This ICCV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

STaR: Seamless Spatial-Temporal Aware Motion Retargeting with Penetration and Consistency Constraints

Xiaohang Yang Qing Wang Jiahao Yang Gregory Slabaugh Shanxin Yuan
Queen Mary University of London, United Kingdom
{xiaohang.yang, qing.wang, jiahao.yang, g.slabaugh, shanxin.yuan}@qmul.ac.uk

Abstract

*Motion retargeting seeks to faithfully replicate the spatio-temporal motion characteristics of a source character onto a target character with a different body shape. Apart from motion semantics preservation, ensuring geometric plausibility and maintaining temporal consistency are also crucial for effective motion retargeting. However, many existing methods prioritize either geometric plausibility or temporal consistency. Neglecting geometric plausibility results in interpenetration, while neglecting temporal consistency leads to motion jitter. In this paper, we propose a novel sequence-to-sequence model for seamless **S**patial-**T**emporal **a**ware motion **R**etargeting (**STaR**), with penetration and consistency constraints. STaR consists of two modules: (1) a spatial module that incorporates dense shape representation and a novel limb penetration constraint to ensure geometric plausibility while preserving motion semantics, and (2) a temporal module that utilizes a temporal transformer and a novel temporal consistency constraint to predict the entire motion sequence at once while enforcing multi-level trajectory smoothness. The seamless combination of the two modules helps us achieve a good balance between the semantic, geometric, and temporal targets. Extensive experiments on the Mixamo and ScanRet datasets demonstrate that our method produces plausible and coherent motions while significantly reducing interpenetration rates compared with other approaches. Code page: https://github.com/XiaohangYang829/STaR.*

1. Introduction

Motion retargeting is a fundamental task in computer vision and graphics. It involves transferring motion from one character to another—preserving the source character's dynamics and style while adapting to the target character's unique structure and geometric constraints. It is widely employed in film production, gaming, and robotic animation. With the rise of digital avatars in virtual environments, rapid advancements in motion generation and digital avatars have further intensified research interest in motion retargeting, which bridges motion capture and character animation. As the demand for authentic visual effects grows, expectations for generated motion have become more stringent. Issues like geometric penetration and unnatural temporal jitter can significantly undermine realism.

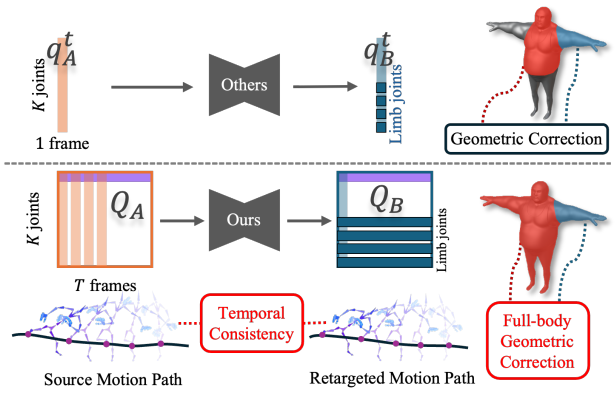


Figure 1. STaR performs sequence-to-sequence motion retargeting, guided by a novel limb penetration constraint and a temporal consistency constraint. Unlike existing methods that focus on limb-body geometric correction, our limb penetration constraint provides full-body geometric correction. The temporal constraint module enhances the smoothness of the entire motion trajectory. STaR effectively reduces the penetration rate while preserving motion semantics and temporal coherence.

Motion inherently carries information in both its spatial and temporal dimensions. The spatial relationships among skeletal joints in each motion frame convey critical information about pose semantics, while their temporal evolution defines the motion trajectory. However, many existing methods primarily focus on the spatial domain, treating motion retargeting as a pose transfer problem. Previous approaches [1, 11, 16] generally optimize sparse skeletal rotations and use Forward Kinematics (FK) to derive joint

For the purpose of open access, the author has applied a Creative Commons Attribution (CC-BY) license to any Author Accepted Manuscript version arising.

positions, ensuring constraints on both. More recent models [12, 17–20] extend this by incorporating skinned motion representations to better preserve semantics and mitigate penetration. Although some methods [1, 11, 16, 19, 20] introduce simple smoothness losses between consecutive frames to improve temporal consistency, temporal glitches persist. Furthermore, neglecting temporal consistency when handling penetration [20] can further disrupt motion trajectories. As illustrated in Fig. 1, existing methods primarily perform motion retargeting frame-by-frame, often overlooking either geometric correction or temporal consistency. We propose to ensure geometric plausibility and temporal consistency across the whole sequence through a sequence-to-sequence design.

To minimize penetration and mitigate temporal artifacts in retargeted motion, we propose a novel motion retargeting network that integrates both spatial and temporal supervision, which consists of two key modules: (1) a spatial module that processes skeletal rotations across all joints at each time step to capture motion semantics and geometric cues; (2) a temporal module that models motion dynamics to ensure smooth and coherent motion transitions over time.

In the spatial module, we first design dense shape representations to align the input with dense spatial supervisions, such as penetration loss [19, 20]. Instead of solely relying on sparse joint rotations, we densify the input by sampling vertices from both the source and target characters. Although existing SDF-based penetration loss can effectively improve geometric plausibility, it presents challenges during training. Since each pose corresponds to a unique SDF, obtaining accurate dynamic SDFs is computationally demanding. To address this, we introduce the limb penetration constraint module, which considers both limb-body penetration and limb-limb penetration. Inspired by the distance function used in point cloud registration [4], we calculate the SDF value for each limb vertex based on the Chamfer Distance. We implemented the limb penetration loss with CUDA [13], allowing us to penalize penetrated vertices efficiently.

