Position Based Skinning of Skeleton-driven Deformable Characters

Nadine Abu Rumman[*] Marco Fratarcangeli[†]

Sapienza University of Rome

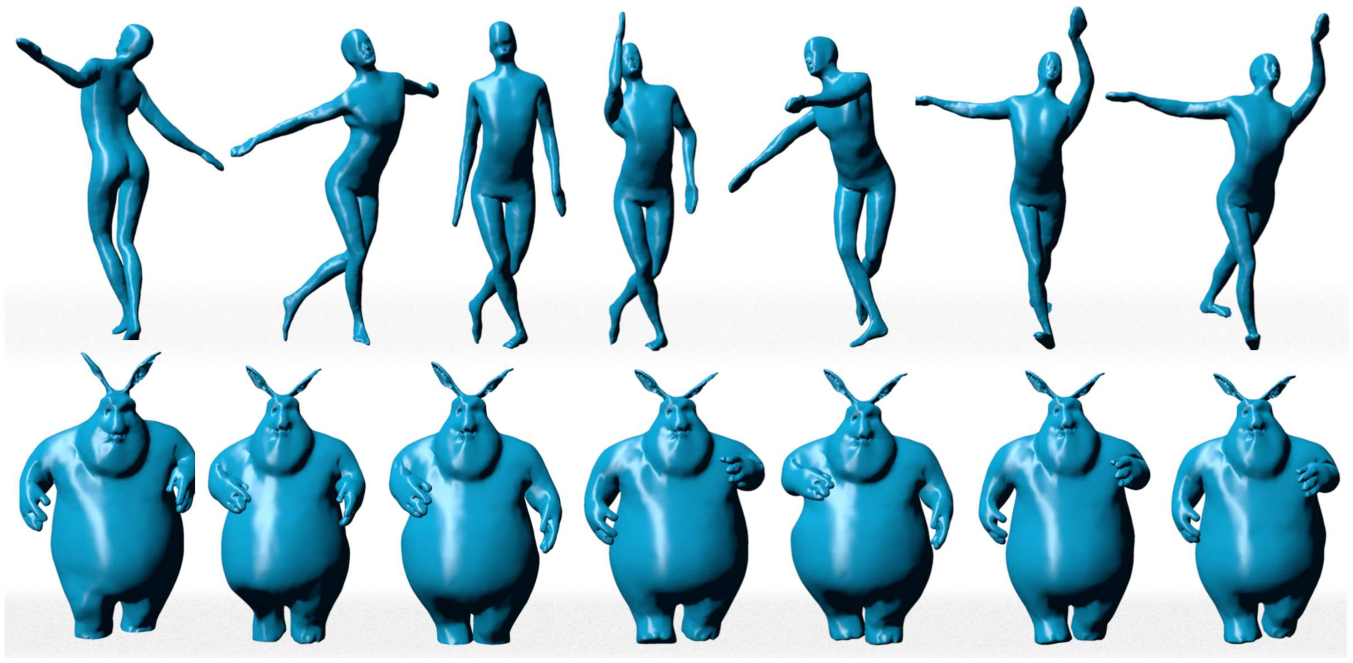

Figure 1: Real-time character animation driven by a kinematic skeleton, including secondary motions and volume preservation.

Abstract

This paper presents a real-time skinning technique for character animation based on a two-layered deformation model. For each frame, the skin of a generic character is first deformed by using a classic linear blend skinning approach, then the vertex positions are adjusted according to a Position Based Dynamics schema. We define geometric constraints which mimic the flesh behavior and produce interesting effects like volume conservation and secondary animations, in particular passive jiggling behavior, without relying on a predefined training set of poses. Once the whole model is defined, the character animation is synthesized in real-time without suffering of the inherent artefacts of classic interactive skinning techniques, such as the "*candy-wrapper*" effect or undesired skin bulging.

CR Categories: 1.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically Based Modeling; 1.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation and Virtual Reality

[*]e-mail:aburumman@dis.uniroma1.it
[†]e-mail:frat@dis.uniroma1.it

Keywords: Skinning, Deformable body, Skeleton, Position Based dynamics, Skeleton-driven deformable body

1 Introduction

In character animation, skinning is the process of defining how geometric surface deforms according to skeletal poses [Jacka et al. 2007; Angelidis and Singh 2007]. By employing a physically based method into the skinning process, the believability and realism of character motions are highly enhanced. Physics-based simulations manage to bring skeleton-driven animation beyond the purely kinematic approach by simulating secondary motions such as jiggling of soft tissues when the character is moving. Those secondary motions enrich the visual experience of the animation and are essential for creating appealing characters for movie productions and virtual reality applications [Larboulette et al. 2005; McAdams et al. 2011].

However, since physical simulation of secondary motions is computationally demanding and complex, it is usually avoided in interactive animations [Shinar et al. 2008; Clutterbuck and Jacob 2010]. Simulating flesh-like deformations is difficult due to the coupling between the skeleton, the soft skin, and the complex inner bio-mechanical structure of the human body [Lee et al. 2009]. The computational process for obtaining believable skin motion must trade-off between these requirements: it must (1) be fast enough to achieve interactive rate (i.e., $>$ 30 fps), (2) produce believable animation to minimize manual post-processing time, and (3) be controllable and stable.

In this paper we present a simple and fast skinning technique for animating virtual characters. The idea is to create a two-layered deformation schema, the result of which approximates the behavior of the skin. In order to simulate the behavior of the skin a tetrahedral mesh is generated from a triangle mesh. Tetrahedral meshes are commonly used for simulating deformable bodies [Müller et al. 2002; Diziol et al. 2009]. All resulting tetrahedrons are assigned to volume constraints (Sec. 3.3), which ensure the conservation of the character body total volume. The original triangle mesh can be assigned to the tetrahedral mesh. The triangle mesh can be used for the purposes of rendering, and the tetrahedral mesh is used for all other aspects of simulation like volume preservation.

