Contact-Aware Retargeting of Skinned Motion

Ruben Villegas [1] Duygu Ceylan [1] Aaron Hertzmann [1] Jimei Yang [1] Jun Saito [1]
[1] Adobe Research

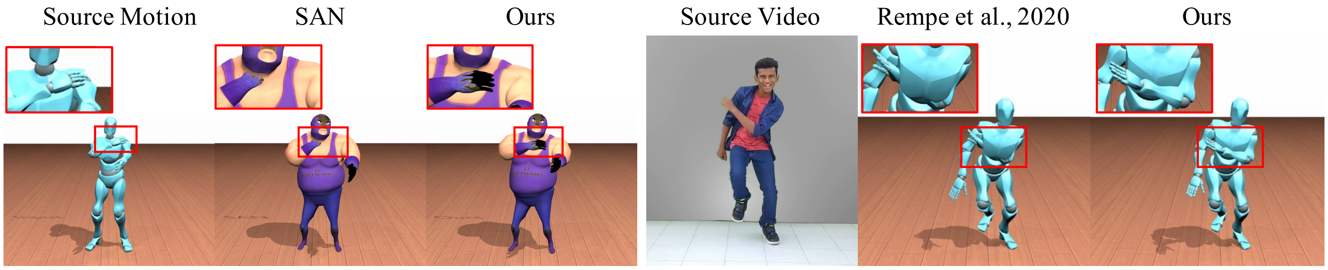

Figure 1. Our contact-aware motion retargeting method reduces interpenetration and preserves self-contacts on characters with different geometries in a unified framework while existing methods, like Skeleton-Aware Networks (SAN) [2], do not (Left). In addition, our method generalizes to unseen motions estimated from videos; qualitatively improving noticeable interpenetration in recent human motion estimation methods.

Abstract

This paper introduces a motion retargeting method that preserves self-contacts and prevents interpenetration. Self-contacts, such as when hands touch each other or the torso or the head, are important attributes of human body language and dynamics, yet existing methods do not model or preserve these contacts. Likewise, interpenetration, such as a hand passing into the torso, are a typical artifact of motion estimation methods. The input to our method is a human motion sequence and a target skeleton and character geometry. The method identifies self-contacts and ground contacts in the input motion, and optimizes the motion to apply to the output skeleton, while preserving these contacts and reducing interpenetration. We introduce a novel geometry-conditioned recurrent network with an encoder-space optimization strategy that achieves efficient retargeting while satisfying contact constraints. In experiments, our results quantitatively outperform previous methods and we conduct a user study where our retargeted motions are rated as higher-quality than those produced by recent works. We also show our method generalizes to motion estimated from human videos where we improve over previous works that produce noticeable interpenetration.

1. Introduction

Self-contact, where one part of a person's body comes into contact with another part, is crucial to how we perceive human motion. Self-contact often indicates different behaviors or emotional states. For example, people might rest their head in their hands if they are concerned or thoughtful, whereas certain dance moves require one's hands to be placed on one's hips. Conversely, implausible self-contacts, such as hands passing through each other, ruin the physical plausibility of a motion. Hence, handling self-contact is crucial for reconstructing, interpreting, and synthesizing human motion. Note that synthesizing contact for a skinned character requires knowledge of the character's 3D geometry; it cannot be accurately determined from skeleton alone.

This paper introduces a motion retargeting algorithm that preserves both self-contact and ground contact, while also reducing interpenetration. Our retargeting algorithm takes an input human motion and a target character, and produces a plausible animation that manifests how that target character might perform the input motion. Our method first identifies self-contact and foot contacts in the input motion. We use an energy function that preserves these contacts in the output motion, while reducing interpenetration in the output. We reason about self-contacts from the character's geometry and foot contacts from the character skeleton, in order to guarantee that contacts will be accurately transferred in the skinned and rendered motion. Our approach generalizes to any mesh geometry and topology regardless of the number of vertices in the mesh.

Due to the difficultly of directly optimizing a full motion sequence, we build on previous work and train a Recurrent Neural Network (RNN) to perform retargeting. We find that an RNN does not perfectly satisfies contact constraints given its efficient inference. Therefore, we propose encoder-space optimization, in which we refine the RNN's predictions by iteratively optimizing the hidden units of RNN's encoder. This process allows our method to efficiently satisfy constraints, by taking advantage of the

RNN's smooth, disentangled encoder-space.

Since hands often participate in meaningful self-contacts (e.g., contacts from hands to head, hands to hand, hands to hip), we focus our analysis on contacts between the hands and the rest of the character geometry. Note that our use of geometry means that the output style will depend on character mesh: a bulkier character is more constrained in their movements, which is reflected in our results. We evaluate our method on various complex motion sequences as well as a wide range of character geometries from skinny to bulky. We show that our method provides a good balance between preserving input motion properties while preserving self-contacts and reducing implausible interpenetration. In our qualitative and quantitative experiments, we outperform the state-of-the-art learning-based motion retargeting methods. In addition, we qualitatively show our method generalizes to real scenarios by retargeting motion extracted from human videos. Our method improves upon methods that only consider the skeleton when estimating the motion.

2. Related Work

Most early methods for motion retargeting perform a constrained optimization problem over the skeleton of a character. Gleicher [10] optimized the output motion sequence guided by kinematic constraints. Several authors include physical plausibility constraints in the optimization [23, 27, 5], including ground contact constraints. Another approach is to first solve an inverse kinematics (IK) problem for each motion frame, and then apply spacetime smoothing to the result [19, 8, 30, 6, 11].

