Automatic shape adjustment at joints for the implicit skinning

Olivier Hachette, Florian Canezin, Rodolphe Vaillant, Nicolas Mellado, Loic Barthe

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Automatic shape adjustment at joints for the Implicit Skinning

Olivier Hachette[a], Florian Canezin[a], Rodolphe Vaillant, Nicolas Mellado[a], Loïc Barthe[a]

[a] *Université de Toulouse, IRIT, CNRS*

ARTICLE INFO

Article history:
Received November 17, 2021

Keywords: Shape deformation, Geometric modeling, Skinning, Computer animation

ABSTRACT

The implicit skinning is a geometric interactive skinning method, for skeleton-based animations, enabling plausible deformations at joints while resolving skin self-collisions. Even though requiring a few user interactions to be adequately parameterized, some efforts have to be spent on the edition of the shapes at joints.

In this research, we introduce a dedicated optimisation framework for automatically adjusting the shape of the surfaces generating the deformations at joints when they are rotated during an animation. This approach directly fits in the implicit skinning pipeline and it has no impact on the algorithm performance during animation. Starting from the mesh partition of the mesh representing the animated character, we propose a dedicated hole filling algorithm based on a particle system and a power crust meshing. We then introduce a procedure optimizing the shape of the filled mesh when it rotates at the joint level. This automatically generates plausible skin deformation when joints are rotated without the need of extra user editing.

1. Introduction

Skeleton-based animation is a widely used approach for animating 3D digital characters. In general, an animation is first created by applying linear transformations (e.g. rotations and scales) on the different skeleton bones. Then, the mesh representing the animated character has to be deformed according to its skeleton transformations and model internal structure. This geometry deformation step, called *skinning*, is still time consuming for artists and scientifically very challenging to address when targeting realistic or visually plausible results.

Over the years, a wide range of skinning approaches have been proposed. On the one hand, physically-based methods provide realistic deformations, eventually including secondary motion effects such as muscle inflation and jiggling [2, 3, 4, 5, 6], while avoiding self-collisions [7, 8, 9, 10]. Example-based learning methods can also be used for skinning the character body [11] or learning the parameters of a fast geometric method from a high quality, computationally expensive deformer [12]. All these approaches often require non-intuitive and tedious parameterization or a large set of varying training data to provide a desired result.

On the other hand, very fast geometric skinning methods blend the bone transformations expressed linearly [13, 14, 15] or as dual quaternions [16, 17] at each mesh vertex, using skinning weights. Delta mush skinning avoids the transformation blending by smoothing a rigid transformation of the mesh and restoring details stored in rest pose [18, 19]. These approaches do not handle self-contact and collisions, and they are subject to visual artefacts such as volume loss or bulge. The automatic deformation can however be improved by experienced artists, which makes them attractive for interactive real-time applications such as video games. When tweaking their parameters and adding ghost bones is not sufficient to improve the final shape, the use of pose space deformations may be considered [20]. It enables the production of realistic or desired transformations by interpolating key poses, at the cost of extensive user edits.

Other approaches improve volume preservation [21] and resolve self-collisions while providing an interactive framerate for rigging and animating a character. Projective skin-

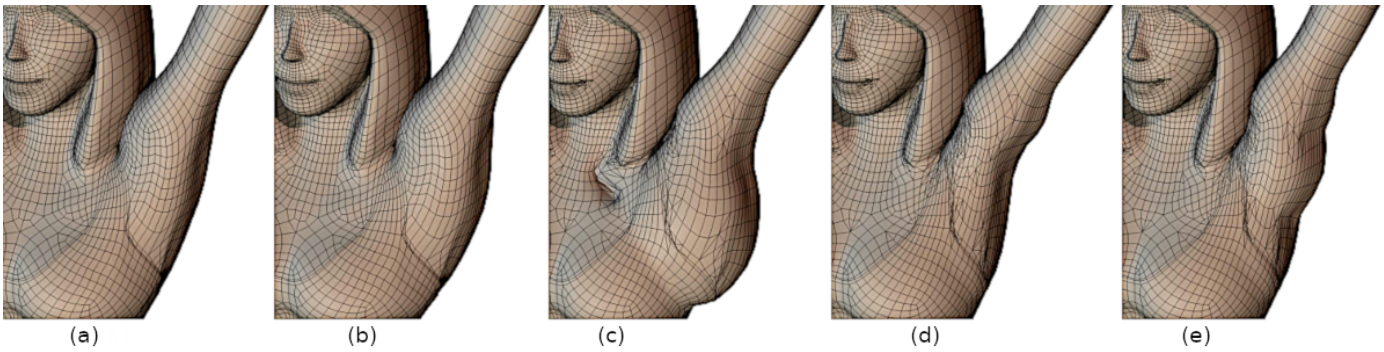


Fig. 1. Result produced by different skinning methods applied to Dana's shoulder. (a) linear blending, (b) dual quaternions, (c) implicit skinning with default settings [1], (d) Ours, (e) implicit skinning [1] with artist edited joint.

ning [22, 23] relies on a mesh tetrahedrisation and a projective dynamics constraint optimization scheme [24] to approximate physical skin behaviors. Also based on a volumetric reconstruction of the character, the implicit skinning technique [25, 1] approximates the mesh with an iso-surface of a 3D scalar field. During motion, the scalar field is deformed according to the skeleton transformations and the mesh is animated by tracking the iso-surface deformations.

The variety of skin deformations at joints produced by the implicit skinning can be enhanced by sketching the skin shapes at different poses using sketch-based blending [26], and the addition of muscles and bones [27] in the implicit skinning framework [28] allows to also produce muscle deformations while still automatically handling skin self-collisions. These make the implicit skinning approach a unified approach very well suited for generating plausible skin deformations for skeleton-based animated characters and its improvement is the topic of this research.

