Egocentric Mapping of Body Surface Constraints

Eray Molla, Henrique Galvan Debarba, and Ronan Boulic, *Senior Member, IEEE*

Abstract—The relative location of human body parts often materializes the semantics of on-going actions, intentions and even emotions expressed, or performed, by a human being. However, traditional methods of performance animation fail to correctly and automatically map the semantics of performer postures involving self-body contacts onto characters with different sizes and proportions. Our method proposes an egocentric normalization of the body-part relative distances to preserve the consistency of self contacts for a large variety of human-like target characters. Egocentric coordinates are character independent and encode the whole posture space, i.e., it ensures the continuity of the motion with and without self-contacts. We can transfer classes of complex postures involving multiple interacting limb segments by preserving their spatial order without depending on temporal coherence. The mapping process exploits a low-cost constraint relaxation technique relying on analytic inverse kinematics; thus, we can achieve online performance animation. We demonstrate our approach on a variety of characters and compare it with the state of the art in online retargeting with a user study. Overall, our method performs better than the state of the art, especially when the proportions of the animated character deviates from those of the performer.

Index Terms—performance animation, retargeting

1 Introduction

Performance animation has a variety of applications such as computer animated puppetry [1], [2], [3], [4], virtual reality based rehabilitation [5] and cognitive neuroscience [6]. Such applications require mapping the motion of the performer onto avatars without losing the purpose and the meaning of the performed postures. Indeed the relative location of human body parts often determines the semantics of on-going actions, intentions, and even emotions expressed by a human being [7], [8], [9]. However, traditional methods of performance animation fail to correctly map the semantics of performer postures involving self-body contacts onto characters with different sizes and proportions. This can be critically adverse to the development of computer games engaged by full-body movements [10]. In the present paper, we focus on the class of movements involving self-interactions. We propose an egocentric normalization of the body-part relative distances to preserve the consistency of self-contacts for a large variety of human-like target characters. We also consider the interaction with the ground as an external entity expressed within the egocentric coordinates; other interactions, for instance, between characters, are beyond the scope of this paper. In the remainder of this paper, we also refer to the target character as an avatar.

Simply assigning the captured performer joint angles to the avatar pose works fine whenever there is no self-body interactions, as is also the case for a normalized posture representation [11]. However, it quickly produces artifacts when there is self-contact. For instance, consider a motion where the hands are initially kept in front of the belly and, then moved towards it. If we naively map this motion onto an avatar with a big belly, the hands will touch the belly much earlier than in the performed motion (Figure 2). On the other hand, if a character is thinner than the performer, then the hands would not meet the belly at all. This causes a visible artifact on the transferred movement. Requesting the performer to adjust his movements to match the target avatar would result in a heavy cognitive load. Other cases of potential self-interaction mismatches include goal-oriented contact with other body parts such as clapping, washing, dressing and some postures used in non-verbal communication. For this reason, we choose to offer a general framework able to transfer the performer self-interaction constraints to any human-like avatars while the movement is being performed.

We propose an egocentric normalization of posture to encapsulate the spatial relationships between the limbs and the other body parts. It relies on an additional offline body surface calibration step after a standard skeleton calibration stage [12]. During the online interaction the calibrated body surface of the performer is used to infer the instantaneous bodycentric coordinates of the limbs. Given any surface-calibrated character, the bodycentric coordinates can be efficiently converted into Cartesian constraints to recover on the fly the avatar character body posture with Inverse Kinematics (IK).

The main technical contributions of this paper are:

1) We introduce a character independent normalization of the posture with respect to the body surface and the flat ground called egocentric posture representation. It explicitly encodes the body parts spatial order, i.e. the fact that body parts might be locally organized into layers, with the novel concept of egocentric planes.
2) We propose a mapping of this representation transferring the original spatial order on any human-like target characters that have gone through a simple offline body surface calibration stage.

• *E. Molla and H. Galvan Debarba were with the Immersive Interaction research Group, Ecole Polytechnique Federale de Lausanne, 1015, Switzerland.*

• *R. Boulic leads the Immersive Interaction research Group, Ecole Polytechnique Federale de Lausanne, 1015, Switzerland.*

Manuscript received to be added; revised to be added. Corresponding author: R. Boulic (email: ronan.boulic@epfl.ch).

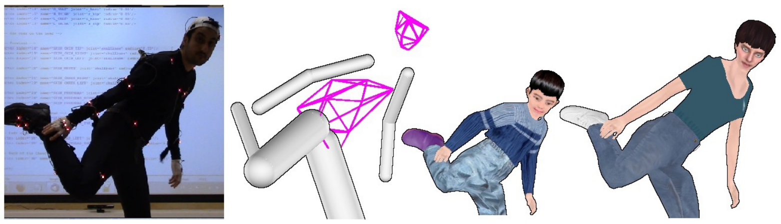

Fig. 1. Our technique normalizes a posture with or without contact with respect to the body surface and maps it onto characters with different size and proportion by preserving their relative location.

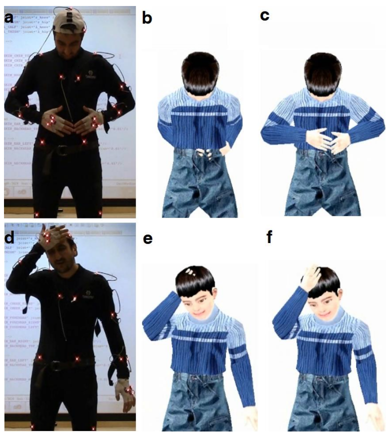

Fig. 2. Variation in proportion and thickness between the performer and the avatar causes artifacts in case of self-body interactions. (a, d) Performed postures. (b, e) Resulting postures with Kulpa et al's [11] technique. (c, f) Our results.

3) We ensure a smooth, history independent, online transfer of the performer motion to a wide variety of human-like avatars.

The next section highlights key prior work in motion retargeting and performance animation. We then provide a technical overview of our approach before describing successively the calibration process, the egocentric normalization and denormalization, and posture adaptation. Various comparative results are presented prior to a conclusion.

2 Related Work

Computer puppetry [1] and motion retargeting [13] have a long history but were seldom united in the same contribution except for the landmark paper from Shin et al. [2]. In their work, the concept of importance was introduced to handle anatomic differences between the performer and the target character when interacting with external constraints such as the ground. However, self-interaction constraints are not taken into account.

On the other hand, motion retargeting is generally an offline process performed by animators in a desktop environment where constraints are scripted, hence known in advance [3], [14], [15], [16], [17]. The animator can explore the design space by trial and error until a satisfactory motion transfer is obtained on the target character. Recent research efforts have also proposed attractive methods based on topological coordinates for the (offline) retargeting of complex multi-agents interactions. However these methods are either costly [18] or not suited for online performance animation [19]. Likewise, techniques allowing the deformation transfer between surface meshes are also still far from real-time [20], [21].

Another challenge is mapping the captured motion onto characters with different topology [22], [23], [24]. Even though these techniques allow the animation of non-humanoid characters from human input, they require a dataset and some significant offline machine learning processes to derive a mapping function. Moreover, they do not take advantage of the a priori knowledge about the morphology of the target characters, human-like topology in our problem, which is exploited in our efficient IK solver.

There are further issues to consider in retargeting when it comes to preserving self-contact [25]. Al-Asqhar et al. [26] introduce an efficient technique to express the position of the joints with respect to the descriptor points computed in the original motion. This approach can adapt close interactions to new geometries and the character morphology. However, considering only the joint relative distances is not sufficient; it could result in changes among the spatial order of the body segments such as side switching in complex poses, e.g. crossed legs or arms [27]. We address this problem by introducing the concept of egocentric plane that materializes the instantaneous separating plane between each pair of body parts. Moreover, the problem of transferring the motion both with and without close interactions for computer puppeteering has not been explored yet, which is the main topic of our research. Considering geometric relationships between body parts also performs well on human action recognition [28]; however its use has not been investigated for the motion retargeting problem.

Recently, a spatial parameterization based on electric coordinates has been proposed [29]. Although this parameterization can be used to define grasp postures and to robustly map them to a variety of target hands and objects [30], the computation of the electric coordinates is not fast enough for interactive applications.

The full-body real-time control of human-like avatars is different in the sense that the system should relieve as much as possible the performer from the burden of the performer-to-avatar mapping process while interacting in real-time in a virtual environment. TV shows and theme park attractions displaying live virtual characters would greatly benefit from a more transparent posture transfer preserving the performer body posture semantics [1], [2].

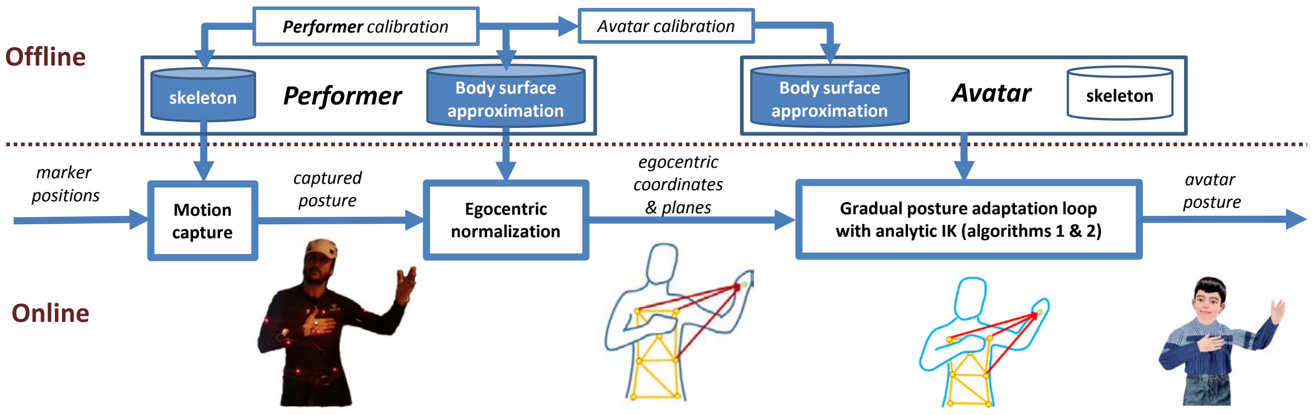

Fig. 3. System overview. First, the offline process consists in the performer calibration that identifies the performer skeleton and builds an approximation of the performer body surface from a set of points sampled by the performer. The avatar calibration builds the avatar body surface approximation from the corresponding set of sampled points on the avatar body surface. Second, our online motion retargeting algorithm is completed in three steps (under the dashed line). The performer's pose is captured (step 1) and converted into a set of morphology independent body surface constraints (step 2). They are mapped onto any human-like avatar through a gradual posture adaptation loop (step 3) aiming at preserving the self-body contacts and the same order of body parts as in the source performance (spatial order).

3 Overview

Figure 3 shows an overview of our retargeting process. It is composed of a short offline stage (above the dashed line) consisting of the performer and avatar calibrations, and the online algorithm ensuring the motion retargeting.