The above methods either use expensive optimization algorithms or else suboptimal spatiotemporal smoothing. More recently, deep learning methods can resolve both of these issues. Villegas et al. [31] train a recurrent neural network with a forward-kinematics layer for unsupervised motion retargeting. Lim et al. [20] train a feed-forward motion retargeting network that decouples the local pose and the overall character movement. Aberman et al. [3] proposed a supervised method for 2D motion retargeting that learns disentangled representations of motion, skeleton, and camera view-angle. Aberman et al. [2] proposed Skeleton-Aware Networks (SAN) which learn to retarget between skeletons with different numbers of joints. However, this method requires retraining for each new set of skeleton topologies, and their method does not take character geometry into account. Moreover, these network-based retargeting methods cannot precisely satisfy constraints, a limitation we address with encoder-space optimization.

Only a few older works consider self-contacts in retargeting. Lyard and Magnenat-Thalmann [22] use a heuristic sequential optimization strategy. Ho and Shum [15] adapt robot motions with a spacetime optimization strategy that prevents self-collisions. More recently, Basset *et al.* [1, 7] proposed a constrained optimization algorithm with attraction and repulsion terms to preserve contacts and avoid interpenetration. These works, however, require exact vertex correspondences between the input and target meshes, expensive constrained optimization over the entire sequence, and smoothing as a post-process. Liu *et al.* [21] requires input/target mesh calibration, and solves for bone, surface smoothness, and volumetric constraints by solving a linear system. Our method is general to any source/target mesh, retargets motion frame-by-frame, and all mesh and skeleton shape constraints are implicitly handled.

3D character geometry has been considered in other animation-related tasks to improve the motion plausibility. Tzionas et al. [29] incorporated collision detection for hand motion capture and Hasson et al. [13] used a mesh-level contact loss for jointly 3D reconstructing the hand and the manipulating object. Body geometry is modeled while estimating and synthesizing plausible humans that interact with 3D scene geometry [12, 33], and with other characters [14]. Interpenetration and character collisions with surrounding objects are handled in character posing [17]. However, it is nontrivial to extend these posing methods to retargeting while maintaining the naturalness of the motion. Our method instead infers character specific geometric constrains (contacts) directly from the mesh and maintains them during retargeting. In work concurrent to our own, Smith et al. [26] demonstrate the importance of self-contact handling for detailed hand tracking.

Recent works have investigated the use of optimization as supervision when ground-truth is not available. Kolotouros et al. [18] proposed a learning scheme for 3D human pose and shape estimation that performs iterative 3D human model fitting which provides weak supervision during learning. Our proposed geometry-aware RNN is trained in a weakly supervised manner using an energy function that preserves self-contacts and ground contacts. In addition, our encoder-space optimization strategy uses the weak supervision provided by our energy function to precisely satisfy constraints at test time.

3. Contact-Aware Motion Retargeting

This section describes our motion retargeting method; the main steps are summarized in Figure 2. The inputs to our method are a source motion represented as the skeleton motion m^A and the skinned motion $\hat{v}^A$, and a target character represented by the skeleton $\bar{s}^B$ and the skinned geometry $\bar{v}^B$ in a reference pose (e.g., T-pose). Then, our goal is to output the skeleton motion m^B as well as the corresponding skinned motion $\hat{v}^B$ that preserves the "content" of the source motion, including self-contacts and ground contacts, while also not introducing interpenetration. The self-contacts and ground contacts in the source motion are automatically estimated by a contact detection step, de-

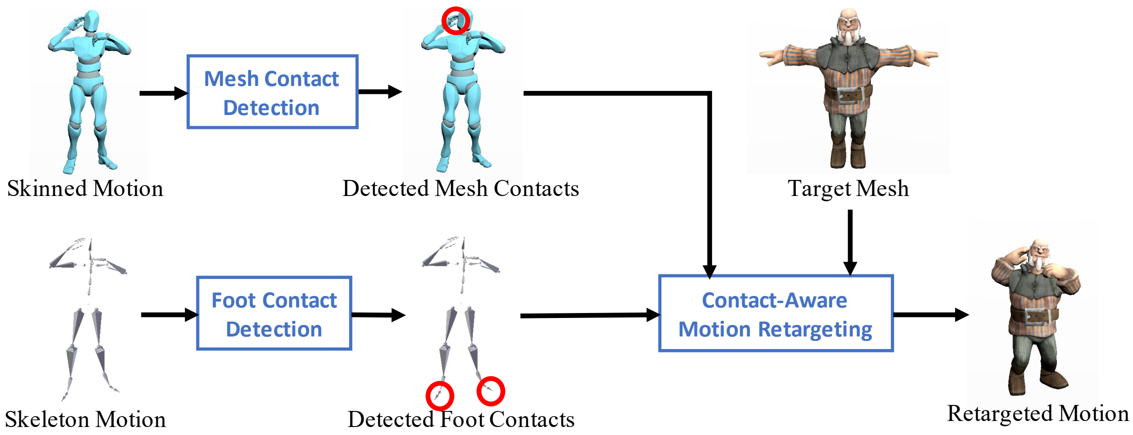

Figure 2. Method overview. Our method first detects hand contacts in the input motion geometry and foot contacts with the floor. The detected contacts are then passed into our contact-aware motion retargeting which retargets the source motion into the target character while preserving the detected contacts.

scribed in Section 3.4. We next describe our energy function, geometry-conditioned RNN, and our encoder-space optimization strategy to satisfy hard constraints.