To enhance temporal smoothness, we integrate multi-level temporal supervision derived from the source motion. The short-term supervision constrains frame-wise motion directions and magnitudes, while the long-term supervision maintains the overall structure of the motion trajectory. This multi-level temporal guidance helps preserve motion smoothness and coherence. To evaluate STaR's performance, we conduct extensive experiments across various scenarios, including unseen motion and characters, and the motion from real humans. Both qualitative and quantitative analyses demonstrate that our method outperforms existing approaches, achieving more natural, coherent, and physically plausible results.

Our contributions are summarized as follows:

- We introduce a novel spatio-temporal network for motion retargeting that concurrently processes pose spatial information and temporal dynamics.
- We apply dense shape representations and propose limb penetration loss to avoid penetration while preserving motion semantics.
- We develop a temporal consistency loss to ensure motion smoothness, which effectively reduces motion jitter.
- We validate our method through comprehensive experiments on Mixamo [2] and ScanRet [18] datasets.

2. Related Work

Optimization-based Motion Retargeting. Early studies [3, 5, 6, 9, 14] explored optimization-based motion retargeting using manually designed kinematic constraints and assumptions. Gleicher [6] introduces a space-time constraint solver to preserve key motion features while adapting movement to characters with different proportions. Lee and Shin [9] combine hierarchical curve fitting with an inverse kinematics solver to smoothly refine motion using multilevel B splines. Choi and Ko [3] develop a real-time motion adaptation technique that dynamically adjusts joint angles for interactive applications. Unlike traditional spacetime optimization, Tak and Ko [14] propose a per-frame Kalman filter framework for stable, physically plausible motion retargeting. Feng *et al*. [5] automate skeleton mapping and motion retargeting, enabling fast character integration. While effective in controlled scenarios, they are constrained by limited training data and simplified motion models.

Neural Motion Retargeting. With the advent of large motion datasets and deep learning, neural motion retargeting methods have gained popularity in recent years. Villegas *et al*. [16] introduce a recurrent neural network with a Forward Kinematics (FK) layer and a cycle consistency-based adversarial training objective to perform unsupervised motion retargeting. Lim *et al*. [11] separate pose retargeting and movement adaptation into two distinct processes, and achieve natural motion transfer. Aberman *et al*. [1] enable unpaired motion retargeting across skeletons with different structures by embedding motion into a common latent space using differentiable skeleton-aware convolution, pooling, and unpooling operators. Li *et al*. [10] propose a deep learning-based iterative method for motion retargeting that avoids adversarial training and instead uses an autoencoder-based latent space optimization to improve generalization to unseen characters and motions. All these methods [1, 10, 11, 16] above are based on the skeletal motion, while [17–20] consider body geometry. Villegas *et al*. [17] employ a geometry-conditioned recurrent network and encoder-space optimization to preserve self-contacts and prevent interpenetration. Zhang *et al*. [20] introduce a residual-based neural retargeting framework that progressively adjusts source motions to fit target skeletons

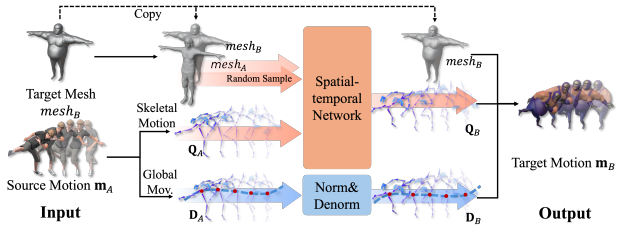


Figure 2. Overview of STaR, which splits the input motion into two modalities: skeletal motion sequence, and global movements. STaR also takes the meshes of the source and target character as input when retargeting the motion.

and shapes, and use distance fields to avoid interpenetration and ensure self-contact. Zhang *et al*. [19] design a motion retargeting framework based on Vision Language Model that extracts and preserves high-level motion semantics using differentiable rendering. Martinelli *et al*. [12] use a transformer-based masked autoencoder for skeletal adaptation and a face-based optimization approach to prevent mesh interpenetration. Zhang *et al*. [18] directly models dense mesh interactions using semantically consistent sensors (SCS) and a Dense Mesh Interaction (DMI) field, enabling accurate motion transfer. Based on these works, our method further enhances the physical plausibility and simultaneously improves the motion smoothness to a level nearly indistinguishable from the source.

3. Methods

The overall framework of STaR is introduced in Sec. 3.1. Since motion retargeting lacks paired ground truth, we leverage additional supervision signals from the source motion sequences: a limb penetration constraint module (Sec. 3.2) to avoid interpenetration and a temporal consistency module (Sec. 3.3) to maintain motion coherence. The overall loss function, integrating these constraints with semantic supervision, is described in Sec. 3.4.

3.1. Spatio-Temporal Aware Motion Retargeting

As shown in Fig. 2, given a source motion $\mathbf{m}_A$ from the source character A and the T-posed meshes of the source character $mesh_A$, and target character $mesh_B$, the goal is to generate the target motion $\mathbf{m}_B$ while preserving motion semantics, geometric plausibility, and temporal consistency. To achieve this, we decompose the input $\mathbf{m}_A$ into skeletal motion, represented by rotation quaternions $\mathbf{Q} \in \mathbb{R}^{T\times K\times 4}$, and global movement $\mathbf{D} \in \mathbb{R}^{T\times 4}$, capturing both local skeletal rotations and the root joint's trajectory. Here, T denotes the length of the motion sequence, and K represents the number of skeletal joints. Because characters vary in height, we compensate for this by normalizing global motion, then restoring its original scale afterward.

At each iteration, we randomly sample a subset of points from $mesh_A$ and $mesh_B$, resulting in two point sets $\mathbf{P}_A$ and $\mathbf{P}_B$. These point sets ($\mathbf{P}_A$ and $\mathbf{P}_B$), along with the skeletal motion ($\mathbf{Q}_A$), are fed into the spatio-temporal model, which predicts a residual motion adjustment ($\Delta\mathbf{Q}$) that aligns the retargeted motion with the target character's structure. The complete framework can be formulated as:

$$\Delta\mathbf{Q} = \mathcal{F}(\mathbf{Q}_A, \mathbf{P}_A, \mathbf{P}_B),\ \mathbf{Q}_B = \Delta\mathbf{Q} \otimes \mathbf{Q}_A, \quad (1)$$

where $\mathcal{F}$ represents the spatio-temporal model.