In the performer calibration step, the skeletal structure and the performer body surface approximation are extracted. The avatar calibration is performed once by picking the vertices that correspond to the performer body surface approximation to have a one-to-one mapping between the performer and the avatar.

Our online retargeting algorithm is completed in three steps. First, the pose of the performer is captured. Second, the egocentric normalization is applied to the captured posture to express the relative location of body parts with respect to each other and to the ground. Third and finally, a gradual posture adaptation loop progressively enforces the position constraints and spatial order of body parts resulting from the denormalized representation with respect to the target character (avatar). This last steps relies on efficient analytical IK techniques to ensure a real-time display of the avatar posture.

4 Calibration

Our method relies on the calibrations of the performer and the avatar.

4.1 Performer Calibration

The performer calibration is completed in three steps: skeleton fitting, hands calibration and the body surface calibration. A complete calibration takes around seven minutes.

Joint Center and Segment Length Estimation: We used the skeleton fitting technique introduced by Silaghi et al. [12]. The performer makes a series of gym motions covering all the Degrees of Freedom (DoF). By using relative motion of the neighboring body segments, a skeleton model matching the performer's morphology is determined.

Calibrating Hands: After the gym motion, the performer places his palms onto a flat target of known position and orientation. The surface of the hands are inferred from the contact with this plane. Note that using an external object for capturing accurate contact has also been exploited by previous work [31]. In their work, they used for registering kinect and magnetic input devices to capture grasping gesture in a small volume.

Calibrating the Body Surface Approximation: The performer brings his hands onto a small number of locations on his body surface as partially shown with yellow dots on the conceptual illustration of Figure 4 left image. This set of manually sampled points is used to build the performer body surface approximation consisting of two crude meshes, one for the trunk and one for the head, visible in magenta on Figure 4 middle image. The approximation of the body limb segments is made through capsules (Figure 4 middle image). We exploit the motion capture markers carried by each limb segment; their average distance to the skeleton segment is retained as the capsule radius.

Body Surface Approximation Strategy: The number and the locations of the manually sampled points in the body surface calibration step need to be chosen to include the most important self interaction locations on the trunk and the head. The density of the sampled points is higher for the head because human are more sensitive to a small mismatch for this perceptually dense region compared to the trunk [32]. That being said the crude approximation illustrated in Figure 4 left and middle images is sufficient for the range of actions we have explored.

4.2 Avatar (Target Character) Calibration

In the avatar calibration step, we mark the correspondences of the sampled points on the avatar mesh (yellow dots on Figure 4 left and middle images, and Figure 5). In addition, we also pick a few points on the limb segments surface (gray dots on Figure 5) to compute their average thickness to constitute the radius of their capsule approximation. This step takes less than a minute.

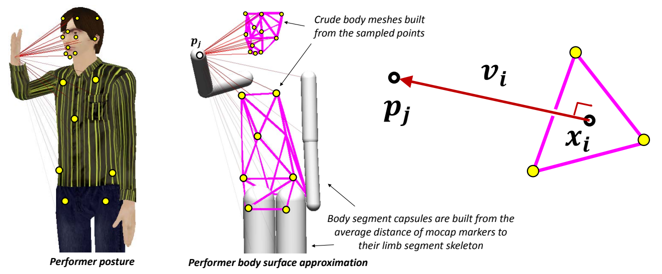
Crude body meshes built from the sampled points
p_j
v_i
p_j
x_i
Body segment capsules are built from the average distance of mocap markers to their limb segment skeleton
Performer posture
Performer body surface approximation

Fig. 4. **Performer body surface approximation**: Conceptual illustration of the performer body (left) with a partial illustration of the manually sampled points on his body surface (yellow dots) to build the two crude meshes approximating the trunk and the head (in magenta, middle). Red vectors are the **surface relative displacement vectors**, $\mathbf{v}_i$, for the right wrist. Each vector originates from a **reference point**. A wrist **reference point**, $\mathbf{x}_i$, is the projection of the wrist's position, $\mathbf{p}_j$, to the corresponding **surface primitive**, i.e., the triangles of the crude body mesh (in magenta, middle and right) or the limb capsules (in gray, middle). The red intensity of the line depicts the normalized importance $\hat{\lambda}_i$ of the relationship.

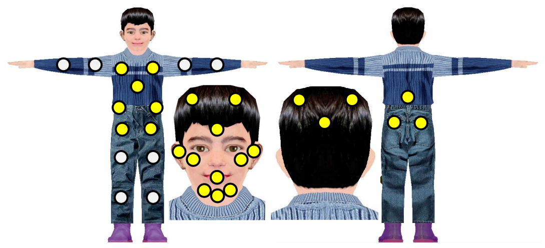

Fig. 5. The yellow dots indicate the sampled points on the avatar's body that have to correspond to the ones sampled on the performer body to build the two crude meshes for the trunk and the head. Additional points, gray dots, are picked on the avatar limbs mesh to compute the average radius of each limb segment capsule approximation.

5 Egocentric Posture Representation

The egocentric representation of a posture at any point in time t is expressed by four families of information:

1) **The captured joint angles of the performer:** Joint angles are the traditional parameters for representing an articulated body posture. They can be captured by any full-body IK solver. We used the one proposed in [33].
2) **The normalized root reference point, $\widehat{\mathbf{g}}$, and the height of the root, $\widehat{h}_{root}(t)$:** We assume that the root node of the performer and avatar skeletons are located at the base of the spine and close to the hip joints, hence making it the best node to characterize the body interaction with the ground. For this reason, we define the **root reference point**, **g**, as the projection of the **root position**, **G**, on the flat ground (Figure 6). We normalize both **g**, and the instantaneous height of root, $h_{root}(t)$; by the height of root in the standing posture, h_{root}:
$$\widehat{\mathbf{g}} = \frac{\mathbf{g}}{h_{root}}, \; \widehat{h}_{root}(t) = \frac{h_{root}(t)}{h_{root}} \tag{1}$$
Therefore, the height and the velocity of the root can be adapted to the size of the target character by reversing Equation 1.
3) **The set of egocentric coordinates of the limbs' joints:** This concept was introduced by Al-Asqhar et al. [26]. In our context, its purpose is to encode the relative location of body parts with respect to each others (Sections 5.1 and 5.2).
4) **The set of egocentric planes of the limbs' segments:** We introduce the concept of egocentric planes to encode the separating planes between each pairs of body parts. Its purpose is to preserve the body parts spatial order in the performer to avatar mapping process (Section 5.3).

5.1 Egocentric Coordinates of a Limb Joint

We decompose the position of each limb joint into a weighted sum of **surface relative displacement vectors v** as shown on Figure 4 right image. These vectors are computed by projecting the joint's position **p** on the m components of the body surface approximation consisting in the set of triangles of the crude trunk and head meshes and the set of the limb segment capsules. We define each projected point as a **reference point x**. The importance λ of a reference point is inversely proportional to the magnitude of the displacement vector (see Appendix A for details). The importance values are normalized linearly such that their sum is equal to one. Therefore, a joint's position, $\mathbf{p}_j$, can be expressed as
$$\mathbf{p}_j = \sum_{i=1}^{m} \hat{\lambda}_i (\mathbf{x}_i + \mathbf{v}_i) \tag{2}$$
We store this representation in a normalized form to allow their adaptation to the size, proportion and body surface of a target character. In particular, each reference point **x** is converted into barycentric coordinates for the crude meshes triangles and into normalized cylindrical coordinates for the limb segment capsules.

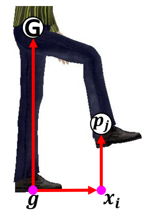

Fig. 6. Foot step normalization

The ground projection of each foot's position is also considered as a reference point for the corresponding foot ($\mathbf{x}_i$ in Figure 6). This allows handling the ground contact. This reference point is stored relative to the root reference point, **g**, and its spatial relationships with **g** and the foot position, $\mathbf{p}_j$, are normalized by the root height so that the footsteps can be adjusted to the avatar's size like in [11] (Figure 6).

Encoding Limb Extremities Orientation: We decompose the surface normals of the limb extremities, i.e., the hands and the feet, into a weighted sum of surface relative angular deviations as
$$\theta_{\mathbf{n}_j} = \sum_{i=1}^{m} \hat{\lambda}_i (\theta_{\mathbf{n}_i} + \beta_i) \tag{3}$$
where $\theta_{\mathbf{n}_j}$ and $\theta_{\mathbf{n}_i}$ are the orientations of the normals of the extremity and a reference

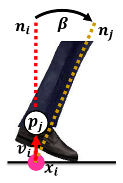

Fig. 7. The decomposition of the foot normal.

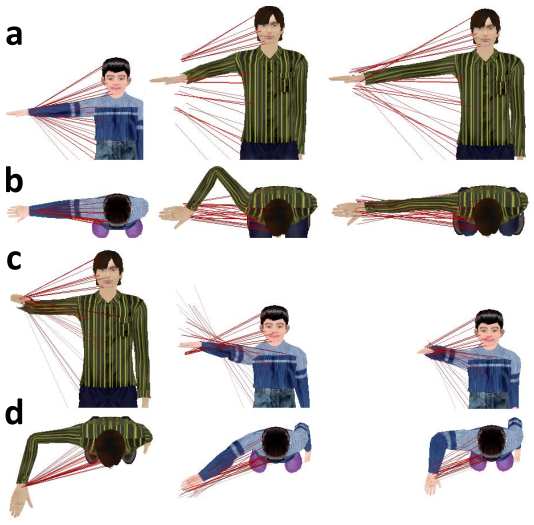

Fig. 8. Mapping the raw captured relative vectors **v** causes artifacts in case of size and proportion differences. Left: The "performer" pose. Middle: Results without normalization on the "target" characters [26]. Right: Our results. (a, b) and (c, d) are the front and top view of the same poses, respectively.

point, respectively; and β_i is the smallest rotation to align them (Figure 7). We keep all β_i values and use them to adapt the extremity's normal onto the target character's body surface, using Equation 3, after mapping the reference points and computing their normals.

Finally, we normalize the surface relative displacement vectors **v** owing to a novel morphology independent approach as detailed in the next section.

5.2 Kinematic Path Normalization

Size and proportion differences between the performer and the avatar cause artifacts if the surface relative displacement vectors **v** (Figure 4) are mapped as they are captured. For instance, a tall character cannot reproduce fully extended arm postures with the surface relative displacement vectors coming from a small performer (Figure 8 (a, b)). Conversely, if the surface relative vectors of a tall performer are mapped onto a small avatar, they point outside the target character reach space for most poses (Figure 8 (c, d)).

To overcome these problems, we propose the kinematic path normalization (Figure 9). We define a kinematic path as the set of joints $\{j_0, j_1, ..., j_n\}$ and the vectors along the bone segments $\{\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_n\}$ between the parent joint j_0 of the reference point $\mathbf{x}$ and the pointed limb joint j_n. $\mathbf{s}_i$ denotes the vector pointing from j_{i-1} to j_i. Hence each relative displacement vector **v** can be expressed as a sum of the segment vectors along the kinematic path (Figure 9).