3.1. Energy Function

Given the source motion, skeleton, and inferred contacts, we formulate an energy function for the output motion, which includes novel terms to preserve the input contacts, and to reduce the introduction of interpenetration. Like previous work, we also include terms to preserve the source motion, represented in terms of skeleton motion, i.e., joint rotations, global trajectory, end-effector velocities, and foot contacts. We define the full energy function as:

$$E_{full}(t) = E_{geo}(t) + E_{skel}(t), \tag{1}$$

where E_{geo} is the geometry based motion modeling term focusing on vertex contacts and interpenetration and E_{skel} is the skeleton based motion modeling term. This energy is defined and optimized for each frame t in an online retargeting fashion. For simplicity, we omit the input t in the remaining of the paper.

3.1.1 Geometry-level modeling terms

Modeling motion at the geometry level requires two parallel objectives working together: modeling mesh contacts and reducing interpenetration. If we model the mesh contacts alone, there is nothing to prevent the retargeted motion from perfectly reaching that contact while introducing large penetrations (e.g., left hand going through the torso to touch the right arm). On the other extreme, if we completely avoid interpenetration, meaningful contacts are likely to be lost as these may cause slight interpenetration. Therefore, our geometry modeling energy is defined by:

$$E_{geo} = \lambda E_{j2j} + \beta E_{int} + \gamma E_{foot}, \tag{2}$$

where E_{j2j} is our self-contact term, E_{int} is our interpenetration reduction term, and E_{foot} is the foot contact term.

Self-contact preservation. This step takes as input a set of vertex contact constraints on the output motion for a particular time frame, identified during the contact detection step (Section 3.4). Specifically, the input is a set of $\mathcal{V} = \{(i,j)\}$ where tuple (i,j) indicates that vertices v_i and v_j of the output mesh should be in contact. This is measured as the average distance between such pairs of vertices:

$$E_{j2j} = \frac{1}{|\mathcal{V}|} \sum_{(i,j)\in\mathcal{V}} \|v_i - v_j\|^2 \tag{3}$$

Interpenetration reduction. We reduce mesh penetrations with a penalty similar to that of Tzionas et al. [29]. Specifically, at each time step, we detect the set of colliding triangles $\mathcal{F} = \{(r,i)\}$ using a bounding volume hierarchy [28]. We further define a local 3D distance field for each triangle in the form of a cone. For a pair of colliding triangles f_r and f_i, the position of one triangle inside the local 3D distance field of the other determines the penetration term. Thus, we define E_{int} as follows:

$$E_{int} = \sum_{(r,i)\in\mathcal{F}} \left(\sum_{v_j \in f_r} w_{r,i} \| -\psi_i(v_j) n_j \|^2 + \sum_{v_k \in f_i} w_{i,r} \| -\psi_r(v_k) n_k \|^2 \right). \tag{4}$$

where v_j's are vertices in one of the triangles, n_j are the corresponding per-vertex normals, and ψ_i and ψ_r are the distance fields for triangles i and r. The distance fields are positive for triangles that interpenetrate, and zero otherwise. See Tzionas et al. [29] for further details.

The weight factors $w_{r,i} = w_{i,r}$ are determined as follows. We observed in initial tests that setting uniform weight created unnatural motions, because of slight interpenetrations near joints such as the underarms or crotch. Such interpenetrations are not objectionable, whereas penetrations between different part of the character are. Hence, we set the weights based on geodesic distance, with a small weight for triangles that are close together on the surface:

$$w_{r,i} = \eta^{-1} \min_{v_j \in f_r, v_k \in f_i} D(v_j, v_k) \tag{5}$$

where $D(v_j, v_k)$ is the geodesic distance from vertex v_j to vertex v_k, and the normalization η is the maximum geodesic distance between all pairs of vertices in the mesh: $\eta = \max_{a,b} D(v_a, v_b)$.

Foot contact preservation. This step takes as input the foot contacts with the ground detected in the source motion as described in Section 3.4. The energy term preserves these contacts by minimizing velocity and height for vertices that should be in contact:

$$E_{foot} = \sum_{j \in \mathcal{C}} \frac{1}{h^B} \left(\|\dot{g}_j^B\|^2 + \| \left(g_j^B\right)^y \|^2 \right), \tag{6}$$

where g^B indicates the global joint position and $\mathcal{C}$ is the set of joint indices (i.e., heels and toes) in contact with the ground at the current time-step t. The first term inside the parenthesis minimizes the foot joint velocities and the second term minimizes the y coordinates of the foot joint positions to be at ground level during a detected contact.

In practice, the heel height from the ground depends on how the character was built. Heel may be at ankle height in some characters, and so, we get the heel height from reference pose, i.e., T-pose skeleton, and add it to the heel coordinates so that ground height of zero corresponds to contact.

3.1.2 Skeleton-level motion transfer term

In order to preserve the style of the input motion, we adopt an energy term that models the motion at the skeleton level similar to previous works. Specifically, we focus on preserving the local motion represented as joint rotations, the global motion represented as the root trajectory and the global motion of the end-effectors (i.e., hands and feet). Hence, our motion energy function is defined as:

$$E_{skel} = E_{weak} + \omega E_{ee}, \tag{7}$$

where E_{weak} models local and global motion and E_{ee} models the end-effector positions. Our first term encourages the retargeted joint rotations to remain close to the source motion in a weakly supervised manner, while also encouraging the target character to follow a global trajectory similar to the source. Our weakly supervised energy is defined by:

$$E_{weak} = \rho \sum_j \|\theta_j^B - \theta_j^A\|^2 + \|o^B - o^{A\to B}\|^2, \tag{8}$$

where θ_j^A is the rotation of joint j in the source motion, θ_j^B is the rotation of joint j in the retargeted motion, o^B is the retargeted root velocity, and ρ is the weight of this term. The source motion's root velocity o^A is scaled by the height ratio between the legs of the source and target characters, producing $o^{A\to B}$.