Motion Encoder and Shape Encoder. As shown in Fig. 3 (a), like most retargeting methods [11, 19, 20], we utilize a motion encoder to process the skeletal motion $\mathbf{Q}_A$ to motion feature $\mathbf{f}^q \in \mathbb{R}^{T\times K\times C}$, where C is the latent length. However, this sparse input cannot provide dense information for motion retargeting when there are dense supervision signals applied.

To address this, we incorporate a frozen off-the-shelf point cloud network [7] as a geometric feature extractor. Since most point cloud learning methods require a fixed number of input points, we randomly sample point sets from the T-posed meshes of both source and target characters at each iteration. Because the sampled points vary in each iteration, the network is exposed to different geometric subsets over time, forcing it to develop a more robust shape awareness rather than overfitting to specific sampled regions. This strategy enhances the network's ability to generalize across different body shapes while ensuring a consistent understanding of detailed geometry.

The output of the dense shape encoder is further processed by a small multi-layer perceptron (MLP), which takes the logits from the point cloud transformer as input and generates a shape feature $\mathbf{f}^s \in \mathbb{R}^{T\times K\times C}$. This feature is designed to align with the representation produced by the motion encoder, ensuring compatibility between shape and motion features. It is particularly beneficial for subsequent spatio-temporal operations, as it enables smooth information exchange across both spatial and temporal dimensions, improving the model's ability to learn coherent motion transformations.

Spatio-Temporal Network. The motion feature $\mathbf{f}^q$ from the motion encoder and the dense shape representations $\mathbf{f}^s_A$ and $\mathbf{f}^s_B$ are first concatenated along the latent dimension, forming a unified representation of size $\mathbb{R}^{T\times K\times(3*C)}$. This combined feature is then processed by the spatial transformer, which captures spatial relationships between skeletal joints at each time step, ensuring structural consistency and geometric plausibility.

After spatial processing, we apply a permutation operation, swapping the temporal and spatial dimensions to reshape the input from $\mathbb{R}^{T\times K\times(3*C)}$ to $\mathbb{R}^{K\times T\times(3*C)}$. This step enables the temporal transformer to model motion evolution for each joint individually, focusing on temporal con-

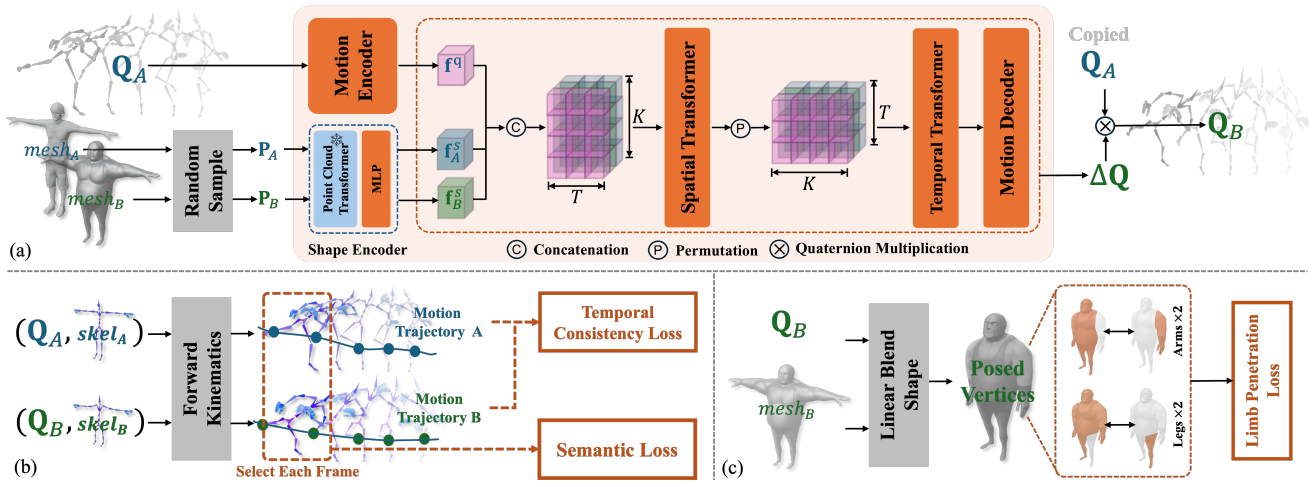


Figure 3. Details of STaR. (a) The detailed architecture of the spatio-temporal model. (b) The constraints on skeletal motion include temporal consistency loss and semantic loss. (c) The limb penetration loss is applied to the skinned motion.

sistency. For each joint, the temporal transformer extracts temporal attention across its motion trajectory, capturing both local smoothness and long-term dependencies. This ensures coherent motion transitions while maintaining temporal consistency throughout the sequence. Finally, the motion decoder predicts a residual motion correction, which is applied to the source motion $\mathbf{Q}_A$ to obtain the retargeted skeletal motion $\mathbf{Q}_B$.

3.2. Limb Penetration Constraint Module

To prevent penetration, we propose *limb penetration loss*. As illustrated in Fig. 4 (a), this loss is designed based on the observation that penetration occurs not only between the limbs and the main body [20], but also *between* the limbs. To mitigate penetration, we penalize each limb's penetration with all other body parts, as illustrated in Fig. 3 (c).