The proposed normalization first computes the contribution of each segment vector $\mathbf{s}_i$ to **v** by projecting it on **v** as $\|\mathbf{s}_i\| \cos(\alpha_i)$ where

$$\cos(\alpha_i) = \frac{\mathbf{v}}{\|\mathbf{v}\|} \cdot \frac{\mathbf{s}_i}{\|\mathbf{s}_i\|} \tag{4}$$

Then we compute a normalization factor τ and a vector $\hat{\mathbf{v}}$ such that:

$$\mathbf{v} = \tau\hat{\mathbf{v}}, \text{ with } \tau = \sum_{i=1}^{n} \|\mathbf{s}_i\| \, |\cos(\alpha_i)| \tag{5}$$

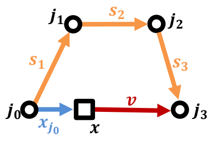

Fig. 9. The kinematic path of the surface relative displacement vector **v** is expressed by $\mathbf{v} = -\mathbf{x}_{j_0} + \sum_{i=1}^{n} \mathbf{s}_i$ where $\mathbf{x}_{j_0}$ is the vector from the parent joint j_0 to the reference point **x**.

Note that in case of near contact, Equation 4 could cause instabilities, because $\|\mathbf{v}\| \approx 0$. However, this case can easily be handled because the mapped vector is also approximately zero. The proposed approach has important properties. First, Equation 5 always leads to a positive τ, since the normalization coefficients are non-negative. It prevents the vectors **v** from switching sides, which could cause penetration. Second, it generalizes the concepts of body and limb scaling approach from [11] as can be seen on Figure 8 right column. Compared to [11], our method avoids the artifacts resulting from proportion differences (e.g. Figure 10).

We keep $\hat{\mathbf{v}}$ and the set of normalization coefficients $C = \{|\cos(\alpha_1)|, |\cos(\alpha_2)|, ...|\cos(\alpha_n)|\}$. Under this normalized form, the relative displacement vectors can be adapted to the segment lengths of the target character by reversing Equation 5.

Summary: We define the egocentric coordinates of a joint E_j as a set of the m tuples

$$E_j = \{\mathbf{e}_{j,1}, \mathbf{e}_{j,2}, ..., \mathbf{e}_{j,m}\} \tag{6}$$

with each tuple composed of five elements: $\mathbf{e}_{j,i} = (\hat{\lambda}_i, \hat{\mathbf{x}}_i, \hat{\mathbf{v}}_i, C_i, \beta_i)$ expressing the normalized parameters with respect to the component i of the body surface approximation.

5.3 Egocentric Planes

Egocentric coordinates encapsulate the relative location of the limb joints with respect to the other body parts. However, they do not include an explicit representation of the **spatial order** of body segments, i.e. the relative organization of two body parts in a simple contact and, more generally, the fact that multiple body parts might be locally organized into layers. Moreover, even with our kinematic path normalization, the set of surface relative displacement vectors **v** obtained for the target character can still be slightly

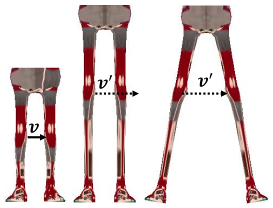

Fig. 10. Left: The source pose. Middle: The target character with the same pelvis width, but longer legs. $\mathbf{v}'$ is obtained with the limb length used as the normalization coefficient. Right: An artifact is obtained by enforcing the $\mathbf{v}'$ constraint. Conversely, with our approach, the legs being orthogonal to **v**, their contribution is null with our kinematic path normalization, resulting in the middle posture for the target character.

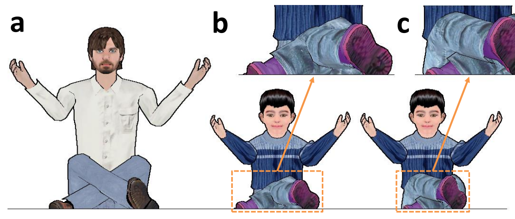

Fig. 11. Using only the egocentric coordinates could cause change in the limbs' spatial order of the mapped posture. a) Performer adopting a yoga pose. b) Mapping only with egocentric coordinates, similarly to Al-Asqhar et al.'s [26] work. c) Our result.

spread in space as can be seen on Figure 8 right column. The resulting target joint position obtained through their weighted average with Equation 2 may violate the spatial order adopted by the performer. For instance, Figure 11a shows a yoga pose considered as the performer posture: the high importance λ of both feet constraints (due to their proximity with the ground) can compromise the enforcement of lower importance constraints such as the knees and result in the legs penetrating each other (Figure 11b). To avoid such artifacts, we introduce the novel concept of egocentric plane and use it to ensure the correct spatial order (Figure 11c).

Egocentric planes associated to two limb segments: Given two limb segments modelled as capsules, ls_i and ls_j, we define the egocentric plane of ls_i against ls_j as the tangent plane of ls_j crossing its closest point to ls_i (Figure 12a). This plane $ep_{i|j}$ builds a half-space where the plane's normal $\mathbf{n}_{ep_{i|j}}$ points towards ls_i. Such an egocentric plane remains stable in the capsule local coordinate system when the capsule closest points switch from one end to the other end of the cylindrical section (Figure 12a-c) or when the capsules become parallel (Figure 12b).

The intersection of all half-spaces from other limb segments corresponds to the convex region within which ls_i can move without changing the spatial order (Figure 12d). Therefore, we map these planes on the avatar and use them to preserve the spatial order. For **mapping the egocentric planes**, we convert each closest point to normalized cylinder coordinates. These coordinates are mapped on the avatar and their tangent planes are used to constrain the segment. Note that for each pair of segments, we obtain two separating planes. We exploit this fact to relax the mapped half-space constraints as detailed in the Appendix B.

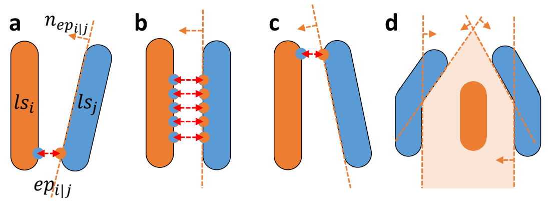

Fig. 12. Egocentric planes explained in 2D. Orange and blue dots are the corresponding closest points. a, b, c) ls_j is barely rotated counterclockwise. The pair of closest points change quickly, although the angular variation is small. However, the separating planes remain continuous. d) Placing the orange segment within the intersection of all half-spaces (light convex region) does not change the spatial order.

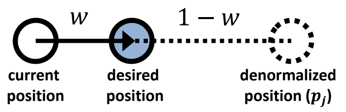

Fig. 13. **Attracting each joint** towards its denormalized position in proportion to w.

6 Posture Adaptation

The previous section has described the output of the **Egocentric normalization** which is the second step of our online process (Figure 3). From this normalized representation of the performer, the present step is in charge of converging, in real-time, to an avatar posture that preserves the relative body parts location displayed by the performer.

An overview of this step is presented in Algorithm 1. The first step is a posture initialization step that prepares the main convergence loop. This main loop is composed of two stages. Firstly, a limb convergence loop that gradually enforces the position constraints and the spatial order of the limbs' segments. This loop is driven by the interpolation parameter w, varying from 0 to 1 (Figure 13). Such a progressive constraints enforcement allow mutually dependent limbs to all contribute and to ensure a smooth convergence. Secondly, the body folding stage is responsible for contributing to resolve the constraints between the upper and the lower limbs, if necessary. This completes one step of the posture adaptation main loop. Once the joint positions have converged, the orientation of the extremities are adjusted to align with the body surface.

```
Data: Avatar self-interaction constraints (Sec. 6)
Result: Avatar Posture
Avatar posture initialization (Sec. 6.1);
while adaptation is not complete do            // main loop
    $lCount = 0$;
    while $lCount{+}{+} \leq maxLCount$ do         // limb loop
        $w = lCount/maxLCount$;
        foreach limb do
            Mapping egocentric coordinates of limb's
            joints (Eq. 2, 6);
            Attracting each joint towards $\mathbf{p}_j$ in
            proportion to $w$ (Fig. 13);
            Adapting limb pose to desired joint
            positions (Sec. 6.2);
            Mapping the egocentric planes of each
            segment (Sec. 5.3);
            Activation of relaxed egocentric planes in
            proportion to $w$ (Fig. 14);
            Ensuring spatial order of each segment by
            the desired (activated) constraint planes
            (Sec. 6.3);
        end
    end
    Body folding (Sec. 6.4);
end
Adapting orientation of extremities to the surface
(Sec. 6.5);
```

Algorithm 1: Posture adaptation on the target character.

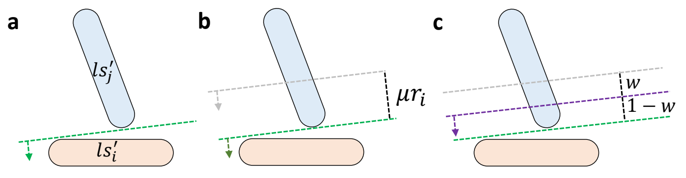

Fig. 14. **Progressive activation of relaxed egocentric planes**. a) The green line is the relaxed half-space constraint as detailed in the Appendix B. b) The gray line is obtained by moving the green line towards the opposite side of its normal by a multiple μ of the radius of ls'_i. c) The desired (activated) constraint plane (purple) is obtained by interpolating the gray and green planes in proportion to w.

6.1 Avatar Posture Initialization

The initialization is crucial to prevent local minima. In our context, a typical local minimum preserves the most important characteristics such as contacts but it fails to reflect secondary ones such as the spatial order of body parts. Consider the seated posture from Figure 15a. In this pose, the source (performer) character hands are in contact and onto each other, in front of the torso and above the legs, but are kept away from them. The proportion differences between the source and the target (avatar) characters produces the avatar posture Figure 15b with the performer captured joint angles. With our approach, by construction, the contact relationship between the hands outweighs the other ones. The posture adaptation algorithm starts from the captured joint angle posture and attempts to preserve the hands' contact while minimizing the arm joints angular deviation from this initial posture. The consequence is the local minimum from Figure 15c. Therefore the goal of the initialization step is to build an initial posture that is not only compatible with the posture subspace defined by the egocentric planes (Figure 15d), but that also tries to take advantage of the posture space redundancy to reflect as much as possible the performer posture (Figure 15e).