For each animation frame, the deformation of the character is decoupled in two steps. First, we apply classic linear blend skinning to the character, then a system of geometric constraints is solved and the vertex positions are adjusted accordingly (Fig. 3). In this second step, we simulate the skin as a soft body and the deformation approach is based on Position Based Dynamics (PBD) [Müller et al. 2007]. We use geometrical constraints for modeling the skin, the inner volumetric structure and the binding of the skin with the skeleton. These constraints are iteratively satisfied leading to primary and secondary motions of the external skin in real-time.

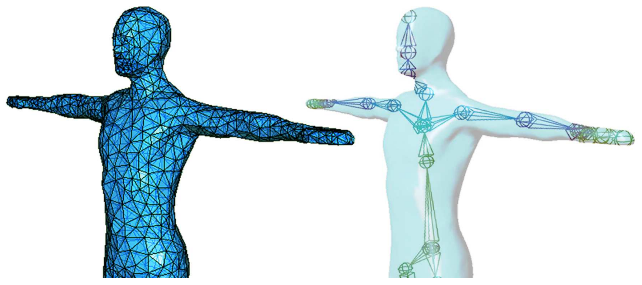
Figure 2: A tetrahedral mesh and the underlying skeleton.

The main contribution of this paper is the development of a skinning system, which provides believable body deformations at interactive rate, including secondary motions (Sec. 3). To achieve this, we introduce a deformation model based on geometric constraints which (1) preserve the skin volume, (2) reproduce passive jiggling behavior and (3) implicitly solve the artefacts of classic interactive skinning techniques, like the "*candy-wrapper*" effect and undesired skin bulging. Being based on PBD, our system is efficient, controllable and unconditionally stable, even when large time steps are employed for advancing the character's dynamics (Sec. 4).

2 Related Work

Linear blend skinning [Magnenat-Thalmann et al. 1988] has been widely used for calculating skin deformations in real time, because it is easy to implement and both time and space efficient, which makes it particularly suitable for real-time applications such as games. Unfortunately, linear skinning suffers from visual artefacts like self-intersection, volume loss or the well-known "*candy-wrapper*" artefact, which are the result of the linear nature of the algorithm, since the linear interpolation of the transformation matrices is not equivalent to the linear interpolation of their rotations. An interesting extension of linear blend skinning called spline skinning comes from [Forstmann and Ohya 2006], which often produces better skinning deformations. An overview survey on linear skinning techniques can be seen in Jacka et al. [Jacka et al. 2007]. These artefacts can be solved by adding pose examples [Lewis et al. 2000; Kry et al. 2002], additional skinning weights [Wang and Phillips 2002; Mohr and Gleicher 2003; Merry et al. 2006] or by replacing the linear blending with a non-linear transformation blending (dual quaternion skinning) [Kavan et al. 2007]. However, dual quaternion blending suffers from an undesired joint-bulging artefact while bending, which requires artistic manual work to be fixed. Because this is a tedious process, automatic skinning techniques [Baran and Popović 2007; Wareham and Lasenby 2008; Chen et al. 2011] are becoming increasingly popular.

Recently, Kavan and Sorkine [Kavan and Sorkine 2012] developed a new skinning method based on the concept of joint-based deformers, which avoids the artefacts of linear blend skinning as well as the bulging artefacts of dual quaternion skinning. All these techniques are either purely kinematic, lacking of secondary motions effects like passive jiggling motion of fatty tissues, or example-based techniques that require the artists to create many different samples by hand for a wide variety of poses.

After the pioneering work of Terzopoulos et al. [Terzopoulos et al. 1987], many physically based methods encouraged to simulate soft bodies or to add dynamic effects to the skin [Chadwick et al. 1989; Pentland and Williams 1989; Gourret et al. 1989; Turner and Thalmann 1993; Lee et al. 1995; Fratarcangeli 2012]. A possible approach for physically-based deformations of soft bodies is to focus on the surface rather than the volume [Bro-nielsen and Cotin 1996; Shi et al. 2008]. In particular Galoppo et al. [Galoppo et al. 2007] presented a fast method to compute the skin deformation on the surface of a soft body including a rigid core. Their formulation only considers the elastic energy from skin-layer deformation, and does not include the deformation inside the volume, which may lead to inaccuracies when capturing pose-dependent deformations. A survey of Botsch and Sorkine [Botsch and Sorkine 2008] provides the comprehensive details about these techniques.

In contrast, Capell et al. [Capell et al. 2002] used volumetric finite element mesh to represent the deformation of skin, driven by the underlying skeleton motion. They extended their method to include rigging forces, which guide the deformation to a desired shape [Capell et al. 2005]. In their method, they effectively handled the effect of skin movement by using skeletal constraints, but using forces that can violate the conservation of momentum may make their simulation unstable under large time steps.

Recently [Kim and Pollard 2011] proposed an approach relying on the finite element method (FEM) to simulate the skin deformation, able to handle both one-way and two-way simulations. [Deul and Bender 2013] introduced a multi-layer character skinning based on shape matching with oriented particles, used to simulate the elastic behavior of a closed triangular mesh as representation of a skin model. They make a use of position-based constraints for coupling the skeleton with the skin, and handling self-collisions.

Models of both [Kim and Pollard 2011] and [Deul and Bender 2013] are more sophisticate and reflect better the inner structure of the human skin than ours. We decided to employ a robust combination of two popular techniques well known in the Computer Graphics community, linear blend skinning and position based dynamics, because it requires a small effort to be programmed and leads to believable animations with a generally faster performance than the above mentioned methods (Sec. 4).