The calculation of the *limb penetration loss* is inspired by Chamfer Distance and point cloud registration [4], which are based on measuring the distance between two point sets. As clarified in Fig. 4, the loss calculation follows these steps (1) deforming the T-posed vertices and normal vectors of the target character according to Q_B using Linear Blend Skinning (LBS) to obtain posed vertices for the limb and the rest of the body, serving as query and reference vertices, respectively; (2) computing the Chamfer Distance between the query and reference vertices; (3) identifying the nearest reference vertex for each query vertex; and (4) multiplying the vector from the query vertex to its nearest reference vertex by the deformed normal vector of the reference vertex, yielding the penetration loss for that query vertex. Specifically, for each query vertex $\mathbf{e}$ located on the limbs, the nearest reference vertex $\mathbf{e}^r$ with normal vector $\mathbf{n}^r$ is identified using a Nearest Neighbor Search (NNS) algorithm. The SDF value for the query vertex $\phi(\mathbf{e})$ is calculated by the dot product $\mathbf{v}^r \cdot \mathbf{n}^r$, where $\mathbf{v}^r = \mathbf{e} - \mathbf{e}^r$. The overall penetration penalty $\mathcal{L}_{lp}$ is defined as follows:

$$\mathcal{L}_{lp} = \frac{1}{\sum_l N_l} \sum_l \sum_{\mathbf{e} \in \mathbf{E}_l} ReLU\left(\phi(\mathbf{e})\right), \tag{2}$$

where l is one of the four limbs (right arm, left arm, right leg, left leg), N_l is the vertex number for each limb, and the vertex set $\mathbf{E}_l = \{\mathbf{e}_i\}_{i=1}^{N_l}$. The reference vertex set is $\mathbf{E}_l^r = \{\mathbf{e}_j^r\}_{j=1}^{N_l^r}$. Sampling the same points for all characters within a batch allows us to process the limb penetration loss for the entire batch simultaneously. In our model, we assume that the SDF value is positive when a vertex is inside the body. We apply the $ReLU$ function to retain all positive SDF values, ensuring that only penetrating vertices contribute to the penetration loss. Additionally, since the LBS algorithm introduces partial interpenetration, we exclude certain body parts, such as the upper arm, from the loss computation.

Please refer to Sec. A in the supplemental materials for the detailed image, which displays the conditions of both penetrating vertices and non-penetrating ones.

3.3. Temporal Consistency Constraint Module

As illustrated in Fig. 5 (a), the motion path may intersect with the body, causing penetration. When applying SDF loss, the spatial module often truncates the motion path to prevent penetration, disrupting its smoothness and contributing to temporal jitter. Ideally, instead of abrupt truncation, the motion path should be warped adaptively based on penetration conditions along its trajectory—even in regions where no penetration occurs. To achieve this, we introduce a *temporal transformer* [15] to enable information exchange across different time frames. Additionally, we propose a multi-level *temporal consistency loss* that constrains both

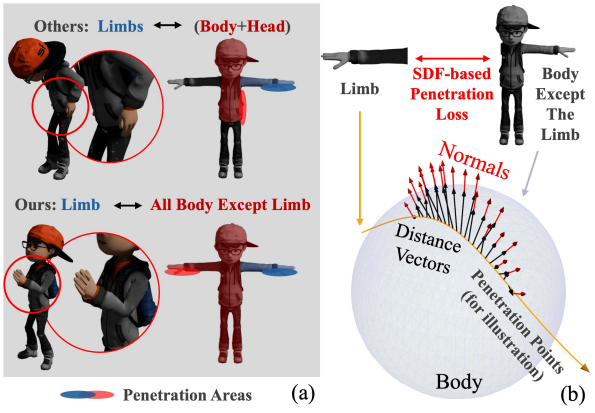


Figure 4. Illustration of different types of interpenetration. Existing methods [20] focus on the interpenetration between limbs and body or head. While for motions like clapping, two hands tend to penetrate each other. We design a novel normal-based SDF loss to include all kinds of interpenetration.

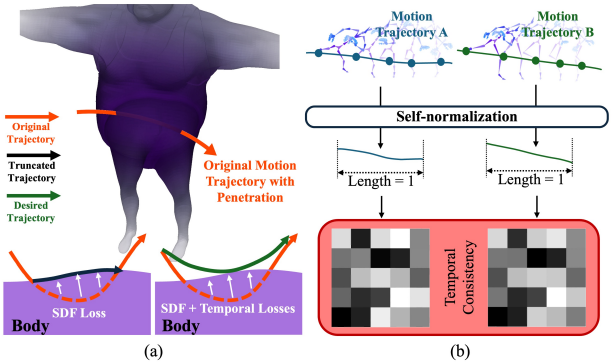


Figure 5. (a): Geometric correction for motion path with penetration will lead to motion jitter. Motion smoothness should be retained while avoiding penetration. The green curve is the desired motion path, balancing penetration avoidance and motion coherence. (b): Illustration of the temporal consistency. The motion paths from different characters are self-normalized to eliminate the effect of different skeleton structures.

short-term motion vectors and long-term motion trajectories, ensuring a smoother and more coherent motion path.

Simple smoothness losses used by other works focus on consecutive frames, whose receptive field is quite limited and fails to perceive the smoothness of the full motion path. To narrow the consistency gap between the generated motion and the source motion, we introduce a multi-level *temporal consistency loss*. We obtain the motion trajectory of each joint by manipulating the skeleton with Forward Kinematics (FK) according to the predicted skeletal rotations $\mathbf{Q}_B$, as demonstrated in Fig. 3 (b). Unlike [19], we do not normalize the motion by the character height or skeleton length. Instead, we normalize the trajectory by itself, scaling it to a unit cube to neglect the influence of different skeleton structures, as displayed in Fig. 5 (b). We maintain the motion dynamics by constraining the pair-wise motion vectors. For the joint k, the normalized trajectory is $\{\mathbf{c}_t^k\}_{t=1}^T$. For each point pair $(\mathbf{c}_i^k, \mathbf{c}_j^k), i, j \in \{1, ...T\}$, we calculate the motion vector $\mathbf{m}_{i,j}^k$ from the start point $\mathbf{c}_i^k$ to the endpoint $\mathbf{c}_j^k$. Stacking the motion vectors together, we create a matrix $\mathbf{M}^k \in \mathbb{R}^{T\times T\times 3}$. As the Fig. 5 (b) shows, we use the difference of the motion matrix $\mathbf{M}^k$ in the *temporal consistency loss*. The motion matrix is calculated for each joint of the source motion, and it acts as the supervision of the motion dynamics of its corresponding joint in the retargeted motion. The *temporal consistency loss* is defined as:

$$\mathcal{L}_{tc} = \frac{1}{K}\sum_{k=1}^{K} \left\|\mathbf{M}_A^k - \mathbf{M}_B^k\right\|_2^2. \tag{3}$$