Initialization Process: We start the initialization by bringing the root of the target character onto its denormalized position (Equation 1). Then, we assign the captured joint angles onto the target character. To avoid the local minima issue expressed above, we compute the initial arm joint target positions $\mathbf{p}_i$ by slightly modifying Equation 2. First, we ignore the spatial relationship between the arms because they tend to dominate the adaption process too

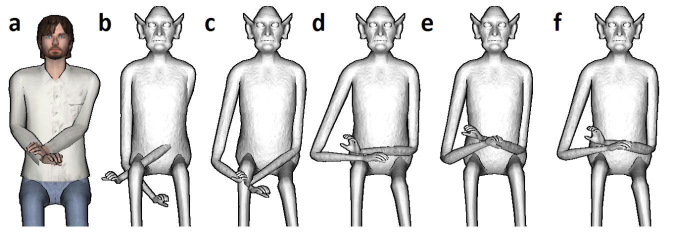

Fig. 15. Initialization and the local minima problem. The target character (b-f) has longer arms than the source. a) The source pose. b) Mapping the joint angles. c) The final pose adapted from b. The spatial order is broken without the egocentric plane constraints. d) c with spatial order constraint. It does not keep any distance between the legs and the hands. e) Our initialization. f) The final pose resulting from e. Notice the distance between the hands and the legs.

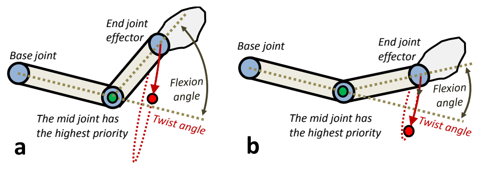

Fig. 16. Limb IK with reversed priorities where the mid joint position constraint is enforced with highest priority whereas the end joint effector position is enforced only with the remaining redundancy: in these examples the base joint has already been rotated to bring the mid joint to its target (green dot). a) Only the base joint twist angle (red) and the mid joint flexion angle (brown) can be used to bring the end joint as close as possible to its desired position (red dot). b) An ill-conditioned configuration is characterized by a small flexion angle potentially inducing a large twist angle to enforce even a small amount of displacement of the end joint (red arrow) ; for this reason the twist angle magnitude is bounded (dotted brown arrow) by the current flexion angle magnitude.

much whenever they are close to each other. We use only the reference points $\mathbf{x}$ and denormalized surface relative vectors $\mathbf{v}$ computed on other body parts. Then, we project each individual component $\mathbf{p}_i$ of the joint position, i.e. $(\mathbf{x}_i + \mathbf{v}_i)$, to the region boundary determined by the leg half-planes if that vector $\mathbf{p}_i$ points outside this region. After computing the initial arm joints' position constraints $\mathbf{p}_j$, from the set of potentially projected components $\mathbf{p}_i$ and Equation 2, we obtain the corresponding initial arms' posture from the approach described in the next section.

6.2 Adapting Limb Pose to Position Constraints

We describe here the inverse Kinematics (IK) approach used to synthesize the limbs' posture from the denormalized position constraints $\mathbf{p}_j$ expressed for their mid (elbow or knee) and end (wrist or ankle) joints. This approach is exploited both for the posture initialization step and within the limb convergence loop in charge of gradually enforcing the constraints. Since satisfying both the mid and end joint constraints is usually not possible, we introduce the following interpolation scheme.

Interpolating standard and reversed limb IK solutions: First, we use a well-known limb IK [34] to compute the posture that enforces the end joint effector location (first priority); we use the arm redundancy to position the mid joint as well as possible but without altering the end effector location (second priority). Second, we reverse the priorities, i.e. positioning the mid joint is given the highest priority whereas positioning the end effector is done as well as possible but without altering the mid joint location (Figure 16a). This is achieved as follows: the base joint, i.e. shoulder or hip, is rotated to bring the mid joint onto the closest point to its target. Then, the remaining two mechanical degrees of freedom (DoF), twist of the base and the flexion of the mid joint, are used to align the end joint with its target. In the third and final step, we slerp these two postures in proportion to the maximum importance values λ_i of the respective end and mid joints as detailed in Appendix C.

The satisfaction of a second priority position goal can suffer from ill-conditioned configurations around nearly full-extended limb postures as noted in [35] and illustrated in Figure 16b. In such a context, the available redundancy provided by the base joint twist angle cannot contribute much to achieve a secondary task and may even cause

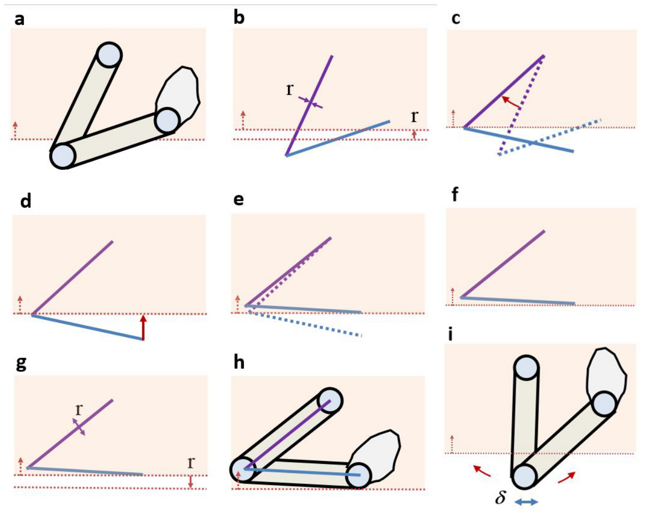

Fig. 17. Handling a single half-space constraint for both segments. The semi-transparent region designates the valid side of the half-space. We assume both segments have the same radius for the ease of illustration. a) Initial configuration where both segments are not completely within the half-space valid side. b) We transform the problem to a half-space constraint for line segments by deflating both the segments and the half-space by the segment radius r. c) The shoulder is rotated to bring the elbow on the half-space plane. d-f) The wrist is brought to the closest point on the half-space using analytic arm IK. This operation makes sure that both joints are contained within the valid region. g) The deflation operation is reversed to observe the final result. h) Both limb segments' capsules are within the half plane. i) example of ill-conditioned context where the upper segment is nearly perpendicular to the half-space constraint plane.

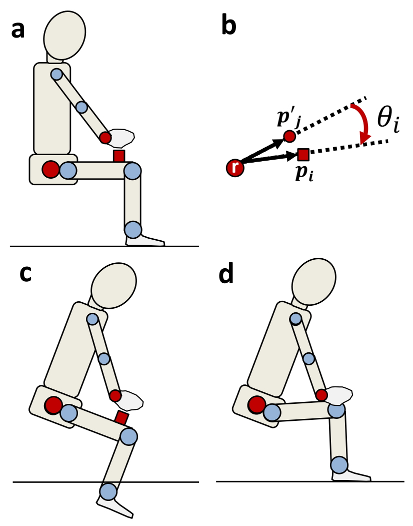

Fig. 18. Body folding considering the relationship between a thigh and a wrist. a) The arm is not long enough to bring the wrist onto its target (square). b) Root **r** needs to be rotated by θ_i to align the position of the wrist obtained from the *limb loop*, $\mathbf{p}'_j$, with its desired position $\mathbf{p}_i$. c) Rotating the root affects the whole body. d) The head is rotated back to preserve its global orientation and the feet are brought into their initial positions by using leg IK.

instability. To overcome this issue, we bound the use of of this twist angle to the magnitude of the flexion angle. Hence, in the fully-extended posture, i.e. the flexion angle is zero, the twist is not allowed to contribute at all. This ensures continuity even if the character has significant proportion differences, i.e. shorter or longer limbs.

6.3 Ensuring Spatial Order with Egocentric Planes

As illustrated in Figure 12d, the intersection of mapped egocentric half-planes forms a convex region for each limb segment. Therefore ensuring the correct spatial order requires bringing each segment within its mapped valid region.

For each half-space constraint, similarly to [36], we first handle the limb upper segment alone prior to constrain the full limb with analytical IK as illustrated on Figure 17. For this, we start by deflating both the capsules and the half-space by the capsule radius of the segment. This results in an equivalent problem in which we deal with line segments rather than volumes (Figure 17a-b). First, we rotate the base-joint, i.e. shoulder or hip, to bring the mid-joint, i.e. elbow or knee, on the surface of the half-space plane (Figure 17c). Then, we use analytical IK to compute the joint angles which bring the end joint, i.e. wrist or ankle, to the closest point on the plane surface (Figure 17d-e). As a consequence the constraint is satisfied as both joints are contained within the valid region (Figure 17f). Figure 17g-h illustrate the solution back with the segment volumes.

It is important to note that when resolving a half-space constraint for the upper segment of the limb, it is sufficient to bring the mid-joint to the boundary (Figure 17a-c). On the other hand, a half-space constraint for the lower segment is resolved when both the mid-joint and the end-joint are contained within the half-space. Therefore, the full-sequence of proposed adjustment needs to be applied.

Ill-conditioned context of the base joint rotation: When the upper segment is nearly perpendicular to the constraint plane, the solution we propose becomes ill-conditioned (Figure 17i). Indeed, in such a singular context, a small displacement δ of the mid joint position, from one update to the next, may induce large rotation of the base joint in opposite directions. To prevent such discontinuities, the base joint rotation is bounded by the angle made between the upper segment direction and the half plane normal. That angle being close to zero in the singular context, the base joint is almost not involved in the resolution of the half space constraint for the upper segment in this context. This approach may lead to some partial penetrations but ensures the motion continuity which we consider much more critical for the quality of the produced avatar movement.

6.4 Body Folding Stage

The first stage of the *main loop* in Algorithm 1 only controls the limbs. As a consequence, the analytic limb IK may not be sufficient to handle the interaction between the arms and the legs. This is the case, for instance, when the avatar has short arms compared to its torso. The body folding stage is in charge of handling such scenarios (Figure 18).

Algorithm 2 presents an overview of the body folding stage. We consider each interaction possibility between the left/right arm joints, i.e. elbow and wrist, and the left/right leg segments, i.e. thigh and calf. Given a leg segment with index i, and an arm joint with index j, we first compute the desired position of the joint $\mathbf{p}_i$ with respect to that segment only by adding the mapped displacement vector $\mathbf{v}_i$ to the reference point on that segment, $\mathbf{x}_i$ (Section 5.1). Then we

```
Data: Egocentric coordinates of the limbs' joints
Result: Adapted Pose with body folding
$\theta = 0$ ;
$\lambda = 0$ ;
foreach legSegment with index $i$ do
    foreach armJoint with index $j$ do
        $\mathbf{p}_i = \mathbf{x}_i + \mathbf{v}_i$ ;
        $\mathbf{rp}_i = \mathbf{p}_i - \mathbf{r}$ ;
        $\mathbf{rp'}_j = \mathbf{p'}_j - \mathbf{r}$ ;
        $\theta_i = PitchRotationFromTo(\mathbf{rp'}_j, \mathbf{rp}_i)$ ;
        $\theta = \theta + \lambda_i \times \theta_i$ ;
        $\lambda = \lambda + \lambda_i$ ;
    end
end
$\theta = \theta / \lambda$ ;
Rotate the root by $\theta$ ;
Rotate the head by $-\theta$ ;
Use leg IK to bring the feet to their initial state ;
```

Algorithm 2: Body folding.

compute the rotation θ_i which aligns the vector connecting the root $\mathbf{r}$ and the position of the arm joint as a result of the *limb loop*, $\mathbf{p'}_j$, with the desired position (Figure 18b). A weighted average, θ, for all possible combinations is computed considering the corresponding importance value, λ_i. θ is used to apply a relative 1DoF forward-backward (pitch) rotation to the root (Figure 18c). Finally, we rotate the head to preserve its initial orientation and use leg IK to bring the feet to their initial positions (Figure 18d). This operation adjusts the distance between the arm joints and the leg segments so that the arm joints can achieve their targets in the next iteration. Note that simply rotating the hips with the opposite value of θ is not sufficient as the root rotation changes the hip positions slightly, that is why we use the leg IK to adjust the whole leg state.