3 Position Based Skinning

The input to our system are: 1) the volumetric tetrahedral mesh of a character in rest pose, and 2) a corresponding skeleton which is

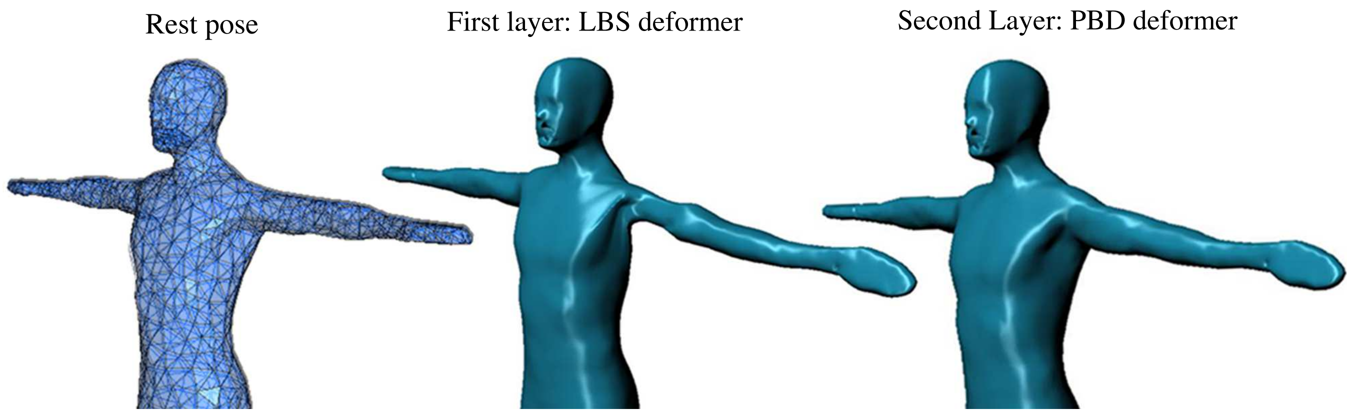

Figure 3: Two-layers deformer. *Left.* Initial rest pose. *Middle.* Step 1: Linear Blend Skinning (LBS) is applied. Note the “candy wrapper” effect on the shoulder. *Right.* Step 2: Vertex positions are adjusted using Position Based Dynamics (PBD).

kinematically animated (Fig. 2). In the initialization phase, geometrical constraints are defined using a Position Based Dynamics (PBD) schema. For each animation frame, as the skeleton moves, the skin vertices are first deformed through a standard linear blend skinning (LBS) process. And then the vertex positions are adjusted by solving the per-defined geometric constraints. In the following sections, we first explain how the weights for LBS are found (Sec. 3.1), then we proceed to briefly describe PBD (Sec. 3.2), and finally provide the definition and the resolution method for the geometric constraints (Sec. 3.3).

3.1 Linear Blend Skinning

We consider a skeleton as defined by its bones. Each bone is expressed as a rotation matrix. Given a bone j, we distinguish between its rest matrix $\mathbf{R}_j$ and its posed matrix $\mathbf{P}_j$ (that is, its configuration in any animation frame), where $j = 1 \ldots n$, and n is the total number of bones. For each animation frame, the motion of the posed bones is defined by given input motion capture data. At rest, the input mesh is associated with its skeleton (Fig. 2). In LBS, each vertex v_i in the input mesh is attached to one or more skeletal bones, each attachment affecting the vertex with a different strength, or *weight*. The final transformed vertex position v'_i is a weighted average of its initial position transformed by each of the attached bones, according to the following equation [Merry et al. 2006]:

$$v'_i = \sum_{j=1}^{n} w_j \cdot \mathbf{P}_j \cdot \mathbf{R}_j^{-1} \cdot v_i \qquad (1)$$

where $\mathbf{P}_j$ is the transformation matrix associated with bone j in its current pose, $\mathbf{R}_j^{-1}$ is the inverse transformation of the same bone in the rest pose and w_j is the scalar weight binding $\mathbf{v}_i$ to the bone j, under the condition $\sum_{j=1}^{n} w_j = 1$. By attaching each vertex to only one bone, the equation above achieves a *rigid* binding. In order to obtain *smooth* skin deformations in the second step of our method (Sec. 3.3), we compute the weights for each vertex $\mathbf{v}_i$ according to this heuristic formula:

$$w_j = \begin{cases} 1.0 & \text{if } \frac{1}{6} \leq \frac{(\mathbf{v}_i - \mathbf{s}_j)}{|\mathbf{d}_j - \mathbf{s}_j|} \cdot \frac{(\mathbf{d}_j - \mathbf{s}_j)}{|\mathbf{d}_j - \mathbf{s}_j|} \leq \frac{5}{6} \\ \frac{1.0}{n_{joints}} & \text{otherwise} \end{cases} \qquad (2)$$

where $\mathbf{s}_j$ and $\mathbf{d}_j$ are the parent and child joint positions (end points) of the bone j (Fig. 4) nearest to $\mathbf{v}_i$, and n_{joints} is the number of bones attached to the end points bone nearest to $\mathbf{v}_i$. In other words, vertices far away from the end points of a bone are rigidly influenced by just that bone. Vertices near the end points are equally influenced by the bones in that area.

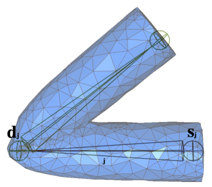

Figure 4: *End points of the bone j: $\mathbf{s}_j$ and $\mathbf{d}_j$*

Using the weights defined in Eq. 2, the LBS step deforms the vertices in a soft way near the articulations of the character. Such a deformation will be then adjusted by the second step of our algorithm, the PBD deformer (Sec. 3.2). Fig. 5 compares the skin configuration at the end of the whole process (LBS and then PBD) by binding each vertex only to the nearest bone (left) with weight equal to one, and by using the heuristics in Eq. 2 (right).

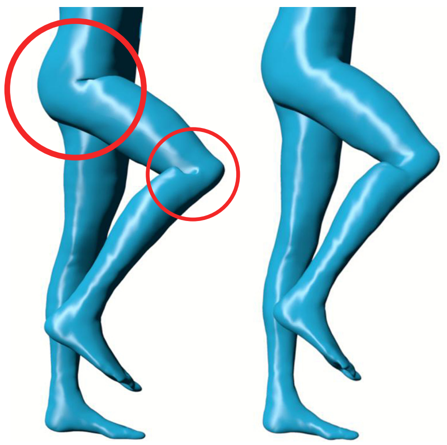

Figure 5: *Left.* Each vertex in the mesh is attached to exactly one bone in the skeleton. *Right.* Each vertex in the mesh is attached to more than one bone according to the weights in Eq. 2. In both the cases the PBD deformer is then applied.