3.4. Loss Functions

As illustrated in Fig. 3, unlike previous multi-stage models [19, 20], we train STaR in an end-to-end manner. And as there are no paired datasets for the motion retargeting task, we exploit as many supervision signals as possible from the source motion to achieve unsupervised learning. Apart from the limb penetration loss and temporal consistency loss introduced in Sec. 3.2 and Sec. 3.3, we utilize several semantic losses to preserve the motion semantics.

Reconstruction Loss. We adopt a self-reconstruction strategy following [11, 20]. The objective of self-reconstruction is to constrain the retargeting of the source motion $\mathbf{Q}_A$ onto the source character A. This strategy assists the network in searching for the static points of the neural function and ensures the stabilization of the model during training. The reconstruction loss $\mathcal{L}_{rec}$ includes skeletal rotation reconstruction and joint position reconstruction, which is defined as:

$$\begin{aligned}\mathcal{L}_{rec} = &\|\mathbf{Q}_A - \hat{\mathbf{Q}}_A\|_2^2 + \\ &\|f_K(\mathbf{Q}_A, \gamma_A) - f_K(\hat{\mathbf{Q}}_A, \gamma_A)\|_2^2,\end{aligned} \tag{4}$$

in which $\hat{\mathbf{Q}}_A$ is the reconstructed motion. f_K is the Forward Kinematics (FK) function, which projects the T-posed skeleton $\gamma \in \mathbb{R}^{K\times 3}$ to the specific pose.

Constraint Loss. In contrast to the reconstruction loss, while retargeting motion to the target character B, we use the constraint loss to limit modification while reducing penetration. To constrain the modification as small as possible, we encourage the retargeted motion $\mathbf{Q}_B$ to be close to the source motion $\mathbf{Q}_A$:

$$\begin{aligned}\mathcal{L}_{con} = &\|\mathbf{Q}_B - \mathbf{Q}_A\|_2^2 + \\ &\|f_K(\mathbf{Q}_B, \gamma_B) - f_K(\mathbf{Q}_A, \gamma_B)\|_2^2,\end{aligned} \tag{5}$$

where γ_B is the T-posed skeleton of the target character, and $f_K(\mathbf{Q}_A, \gamma_B)$ is the joint position after copying motion $\mathbf{Q}_A$ to the target character B.

Joint Orientation Loss. Under the joint influence of multiple losses, the network sometimes predicts motion that appears consistent with the source motion but results in an inward-outward inversion of the arms. In such cases, while the skeletal motion and penetration rate remain reasonable, the semantic meaning of the action is altered. To address this, we bind vectors $\mathbf{J}$ to all skeletal joints, which transforms along with the Forward Kinematics process. By monitoring the direction of this vector, we ensure that each skeletal joint maintains a reasonable orientation. The joint orientation loss is formulated as:

$$\mathcal{L}_j = \|f_K(\mathbf{Q}_A, \mathbf{J}) - f_K(\mathbf{Q}_B, \mathbf{J})\|_2^2, \tag{6}$$

where $\mathbf{J}$ is identical for every character.

The overall loss function is:

$$\mathcal{L}_{all} = \lambda_{rec}\mathcal{L}_{rec} + \lambda_{con}\mathcal{L}_{con} + \lambda_{lp}\mathcal{L}_{lp} + \lambda_{tc}\mathcal{L}_{tc} + \lambda_j\mathcal{L}_j. \tag{7}$$

Global motion. According to our observation, the dataset [2] lacks sufficient diversity to effectively train a stable global translation decoder, particularly for predicting translations of unseen characters. The neural function will overfit the training set. Thus, we follow the normalization method from [20].

4. Experiments

Datasets. We train and evaluate STaR, along with comparative methods, on the Mixamo dataset [2], a collection of diverse motion clips performed by artificial characters with various skeleton configurations and body shapes. We collect 2,400 motion instances from 11 characters, splitting them into training and testing sets. The training set comprises 1,905 motion sequences, with source and target characters sampled from a pool of 7 characters per sequence. Each source motion is randomly sampled for 60 frames to facilitate motion retargeting training. For testing, we follow the evaluation setup of previous works [1, 11, 16, 20]. The test set includes both Seen and Unseen Characters (SC, UC), as well as Seen and Unseen Motion clips (SM, UM), forming four subsets: SC+SM, UC+SM, SC+UM, and UC+UM. Only SC+SM is encountered during training. We sample 100 pairs per subset, segment each motion into 120-frame sequences, and report final results as the average across all subsets. To assess performance on real-human motion retargeting, we also test on the ScanRet dataset [18], which contains 8,298 motion clips from 100 human actors. During inference, we follow the official split from [18], consisting of 9 motion clips and 10 human actors.

Implementation details. We select $K = 22$ nodes per character and sample $N = 1024$ vertices for geometric embedding in each forward pass of the network. The feature length is set to $S = 64$. The hyperparameters are configured as follows: $\lambda_{rec} = 0.1$, $\lambda_{con} = 0.1$, $\lambda_{lp} = 5.0$, $\lambda_{tc} = 1.0$, and $\lambda_j = 1.0$. The limb penetration loss is partially implemented with CUDA [13] for acceleration. We set N_l and N_l^r to $(50, 100, 200, 400)$, and $(500, 1{,}000, 2{,}000, 4{,}000)$ respectively, and the corresponding results can be found in Sec. 4.3. We train the network using the Adam optimizer [8] and incorporate a frozen point cloud model from PCT [7]. The entire training process runs for 50 epochs.