We further define our second term which models the motion of the end-effectors as follows:

$$E_{ee} = \sum_{j \in ee} \| \frac{1}{h^B} \dot{g}_j^B - \frac{1}{h^A} \dot{g}_j^A \|^2, \tag{9}$$

where $\dot{g}_j^A$ and $\dot{g}_j^B$ respectively are the velocities of the source and output character joint j in global coordinates and ee is the set of joint indices of the hands and feet end-effectors. E_{ee} minimizes the difference between the global velocities of end-effectors, scaled by the respective character heights h^A and h^B.

3.2. Geometry-Conditioned RNN

Directly optimizing the output motion is very expensive and impractical, which has motivated the development of RNN-based methods [9]. Direct optimization would be especially impractical for our case, since geometric collision detection on full-body meshes is memory-intensive.

Instead, we train a recurrent neural network (RNN) that retargets a source motion to a target character by minimizing the energy function defined in the previous section. As shown in Figure 3, the RNN is composed of an encoder that encodes the source skeleton motion m^A into RNN states h^{enc} and a decoder that outputs motion features in the hidden states that are used to predict the target skeleton motion, m^B, and skinned motion, $\hat{v}^B$. In order to decode hidden motion features, the decoder takes as input the encoded source motion features in the current frame, the local motion (p^B) and root velocity (o^B) of the target character in the previous frame as well as the skeleton ($\bar{s}^B$) and the geometric encoding of the target character in reference pose (i.e., T-pose). We utilize the PointNet [24] architecture to obtain a geometric encoding of the target character given its geometry and skinning weights in the reference pose. This enables our architecture to be invariant to the topology of the target character mesh. Given the hidden motion features, we utilize linear layers to decode the local joint rotations and the root velocity. A Forward kinematics (FK) layer converts the local joint rotations (θ^B) to local joint positions and by adding the root velocity we obtain the global

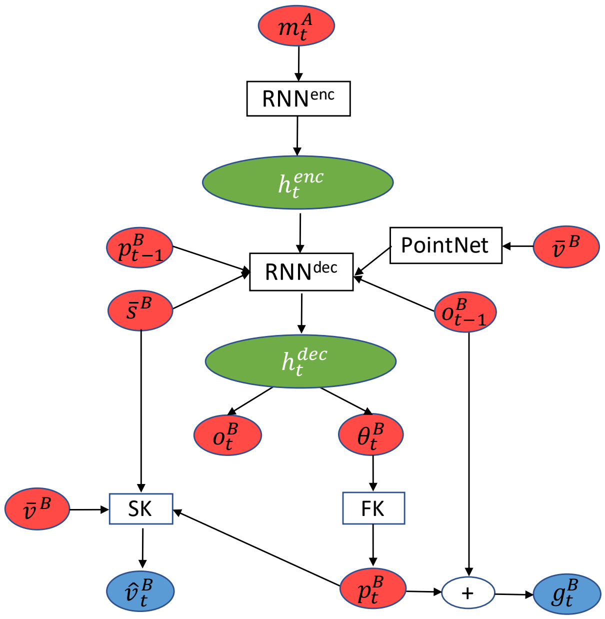

Figure 3. Geometry-conditioned RNN. See text for details.

skeleton joint coordinates, g^B. We also incorporate a skinning layer (SK) that deforms the target character geometry based on the skeleton motion and outputs the local skinned motion, $\hat{v}^B$. We refer the reader to the supplementary material for details on the network architecture.

3.3. Encoder-Space Optimization

While the RNN produces good output motions, they often do not adequatenly satisfy contacts and penetration constraints. Therefore, at test-time, we take the RNN output as the initial solution to our energy function and continue optimizing it. As directly optimizing the pose representation remains impractical, we propose *encoder-space optimization*, in which the motion is updated by updating the hidden encoding h^{enc} and root velocity o. This allows us to update the motion frame-by-frame, while also taking advantage of the smooth, low-dimensional, decoupled embedding learned by the network. Specifically, for each time-step t of the output motion, we perform $N = 30$ gradient descent updates to the encoder hidden units and root velocity as follows:

$$h^{\text{enc}}_{t,n+1} \leftarrow h^{\text{enc}}_{t,n} - \alpha \frac{\partial E_{full}(t)}{\partial h^{\text{enc}}_{t,n}}, \tag{10}$$

$$o_{t-1,n+1} \leftarrow o_{t-1,n} - \alpha \frac{\partial E_{full}(t)}{\partial o_{t-1,n}}, \tag{11}$$

where $1 \leq n \leq N$ indicates the iteration number. Note that these updates only consider the $E(t)$ terms that depend on the current time-step t. The output motion can then be generated from the updated hidden units and root velocity.

Attempting to sequentially update the pose parameters θ frame-by-frame does not improve the motion, since these parameters are tightly coupled in the energy.

3.4. Input contact detection

We now describe how we estimate self-contacts and ground contact in the source motion.

Self-contacts. There are two steps to our self-contact detection process. First, we identify instances in the source motion where either of the character's hands intersect any other body part, including the other hand. Then, we convert these intersections into contact constraints for use in the output motion.