Implicit Skinning. The implicit skinning method takes as input a partitioned mesh with respect to the skeleton bones and a set of default skinning weights (Figure 2-a). Each mesh part is approximated by a 3D scalar field defined by an Hermite Radial Basis Function (HRBF) [29, 30] and these scalar fields are combined in a composition tree [31] to define a final scalar field whose 0.5-iso-surface represents the character skin. During a rotation at a joint, each part's scalar field is rigidly transformed and the shape of the skin at the joint is defined by the way the 0.5-iso-surfaces of the parts slide over each other (Figures 2-e and 6). Even though a great advantage of the implicit skinning is the significant reduction of the user interactions required to rig a character, a specific effort still has to be spent to manually adjust the shape of the parts' iso-surfaces at joints to get plausible results when they slide over each other during rotations. This is especially necessary at complicated articulations such as shoulders, as illustrated in Figures 1-c,e. In addition, the interactive editing of the iso-surface through control points and normals to adjust the character skin shape at joints during its animation appears unnatural for artists that are used to mesh editing tools.

Contribution. Starting from the mesh parts (Figures 2-b,c) and their approximating HRBFs (Figure 2-d), we first use particles and a Power Crust algorithm to close the meshes (Section 3, Figure 2-f and Figure 3). We then propose a dedicated optimization scheme automatically adjusting the mesh vertices' position where parts slide when joints are rotated (Section 4). It is composed of three steps: (1) a pre-fairing step (Section 4.1, Figure 2-f), (2) a sliding optimisation step (Section 4.2, Figure 2-g) and (3) a post-fairing step (Section 4.3, Figure 2-h).

We thus introduce two scientific contributions: A new dedicated pipeline based on specifically adapted and parameterized state of the art techniques (i.e. a particle system and a Power Crust meshing), and a new optimization scheme automatically adjusting the animated shapes around the joints.

In Section 5, we validate our approach by showing the improvement of our automatically generated deformation at joints (Figure 1-d) on those produced automatically by Vaillant et al. [1] (Figure 1-c). We also illustrate the close similarity of our results with those manually edited by experienced artists (Figure 1-e) and those automatically produced by the Fast Projective Skinning [23] (Figure 10).

2. Related works

Our first requirement is to close the open meshes provided by the initial animated model partition. Then, we adjust the shape of the closed parts where they slide over each other when they are rotated around a joint. We thus review the main mesh hole filling techniques before presenting our approach for controlling the animated mesh deformation at joints. Hole filling in meshes is a long standing problem and many solutions have been proposed over the years as described by Hernández et al. [32]. Two main families of methods can be considered as suggested by Guo et al. [33]: geometric and volumetric approaches.

Geometric methods. They leverage the knowledge provided by the topology of the model to produce appropriate filling. For instance, the advancing front algorithm gradually adds new faces from the mesh bounding edges [34]. The same kind of method for point clouds adds new points from an initial point cloud boundaries [35]. These methods generate water tight meshes or closed point clouds. Other methods rather first coarsely triangulate the hole, then refine the triangulation to generate a denser