Evaluation metrics. We evaluate the effectiveness of our model across three key aspects: joint accuracy, geometric penetration, and motion smoothness. For joint accuracy, we compute the Mean Squared Error (MSE) and local MSE (MSE^{lc}) to assess how closely the retargeted joint coordinates align with the ground truth (GT). Local MSE isolates local motion accuracy by eliminating the influence of global motion. These errors are normalized by character height. From a geometric perspective, we compare the penetration rate (Pen%), which measures the percentage of limb vertices that penetrate other body parts. Unlike most existing methods [19, 20], which consider only the body and head as penetrable regions, we also account for interpenetration between limbs for a more comprehensive evaluation. Few existing methods have established quantitative metrics for evaluating retargeted motion. We introduce a curvature metric (Curv), which quantifies the smoothness of the discrete motion trajectory of each joint. The final curvature score is computed as the average curvature across all limb joints. Further details on these evaluation metrics are provided in Sec. B of the supplementary materials.

4.1. Qualitative Results

We compare our method with "Copy", SAN [1], R^2ET [20], and MeshRet [18], and visualize the skinned results in Fig. 6. The first row shows that when the source and target characters have similar body shapes, all methods preserve motion semantics well, but STaR produces results closest to the source motion. In the second and third rows, when retargeting motion to characters with significantly different body shapes, penetration issues become more pronounced. Copy and SAN [1] struggle to adapt motion naturally when transferring motion from slim to fat, leading to severe penetration. MeshRet [18], though designed for geometric constraints, fails when the source motion already contains interpenetration. R^2ET produces competitive results but suffers from severe motion jitter. Despite body shape constraints, STaR effectively preserves motion semantics while significantly reducing penetration artifacts through full-body geometric correction. The last row, which is the preceding motion frame relative to the third row, partially reflects temporal consistency, showing that STaR produces a more stable and natural motion trajectory. Unlike other methods that aligned frame-wise arm movements, STaR maintains a larger motion space and refines motion trajectories even when no interpenetration occurs, demonstrating its ability

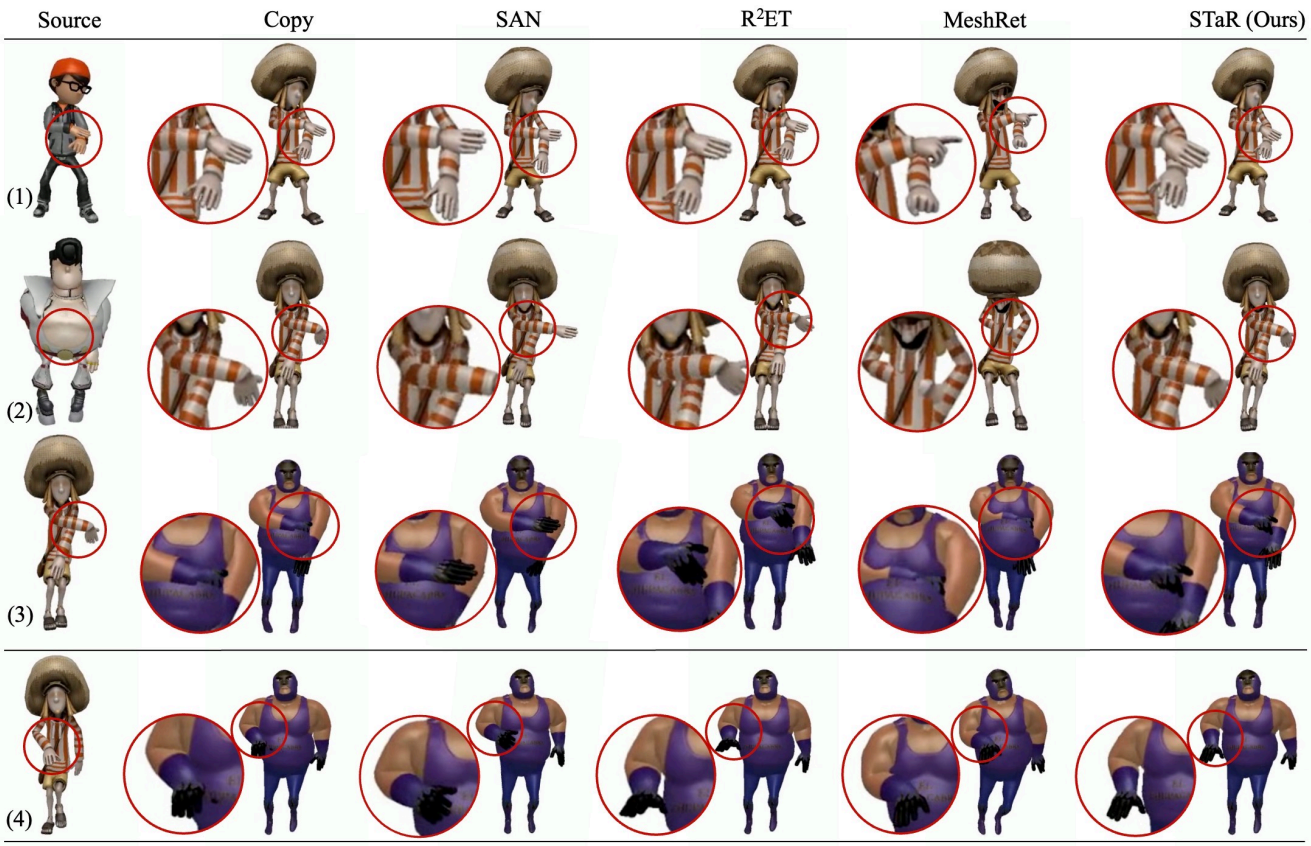


Figure 6. Qualitative comparison with other methods. The top three rows compare motion retargeting across different body shapes: same body shape, fat-to-slim, and slim-to-fat. The bottom row presents a preceding frame relative to the frame of the third row, highlighting the impact of our temporal consistency module on the spatial results. Please ignore hand details.

to adjust trajectories across the entire sequence. We further infer STaR on the real-human dataset, ScanRet [18], and the results are shown in Sec. C in the supplemental materials.