As a preprocess, we group the vertices associated with skinning weights over 0.5 for each joint. For example, the left hand group is the set of vertices for which the skinning weight from the left wrist joint is at least 0.5. Each group also includes the triangles formed by its vertices.

In a given input frame, the algorithm detects if there is an intersection between one of the hand groups and another group on the body, using a Bounding Volume Hierarchy (BVH) [28] on the triangles in the groups. The two groups are determined to be in contact if the average cosine similarity of the per-vertex velocities in global coordinates is greater than 0.9, or if the distance between their nearest vertices is less than 0.2 cm where the shortest character in our dataset is 138 cm. For each detected contact, we identify the top 3 closest pairs of vertices between the two groups. The same process is repeated for all pairs of intersecting groups.

Since the input and output characters may have very different mesh geometry and topology, we next need to transfer contact constraints to the output mesh. Let v^A be a contact vertex in the input shape; we first must identify its corresponding vertex v^B in the output shape. As in many previous works, we use feature matching for mesh correspondence. Specifically, we define a per-vertex feature vector that combines skinning weights and offset vectors from the corresponding joint positions in the reference pose; we found that skinning weights alone were inadequate for correspondence. Then, the corresponding output mesh vertex v^B is the output vertex with the most-similar feature vector to the input vertex [32]. Given a pair of input vertices in contact v_i^A and v_j^A on the input, we generate a constraint that the corresponding vertices v_i^B and v_j^B should be in contact on the target mesh. Details of the feature vector are in the supplemental material.

Foot contacts. We find that simple thresholds similar to previous work, e.g., [25] are sufficient for detecting foot contacts with the ground. The feet are defined by the toe and heel joints, which we threshold in different ways. The toe joint is determined to be in contact if its height from the ground is at most 3cm and displacement from previous time-step is at most 1cm, all at 180 cm scale. The heel is determined to be in contact only if its displacement from the previous time-step is at most 1 cm at 180 cm scale. We treat the heel differently from the toe because some of the motions were captured with subjects wearing high heel shoes which makes our threshold based on heights from the ground inaccurate. We perform an additional step to remove any false negatives by a sliding window approach that fills in contacts if more than half of the frames in a window of 5 frames are in contact.

4. Experiments

Dataset, training and testing. We evaluate our method on the Mixamo dataset [4], which consists of sequences of the same motions performed by multiple characters with different body shapes and skeleton geometries.

A shortcoming of the Mixamo dataset is that it does not provide clean ground truth: many of these motions exhibit the kind of interpenetration and ground contact violations that we aim to correct, and no perfectly clean dataset is available. These issues may not be a problem during training because contacts are enforced by our loss terms rather than being purely learned. However, evaluation is difficult for these motions; it is very difficult to visually evaluate the quality of motion transfer when the input suffers from contact and penetration issues. Hence, for all of our tests, we use the "Y-bot" character motions as source motion, because these motions do have accurate self-contacts. The "Y-bot" character is not seen during training.

We train on the training subset used in [31], which contains distinct motion sequences performed by 7 characters, and contains 1646 motions in total. Similar to [31, 20], we train our method by extracting clips of 60 frames long, which we retarget to characters in the training. In contrast to [31, 20], we evaluate our method and baselines on full motion capture sequences of variable length. Please see the supplementary materials for more training and testing details.

Preprocessing. We preprocess each motion sequence similar to [31, 20, 16] by separating into local pose and global pose. While local pose consists of joint rotations relative to their parents, global pose consists of the velocity of the root (i.e., hip) in x, y, and z coordinates. We use the same 22 joint subset modeled in [31, 20]. Please see the supplementary materials for more details.

Baselines. We compare against the unsupervised motion retargeting methods from [31] and [20] which both involve a forward kinematics layer in their network architectures. We also compare with Skeleton-Aware Networks [2] (SAN) which uses skeleton pooling-based encoder and decoder networks to learn a skeleton invariant representation via an adversarial cycle consistency method. Note that SAN is an offline motion retargeting method (i.e., it requires the whole source motion sequence to be seen before retargeting), and applies inverse-kinematics (IK) as post-processing to maintain foot contacts in the final output. Following SAN, we also include a baseline where we take our network outputs and perform the same IK as a post-processing step as in [2] instead of our encoder-space optimization.

Evaluation metrics. We evaluate results based both on the geometry and the skeleton. We evaluate skeletal motion, by measuring how close the retargeted motions are to the ground-truth provided in the Mixamo dataset. The global joint position error is normalized by character height.

Interpenetration is scored using the interpenetration penalty in Equation 4 on the normalized vertices. We evaluate how well self-contacts are preserved by measuring the distance between the vertices involved in each contact. We use continuous scores rather than binary losses, because some characters are unable to perfectly reach a contact point due to their geometries (e.g., bulky vs skinny character), and slight contact errors are preferable to large ones. Our foot contact metric is based on a binary classification of the foot contacts detected in the retargeted motion against the contacts detected in the source motion. We use a binary metric since, in contrast to our vertex distance metric, there is nothing preventing feet from reaching the ground.

Finally, we conduct a user evaluation where we show two retargeting results of a given source motion, our result versus the result of the best performing baseline method, and ask the users to choose the more plausible result.

4.1. Results

In this section, we present quantitative and qualitative experiments. Please see our supplementary material for an ablation of the different components in our method.