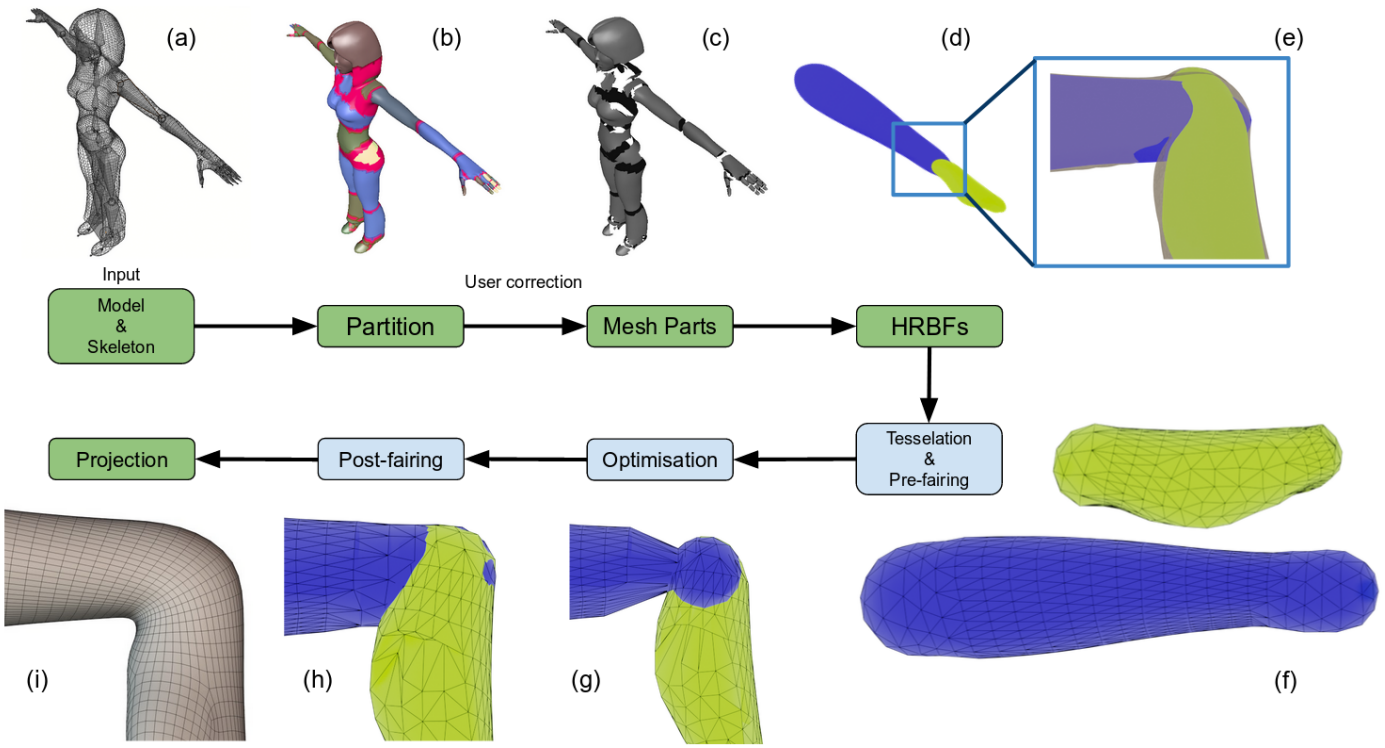


Fig. 2. Overview of our approach. (a) Input mesh and its associated skeleton. (b) Model segmentation with respect to the skinning weights. (c) Mesh parts resulting from the input mesh segmentation. (d) 0.5-iso-surfaces interpolating the mesh parts of the forearm in blue and the hand in green. (e) Bulging shape generated at the wrist joint when parts are rotated. (f) Closure of the mesh parts following the 0.5-iso-surfaces and pre-fairing of the extremity shapes. (g) Iterative shape optimization removing the bulge when limbs rotate. (h) Post-fairing generating the final shape for rigging the model. (i) Result of the implicit skinning applied on our optimized shapes at joints.

topology and finally apply some geometric operators to smooth the new mesh filling the hole, eventually based on the harmonic umbrella operator [36]. In these approaches, the locality of the algorithm does not ensure the mesh global continuity. Whatever the way the hole has been filled, the quality of the new mesh can be further improved by fairing the surface normals solving an harmonic equation and then computing the vertex positions using the Poisson equations [37]. In fact, depending on the hole size, different methods may have varying efficiency and Feng et al. [38] propose to adapt the filling strategy to the hole size.

Volumetric methods. Such approaches rely on 3D field functions to fill the model holes. Their main advantage is the capacity of field functions to automatically fill holes of arbitrary topology with smooth surfaces, though they still have to be meshed and they do not provide fine and local control of the surface shape. This idea was first introduced to complete scanned data [39] with weighted distance fields. A diffusion on a distance field [40] or a biharmonic field [41] discretized in a 3D grid can also be used to fill holes. These approaches can be improved using an octree and Hermite data to reconstruct edges [42]. The use of a discrete structure can be avoided with continuous smooth interpolation of field functions such as Radial Basis Functions [43] or Algebriac Point Set Surfaces [44]. These approaches may fail in providing an adequate topology and better guarantees are provided by the integration of a Poisson reconstruction and restricted Delaunay triangulations [45]. A more recent method uses an advancing front guided by field functions to achieve hole filling in specified areas [46].

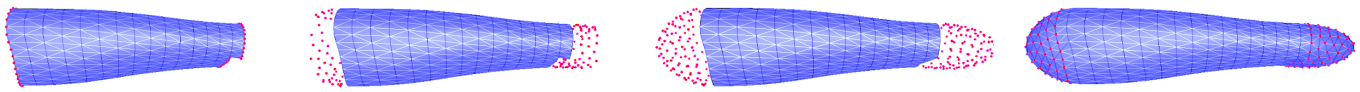
In this research, we need to both close the mesh parts (illustrated on the Dana model in Figure 2-c) and adjust the shape of the closure (Figure 2-e) in order to provide a plausible deformation at joints where closed parts slide on each other (Figures 2-h,i). We thus take inspiration from the conjoint use of field functions and mesh processing techniques to provide both an efficient hole filling and a mesh shape optimization.

While the use of HRBF enables the automatic generation of closed implicit surfaces approximating the mesh parts, it is not computationally well suited for optimizing the closure shapes around joints. We thus close the mesh parts following the closed approximating implicit surfaces to rely on mesh processing techniques for the shape optimization. Once the mesh shape optimized, we generate the final optimized implicit surface required by the implicit skinning framework by interpolating the optimized closed mesh vertices (as done for the initial open mesh parts). This leads us to the processing pipeline presented in Figure 2.

3. Parts closure

3.1. Partitioning

Following the standard interactive skinning pipeline, we start from the input skeleton and character mesh (Figure 2-a). The mesh is split into mesh parts corresponding to the skeleton



(a) (b) (c) (d)

Fig. 3. Illustration of the closure procedure. (a) We first add particles at the holes boundaries. (b-c) Particles are then iteratively displaced, subdivided and/or removed along the HRBF 0.5-iso-surface following the minimization of an energy based on their density. (d) Once stabilized, the particles are meshed with a Power Crust method.

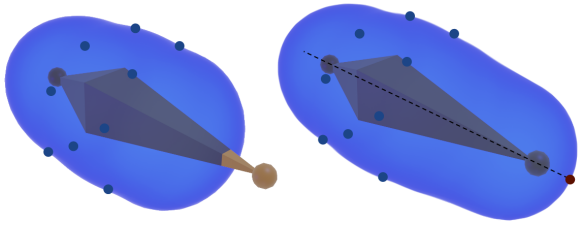


Fig. 4. The blue dots are the constraints interpolated by the HRBF whose 0.5-iso-surface is shown in blue. On the left we can see that a boundary of the bone is outside of the implicit surface, and the value of the scalar field is thus lower than 0.5 at this bone extremity. On the right, adding the new red constraint aligned to the bone solves this issue.

bones, either based on the skinning weights (that may be default weights used for geometric skinning), on the bone proximity, or following a user provided partitioning. In Figure 2-b the removed vertices at joints are depicted in pink and Figure 2-c shows the resulting mesh parts produced on the Dana Model. In this example, we use the skinning weights for partitioning the model.

As suggested by Vaillant et al. [25], we apply a Poisson disk sampling [47] on each mesh part vertices, from which we compute an interpolating HRBF field function (Figure 2-d). A HRBF generates a field function per mesh part, whose 0.5-iso-surface is a closed surface closely approximating the mesh part vertices. The automatic closure of the mesh parts may be far from the closure required by the implicit skinning to provide a plausible skin deformation at joints. Vaillant et al. [25] thus suggested to add a control sample on each side of the bone at a distance of the bone extremity equal to the average of the mesh part boundary vertices in order to produce a result closer to an expected solution (even though often not fully satisfactory as illustrate in Figures 1-c and 8-1st-column).