Exploiting the full 3D root rotation could cause some side effects like jerkiness if the performer does not actually intend to bend, e.g., walking. Eliminating such artifacts elegantly requires a higher level knowledge about the context of the performed task and time coherence, which is out of the scope of this paper. On the other hand, we propose two efficient techniques to improve the results, although they are action-dependent. First, we limit the use of the root to only the pitch rotation so that the bending is restricted to the sagittal plane. Second, we assume that the body folding stage is not suited for retargeting the class of postures close to standing postures, e.g. locomotion. For this reason we bound the root rotation involved in body folding by the minimal angular variation necessary to bring the hips to the standing posture. As a result, the root rotation is prevented in the standing posture as this variation is null whereas it is progressively allowed as the posture is closer to a seated posture.

6.5 Adapting Orientation of Extremities

After the positional goals are handled, we apply the smallest rotation to each end joints to align their extremity's surface normal with the avatar's one, computed using Equation 3.

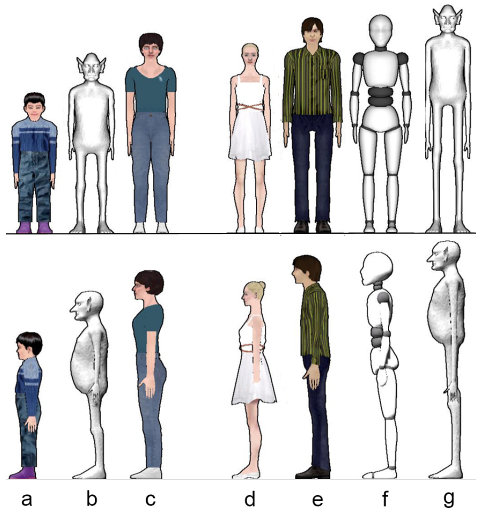

Fig. 19. Set of test characters with different body height and proportions, including for the user study (a, b, c).

7 Results

We have evaluated the performance of our technique in two ways. First, some performance animation tests are presented that include a variety of self-contacts and characters (Figure 19). Second, we have conducted a study comparing our approach to two other state of the art methods with respect to their ability to correctly reflect the semantic expressiveness of a set of nine movements.

Performance: for our tests we have used a computer with Core 2.67 GHz CPU with 4GB of memory. We have set $maxLCount = 3$ and limited the full body adaptation to two iterations. It runs about 40FPS without any performance optimization. Since we consider the interaction of the characters only with their own body and the ground surface, our technique can easily be parallelized for handling multiple characters.

Effect of maxLCount: as noted in Algorithm 1 maxLCount affects how much a joint is attracted towards its target at every step ($w = lCount/maxLCount$). Therefore, a greater maxLCount yields smoother convergence whereas it increases the computation cost linearly. Moreover, as increasing maxLCount results in smaller steps for w, and as we process the limbs sequentially, it allows two interacting limbs to contribute more equally to the satisfaction of that relationship.

To illustrate this, let's assume that the left arm is processed before the right arm and w value is 1. If the goal is to bring both hands together and they are distant from each other initially, only the left arm would contribute to align the left hand with the right one. Besides, the big deviation in the pose at one step would produce a jerky outcome (and possible divergence). Increasing maxLCount results in small w values that offers all the limbs the opportunity to contribute.

Memory cost: Appendix D provides a detailed storage cost analysis of our method.

Performance Animation: We have tested our technique on a set of motions including postures with contact of several body parts and also non-contact postures (Figure 1,

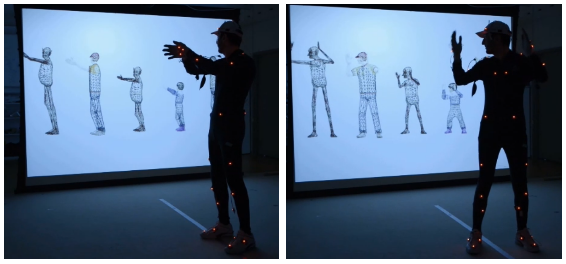
Fig. 20. Embodying multiple characters simultaneously.

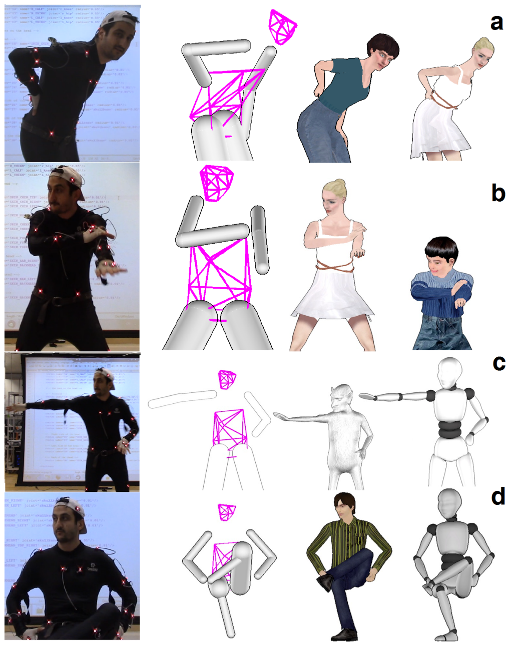

Fig. 21. Performance Animation. Left: Performed posture. Second column: Captured motion. Last two columns: Our results. a) Back pain. b) Haka dance. c) Contact free dance. d) Leg crossing.

20, 21 and the supplementary video material) on a variety of characters with different size and proportion. We have used an optical marker based system[1] for capturing the motion. We attach 38 markers on the performer. To deal with the occluded markers in real time we have used the method proposed by Aristidou et al. [37].

Our technique results in continuous motion within the reach space and reconstructs natural poses both in contact and no-contact cases. Moreover, it is very responsive, thus can be used to retarget highly dynamic motion and it can even retarget the twist motion of the limbs successfully (see the Haka dance in the supplementary video material), which is hard to handle with particle based IK systems [26], [36].

7.1 Study on Semantic Expressiveness

In order to evaluate the quality of the proposed technique, we compared our approach, MDB, to two state of the art real-time retargeting techniques: AKC [26] and KMA [11].

We implemented a variation of AKC within our framework by simply using the captured surface relative displacement vectors to adapt the target character's pose, i.e., without kinematic path normalization (Section 5.2), by disabling the adaptation through the egocentric planes and using the same number of full-body iterations and $maxLCount$.

We implemented KMA by following the Morphological Adaptation, Ground Adaptation and Posture Conversion steps introduced in Kulpa et al.'s paper [11]. There are two important points to note about our implementation of KMA. First, we used the captured spine joint angles and Forward Kinematics to obtain the spine pose of the target characters, whereas they propose the use of a spline based solution. Second, the three steps to obtain the full-body pose are all analytic solutions, and therefore do not rely on user chosen parameters.

Expectation from the User Study: Our technique can retarget self-body contacts without compromising the overall motion dynamics. Furthermore we expect that the performer's motion can be mapped onto characters with different sizes and proportions by preserving the purpose and the meaning. For example, the semantic of a classic dance figure is often carried out through precise self-contacts whereas an expressively angry character performance is mostly conveyed through a specific motion dynamics. Therefore we anticipate that the perception of a given motion remain consistent throughout a large range of target characters such as a child, an adult or any character with differing proportions as the original performer.

Experimental setup: We hired a professional actor to perform ten different motions with a variety of gestures and body movements. We used a PhaseSpace motion capture system to track the attached marker trajectories and we also recorded the performances with an RGB camera. From the captured marker trajectories we reconstructed the pose of the actor. We used three characters with different body size and proportions, a child, a woman and an alien (respectively Figure 19a, c, b) whose body size and proportion measurements are provided in Appendix E. We retargeted each captured motion onto these characters with different body size and proportions by using three mapping techniques, i.e., MDB, KMA and AKC. Each outcome was stored as video sequences. We used the same set of sampled points and locations at the surface calibration stage (Figure 5) for all performed motions, i.e. the body surface approximation was generic (not specifically adjusted to the set of performed motions).

Subjects were seated in front of a large TV screen ($71cm$x$126cm$) for the evaluation. We displayed the recorded actor's performance on the left side of the screen and the outcome of the three retargeting methods on the same character side by side ($28.5cm$x$28.5cm$ each), simultaneously and with a randomized screen placement (Figure 22). We asked the subjects to carefully analyse each animated motion clip and decide how faithfully it replicates the performed pose and action in the provided video clip. A slider was placed under each animation for scoring and they were initially positioned at mid-scale. One button was provided to pause and play all motion sequences simultaneously including the original video for allowing frame by frame comparisons. No time limit was enforced to the subjects for providing the scores.

The subjects assessed the same performance on three different characters consecutively with the same order, i.e., child, woman and alien (respectively Figure 19a, c and b).

1. PhaseSpace Impulse X2: http://www.phasespace.com/

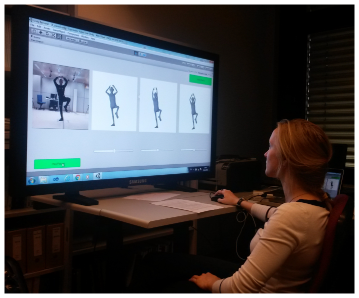
Fig. 22. The user interface of the user study.

The performances were evaluated with the same order by all subjects. Each subject scored the outcome of each retargeting technique on all three characters for nine different performances, after a training phase with a different motion that was excluded from the data analysis. In other words, 3x3x9=81 scores were used for the analysis.

We chose a variety of motions with different semantic meanings, dynamics and body parts interacting with each other for the validation study. A representative frame from each motion sequence is presented on Figure 23. The length of the motion clips were between one and three seconds. These motions sequences can be summarized as follows:
M1: Hands-up ballet figure
M2: Holding an ankle ballet figure
M3: Grabbing a gun from the pocket and shooting
M4: Disappointment
M5: Stretching the body while seated
M6: Soldier salute
M7: Thoughtful with crossed legs
M8: Interacting with a watch
M9: Washing the legs

Evaluation Results: A total of 21 subjects participated in the study (7 females, aged from 18 to 34).