3.2 Position-based Dynamics

In this paper, we simulate the character skin inside the Position Based Dynamics (PBD) [Müller et al. 2007] framework. We choose PBD for its unconditionally stable time integration and robustness. The most popular methods for simulating dynamics in computer graphics are force based. Force based method computes the accelerations based on Newton's second law of motion. Then a time integration method is uses to update the velocities and finally the positions of the object. Unlike force based methods, PBD omits the velocity and immediately works on the positions. By solving a set of geometrical constraints inside the PBD framework. Our method allows simulating secondary motions like jiggling and inherits the stability of the position based dynamics method. Allowing large time steps suitable for real-time applications.

For each animation frame, we perform two steps: the first one uses LBS to deform kinematically the vertices according to the motion of the underlying skeleton. The second step uses a soft body deformer for adjusting the vertex position, fixing the drawbacks inherent to LBS such as various types of skin collapsing effects. We use PBD to model the elastic behavior of the skin. The basic idea of PBD is to model the soft body as a particle system, simulate the dynamics according to external forces (if present), and then solve non-linear constraints $C(\mathbf{x}) = 0$ which express a set of geometric relationships between the particles, where $\mathbf{x}$ is the vector of all the positions of the particles. In Sec. 3.3, we define the geometric constraints used in our model and how they are solved. In the following, we introduce the basic concepts of PBD which are useful to understand our method. The interested reader can refer to [Müller et al. 2007] for more information.

The soft body is represented by a set of N particles and a set of M constraints. Each particle i has three attributes, a mass m_i, a position $\mathbf{x}_i$ and a velocity $\mathbf{v}_i$. In our model, each particle p_i corresponds to a vertex of the input mesh with $m_i = 1$. A geometric constraint j is a mathematical relationship between particles $C_j(\mathbf{x}) = 0$. The set of constraints must be always satisfied, or at least, the error should be as small as possible. During the simulation, if the particles configuration $\mathbf{x}^k$ does not satisfy the set of constraints, then the solver *projects* the particle positions in a valid state by finding a displacement $\Delta\mathbf{x}^k$ such that $C(\mathbf{x}^k + \Delta\mathbf{x}^k) = 0$.

Given the set of N particles and of M constraints, the simulation proceeds as described by Algorithm 1.

Algorithm 1 Position Based skinning

```
1:  for all particles $i$ do
2:      $\mathbf{v}_i = \mathbf{v}_i^0$
3:      $\mathbf{x}_i = \mathbf{x}_i^0$
4:  end for
5:  loop
6:      for all particles $i$ do
7:          $\mathbf{v}_i = \mathbf{v}_i + \mathbf{a}_{ext}\Delta t$
8:          $\mathbf{p}_i = \mathbf{x}_i + \mathbf{v}_i\Delta t$
9:      end for
10:     for $k \leftarrow 1, N_{iterations}$ do
11:         projectBindConstraints($\mathbf{p}_i$)
12:         projectStretchConstraints($\mathbf{p}_i$)
13:         projectVolumeConstraints($\mathbf{p}_i$)
14:     end for
15:     for all particles $i$ do
16:         $\mathbf{v}_i = (\mathbf{p}_i - \mathbf{x}_i)\,/\Delta t$
17:         $\mathbf{x}_i = \mathbf{p}_i$
18:     end for
19: end loop
```

where Δt is the time step.

The positions and velocities of the particles are initialized in (1)-(4) before the simulation loop starts. Lines (6)-(9) perform simple explicit forward Euler integration step on the velocities and the positions, where we apply gravity as external acceleration $\mathbf{a}_{ext}$. The geometric constraints are iteratively solved in (10)-(14). The idea is to repeatedly solve each constraint sequentially one after the other, in a Gauss-Seidel type fashion. The process is repeated $N_{iterations}$ times. The corrected positions $\mathbf{p}_i$ are finally used to update the positions and the velocities in (15)-(18).

Constraint projection: All the constraints in our method are functions $C_i(\mathbf{p}) = 0$. The stiffness parameter k_j defines the strength of the constraint in a range from zero to one. Projecting a set of particles according to a constraint means moving the particles such that they satisfy the constraint. The most important issue in connection with moving particles directly inside a simulation loop is the conservation of linear and angular momentum.

Linear momentum is conserved if

$$\sum_i m_i \Delta\mathbf{p}_i = \mathbf{0} \tag{3}$$

where $\Delta\mathbf{p}_i$ is the displacement of the particle i due to the projection.

Angular momentum is conserved if

$$\sum_i \mathbf{r}_i \times m_i \Delta\mathbf{p}_i = \mathbf{0} \tag{4}$$

where the $\mathbf{r}_i$ are the distances of the $\mathbf{p}_i$ to an arbitrary common rotation center.

Given $\mathbf{p}$ we want to find a correction $\Delta\mathbf{p}$ such that $C(\mathbf{p} + \Delta\mathbf{p}) = 0$. This equation can be linearized:

$$C(\mathbf{p} + \Delta\mathbf{p}) \approx C(\mathbf{p}) + \nabla_{\mathbf{p}} C(\mathbf{p}) \cdot \Delta\mathbf{p} = 0 \tag{5}$$

If $\Delta\mathbf{p}$ is chosen to be parallel to $\nabla_p C(\mathbf{p})$ (which is perpendicular to rigid body modes), then both linear and angular momenta are conserved. Therefore it can be imposed:

$$\Delta\mathbf{p} = \lambda \nabla_{\mathbf{p}} C(\mathbf{p}) \tag{6}$$

Substituting Eq. 6 into Eq. 5, solving for λ and substituting it back into Eq. 6 yields the final formula for $\Delta\mathbf{p}$:

$$\Delta\mathbf{p} = -\frac{C(\mathbf{p})}{\left|\nabla_{\mathbf{p}} C(\mathbf{p})\right|^2} \nabla_{\mathbf{p}} C(\mathbf{p}) \tag{7}$$

For the correction of an individual particle $\mathbf{p}_i$:

$$\Delta\mathbf{p}_i = -s \nabla_{\mathbf{p}_i} C(\mathbf{p}) \tag{8}$$

where the scaling factor is

$$s = \frac{C(\mathbf{p})}{\sum_j \left|\nabla_{\mathbf{p}_j} C(\mathbf{p})\right|^2} \tag{9}$$

In the following section we show how to apply the above formulas for resolving the geometric constraints in our model.