In the implicit skinning [25, 1], this is the only shape control provided at joints and any incorrect (Figure 1-c) or approximative (Figures 2-e and 6) shape has to be manually edited by displacing / adding / removing HRBF control samples. In our case, we only require that the 0.5-iso-surface of the HRBF encompasses the bone extremities. This is enforced by computing the HRBF value at the bone extremities and checking if the value is greater than 0.5 (we use the convention in which the field function is greater than 0.5 inside the implicit surface defined by the 0.5-iso-surface). If it is not the case, we then add an HRBF control sample aligned with the bone, outside its extremity as illustrated in Figure 4. This provides the closed field functions from which we close the mesh parts.

3.2. Particles

We now populate the holes filled by HRBFs with particles by adapting to our need existing particles systems over implicit surfaces [48, 49]. These particles represent the new vertices used to create the final mesh at the end of the closure process (Section 3.3). To do this, we use the mesh part vertices as fixed particles that repulse new moving particles initially added at the part boundaries. When moving, the new particles slide along the 0.5-iso-surface of the HRBF and fill the holes.

Particles are thus initialized by the part mesh vertices and two additional particles are located at each part mesh boundary vertices (Figure 3-a). We add two particles by vertex located at the part boundary rather than a single in order to start with a number of particles closer to the required final one. A density function ρ_i and an energy E_i is associated to each particle i. For N particles, they are defined as follows:

$$\rho_i(\mathbf{p}) = \frac{1}{\sigma_i\sqrt{2\pi}} e^{-\left(\frac{\|\mathbf{p}-\mathbf{p_i}\|}{\alpha}\right)^2}, \quad (1)$$

and

$$E_i = \sum_{j\neq i}^{N} \rho_j(\mathbf{p_i}). \quad (2)$$

In Equation 1, α and σ_i are defined as explained below.

The parameter α allows the adjustment of the particle density to the approximate target number of particles, here experimentally set to $N_t = 500$. It is defined as:

$$\alpha = \frac{4r}{\sqrt{-\log(0.1)N_t}}, \quad (3)$$

where r is the maximal distance between a moving particle and the axis passing by the barycenter of all particles in the direction of the first eigenvector of their PCA.

The parameter σ_i is initialized to 1 for all particles. Then, energies E_i are computed with $\sigma_i = 1$ and, once these energies computed, σ_i is set to $\sigma_i = \frac{E_i}{\sqrt{2\pi}}$ for all particles coming from the mesh part. These mesh part particles are fixed during all the hole filling procedure and this value of σ_i, set empirically during our experiments, ensures that they adequately repulse the new particles covering the holes.

Once initialized, only the new particles placed at mesh boundaries are displaced, subdivided or removed, driven by the

minimization of their energy E_i implemented with the following iterative process. At each iteration, energies E_i are computed for each particle and, if a particle has an energy lower than E_{min}, it is subdivided in two particles, while if it has an energy greater than E_{max}, it is removed. E_{min} and E_{max} are experimentally set respectively to 0.05 and 0.1. The energy of the subdivided particles is recomputed and particles are displaced following the gradient of the energy and a projection onto the 0.5-iso-surface of the HRBF field. The gradient is given by:

$$\nabla E_i = \sum_{j \neq i}^{N} \nabla \rho_j(\mathbf{p_i}) = \sum_{j \neq i}^{N} \frac{-2}{\alpha^2} \rho_j(\mathbf{p_i}) \, (\mathbf{p_i} - \mathbf{p_j}) \, , \tag{4}$$

and, following Witkin and Heckbert [48], the addition of the projection on the HRBF leads us to the following update of the particle position $\mathbf{p_i}$:

$$\mathbf{p_i} = \mathbf{p_i} + \alpha \left(\nabla E_i - \frac{(\nabla f(\mathbf{p_i}) \cdot \nabla E_i) \, \nabla f(\mathbf{p_i})}{\nabla f(\mathbf{p_i}) \cdot \nabla f(\mathbf{p_i})} \right) . \tag{5}$$

This process of energy computation, subdivision/removal and displacement is repeated (Figure 3-b) while the variation of the number of particles does not exceed 5% in 5 successive steps. Once this criterion is reached, we perform a few more displacements (5 in our implementation) without subdividing or removing particles to produce a more uniform final sampling (Figure 3-c).

3.3. Re-Meshing

Once the particle system has converged, we apply the Power Crust meshing algorithm [50] to compute a mesh topology taking as vertices the particles, and we perform a union of this closing mesh with the initial part mesh (Figures 2-f and 3-d). The Power Crust proposed by Amenta et al. [50] first reconstructs an approximate medial axis and a medial axis transform from the set of vertices. It then uses the inverse of this medial axis transform with a weighted Voronoi diagram to reconstruct a topology over the input vertices. In our settings, we directly use the bones as approximation of the medial axis and we take benefit of the simplicity and the robustness of this power-crust algorithm to compute the mesh topology over the closing particles provided by the particle system.

4. Shape fitting

We now optimize the shape of the closed mesh representing each part. During this step, we do not modify the position of the initial part mesh vertices and we focus on those added to fill the holes.