4.2. Quantitative Results

Table 1 compares our method with current state-of-the-art approaches. The "Copy" baseline achieves the lowest MSE and local MSE, since Mixamo [2] may generate dataset samples by copying and slightly modifying motions—yielding artificially low error metrics but at the cost of semantic accuracy and increased interpenetration, especially for characters with differing body shapes, as shown in Fig. 6. Among all motion retargeting methods, R^{2}ET [20] achieves the lowest MSE, demonstrating its ability to preserve motion semantics. Our method, STaR, maintains similarly low MSE while adjusting metric weights to prioritize reduced penetration rate and curvature.

Regarding geometric plausibility, skeletal-only retargeting methods such as PM-Net [11] and SAN [1] exhibit high penetration rates—typically exceeding 11%—because they disregard character geometry during motion retargeting. While these methods generally preserve motion smoothness. Methods incorporating geometric constraints, such as R^{2}ET [20] and MeshRet [18], effectively reduce penetration rate but introduce excessive motion jitter, as indicated by elevated curvature. R^{2}ET† [20] addresses this by predicting balancing weights to smoothen the motion, which significantly lowers curvature at the expense of a slight increase in penetration rate. Our method, STaR (Curv), achieves lower penetration rate (8.70%), outperforming R^{2}ET [20] while also producing lower curvature than R^{2}ET† [20], with only a slight increase in MSE. During the process of preventing interpenetration, curvature inevitably increases slightly, and this modest increase does not significantly affect motion smoothness. By reducing the penetration rate instead, we further enhance motion plausibility. We relax the smoothness constraint and achieve the lowest penetration rate of 7.99%, which is 16.4% lower than R^{2}ET [20]. These results demonstrate that STaR effectively balances motion semantics, geometric plausibility, and temporal consistency, highlighting the benefits of our spatio-temporal model, limb penetration constraint, and temporal consistency constraint.

For human perception evaluation, please refer to Sec. D.

Table 1. Quantitative comparisons with state-of-the-art methods. MSE^{lc} is the local MSE. $R^2ET^{\dagger}$ [20] refers to the debugged version of R^2ET [20]. **STaR** (Curv) assigns a higher weight to motion smoothness. For MSE, MSE^{lc}, and Pen%, lower values indicate better performance. For Curv, closer to GT's Curv is better.

Methods	MSE ↓	MSE^{lc} ↓	Pen% ↓	Curv ↳
Source	-	-	-	6.54
GT	-	-	11.26	7.64
Copy	0.0308	0.0102	11.23	**7.66**
Skeletal-only motion retargeting methods				
PM-Net [11]	0.2469	0.0287	11.46	12.06
SAN [1]	0.1664	0.0256	12.43	8.16
Skinned motion retargeting methods				
R^2ET [20]	0.0335	0.0138	9.55	35.67
$R^2ET^{\dagger}$ [20]	0.0331	0.0132	9.92	7.87
MeshRet [18]	0.0691	0.0325	9.39	17.67
STaR (Curv)	0.0353	0.0158	8.70	7.76
STaR (Ours Final)	0.0368	0.0174	**7.99**	10.66

4.3. Ablation Study

Table 2. Ablation Study. Baseline is a model which retargets motion frame-by-frame, training with $\mathcal{L}_{sdf}$ from [20]. ST is the spatio-temporal model, and DSR stands for dense shape representation. $\mathcal{L}_{lp}$ is the limb penetration constraint, and $\mathcal{L}_{tc}$ is the temporal consistency constraint. $\mathcal{L}_t$ is a basic motion smoothness loss. ✓✓ stands for a higher weight for $\mathcal{L}_{lp}$. In the second part of the table, we sample different numbers of vertices for the limb penetration constraint.

ST	$\mathcal{L}_{lp}$	DSR	$\mathcal{L}_{tc}$	MSE ↓	MSE^{lc} ↓	Pen% ↓	Curv ↳
✗	$\mathcal{L}_{sdf}$	✗	✗	0.0479	0.0321	8.74	66.35
✓	$\mathcal{L}_{sdf}$	✗	✗	0.0371	0.0179	9.45	**7.87**
✓	✓	✗	✗	0.0361	0.0167	8.20	35.74
✓	✓	✗	$\mathcal{L}_t$	0.0361	0.0167	8.40	35.74
✓	✓	✓	$\mathcal{L}_t$	0.0368	0.0167	8.20	21.29
✓	✓✓	✓	$\mathcal{L}_t$	0.2206	0.2082	**3.63**	31.60
✓	✓	✓	✓	0.0368	0.0174	7.99	10.66
Ablation study on vertex number for $\mathcal{L}_{lp}$							
4,000 (Ours)				0.0368	0.0174	7.99	10.66
2,000				**0.0355**	**0.0162**	8.41	8.66
1,000				0.0356	0.0163	8.38	8.55
500				0.0378	0.0190	8.49	8.66

We conduct extensive experiments to evaluate the significance of each proposed component, and the results are shown in the first part of Tab. 2. The baseline follows R^2ET [20], which applies an SDF loss for training but retargets motion frame-by-frame without considering temporal dependencies. Introducing the spatio-temporal (ST) model significantly reduces curvature, demonstrating the effectiveness of temporal modeling in reducing motion jitter. However, the penetration rate increases slightly. Adding the limb penetration constraint ($\mathcal{L}_{lp}$) effectively mitigates interpenetration, reducing the penetration rate from 9.45% to 8.20%. Further incorporating the basic smoothness loss ($\mathcal{L}_t$) improves temporal consistency without affecting penetration.