3.3 Geometric Constraints

As mentioned in the previous section, we define a particle $\mathbf{p}_i$ for each vertex of the input mesh. We define a set of constraints M as depicted in the following subsections.

3.4 Stretch Constraint

We define one *stretch* constraint for the particles $(\mathbf{p}_1, \mathbf{p}_2)$ at the end points of each edge of the mesh, including the edges of the internal tetrahedrons:

$$C(\mathbf{p}_1, \mathbf{p}_2) = |\mathbf{p}_1 - \mathbf{p}_2| - d = 0 \tag{10}$$

where d is the rest length of the edge.

In order to represent how much the edge can be stretched or compressed, we provide a stiffness parameter $k_{stretch}$, which controls the elasticity of the tissues.

For a full mathematical description of this constraint, the reader may refer to [Müller et al. 2007; Bender et al. 2013].

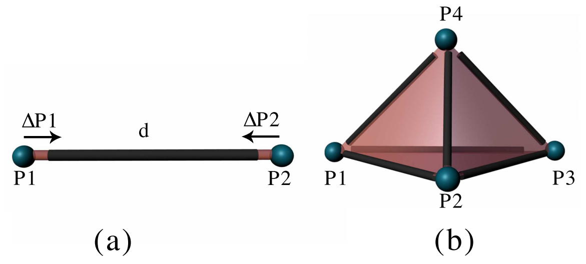

Figure 6: Shows a stretch and volume constraint (a) illustrates the stretch constraint on one edge (b) illustrates six stretch constraint on six edges of a single tetrahedron in addition to volume constraint.

3.5 Tetrahedral Volume Constraint

We define one *volume* constraint for the particles $(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4)$ at the corners of each tetrahedral of the mesh [Aldrich et al.]:

$$C(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4) = \frac{1}{6}(\mathbf{p}_{2,1} \times \mathbf{p}_{3,1}) \cdot \mathbf{p}_{4,1} - V_0 \tag{11}$$

where $\mathbf{p}_{i,j}$ is the short notation for $\mathbf{p}_i - \mathbf{p}_j$ and V_0 is the rest volume of the tetrahedral.

The gradient with respect to the particles are:

$$\nabla_{p2} C(\mathbf{p}_{2,3}) = \frac{1}{6}(\mathbf{p}_2 \times \mathbf{p}_3) \tag{12}$$

$$\nabla_{p3} C(\mathbf{p}_{3,4}) = \frac{1}{6}(\mathbf{p}_3 \times \mathbf{p}_4) \tag{13}$$

$$\nabla_{p4} C(\mathbf{p}_{4,2}) = \frac{1}{6}(\mathbf{p}_4 \times \mathbf{p}_2) \tag{14}$$

$$\nabla_{p1} C(\mathbf{p}_1) = -(\nabla_{p2} C(\mathbf{p}_{2,3}) + \nabla_{p3} C(\mathbf{p}_{3,4}) + \nabla_{p4} C(\mathbf{p}_{4,2})) \tag{15}$$

which means:

$$\nabla_{p1} C(\mathbf{p}_1) = -(\frac{1}{6}(\mathbf{p}_2 \times \mathbf{p}_3) + \frac{1}{6}(\mathbf{p}_3 \times \mathbf{p}_4) + \frac{1}{6}(\mathbf{p}_4 \times \mathbf{p}_2)) \tag{16}$$

Therefore, for the correction of each particle inside the tetrahedron:

$$\Delta\mathbf{p}_1 = -s \cdot k_{volume} \cdot \left(\frac{1}{6}(\mathbf{p}_2 \times \mathbf{p}_3) + \frac{1}{6}(\mathbf{p}_3 \times \mathbf{p}_4) + \frac{1}{6}(\mathbf{p}_4 \times \mathbf{p}_2)\right) \tag{17}$$

$$\Delta\mathbf{p}_2 = s \cdot k_{volume} \cdot \frac{1}{6}(\mathbf{p}_2 \times \mathbf{p}_3) \tag{18}$$

$$\Delta\mathbf{p}_3 = s \cdot k_{volume} \cdot \frac{1}{6}(\mathbf{p}_3 \times \mathbf{p}_4) \tag{19}$$

$$\Delta\mathbf{p}_4 = s \cdot k_{volume} \cdot \frac{1}{6}(\mathbf{p}_4 \times \mathbf{p}_2) \tag{20}$$

where k_{volume} is the stiffness parameter and s is the scaling factor from Eq. 9:

$$s = \frac{\frac{1}{6}(\mathbf{p}_{2,1} \times \mathbf{p}_{3,1}) \cdot \mathbf{p}_{4,1} - V_0}{\Sigma_{i=1}^{4} \mid \nabla_{\mathbf{p}_i} C(\mathbf{p}_i) \mid^2} \tag{21}$$

3.6 Bind Constraint

We define a *bind* constraint between each particle and its nearest bone. Basically, a bind constraint is a stretch constraint between a particle and its projection on the nearest skeleton bone. When the skeleton moves and the joints rotate, the projection point for each particle is updated accordingly and the bind constraint push or pull the particle to maintain the rest distance. This mechanism is depicted in Fig.7.

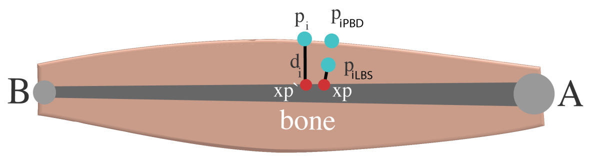

Figure 7: *Bind* constraint. The particle position p$_i$ is projected on the bone. While moving, the distance of the particle from the bone is compared to the distance of the rest position d$_i$ in order to maintain the distance to the bone, where p$_{iLBS}$ is the particle's position p$_i$ after LBS and p$_{iPBD}$ is the particle's position after PBD.