4.1. Pre-fairing

We start by computing a shape of low curvature variation by displacing the mesh vertices so that they solve the Laplace equation expressed on their distance to the bone (Figure 2-f). We thus define $\mathbf{p}'_i$, the projection of a particle position $\mathbf{p}_i$ onto its associated bone as:

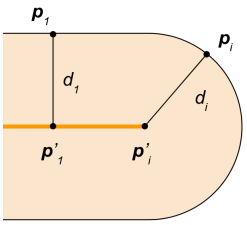


Fig. 5. 2D illustration of points $\mathbf{p_i}$, their projection $\mathbf{p'_i}$ and the distance $d_i = \|\mathbf{p}_i - \mathbf{p}'_i\|$.

$$\mathbf{p}'_i = \mathbf{A} + \max(\min(\frac{(\mathbf{p}_i - \mathbf{A}) \cdot (\mathbf{B} - \mathbf{A})}{\|(\mathbf{B} - \mathbf{A})\|^2}, 1), 0)(\mathbf{B} - \mathbf{A}) \, , \tag{6}$$

where $\mathbf{A}$ and $\mathbf{B}$ are the bone extremities. We define $d_i = \|\mathbf{p}_i - \mathbf{p}'_i\|$ as illustrated in Figure 5, and compute the discreet Laplace-Beltrami operator with cotangent weights [51] as:

$$\Delta = \sum_{j \in \mathcal{N}_i} \omega_{ij} \left(\mathbf{p_i} - \mathbf{p_j} \right) \, , \tag{7}$$

where $\mathcal{N}_i$ is the set of indices of vertices in the one ring neighborhood of $\mathbf{p_i}$ and $\omega_{ij} = (\cot \alpha_{ij} + \cot \beta_{ij})$ with α_{ij} and β_{ij} being the angles opposite to the edge $[\mathbf{p_i}, \mathbf{p_j}]$. We then solve the Laplace equation on $d = \{d_i\}$ with as constraints the set of distances $\hat{d}$ of the initial part mesh boundary vertices to their skeleton:

$$\Delta \, d = \begin{bmatrix} L00 & L10 \\ L01 & L11 \end{bmatrix} \begin{bmatrix} \hat{d} \\ \tilde{d} \end{bmatrix} = \begin{bmatrix} D \\ 0 \end{bmatrix} \, , \tag{8}$$

where L_{kl}, $k, l \in \{0, 1\}$, are matrices of cotangent weights, $\tilde{d}$ are the unknown, and D and 0 are respectively the column vector of computed constraints and the zero values of the Laplace equation on distances $\tilde{d}$ to the bone at free vertices. This equation can be reduced and solved as:

$$L11 \, \tilde{d} = -L01 \, \hat{d} \, . \tag{9}$$

When starting from a cylindrical initial mesh part, this process fills the holes on each side with an half sphere.

4.2. Sliding optimization

We now optimize the vertices position when bones are rotated around joints. Our goal is to avoid the bulge that is often produced when the rotation angle increases, as shown, for example, in Figures 6, 1-c, 8-left. To do so, we define a bulging energy E^B for minimizing the bulging area produced in the different poses for a range of rotation angles defined between two bones at a joint. During optimization at a joint, we consider one bone fixed (the one of $\mathbf{p_i}$) and the other one in rotation.

$$E^B = \sum_{R \in \mathcal{N}_R} E_R \text{ with } E_R = \sum_{i}^{N_h} \phi_i(R) \frac{(\mathbf{p}_i - \tilde{\mathbf{p}}_{i,R}) \cdot (\mathbf{p}_i - \mathbf{p}'_i)}{\left\|\mathbf{p}_i - \mathbf{p}'_i\right\|} \, , \tag{10}$$

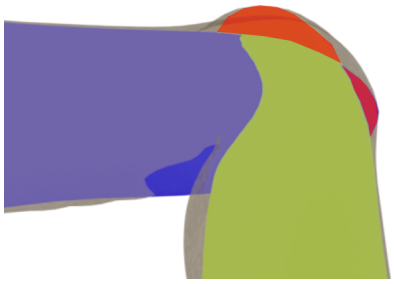


Fig. 6. In red, the bulge generated during the rotation of the wrist.

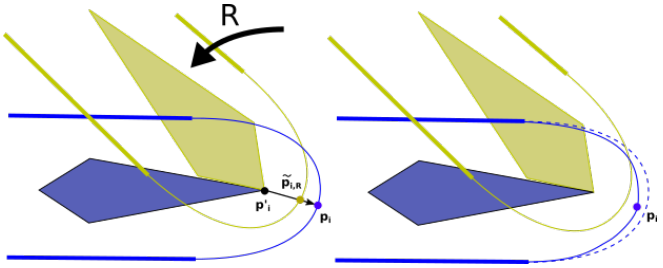


Fig. 7. Illustration of the displacement of a mesh closure point at a rotation R during the bulge removal optimisation. The point $\mathbf{p_i}$ (left-blue) is moved (right-blue) towards its projection onto the rotated mesh $\tilde{\mathbf{p}}_{i,R}$ (left-yellow).

where $\mathcal{N}_R$ is the set of rotations discretizing the range of possible rotations around the joint (we found that 20 rotations per free direction of rotation is a minimum to obtain a plausible result), E_R is the energy per rotation R, N_h is the number of points filling the holes of the optimized mesh around the joint, and $\tilde{\mathbf{p}}_{i,R}$ is the closest point to the projection of the point $\mathbf{p_i}$, in the radial direction $(\mathbf{p}_i - \mathbf{p}'_i)$, onto the mesh of the other bone as illustrated in Figure 7-left. The use of the function $\phi_i(R)$ avoids the optimisation of rotated points whose projection onto the other mesh is outside the optimized area: when the projected point is not in direct proximity of the filled part of the other mesh or when $\mathbf{p}'_i$ is further than a third of the bone from the current joint center. We thus set $\phi_i(R) = 0$ when the projection is onto the initial part of the other mesh and the point is inside this part. Otherwise, $\phi_i(R) = 1$.