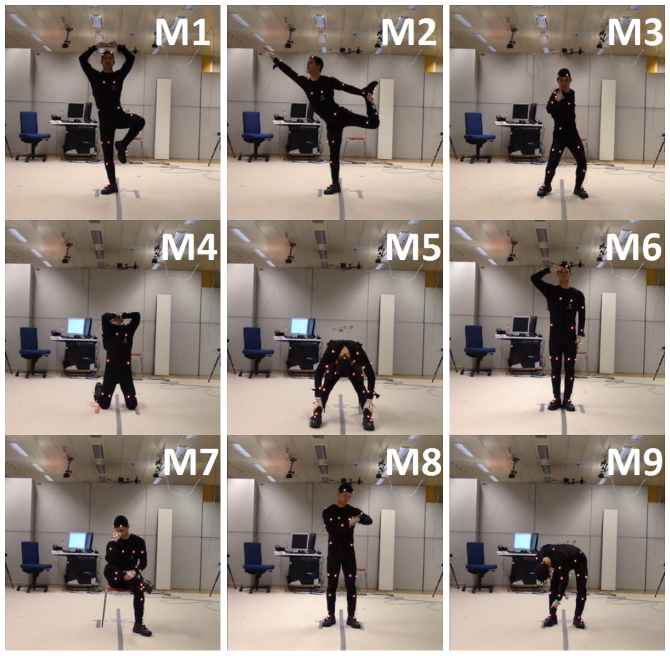

Fig. 23. The performances used for the evaluation.

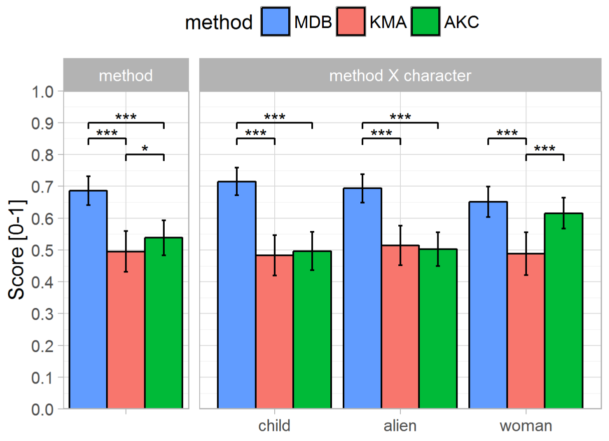

Fig. 24. The mean score obtained for each method and for the interaction between method and character. The left frame shows results by method, and the right frame shows the interaction of method and character (Figure 19a, b, c). Error bars represent the standard error of the mean. The '$*$', '$**$' and '$***$' indicates $p < .05$, $p < .01$ and $p < .001$ respectively.

We compare the performances of the algorithms with respect to two factors: character and motion. First, we investigate the study outcome on target characters with different sizes and proportions. Then, we focus on the methods performance for each of the motion sequences. We perform statistical analysis with ANOVA and post-hoc analysis with pairwise t-test and Holm-Bonferroni correction for multiple comparisons. We set the significance threshold α to 0.05.

Analysis of Method and Character: two-way ANOVA of the within subjects factors *method* and *character* yields the significant main effect of *method* ($F[2, 38] = 63.9$, $p < .001$) and of the interaction between both factors ($F[4, 76] = 15.5$, $p < .001$). Post-hoc analysis of the *method* main effect indicates that our method (MDB) received significantly higher scores than KMA and AKC, thus performing better than the related methods overall (left frame of Figure 24). On the post-hoc analysis of the *method* and *character* interaction we compared the levels of the first while keeping the second constant, for a total of 9 comparisons (3 per character). Results of the paired comparisons suggest that MDB scores are higher than KMA and AKC when the character proportions deviate from those of the performer (child and alien) and fail to reject equivalency to AKC when character and performer proportions are similar (the woman character has adult proportions). The right frame of Figure 24 shows the post-hoc results of the *method* and *character* interaction.

Analysis of Method and Motion: We are interested in examining how the methods perform for each of the nine motion sequences, but we are not interested in comparing one motion to another. Thus, we carry individual one-way ANOVA tests for each motion with *method* as the only factor. The tests yielded significant effect of *method* for all motions except M3. We present the overview of the (post-hoc) statistically significant differences in Figure 26.

The nature of the motion can induce different ranking among method scores (Figure 26). However, the method AKC is never scoring higher than our method MDB. In some cases that we discuss below, the method KMA is scoring

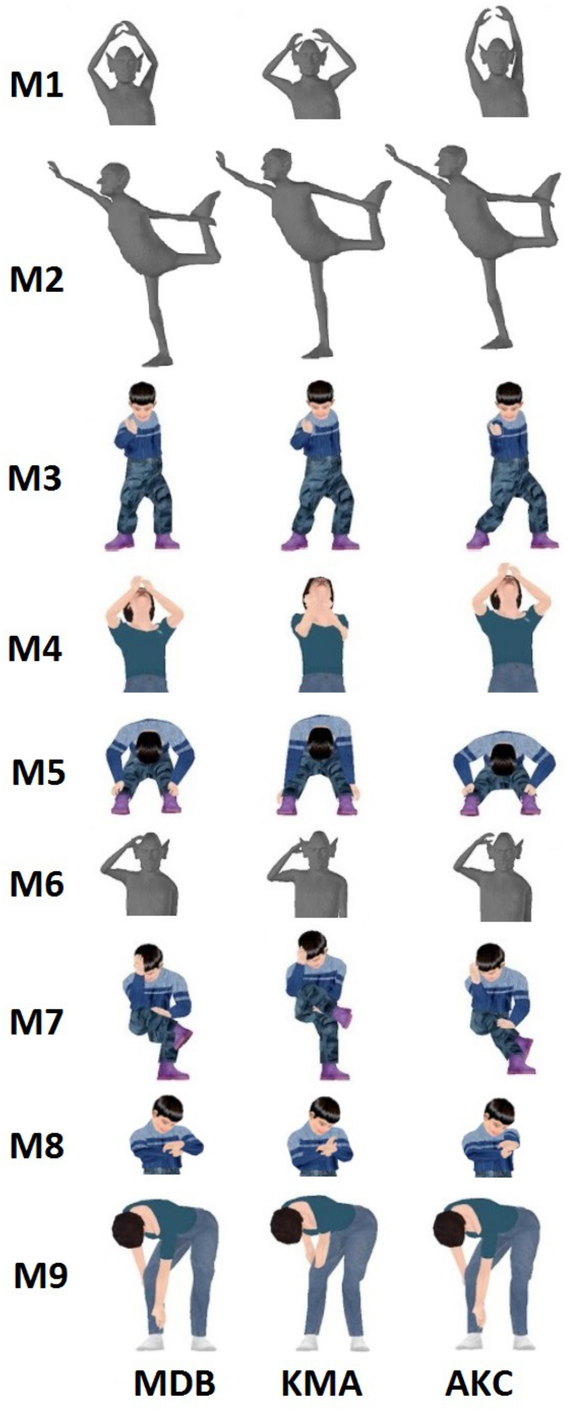

Fig. 25. Examples of target character poses for the nine motion clips. The complete set of motion sequences for each character is presented in the supplementary video material.

better than the other two methods despite - or owing to - its lack of preserving the relative location of body parts from the original motion.

Let us now review the reasons that may explain the user evaluations in Figure 26. We made a snapshot selection from the animation sequences to highlight potential plausibility issues (Figure 25). For the ballet pose M1 we believe that viewers are sensitive to both the contact and the relative orientation of body parts; KMA does not enforce the contact

	AKC	KMA	MDB
Self Contact	✓	X	✓
Env. Contact	✓	✓	Ground
Non-Contact	X	✓	✓
Spatial Order	X	X	✓
Embodiment	X	✓	✓

TABLE 1
Comparison with the state of the art.

whereas AKC overstretches the arm hence violating the relative orientation of the upper body parts. For the second dance pose in M2, the contact of the hand and the leg appears to be enforced (even if by chance for KMA resulting in a greater standard error across characters). M3 is an example of a fast gun shooting motion for which the viewers seem to be less sensitive to the arm penetration in the body (for KMA). The feet placement is also not faithful to the original performance for the method AKC but it seems like this is not decisive in the evaluation. Hand-face interactions are crucial in M4 (disappointment) and this is the reason why we think that viewers are highly sensitive to an incorrect transfer as even a small deviation of hand's placement may change the meaning. The stretching movement M5 induces important limb penetrations with KMA. M6 is a soldier salute movement for which the simpler normalization from KMA may be preferred to the one proposed in MDB and AKC. M7 is a kind of thinker pose inducing many body part contacts that are difficult to transfer for KMA and AKC. In M8 the watch checking movement involves both arms and the head; the fact that AKC does not include the path normalization, we propose, induces an implausible posture. Finally, the washing leg motion M9 produces arm-leg penetrations that are severely evaluated for KMA.

Comparison Summary: Table 1 presents a concise comparison between our method and the other two state of the art real-time retargeting techniques, AKC [26] and KMA [11]. The key strength of AKC is its ability to handle all types of contact cases correctly, both self-contacts and with the environment. Nevertheless it may produce artifacts in no-contact cases for the characters with different size and proportion as we highlighted in Section 6.3. One critical feature of KMA is its low computing cost and its ability to handle a large variety of environment contacts if the constraints are set explicitly (i.e., they are not extracted automatically). However KMA does not handle postures with self interaction. By construction, our approach is designed to uniformly handle free space gestures and self-interaction while ensuring the correct spatial order of body parts. It focuses on producing plausible postures consistent with the full range of motion of the target character. Its limitation of handling only the interaction with a flat ground can be partly alleviated if the performer is provided with the first-person viewpoint of such an avatar. With its visual feedback the performer could consistently embody the target character to interact with surrounding virtual objects.

8 Discussion and Conclusion

Although our technique can handle a broad range of self-interaction scenarios, it has some limitations. First of all,

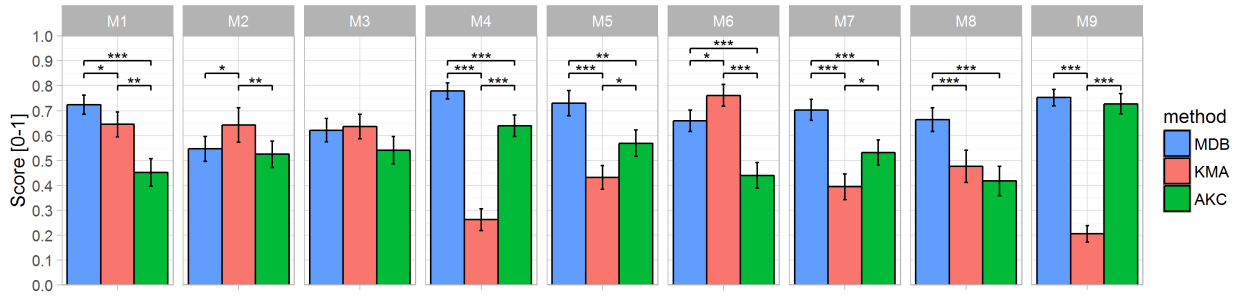

Fig. 26. The mean score obtained for each method in each of the 9 motion sequences. Error bars represent the standard error of the mean. The '$*$', '$**$' and '$***$' indicates $p < .05$, $p < .01$ and $p < .001$ respectively.

as already highlighted in Sections 6.4 and when discussing motion M6 (soldier salute), some postures may have multiple meanings or purposes and hence become ambiguous when mapping it on a target character. For instance, think about a standing posture where the hand palms lie along the external leg side. We can interpret it in two ways. First, we can consider it as the desired standing posture (i.e. like in M6 for a soldier) that should be transferred to the target character as raw joint angles. Second, it can also be a case where the character needs to reach a weapon or a tool located at a precise location in proportion to the nearby limbs. In our framework, postures are always handled according to the second approach because we consider this context to be much more frequent. It relieves the performer from the cognitive load of adapting his own posture to achieve target character poses involving self-interactions. In summary, our approach only cares about the consistent transfer of self-interactions without interfering with the style of the motion ; the performer remains entirely responsible of personifying the motion to fit a desired production goal. It could be an appealing direction to extend our framework for handling ambiguities caused by the potential multiple semantic meanings of a single performer posture.