4 Results

We tested our proposed technique on three standard open-source characters (Fig. 1, 8, 9, 10). The Man and Lady surface meshes from the MakeHuman open-source software, bunny character is from Blender open-source software. The tetrahedral meshes were generated using CGAL [Alliez et al. 2013], and the motion capture data is from the Carnegie Mellon University Motion Capture DataBase [Gross and Shi 2001].

Visual quality: In this section, as in the accompanying video, we compare our method with both linear blend skinning (LBS) and dual quaternion skinning (DQS). Our method successfully overcomes two main types of artefacts (Fig.8). The first one is the well-known "*candy-wrapper*" artefact of LBS, the second one is the joint-bulging artefact of DQS.

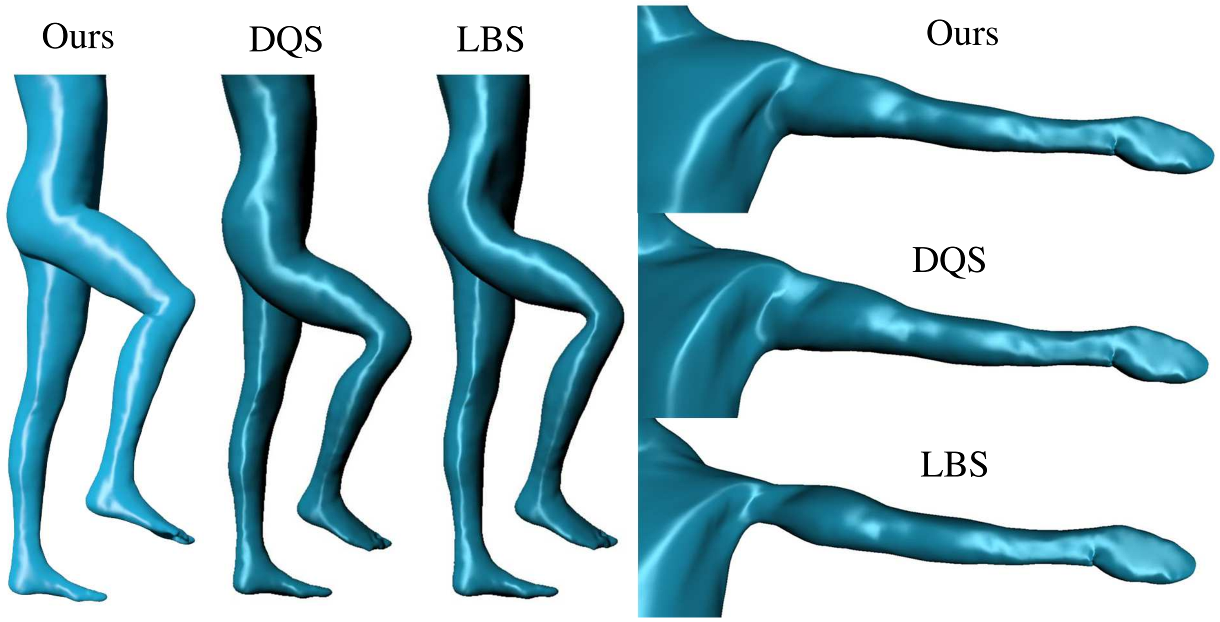

Figure 8: The candy-wrapper artefacts of linear blend skinning and the bulging artefacts of dual quaternion skinning do not occur in our method.

The first step in the animation process, the LBS deformer, improves the convergence speed of the PBD solver (Fig. 10), while maintaining the elastic nature of the character body.

To measure the accuracy of our position based skinning approach, we compute geometric error relative to the volume constraint. First we compute the volume of each tetrahedron inside the volumetric mesh in the rest pose V_0. In each frame, after preform both linear blend skinning and position based skinning, we compute the volume again V'. Therefore, the error is the amount of losing volume E= V_0-V'. While the LBS results suffer under a volume loss of about 14% for the whole mesh, our model successfully preserves the volume and differs only by 0.5 % from the initial volume by iterating 24 times iterations in PBD every time step.

Jiggling: We consider the vertices that have rest distance from their nearest bone greater than the average distances of all vertices to their nearest bone as belonging to the *jiggling* zone. For these vertices, we disable the LBS deformer and tune the $k_{stretch}$ stiffness for all the stretch constraints belonging to this area. When the skeleton moves, this are exhibit passive jiggling deformations, like depicted in Fig. 9.

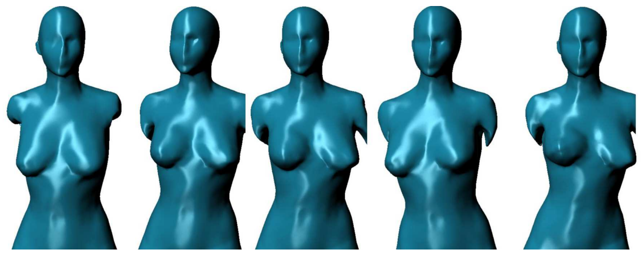

Figure 9: A realistic jiggling behavior of soft tissues for a walking lady.

Performance: In our method the tetrahedral meshes have roughly 2K vertices and 10K tetrahedral elements. To advance the dynamics, we used a 10 ms time step and 24 iterations per frame. We measured the time performances on a set of selected animations on a mass-market laptop equipped with an Intel i5 2.50 GHz processor and 4GB RAM. The mean computation times are reported in Table 1. The animations are shown in the accompanying video.

5 Conclusion and Future Work

We have presented a simple and fast skinning algorithm handling the general problem of skeleton-influenced deformations. Characters with different polygonal resolutions and topologies can be easily accommodated without programming or considerable setup efforts. During the animation, the deformation model preserves the volume and allows for passive jiggling behavior. The artist can control the amount of jiggling by tuning a single scalar stiffness parameter.