To minimize this energy E^B, we first compute $\tilde{\mathbf{p}}_{i,R}$ as:

$$\tilde{\mathbf{p}}_{i,R} = \arg\max_j \frac{\mathbf{p}_i - \mathbf{p}'_i}{\left\|\mathbf{p}_i - \mathbf{p}'_i\right\|} \cdot \frac{\mathbf{q}_{j,R} - \mathbf{p}'_i}{\left\|\mathbf{q}_{j,R} - \mathbf{p}'_i\right\|}\,, \tag{11}$$

where $\mathbf{q}_{j,R}$ are points on the rotated mesh. We then evaluate the energies E_R for the different rotations in $\mathcal{N}_R$. We use these energy values to randomly select a rotation with a probability equal to E_R/E^B for each rotation. For the selected rotation R, we displace the points $\mathbf{p}_i$ generating a bulge (those located outside the other mesh, i.e. when $\left\|\mathbf{p}_i - \mathbf{p}'_i\right\| > \left\|\tilde{\mathbf{p}}_{i,R} - \mathbf{p}'_i\right\|$) inward following Equation 12:

$$\mathbf{p}_i = \mathbf{p}_i - \kappa\,\phi_i(R)\,(\left\|\mathbf{p}_i - \mathbf{p}'_i\right\| - \left\|\tilde{\mathbf{p}}_{i,R} - \mathbf{p}'_i\right\|)(\mathbf{p}_i - \mathbf{p}'_i)\,, \tag{12}$$

where κ is experimentally set to 0.2 to avoid large steps that would introduce instabilities in the minimization process. We then compute the new value of the bulging energy E_B. We iterate this process of computation of E_R, selection of a rotation, and displacement of points $\mathbf{p}_i$ until the bulging energy E^B is stabilized.

4.3. Post-fairing

Even though the sliding optimization step provides satisfactory results with respect to the joint rotation, it does not preserve the continuity of the shape at the boundary between the filling vertices and the original part mesh vertices, as can be seen on Figure 2-g. This is easily corrected by linearly interpolating between the mesh produced by the pre-fairing step (Section 4.1) at the boundary of the initial part mesh, and the mesh produced by the sliding optimization (Section 4.2) at four triangle rings from the boundary, as illustrated in Figure 2-h.

Once the final mesh produced, it is included in the implicit skinning by applying a Poisson sampling (as done in the standard implicit skinning method, see Section 1) and computing its interpolating HRBF.

5. Results and discussions

We ran our experiments on an 4 cores intel i7-3770K CPU @ 3.50GHz with 16 GB RAM. Our implementation is in C++ for the Implicit Skinning and the closure (Section 3) and we use Matlab for the shape fitting steps (Section 4). We apply our algorithm on several joints of different models, as soon as the default solution is not satisfactory, i.e. when an editing is required by an artist. Overall, we handle a joint in around 10s and all the required joints of a model in a few minutes (Table 1) on a standard computer. This optimisation is done once for each joint during the rigging phase and it has no impact on the implicit skinning algorithm timings during animation.

The automatically produced results are close enough to what an experienced artist produces to avoid the need for extra editing (Figures 1, 8, 9 and 10). It may happen that the solution produced by the artist looks less natural, as in Figure 1-e. This is however the desired deformation for this joint and another artist may have generated a slightly different solution. What is thus important for an automatic solution is not to exactly reproduce a solution produced by an artist, but rather to be close enough while remaining visually plausible (i.e. by approximating physical body properties). In addition, if some extra editing is necessary, the artist can now directly edit the final optimized meshes with the standard mesh deformation tools provided by commercial modeling / rigging softwares in a few interactions.

We also compare our deformations to those produced by the Fast Projective Skinning [23] in Figures 9 and 10. This constraint-based method also preserves the volume of the character body and generates skin contact deformations. Overall, the results produced by this method and ours are quite similar. However, our optimized implicit skinning tends to produce less deep contacts and it better preserves the limbs thickness.

Discussions and limitations. We have tested our method on models having different morphology and on the complicated

Table 1. Average timings (in seconds) on the optimization of different joints for different models.

Model (#parts)	Part Closure		Shape Fitting			Total
	Particles Spreading	Re-meshing	Pre-fairing	Sliding Optimization	Post fairing	
Hand (21)	0.33	0.23	0.30	10.5	<0.01	11.36
Male (24)	1.00	0.42	0.08	9.4	<0.01	10.90
Armadillo (35)	0.26	0.23	0.31	12	<0.01	12.80
Dana (54)	1.91	0.20	0.17	8	<0.01	10.28

case of a hand. It always provided the expected results. In rare cases, the process may fail because the initial mesh part is segmented too close from the joint. In that case, a very fast user interaction with a painting tool is enough to reduce the mesh part.

Another limitation would be the use of our optimisation on a non-humanoid model having large shape details at the joint. In that case, a specific user edition is required to adapt the deformation to the specificity of the model.

We notice an increase in the mean time spent in the particle system on the Dana model. This is due to a set of mesh parts for which the HRBF closure is larger and the particle system requires more iterations to converge.

Our method generates a "rounded" support shape at the joint, augmented by the character mesh details captured by the implicit skinning. It does not automatically generate bone shapes as could be desired at a bent elbow. The use of additional bone primitives at joints may be a future direction of investigation.

6. Conclusions

We introduced a new approach automatically generating plausible skin deformations at joints when an animated character is rigged with the implicit skinning [25, 1]. This method avoids the user edits required to adequately parameterize the implicit skinning field functions controlling the skin deformations at joints. The quality of the deformation is similar to those produced by experienced artists and the Fast Projective Skinning. It thus makes the implicit skinning an almost fully automatic technique providing high quality rigs.

Future promising directions of research may include the generation of bone shapes at joints when they bend and the improvement of the implicit skinning frame-rate during animation. On the one hand, adding bone shapes may require the study of specific composition operators at joints and it would increase the versatility of possible skin deformations produced by the implicit skinning, allowing to reach realistic results at joints. On the other hand, decreasing the overall computations required to deform a character with the implicit skinning would increase the versatility of practical applications.

Acknowledgments

This research has been partially supported by the FOLD-Dyn project (ANR-16-CE33-0015-01). We thank Garrett Pond from Brigham Young University (BYU) who modeled the Jeff model which is the big guy from the short movie 'Owned', visible in the supplementary materials.