4.1.1 Comparisons

In this section, we compare our full method against the recently proposed motion retargeting methods. As shown in Table 1, our method outperforms all the baselines not only in terms of self-contact handling and interpenetration, the focus of our method, but also in terms of motion quality. A significant advantage of our method is the encoder-space optimization to enforce constraints. In contrast, non of the baseline methods explicitly model the contacts. The SAN method [2] presents an additional post-processing step based on Inverse Kinematics (IK) to ensure foot contacts are modeled (SAN + IK). While it does improve foot contact accuracy and global position MSE, it sacrifices the geometry quality. To have a fair comparison, we perform the

Method	Geometry evaluation		Motion evaluation	
	Inter-Penetrations ↓	Vertex Contact MSE ↓	Foot Contact Accuracy ↑	Global Position MSE ↓
Ours	**0.81**	**3.87**	**0.97**	**0.48**
E_{full} + IK	1.19	4.52	0.82	1.58
SAN [2]	1.43	5.26	0.63	0.82
SAN + IK [2]	1.32	5.78	0.73	0.74
PMnet [20]	2.94	23.11	0.70	3.67
NKN [31]	3.20	14.86	0.71	8.15

Table 1. We evaluate the retargeted motion at the geometry and skeletal level. We evaluate the amount of self-penetrations, geometry contacts distance, foot contacts with the floor, and global joint positions in the retargeted motion.

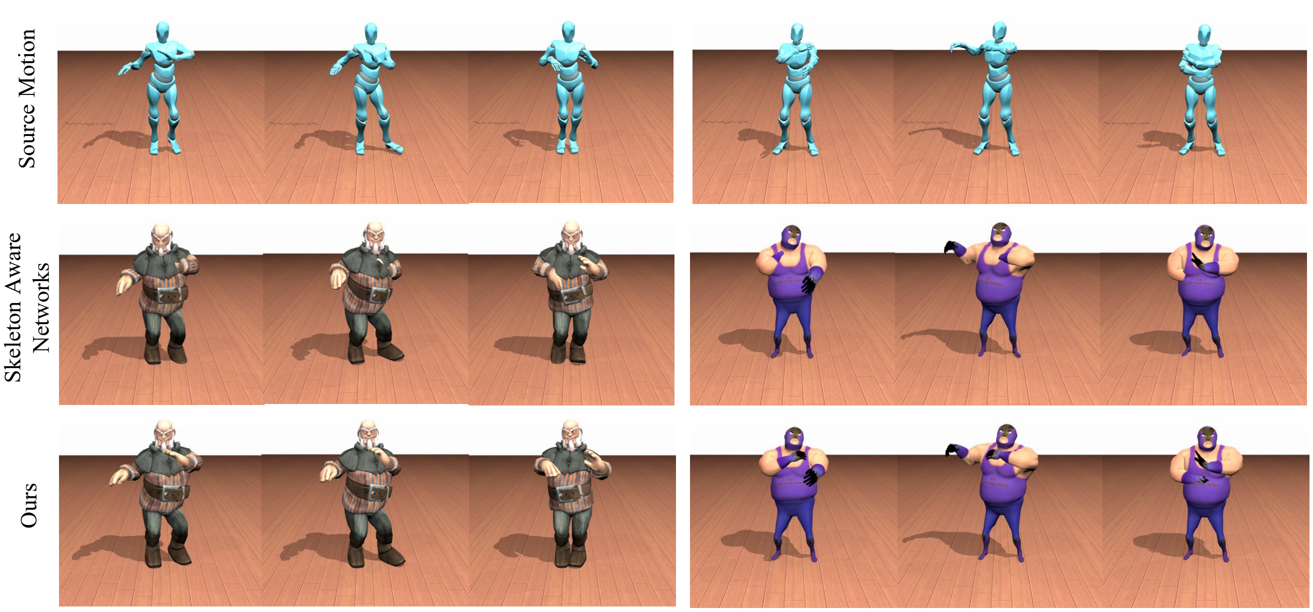

Figure 4. Qualitative comparison. We show an example where our method is able to reduce interpenetration without the need of contact modeling (left side), and an example where our method is able to model self-contacts while reducing self-penetrations (right side). We invite readers to watch the supplementary videos for more detailed visual comparisons.

Expert User Ours / SAN + IK	Non-Expert User Ours / SAN + IK	Overall Ours / SAN + IK
0.86 / 0.14	**0.74** / 0.26	**0.80** / 0.20

Table 2. We ask human subjects to compare our retargeting results to that of the best performing baseline in Table 1. Overall 80% of the users prefer our results.

same IK on the outputs of our network without encoder-space optimization (E_{full} + IK). It is clear that even though foot contact accuracy is improved with IK, our complete method (E_{full} + ESO) outperforms it by a large margin. We invite readers to watch the supplementary videos for visual comparisons.

User study. We conduct a user study to qualitatively evaluate the performance of our method against SAN + IK, the best performing baseline. For each question, human subjects are given three videos, namely the source motion, and the two motions retargeted using either our method or SAN + IK. The retargeted results are randomly placed on either side of the source motion and anonymously labeled. We ask the subjects to choose the retargeting result that looks closest to the source motion. Each subject is shown 20 questions where the motion triplets are randomly sampled out of 180 test motion sequences. We run our user study on a total of 17 subjects where 8 subjects have animation expertise (animation artist or developer) and 9 are not, collecting a total of 340 comparisons. Further details of how we conduct the user evaluation is provided in the supplemental material.