Differently from other methods, in particular [Kim and Pollard 2011] and [Deul and Bender 2013], our system does not model the inner structure of the human skin. However, using a combination of widely known (and relatively simple) techniques, linear blend skinning and position-based dynamics, we achieved believable animations with a generally faster performance of the above mentioned methods.

Being based on Position Based Dynamics, the elastic behavior of the soft body deformer is influenced by the number of iterations employed in the iterative Gauss-Seidel solver (Sec. 3.2). For all of our test meshes, we used 24 iterations; in general the artist has to heuristically choose a value which depends from the topology and the polygonal resolution of the input mesh. Our implementation does not currently detect and resolve self-collisions, which may lead to geometry overlapping, therefore our method cannot guarantee self-intersection free deformations. We plan to implement

Table 1: Skinning performance. S: number of stretch constraints, T: number of volume constraints, B: number of bind constraints, fps: avg. frame rate, CT: avg. skinning computation time for running a 1 sec simulation.

Model	# vertices	S	T	B	CT stretch [ms]	CT volume [ms]	CT total [ms]	# iterations	jiggling	fps
Man	1654	6342	4998	1654	33.0	46.0	52.0	24	0.9	90.5
ManHQ	2645	13223	10516	2645	74.2	69.5	72.1	24	0.9	71.1
bunny	2108	9732	7827	2108	63.3	65.1	68.3	24	0.6	76.8
bunnyHQ	3190	17367	14022	3190	83.5	73.5	76.4	24	0.6	65.8
Lady	1829	9387	6032	1829	57.2	61.2	64.5	24	0.7	79.5

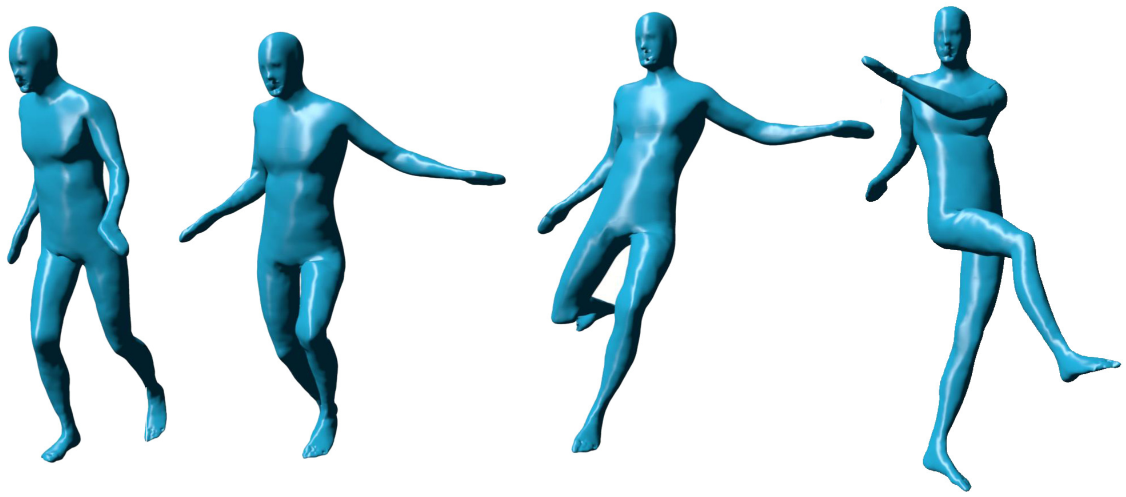
Figure 10: Different poses of character kicking a ball.

self-collisions by generating temporary constraints on-the-fly and including them into our position-based skinning framework. In order to improve the computational performance, we also plan to use a parallel schema for solving the non-linear system of constraints and implementing the position based skinning system on the GPU.

References

ALDRICH, G., PINSKIY, D., AND HAMANN, B. Collision-Driven Volumetric Deformation on the GPU. Eurographics Association, Llandudno, UK, 9–12.

ALLIEZ, P., RINEAU, L., TAYEB, S., TOURNOIS, J., AND YVINEC, M. 2013. 3D mesh generation. In *CGAL User and Reference Manual*, 4.3 ed. CGAL Editorial Board.

ANGELIDIS, A., AND SINGH, K. 2007. Kinodynamic skinning using volume-preserving deformations. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, SCA '07, 129–140.

BARAN, I., AND POPOVIĆ, J. 2007. Automatic rigging and animation of 3d characters. In *ACM SIGGRAPH 2007 Papers*, ACM, New York, NY, USA, SIGGRAPH '07.

BENDER, J., MÜLLER, M., OTADUY, M. A., AND TESCHNER, M. 2013. Position-based methods for the simulation of solid objects in computer graphics. In *EUROGRAPHICS 2013 State of the Art Reports*, Eurographics Association.

BOTSCH, M., AND SORKINE, O. 2008. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics 14*, 1 (Jan.), 213–230.

BRO-NIELSEN, M., AND COTIN, S. 1996. Real-time volumetric deformable models for surgery simulation using finite elements and condensation. In *Computer Graphics Forum*, 57–66.

CAPELL, S., GREEN, S., CURLESS, B., DUCHAMP, T., AND POPOVIĆ, Z. 2002. Interactive skeleton-driven dynamic deformations. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, ACM, New York, NY, USA, SIGGRAPH '02, 586–593.

CAPELL, S., BURKHART, M., CURLESS, B., DUCHAMP, T., AND POPOVIĆ, Z. 2005. Physically based rigging for deformable characters. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM, New York, NY, USA, SCA '05, 301–310.

CHADWICK, J. E., HAUMANN, D. R., AND PARENT, R. E. 1989. Layered construction for deformable animated characters. *SIGGRAPH Comput. Graph. 23*, 3 (July), 243–252.

CHEN, C.-H., LIN, I.-C., TSAI, M.-H., AND LU, P.-H. 2011. Lattice-based skinning and deformation for real-time skeleton-driven animation. In *Proceedings of the 2011 12th International Conference on Computer-Aided Design and Computer Graphics*, IEEE Computer Society, Washington, DC, USA, CADGRAPHICS '11, 306–312.