References

[1] Vaillant, R, Guennebaud, G, Barthe, L, Wyvill, B, Cani, MP. Robust iso-surface tracking for interactive character skinning. ACM Trans Graph 2014;33(6).

[2] Teran, J, Sifakis, E, Irving, G, Fedkiw, R. Robust quasistatic finite elements and flesh simulation. In: Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. SCA '05; New York, NY, USA: Association for Computing Machinery. ISBN 1595931988; 2005, p. 181–190. doi:10.1145/1073368.1073394.

[3] Deul, C, Bender, J. Physically-based character skinning. In: Virtual Reality Interactions and Physical Simulations (VRIPhys). Lille, France: Eurographics Association; 2013,.

[4] Hahn, F, Thomaszewski, B, Coros, S, Sumner, RW, Gross, M. Efficient simulation of secondary motion in rig-space. In: Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation. SCA '13; New York, NY, USA: Association for Computing Machinery. ISBN 9781450321327; 2013, p. 165–171. doi:10.1145/2485895.2485918.

[5] Smith, B, Goes, FD, Kim, T. Stable neo-hookean flesh simulation. ACM Trans Graph 2018;37(2).

[6] Wang, Y, Weidner, NJ, Baxter, MA, Hwang, Y, Kaufman, DM, Sueda, S. Redmax: Efficient & flexible approach for articulated dynamics. ACM Trans Graph 2019;38(4). doi:10.1145/3306346.3322952.

[7] Capell, S, Burkhart, M, Curless, B, Duchamp, T, Popović, Z. Physically based rigging for deformable characters. In: Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. SCA '05; New York, NY, USA: Association for Computing Machinery. ISBN 1595931988; 2005, p. 301–310. doi:10.1145/1073368.1073412.

[8] McAdams, A, Zhu, Y, Selle, A, Empey, M, Tamstorf, R, Teran, J, et al. Efficient elasticity for character skinning with contact and collisions. ACM Trans Graph 2011;30(4). doi:10.1145/2010324.1964932.

[9] Teng, Y, Otaduy, MA, Kim, T. Simulating articulated subspace self-contact. ACM Trans Graph 2014;33(4). doi:10.1145/2601097.2601181.

[10] Brunel, C, Bénard, P, Guennebaud, G. A Time-independent Deformer for Elastic Contacts. ACM Transactions on Graphics 2021;doi:10.1145/3450626.3459879.

[11] Pons-Moll, G, Romero, J, Mahmood, N, Black, MJ. Dyna: a model of dynamic human shape in motion. ACM Trans Graph 2015;34:120:1–120:14.

[12] Bailey, SW, Otte, D, Dilorenzo, P, O'Brien, JF. Fast and deep deformation approximations. ACM Transactions on Graphics 2018;37(4):1–12.

[13] Magnenat-Thalmann, N, Laperrière, R, Thalmann, D. Joint-dependent local deformations for hand animation and object grasping. In: Proceedings on Graphics Interface '88. 1988, p. 26–33.

[14] Alexa, M. Linear combination of transformations. ACM Trans Graph 2002;21(3):380–387. doi:10.1145/566654.566592.

[15] Thalmann, NM, Cordier, F, Seo, H, Papagianakis, G. Modeling of bodies and clothes for virtual environments. In: Proceedings of the 2004 International Conference on Cyberworlds. CW '04; USA: IEEE Computer Society. ISBN 0769521401; 2004, p. 201–208. doi:10.1109/CW.2004.47.

[16] Kavan, L, Collins, S, Žára, J, O'Sullivan, C. Geometric skinning with approximate dual quaternion blending. ACM Trans Graph 2008;27(4):105:1–105:23.

[17] Le, BH, Hodgins, JK. Real-time skeletal skinning with optimized centers of rotation. ACM Transactions on Graphics 2016;35:1–10.

[18] Mancewicz, J, Derksen, ML, Wilson, CA. Delta mush: Smoothing deformations while preserving detail. In: ACM SIGGRAPH 2014