A related production issue is our use of the same canonical pose for all the animated characters. We would advocate to introduce an additional stage of skeleton alignment as in [38] to broaden the range of target characters to those with a differing default posture. Likewise, we do not explicitly handle joint limits in the interest of computing performance and not to restrict the artistic performance. Nevertheless the relative surface vectors handle this implicitly by adjusting the target joint positions. This may cause some minor torso penetrations in the final posture.

Integrating face and finger animation to our framework could be an interesting future work. Facial retargeting methods [39] can be used to deform the face surface and the resulting mapping could be utilized to denormalize sampled points as it is done for the torso surface. For now, since we use the wrists to approximate the hands' contact, the finger-level-contact information is not exploited by our system. However, the egocentric normalization/denormalization could also be used for the finger joints. Another interesting future work would be to integrate the interaction with external objects by taking advantage of the work of [26] and [40] for large environment changes.

To conclude, we have introduced a novel type of normalized representation of posture, named egocentric coordinates, which intrinsically encodes the self-contact information while still handling the non-contact postures continuously. We have shown how egocentric coordinates can be mapped onto characters with different size and proportions by ensuring the correct spatial order through the use of egocentric planes. We have presented our posture adaptation technique which reconstructs postures in real-time. Our mapping technique is path independent, i.e. each frame is processed independently without relying on previous frames, thus a given egocentric normalized posture always maps to the same target pose. We highlighted the drawbacks of this approach for continuity and we proposed solutions in Sections 6.3 and 6.4. Finally, we have presented several use cases of our method for online performance animation and compared it with other state of the Art methods. Results demonstrate its clear potential for consistently transferring the semantics across characters with differing proportions for most of the studied self-interactions.

Acknowledgments

Eray Molla acknowledges the support of the SNF project 200020-146827. We are grateful to Paul Kry, Taku Komura and Richard Kulpa for useful feedback at various stages of this submission and to Simon Labarrière for his great performing style. We would also like to thank the anonymous reviewers for the constructive and detailed feedback.

References

[1] D. J. Sturman, "Computer puppetry," *IEEE Computer Graphics and Applications*, vol. 18, no. 1, pp. 38–45, 1998.
[2] H. J. Shin, J. Lee, S. Y. Shin, and M. Gleicher, "Computer puppetry: An importance-based approach," *ACM Transactions on Graphics*, vol. 20, pp. 67–94, 2001.
[3] R. Kulpa, L. Hoyet, T. Komura, and F. Multon, "Interactive animation of virtual humans based on motion capture data," *Computer Animation and Virtual Worlds*, vol. 20, no. 5-1, pp. 491–500, 2009.
[4] B. Spanlang, X. Navarro, J.-M. Normand, S. Kishore, R. Pizarro, and M. Slater, "Real time whole body motion mapping for avatars and robots." in *VRST*. ACM, 2013, pp. 175–178.
[5] E. Molla and R. Boulic, "A two-arm coordination model for phantom limb pain rehabilitation," in *Proc. Symposium on Virtual Reality Software and Technology*. ACM, 2013, pp. 35–38.
[6] K. Kilteni, I. Bergstrom, and M. Slater, "Drumming in immersive virtual reality: The body shapes the way we play." in *VR*. IEEE, 2013, p. 1.
[7] C. Darwin, *The Expression of the Emotions in Man and Animals*. New York: D. Appleton and Company, 1872.
[8] D. Morris, *Manwatching: A Field Guide to Human Behavior*. Abrams, 1977.

[9] A. Kleinsmith and N. Bianchi-Berthouze, "Affective body expression perception and recognition: A survey," *Affective Computing, IEEE Transactions on*, vol. 4, no. 1, pp. 15–33, Jan 2013.

[10] N. Bianchi-Berthouze, "Understanding the role of body movement in player engagement," *HumanComputer Interaction*, vol. 28, no. 1, pp. 40–75, 2012.

[11] R. Kulpa, F. Multon, and B. Arnaldi, "Morphology-independent representation of motions for interactive human-like animation," *Computer Graphics Forum, Eurographics 2005 special issue*, vol. 24, pp. 343–352, 2005.

[12] M.-C. Silaghi, R. Plaenkers, R. Boulic, P. Fua, and D. Thalmann, "Local and global skeleton fitting techniques for optical motion capture," in *Workshop on Modelling and Motion Capture Techniques for Virtual Environments*, 1998, pp. 26–40.

[13] M. Gleicher, "Retargetting motion to new characters," in *Proc. of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '98. ACM, 1998, pp. 33–42.

[14] K.-J. Choi, "Online motion retargetting," *The Journal of Visualization and Computer Animation*, vol. 11, no. 5, pp. 223–235, 2000.

[15] C. Hecker, B. Raabe, R. W. Enslow, J. DeWeese, J. Maynard, and K. van Prooijen, "Real-time motion retargeting to highly varied user-created morphologies," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 27:1–27:11, Aug. 2008.

[16] S. Guo, R. Southern, J. Chang, D. Greer, and J. Zhang, "Adaptive motion synthesis for virtual characters: a survey," *The Visual Computer*, pp. 1–16, 2014.

[17] U. Celikcan, I. O. Yaz, and T. Capin, "Examplebased Retargeting of Human Motion to Arbitrary Mesh Models," *Computer Graphics Forum*, 2015.

[18] E. S. L. Ho, T. Komura, and C.-L. Tai, "Spatial relationship preserving character motion adaptation," *ACM Trans. Graph.*, vol. 29, no. 4, pp. 33:1–33:8, Jul. 2010.

[19] E. S. L. Ho, H. Wang, and T. Komura, "A multi-resolution approach for adapting close character interaction," in *Proc. of the ACM Symposium on Virtual Reality Software and Technology*, 2014, pp. 97–106.

[20] K. Zhou, W. Xu, Y. Tong, and M. Desbrun, "Deformation transfer to multi-component objects," *Computer Graphics Forum*, vol. 29, no. 2, pp. 319–325, 2010. [Online]. Available: http://dx.doi.org/10.1111/j.1467-8659.2009.01601.x

[21] Q. Avril, D. Ghafourzadeh, S. Ramachandran, S. Fallahdoust, S. Ribet, O. Dionne, M. d. Lasa, and E. Paquette, "Animation Setup Transfer for 3D Characters," *Computer Graphics Forum*, 2016.

[22] M. Dontcheva, G. Yngve, and Z. Popović, "Layered acting for character animation," in *Proceedings of ACM SIGGRAPH '03*, 2003, pp. 409–416.

[23] K. Yamane, Y. Ariki, and J. Hodgins, "Animating non-humanoid characters with human motion data," in *Proceedings of SCA '10*, 2010, pp. 169–178.

[24] Y. Seol, C. O'Sullivan, and J. Lee, "Creature features: online motion puppetry for non-human characters," in *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. ACM, 2013, pp. 213–221.

[25] Y. Teng, M. A. Otaduy, and T. Kim, "Simulating articulated subspace self-contact," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 106:1–106:9, 2014.

[26] R. A. Al-Asqhar, T. Komura, and M. G. Choi, "Relationship descriptors for interactive motion adaptation," in *Proceedings of SCA '13*, 2013, pp. 45–53.

[27] E. S. L. Ho and T. Komura, "Character motion synthesis by topology coordinates," *Comput. Graph. Forum*, vol. 28, no. 2, pp. 299–308, 2009.

[28] R. Vemulapalli, F. Arrate, and R. Chellappa, "Human action recognition by representing 3d skeletons as points in a lie group," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '14, 2014, pp. 588–595.

[29] H. Wang, K. A. Sidorov, P. Sandilands, and T. Komura, "Harmonic parameterization by electrostatics," *ACM Trans. Graph.*, vol. 32, no. 5, pp. 155:1–155:12, Oct. 2013. [Online]. Available: http://doi.acm.org/10.1145/2503177

[30] P. Sandilands, V. Ivan, T. Komura, and S. Vijayakumar, "Dexterous reaching, grasp transfer and planning using electrostatic representations," in *Proceedings of IEEE-RAS International Conference on Humanoid Robots*, 2013.

[31] P. Sandilands, M. G. Choi, and T. Komura, "Capturing close interactions with objects using a magnetic motion capture system and a rgbd sensor," in *Motion in Games*. Springer Berlin Heidelberg, 2012, vol. LNCS 7660, pp. 220–231.

[32] B. Reinert, T. Ritschel, and H.-P. Seidel, "Homunculus warping: Conveying importance using self-intersection-free nonhomogeneous mesh deformation," *Comp. Graph. Forum*, vol. 31, no. 7pt2, pp. 2165–2171, 2012.

[33] L. Unzueta, M. Peinado, R. Boulic, and A. Suescun, "Full-body performance animation with sequential inverse kinematics," *Graphical Models*, vol. 70, no. 5, pp. 87 – 104, 2008.

[34] D. Tolani, A. Goswami, and N. I. Badler, "Real-time inverse kinematics techniques for anthropomorphic limbs," *Graphical Models*, vol. 62, no. 5, pp. 353 – 388, 2000.

[35] E. Molla and R. Boulic, "Singularity free parametrization of human limbs," in *Proceedings of Motion on Games*. ACM, 2013, pp. 165:187–165:196.

[36] T. Jakobsen, "Advanced character physics," in *Game Developers Conference Proceedings*, 2001, pp. 383 – 401.

[37] A. Aristidou, J. Cameron, and J. Lasenby, "Real-time estimation of missing markers in human motion capture," in *The International Conference on Bioinformatics and Biomedical Engineering*, 2008, pp. 1343–1346.

[38] A. Feng, Y. Huang, Y. Xu, and A. Shapiro, "Fast, automatic character animation pipelines," *Computer Animation and Virtual Worlds*, vol. 25, no. 1, pp. 3–16, 2014.

[39] T. Weise, S. Bouaziz, H. Li, and M. Pauly, "Realtime performance-based facial animation," in *ACM Transactions on Graphics (TOG)*, vol. 30, no. 4, 2011, p. 77.

[40] S. Tonneau, R. A. Al-Ashqar, J. Pettré, T. Komura, and N. Mansard, "Character contact re-positioning under large environment deformation," *Comput. Graph. Forum*, vol. 35, no. 2, pp. 127–138, 2016.