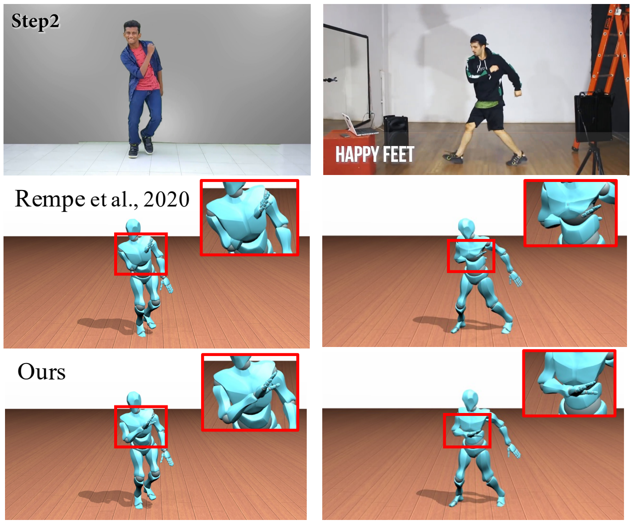

Figure 5. We retarget two motion clips estimated using the method by [25], and synthesize motion that avoids interpenetration present in the original input. We invite readers to watch the supplementary videos for comparisons.

Our results are summarized in Table 2. We find that overall 80% of users prefer our retargeted motion over the baseline. Furthermore 86% of the expert subjects prefer our results demonstrating the superior performance of our approach. We also provide a qualitative comparison of our method against the baseline in Figure 4 where we highlight the penetration reduction and self-contact capabilities of our method in comparison to the baseline.

4.2. Retargeting motion from human videos

In this section, we test our method for retargeting motion estimates from human videos. We get the human pose estimates from [25], and feed it to our method to retarget motion that takes the character mesh into account. In Figure 5, we show how our method (bottom row) is able to avoid interpenetration present in the motion estimated by [25]. This happens because their method processes character skeleton while completely ignoring the character mesh. In addition, we test our method for retargeting human motion into characters with a significant body shape difference. In Figure 6, we show how the motion retargeted using our method follows the input video closely while moving within the target character body shape constraints. Please refer to our supplementary material for video results.

5. Conclusion

We present a motion retargeting method that detects and preserves self and ground contacts while also reducing interpenetration. At the core of our method is an energy formulation that enforces such geometric constraints while also preserving the input motion quality. We train a recurrent neural network (RNN) that minimizes this energy function to retarget motion between source and target characters that vary in terms of skeleton lengths, proportions, as well as character geometries. We further propose an encoder-space optimization strategy to refine the hidden encoding of the RNN to satisfy the contact constraints.

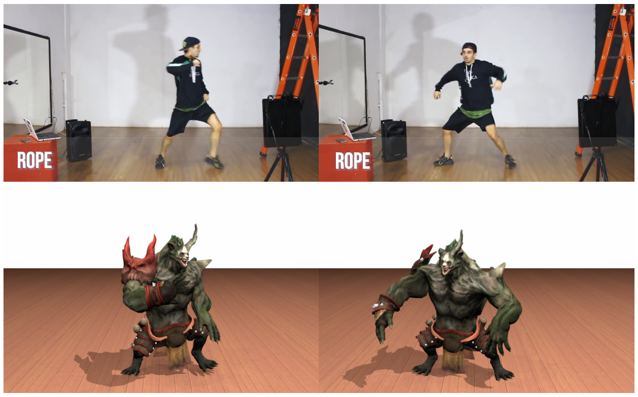

Figure 6. We retarget motion estimated by [25] into a character with significant body shape differences. We invite readers to watch the supplementary video for more results.

While our method outperforms recently proposed retargeting solutions both quantitatively and qualitatively, there still remain some limitations that we would like to address in future work. The different energy terms in our formulation quantify various qualities of a good retargeting output. However, occasionally different terms might be conflicting, e.g., the retargeting might sacrifice preserving the input motion in order to reduce interpenetration. This might result in the loss of the characteristic motion style of the input. A thorough analysis of the factors that affect the perception of a motion and how they can be incorporated in a retargeting method is an interesting future research direction. In our current implementation, we use a heuristic approach to detect the self-contacts in the input motion. Provided with annotated data, such contacts can be learned automatically. Enforcing the interpenetration term aggressively may result in stiff retargeted motion. We utilize geodesic weights to softly penalize intersections in regions such as underarms. Providing the user with the ability to paint regions that allow slight penetrations is another future direction. Finally, even though our contact-aware formulation is general, our network cannot handle arbitrary number of joints. This is an additional future direction we plan to pursue.

Acknowledgements

We would like to thank Minhyuk Sung for his help during initial stages of the project. We would also like to thank Stefano Corazza and everyone in Adobe Aero and Adobe Research for their helpful feedback. Last but not least, we would like to thank the participants in our user study.