CLUTTERBUCK, S., AND JACOB, J. 2010. A physically based approach to virtual character deformations. In *SIGGRAPH 2010 Talks*.

DEUL, C., AND BENDER, J. 2013. Physically-based character skinning. In *Virtual Reality Interactions and Physical Simulations (VRIPhys)*. accepted.

DIZIOL, R., BENDER, J., AND BAYER, D. 2009. Volume conserving simulation of deformable bodies. In *Short Paper Proceedings of Eurographics*.

FORSTMANN, S., AND OHYA, J. 2006. Fast Skeletal Animation by skinned Arc-Spline based Deformation . 1–4.

FRATARCANGELI, M. 2012. Position-based facial animation synthesis. *Computer Animation and Virtual Worlds 23*, 3-4, 457–466.

GALOPPO, N., OTADUY, M. A., TEKIN, S., GROSS, M. H., AND LIN, M. C. 2007. Soft articulated characters with fast contact handling. *Comput. Graph. Forum 26*, 3, 243–253.

GOURRET, J.-P., THALMANN, N. M., AND THALMANN, D. 1989. Simulation of object and human skin formations in a grasping task. In *Proceedings of the 16th Annual Conference on Computer Graphics and Interactive Techniques*, ACM, New York, NY, USA, SIGGRAPH '89, 21–30.

GROSS, R., AND SHI, J. 2001. The cmu motion of body (mobo) database. Tech. Rep. CMU-RI-TR-01-18, Robotics Institute, Carnegie Mellon University.

JACKA, D., REID, A., MERRY, B., AND GAIN, J. 2007. A comparison of linear skinning techniques for character animation. In *Proceedings of the 5th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, ACM, New York, NY, USA, AFRIGRAPH '07, 177–186.

KAVAN, L., AND SORKINE, O. 2012. Elasticity-inspired deformers for character articulation. *ACM Trans. Graph. 31*, 6 (Nov.), 196:1–196:8.

KAVAN, L., COLLINS, S., ŽÁRA, J., AND O'SULLIVAN, C. 2007. Skinning with dual quaternions. In *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games*, ACM, New York, NY, USA, I3D '07, 39–46.

KIM, J., AND POLLARD, N. S. 2011. Fast simulation of skeleton-driven deformable body characters. *ACM Trans. Graph. 30*, 5 (Oct.), 121:1–121:19.

KRY, P. G., JAMES, D. L., AND PAI, D. K. 2002. Eigenskin: Real time large deformation character skinning in hardware. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM, New York, NY, USA, SCA '02, 153–159.

LARBOULETTE, C., CANI, M.-P., AND ARNALDI, B. 2005. Dynamic skinning: adding real-time dynamic effects to an existing character animation. In *SCCG*, 87–93.

LEE, Y., TERZOPOULOS, D., AND WATERS, K. 1995. Realistic modeling for facial animation. In *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques*, ACM, New York, NY, USA, SIGGRAPH '95, 55–62.

LEE, S.-H., SIFAKIS, E., AND TERZOPOULOS, D. 2009. Comprehensive biomechanical modeling and simulation of the upper body. *ACM Trans. Graph. 28*, 4 (Sept.), 99:1–99:17.

LEWIS, J. P., CORDNER, M., AND FONG, N. 2000. Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, SIGGRAPH '00, 165–172.

MAGNENAT-THALMANN, N., LAPERRIÈRE, R., AND THALMANN, D. 1988. Joint-dependent local deformations for hand animation and object grasping. In *Proceedings on Graphics Interface '88*, Canadian Information Processing Society, Toronto, Ont., Canada, Canada, 26–33.

MCADAMS, A., ZHU, Y., SELLE, A., EMPEY, M., TAMSTORF, R., TERAN, J., AND SIFAKIS, E. 2011. Efficient elasticity for character skinning with contact and collisions. *ACM Trans. Graph. 30*, 4 (July), 37:1–37:12.

MERRY, B., MARAIS, P., AND GAIN, J. 2006. Animation space: A truly linear framework for character animation. *ACM Trans. Graph. 25*, 4 (Oct.), 1400–1423.

MOHR, A., AND GLEICHER, M. 2003. Building efficient, accurate character skins from examples. 562–568.

MÜLLER, M., DORSEY, J., MCMILLAN, L., JAGNOW, R., AND CUTLER, B. 2002. Stable real-time deformations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM, New York, NY, USA, 49–54.

MÜLLER, M., HEIDELBERGER, B., HENNIX, M., AND RATCLIFF, J. 2007. Position based dynamics. *J. Vis. Comun. Image Represent. 18*, 2 (Apr.), 109–118.

PENTLAND, A., AND WILLIAMS, J. 1989. Good vibrations: model dynamics for graphics and animation. In *SIGGRAPH*, 215–222.

SHI, X., ZHOU, K., TONG, Y., DESBRUN, M., BAO, H., AND GUO, B. 2008. Example-based dynamic skinning in real time. *ACM Trans. Graph. 27*, 3 (Aug.), 29:1–29:8.

SHINAR, T., SCHROEDER, C., AND FEDKIW, R. 2008. Two-way coupling of rigid and deformable bodies. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, SCA '08, 95–103.

TERZOPOULOS, D., PLATT, J., BARR, A., AND FLEISCHER, K. 1987. Elastically deformable models. *SIGGRAPH Comput. Graph. 21*, 4 (Aug.), 205–214.

TURNER, R., AND THALMANN, D. 1993. The elastic surface layer model for animated character construction. In *PROCEEDINGS OF COMPUTER GRAPHICS INTERNATIONAL '93*, SpringerVerlag, 399–412.

WANG, X. C., AND PHILLIPS, C. 2002. Multi-weight enveloping: Least-squares approximation techniques for skin animation. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM, New York, NY, USA, SCA '02, 129–138.

WAREHAM, R., AND LASENBY, J. 2008. Bone glow: An improved method for the assignment of weights for mesh deformation. In *AMDO*, 63–71.