological reviews*, vol. 69, no. 4, pp. 1199–1227, 1989.

[21] X. B. Peng, P. Abbeel, S. Levine, and M. Van de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Transactions On Graphics (TOG)*, 2018.

[22] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, "Amp: Adversarial motion priors for stylized physics-based character control," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–20, 2021.

[23] X. B. Peng, Y. Guo, L. Halper, S. Levine, and S. Fidler, "Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters," *ACM Transactions on Graphics (TOG)*, vol. 41, no. 4, pp. 1–17, 2022.

[24] Z. Chen, M. Ji, X. Cheng, X. Peng, X. B. Peng, and X. Wang, "Gmt: General motion tracking for humanoid whole-body control," *arXiv:2506.14770*, 2025.

[25] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[26] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019. [Online]. Available: https://openreview.net/forum?id=Bkg6RiCqY7

[27] C. M. Kim, B. Yi, H. Choi, Y. Ma, K. Goldberg, and A. Kanazawa, "Pyroki: A modular toolkit for robot kinematic optimization," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025. [Online]. Available: https://arxiv.org/abs/2505.03728

[28] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[29] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 611–625, 2018.

[30] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.