Eray Molla was a Research and Teaching assistant at the Immersive Interaction research group (IIG), EPFL (Ecole Polytechnique Fédérale de Lausanne), Switzerland. He currently works in the video game industry. He received the PhD degree in Computer and Communication Sciences from EPFL in 2016, his MSc degree in Computer Science from EPFL in 2011, and his BSc in Computer Engineering from METU (Middle East Technical University), Turkey, in 2009.

Henrique Galvan Debarba was a research assistant at the Immersive Interaction research group (IIG), EPFL (Ecole Polytechnique Fédérale de Lausanne), Switzerland. He received a PhD degree in Robotics, Control, and Intelligent Systems from EPFL in 2017, and a MSc degree in Computer Science from UFRGS (Universidade Federal do Rio Grande do Sul), Brazil, in 2012. Henrique currently works as a researcher at Artanim Foundation, Switzerland.

Ronan Boulic is a Senior Scientist and PhD Advisor at the EPFL (Ecole Polytechnique Fédérale de Lausanne), Switzerland. He currently leads the Immersive Interaction research group (IIG) from the School of Computer and Communication Sciences. He received the PhD degree in Computer Science in 1986 from the University of Rennes, France, and the Habilitation degree in Computer Science from the University of Grenoble, France, in 1995. He is senior member of IEEE and of ACM, and member of Eurographics.

Egocentric Mapping of Body Surface Constraints

Eray Molla, Henrique Galvan Debarba, and Ronan Boulic, *Senior Member, IEEE*

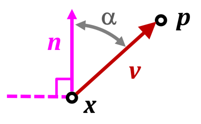

Fig. 1. The projection of the joint position **p** on each element of the body surface approximation produces a reference point, **x**. Such an element can be a triangle from the crude mesh approximation of the trunk or the head, or a limb segment capsule. In this illustration, we highlight the case where an angle α exists between the surface element normal **n** and the surface relative displacement vectors **v**.

Appendix A
Importance λ of a Reference Point

We define an importance metric, λ, relying on two properties: proximity, λ_p, and orthogonality, $\lambda_\perp$ where $\lambda = \lambda_p\, \lambda_\perp$. It analyzes the surface relative displacement vectors **v** that is defined with respect to each element constituting the body surface approximation (Figure 1); these elements are either triangles from the crude meshes approximating the trunk or the head, or limb segment capsules. The proposed metric expresses that nearby surface elements lying perpendicular to the joint are more reliable references to express the joint's location.

Proximity (λ_p): λ_p is inversely proportional to the distance, $\|\mathbf{v}\|$, hence close surfaces are assigned a higher importance. To avoid numeric instability, we compute the proximity as

$$\lambda_p = \begin{cases} \frac{1}{\epsilon} & \|\mathbf{v}\| \leq \epsilon \\ \frac{1}{\|\mathbf{v}\|} & \|\mathbf{v}\| > \epsilon \end{cases} \tag{1}$$

where ϵ is a small positive number close to zero.

Orthogonality ($\lambda_\perp$): We measure the orthogonality $\lambda_\perp$ by exploiting the angle α between the element surface normal **n** and **v** (Figure 1)

$$\lambda_\perp = \begin{cases} \cos\epsilon & \cos\alpha \leq \epsilon \\ \cos\alpha & \cos\alpha > \epsilon \end{cases} \tag{2}$$

Measuring orthogonality is critical for two reasons. First, if the joint projects along the surface normal, it is likely that they are interacting. Second, such interaction context can even carry semantic information in postures such as holding a hand in front of the mouth, eyes, etc.

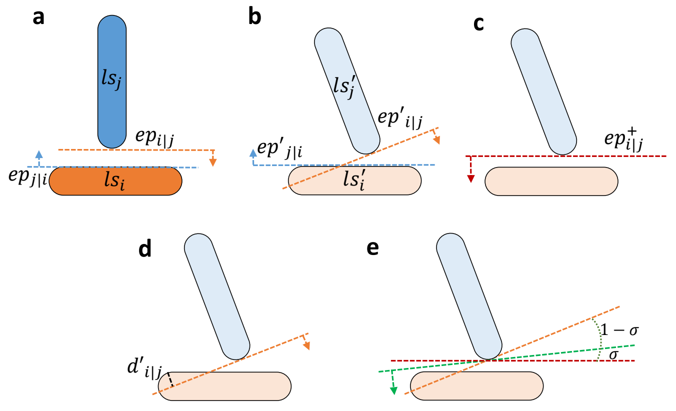

Fig. 2. Half-space constraint relaxation scheme. a) The relationship of the segments in the source. b) Possible configuration on the target character. ls'_i violates $ep'_{i|j}$ even though the spatial order is correct. c) $ep'_{j|i}$ is projected on ls'_j and $ep^+_{i|j}$ is obtained. d) The signed distance of ls'_i to $ep'_{i|j}$ is negative because of the penetration. e) The relaxed constraint (green plane) is obtained when $ep'_{i|j}$ and $ep^+_{i|j}$ are interpolated by σ. In this specific example, ls'_i does not violate the relaxed half-space constraint.

Appendix B
Half Space Constraint Relaxation

As explained in Section 5.3, we compute two separating planes between a pair of source limb segments, ls_i and ls_j, and use them as half-space constraints to preserve the spatial order (Figure 2a).

However, enforcing both constraints can over-constrain the problem in case of a different relative orientation between the target segments, ls'_i, ls'_j, and the source segments (Figure 2b). We make use of a planar interpolation scheme to relax the constraints.

The half-space relaxation for ls'_i works as follows. First, we project $ep'_{j|i}$[1] onto ls'_j, which corresponds to the tangent plane of ls'_j parallel to $ep'_{j|i}$ but pointing at the opposite side. This plane is denoted by $ep^+_{i|j}$ (Figure 2c). Then we calculate the signed distances of ls'_i relative to $ep'_{i|j}$ and $ep^+_{i|j}$ (Figure 2d).

We compute a weight coefficient for each signed distance by using an S-shaped curve (Figure 3). This continuous curve penalizes the plane causing penetration ($d < 0$)

1. Read as the denormalized egocentric plane of segment j built on the segment i.

and encourages the use of the safe one ($d > 0$). These corresponding weights, k' and k^+, are converted to an interpolation coefficient as follows:

$$\sigma = \frac{k'}{k' + k^+} \tag{3}$$

This coefficient is used to interpolate $ep'_{i|j}$ and $ep^+_{i|j}$. As a result, we use that interpolated plane as a relaxed half-space constraint for ls'_i (Figure 2e).

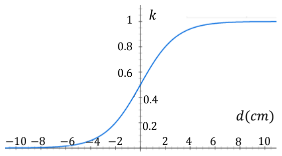

Fig. 3. Curve used for the signed distance weights $k = \frac{1}{1+2^{-d}}$

Appendix C
Slerping Standard and Reversed Limb IK Solutions

This appendix explains how the resulting postures from the standard and reversed limb IK are interpolated. First, we calculate the maximum importance values λ_i of the end and mid joints (respectively $max(\lambda_i^{end})$ and $max(\lambda_i^{mid})$). We use these values to compute a coefficient γ such that $\gamma = max(\lambda_i^{mid})/(max(\lambda_i^{end}) + max(\lambda_i^{mid}))$, where $0 \leq \gamma \leq 1$. We then compute an intermediate limb posture between the standard ($\gamma = 0$) and the reversed ($\gamma = 1$) solutions by slerping the state of each joint in the joint angle space according to the coefficient γ. In this way,

- if the end joint has close contact while the mid joint doesn't have, i.e., $max(\lambda_i^{end}) >> max(\lambda_i^{mid})$, γ will be close to zero. Hence, the slerped pose will be close to the standard limb IK solution which considers the end joint as the first priority.
- if the mid joint has close contact while the end joint doesn't have, i.e., $max(\lambda_i^{mid}) >> max(\lambda_i^{end})$, γ will be close to one. Hence, the slerped pose will be close to the reversed limb IK solution which considers the mid joint as the first priority.
- other combinations will result in compromised poses with continuous transitions.

Appendix D
Storage Cost Analysis

This appendix analyses the storage cost of a limb joint's egocentric coordinates and of the egocentric planes.

Expressing the pose of a character with 86 joints requires approximately 1.4kilobytes by using unit quaternions for rotations and a three dimensional vector for the root position. The cost of egocentric representation of pose on the same character in our implementation is about 21kilobytes where the body mesh contains 34 triangles. Hence, although our representation significantly increases the memory cost compared to the standard posture representation, its absolute value remains negligible for real-time applications.

As noted in Section 5.1, to express each relative displacement, we store an importance scalar λ_i, a 3-DoF reference point $\hat{\mathbf{x}}_i$, a 3-DoF displacement vector $\hat{\mathbf{v}}$ and an orientation deviation β_i, 4 scalars as a unit quaternion, and the set of n coefficients C_i where n is the length of the kinematic path. One for the reference primitive and one for the expressed joint, two identifier scalars are held, as well. Therefore, $1 + 3 + 3 + 4 + n + 2 = 13 + n$ scalars need to be stored to express the relative coordinates of an extremity joint with respect to a single surface primitive.

To consider the relationship with m surface primitives, $\sum_{r=1}^{m} 13 + n_r = 13m + \sum_{r=1}^{m} n_r$ scalars need to be stored, where n_r is the length of the kinematic path between the joint and the corresponding reference point, r. Given that the length of the longest kinematic path in the body is N, e.g., from right hand to left ankle, $O(m(13 + N))$ scalars are required to store the egocentric coordinates of an extremity joint. As a humanoid has four such limbs and the same amount of data needs to be stored twice for each of them, $O(4 \times 2 \times m(13 + N)) = O(8m(13 + N))$ scalar space is used in total.

For the egocentric planes, we compute a separating plane from each limb segment to all other limbs' segments. As there are three other limbs and each of them has two segments, the data of six planes are stored per segment. As the same operation is repeated for both segments of each limbs, the data of $4 \times 2 \times 6 = 48$ egocentric planes are stored in total which is considerably less than the storage cost of the egocentric coordinates.

Appendix E
Body size and proportion measurements of the characters used in the study

Table 1 presents the body measurements of the characters used in the study: Child (C), Alien (A), Woman (W). The first three rows present the absolute measurements for each character in centimeters. The last three rows give the corresponding normalized measurements with respect to the body height. The columns present the data in the following order: body height (H), arm length from the shoulder to the wrist joints (AL), upper arm radius (UAR), lower arm radius (LAR), leg length from the hip to the ankle joints (LL), upper leg radius (ULR), lower leg radius (LLR), belly thickness (BT).

Char.	H	AL	UAR	LAR	LL	ULR	LLR	BT
C (cm)	120	38	5	4	48	6	5	17
A (cm)	152	42	3	2	66	5	2	27
W (cm)	168	51	5	4	80	8	4	16
C (%)	100	32	4	3	40	5	4	14
A (%)	100	28	2	1	43	3	1	18
W (%)	100	30	3	2	48	5	2	10

TABLE 1
Body measurements of the characters