References

[1] Contact preserving shape transfer: Retargeting motion from one shape to another. *Computers & Graphics*, 2020. 2

[2] Kfir Aberman, Peizhuo Li, Dani Lischinski, Olga Sorkine-Hornung, Daniel Cohen-Or, and Baoquan Chen. Skeleton-aware networks for deep motion retargeting. *SIGGRAPH*, 2020. 1, 2, 6, 7

[3] Kfir Aberman, Rundi Wu, Dani Lischinski, Baoquan Chen, and Daniel Cohen-Or. Learning Character-Agnostic Motion for Motion Retargeting in 2D. In *SIGGRAPH*, 2019. 2

[4] Adobe Mixamo. https://www.mixamo.com. Accessed: 2019-11-06. 6

[5] Mazen Al Borno, Ludovic Righetti, Michael J Black, Scott L Delp, Eugene Fiume, and Javier Romero. Robust physics-based motion retargeting with realistic body shapes. In *Computer Graphics Forum*, volume 37, pages 81–92. Wiley Online Library, 2018. 2

[6] Andreas Aristidou and Joan Lasenby. Fabrik: A fast, iterative solver for the inverse kinematics problem. *Graphical Models*, 2011. 2

[7] Jean Basset, Stefanie Wuhrer, Edmond Boyer, and Franck Multon. Contact preserving shape transfer for rigging-free motion retargeting. In *Motion, Interaction and Games*, MIG '19, New York, NY, USA, 2019. Association for Computing Machinery. 2

[8] Kwang-Jin Choi and Hyeong-Seok Ko. Online motion retargetting. In *Proceedings. Seventh Pacific Conference on Computer Graphics and Applications*, pages 32–42. IEEE, 1999. 2

[9] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *ICCV*, pages 4346–4354, 2015. 4

[10] Michael Gleicher. Retargetting motion to new characters. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 33–42. ACM, 1998. 2

[11] Pawan Harish, Mentar Mahmudi, Benoît Le Callennec, and Ronan Boulic. Parallel inverse kinematics for multithreaded architectures. *TOG*, 2016. 2

[12] Mohamed Hassan, Vasileios Choutas, Dimitrios Tzionas, and Michael J. Black. Resolving 3D human pose ambiguities with 3D scene constraints. In *ICCV*, Oct. 2019. 2

[13] Yana Hasson, Gül Varol, Dimitris Tzionas, Igor Kalevatykh, Michael J. Black, Ivan Laptev, and Cordelia Schmid. Learning joint reconstruction of hands and manipulated objects. In *CVPR*, 2019. 2

[14] Edmond S. L. Ho, Taku Komura, and Chiew-Lan Tai. Spatial relationship preserving character motion adaptation. *TOG*, 29(4), July 2010. 2

[15] Edmond S. L. Ho and Hubert Shum. Motion adaptation for humanoid robots in constrained environments. In *ICRA*, 2013. 2

[16] Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. *SIGGRAPH*, 2016. 6

[17] Jongmin Kim, Yeongho Seol, Hoemin Kim, and Taesoo Kwon. Interactive character posing with efficient collision handling. *Computer Animation and Virtual Worlds*, 31(3):e1923, 2020. 2

[18] Nikos Kolotouros, Georgios Pavlakos, Michael J. Black, and Kostas Daniilidis. Learning to reconstruct 3d human pose and shape via model-fitting in the loop. In *ICCV*, 2019. 2

[19] 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. ACM Press/Addison-Wesley Publishing Co., 1999. 2

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

[21] Zhiguang Liu, Antonio Mucherino, Ludovic Hoyet, and Franck Multon. Surface based motion retargeting by preserving spatial relationship. In *Proceedings of the 11th Annual International Conference on Motion, Interaction, and Games*, MIG '18, New York, NY, USA, 2018. Association for Computing Machinery. 2

[22] Etienne Lyard and Nadia Magnenat-Thalmann. Motion adaptation based on character shape. *Computer Animation and Virtual Worlds*, 19(3-4):189–198, 2008. 2

[23] Zoran Popović and Andrew Witkin. Physically based motion transformation. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH, pages 11–20, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co. 2

[24] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 4

[25] Davis Rempe, Leonidas J Guibas, Aaron Hertzmann, Bryan Russell, Ruben Villegas, and Jimei Yang. Contact and human dynamics from monocular video. In *ECCV*, pages 71–87. Springer, 2020. 5, 8

[26] Breannan Smith, Chenglei Wu, He Wen, Patrick Peluse, Yaser Sheikh, Jessica Hodgins, and Takaaki Shiratori. Constraining dense hand surface tracking with elasticity. *SIGGRAPH Asia*, 2020. To appear. 2

[27] Seyoon Tak and Hyeong-Seok Ko. A physically-based motion retargeting filter. *TOG*, 24(1):98–117, 2005. 2

[28] M. Teschner, S. Kimmerle, B. Heidelberger, G. Zachmann, L. Raghupathi, A. Fuhrmann, M.-P. Cani, F. Faure, N. Magnenat-Thalmann, W. Strasser, and P. Volino. Collision Detection for Deformable Objects. In *Eurographics*, 2004. 3, 5

[29] Dimitrios Tzionas, Luca Ballan, Abhilash Srikantha, Pablo Aponte, Marc Pollefeys, and Juergen Gall. Capturing hands in action using discriminative salient points and physics simulation. *IJCV*, 118(2):172–193, June 2016. 2, 3, 4

[30] Luis Unzueta, Manuel Peinado, Ronan Boulic, and Ángel Suescun. Full-body performance animation with sequential inverse kinematics. *Graphical Models*, 2008. 2

[31] Ruben Villegas, Jimei Yang, Duygu Ceylan, and Honglak Lee. Neural Kinematic Networks for Unsupervised Motion Retargetting. In *CVPR*, June 2018. 2, 6, 7

[32] Lingyu Wei, Qixing Huang, Duygu Ceylan, Etienne Vouga, and Hao Li. Dense human body correspondences using convolutional networks. In *CVPR*, 2016. 5

[33] Yan Zhang, Mohamed Hassan, Heiko Neumann, Michael J. Black, and Siyu Tang. Generating 3d people in scenes without people. In *CVPR*, June 2020. 2