Next, we analyze the impact of dense shape representation (DSR). Adding DSR alongside $\mathcal{L}_{lp}$ and $\mathcal{L}_t$ further reduces curvature (from 35.74 to 21.29), indicating that shape-aware features help the model predict smoother motion. However, an excessive penalty on penetration, such as applying a large $\mathcal{L}_{lp}$, reduces penetration significantly (to 3.63%) but leads to poor motion generalization, as reflected by the sharp increase in MSE and local MSE. Our full model, which balances spatial, temporal, and geometric constraints, achieves a pretty low penetration rate (7.99%) while maintaining low curvature (10.66), demonstrating its overall effectiveness.

In the second part of Tab. 2, we examine the effect of varying the number of sampled vertices for the limb penetration constraint. Reducing the number of sampled vertices from 4,000 (ours) to 500 slightly increases penetration and curvature, but the differences remain minimal, suggesting that our method is robust to different sampling densities. Considering the balance between computational efficiency and accuracy, we choose 4,000 vertices in our final model. We conduct further ablation experiments for other modules, and please refer to the supplemental materials.

5. Conclusion

In this paper, we aim to achieve three key objectives in motion retargeting: semantics preservation, geometric plausibility, and temporal consistency. We propose STaR, a novel spatio-temporal skinned motion retargeting framework, embedded with a limb penetration constraint and a temporal consistency constraint. These constraints modify the motion manifold for characters of different shapes, while our spatio-temporal model design helps the optimizer find better parameters for this task. Experimental results demonstrate that our STaR achieves a superior balance across all three objectives compared to previous methods.

Limitations and Future Work. LBS is a major contributor to interpenetration. Resolving skinning artifacts could significantly lower penetration rates, since some issues stem from rigid-body assumptions that real skin deformation would naturally avoid. We also do not tackle missing contact in this paper, due to frequent false positives. Addressing these limitations will be a focus of our future research.

Acknowledgements

GS acknowledges support by Engineering and Physical Sciences Research Council [grant number EP/Y009800/1], through Keystone project funding from Responsible AI UK (KP0016). This research utilized Queen Mary's Andrena HPC facility, supported by QMUL Research-IT.

References

[1] Kfir Aberman, Peizhuo Li, Dani Lischinski, Olga Sorkine-Hornung, Daniel Cohen-Or, and Baoquan Chen. Skeleton-aware networks for deep motion retargeting. *ACM Transactions on Graphics (TOG)*, 39(4):62–1, 2020. 1, 2, 6, 7, 8, 11, 12

[2] Adobe. Mixamo. https://www.mixamo.com. Accessed in March, 2024. 2, 6, 7, 12

[3] Kwang-Jin Choi and Hyeong-Seok Ko. Online motion retargetting. *The Journal of Visualization and Computer Animation*, 11(5):223–235, 2000. 2

[4] Zhi Deng, Yuxin Yao, Bailin Deng, and Juyong Zhang. A robust loss for point cloud registration. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6138–6147, 2021. 2, 4

[5] Andrew Feng, Yazhou Huang, Yuyu Xu, and Ari Shapiro. Automating the transfer of a generic set of behaviors onto a virtual character. In *Motion in Games: 5th International Conference, MIG 2012, Rennes, France, November 15-17, 2012. Proceedings 5*, pages 134–145. Springer, 2012. 2

[6] Michael Gleicher. Retargetting motion to new characters. In *Proceedings of the 25th conference on computer graphics and interactive techniques*, 1998. 2

[7] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *Computational Visual Media*, 7:187–199, 2021. 3, 6, 12

[8] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[9] Jehee Lee and Sung Yong Shin. A hierarchical approach to interactive motion editing for human-like figures. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 39–48, 1999. 2

[10] Shujie Li, Lei Wang, Wei Jia, Yang Zhao, and Liping Zheng. An iterative solution for improving the generalization ability of unsupervised skeleton motion retargeting. *Computers & Graphics*, 104:129–139, 2022. 2

[11] Jongin Lim, Hyung Jin Chang, and Jin Young Choi. Pmnet: Learning of disentangled pose and movement for unsupervised motion retargeting. In *BMVC*, page 7, 2019. 1, 2, 3, 5, 6, 7, 8

[12] Giulia Martinelli, Nicola Garau, Niccoló Bisagno, and Nicola Conci. Moma: Skinned motion retargeting using masked pose modeling. *Computer Vision and Image Understanding*, 249:104141, 2024. 2, 3

[13] NVIDIA Corporation. CUDA Toolkit Documentation, 2024. 2, 6

[14] Seyoon Tak and Hyeong-Seok Ko. A physically-based motion retargeting filter. *ACM Transactions on Graphics (ToG)*, 24(1):98–117, 2005. 2

[15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. 4

[16] Ruben Villegas, Jimei Yang, Duygu Ceylan, and Honglak Lee. Neural kinematic networks for unsupervised motion retargetting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8639–8648, 2018. 1, 2, 6

[17] Ruben Villegas, Duygu Ceylan, Aaron Hertzmann, Jimei Yang, and Jun Saito. Contact-aware retargeting of skinned motion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9720–9729, 2021. 2

[18] Zijie Ye, Jia-Wei Liu, Jia Jia, Shikun Sun, and Mike Zheng Shou. Skinned motion retargeting with dense geometric interaction perception, 2025. 2, 3, 6, 7, 8, 10, 11, 12, 13

[19] Haodong Zhang, Zhike Chen, Haocheng Xu, Lei Hao, Xiaofei Wu, Songcen Xu, Zhensong Zhang, Yue Wang, and Rong Xiong. Semantics-aware motion retargeting with vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2155–2164, 2024. 2, 3, 5, 6, 12

[20] Jiaxu Zhang, Junwu Weng, Di Kang, Fang Zhao, Shaoli Huang, Xuefei Zhe, Linchao Bao, Ying Shan, Jue Wang, and Zhigang Tu. Skinned motion retargeting with residual perception of motion semantics & geometry. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13864–13872, 2023. 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 13