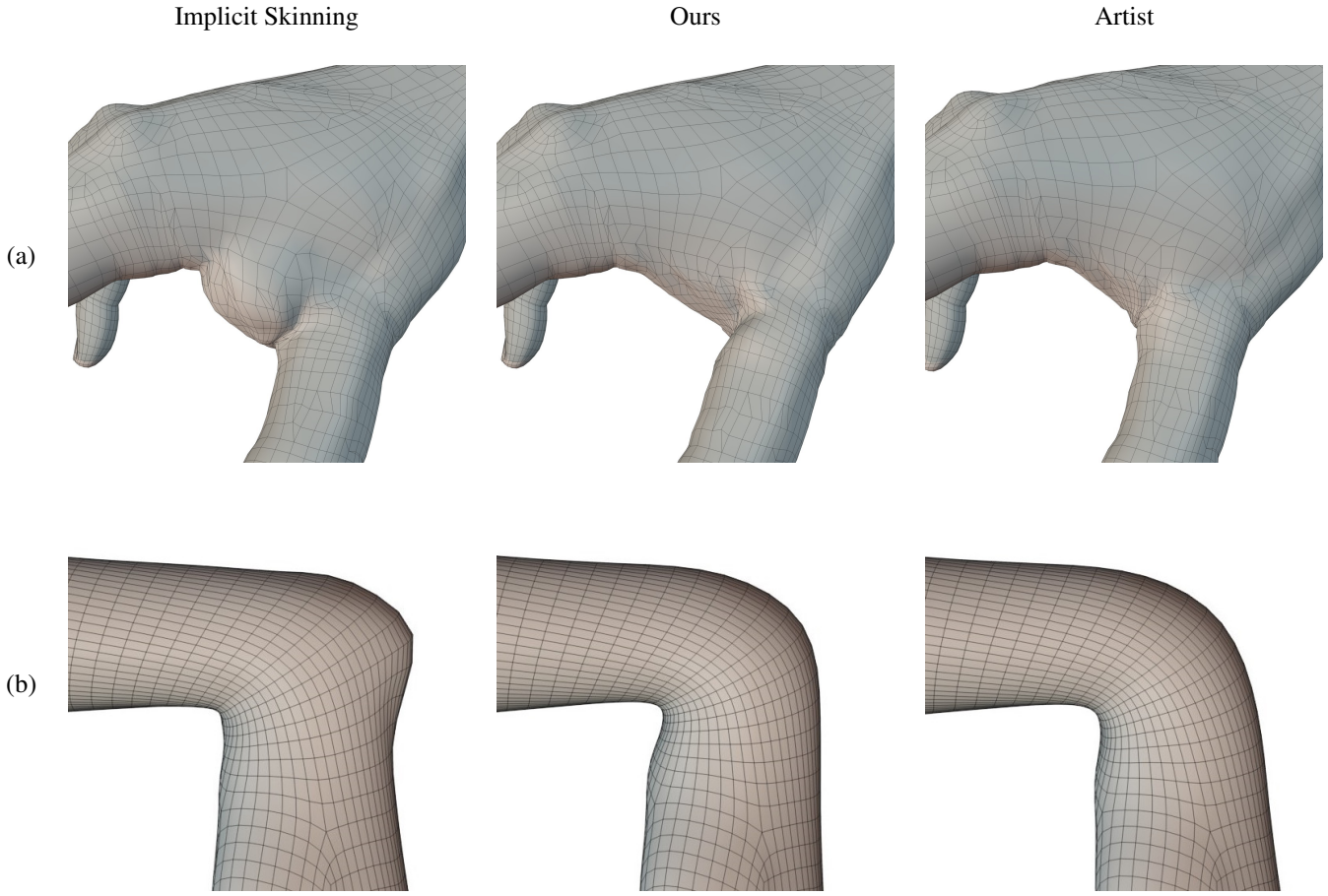


Fig. 8. Comparison of the results produced on different problematic joints of different models by (left) the implicit skinning with the default HRBFs [1], (middle) our optimized meshes and (right) HRBFs adjusted by an experienced user. (a) The thumb of a detailed hand, (b) Dana's wrist.

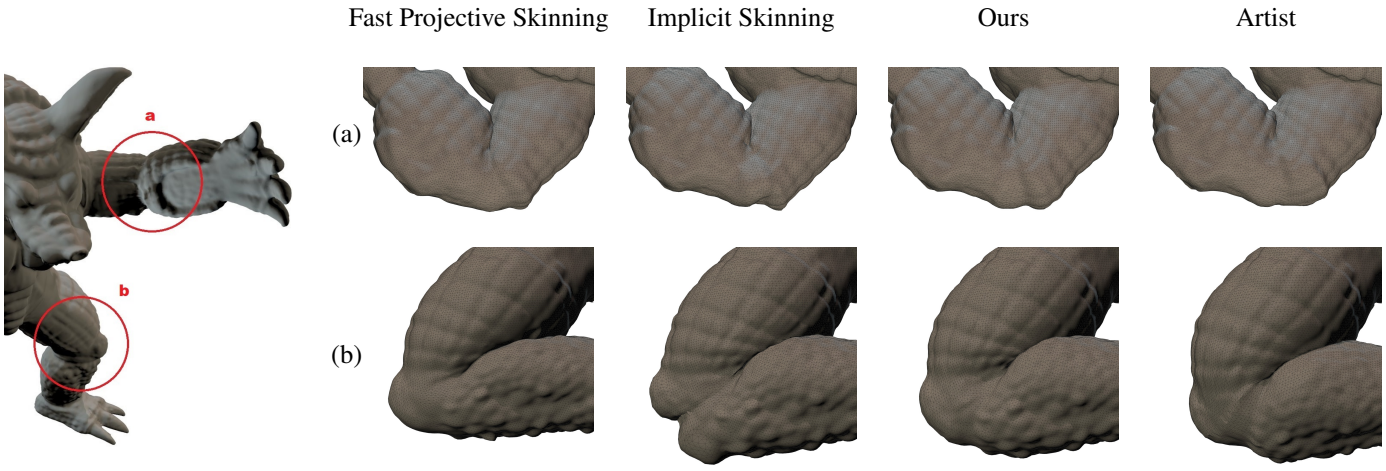


Fig. 9. Comparison of the deformations produced on the Armadillo (a) elbow and (b) knee. The deformations generated by the Fast Projective Skinning [23], our optimized Implicit Skinning and the artist are similar and plausible, while in (b) the one automatically produced by the original Implicit Skinning [1] is to be improved. In (a) the original Implicit Skinning provides an acceptable solution that is still slightly improved by our method.

Fig. 10. Comparison of the results produced, for different poses and joints of the male model, with the Linear Blend Skinning (LBS), Fast Projective Skinning [23], Implicit Skinning [1] and our method.

Talks. SIGGRAPH '14; New York, NY, USA: Association for Computing Machinery. ISBN 9781450329606; 2014,URL: https://doi.org/10.1145/2614106.2614144. doi:10.1145/2614106.2614144.

[19] Le, BH, Lewis, JP. Direct delta mush skinning and variants. ACM Trans Graph 2019;38(4).

[20] Lewis, JP, Cordner, M, Fong, N. 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. SIGGRAPH '00; USA: ACM Press/Addison-Wesley Publishing Co. ISBN 1581132085; 2000, p. 165–172. doi:10.1145/344779.344862.

[21] Rohmer, D, Hahmann, S, Cani, MP. Exact volume preserving skinning with shape control. In: Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. SCA '09; New York, NY, USA: Association for Computing Machinery. ISBN 9781605586106; 2009, p. 83–92. doi:10.1145/1599470.1599481.

[22] Komaritzan, M, Botsch, M. Projective Skinning. Proceedings of the ACM on Computer Graphics and Interactive Techniques 2018;1(1):1–19.

[23] Komaritzan, M, Botsch, M. Fast projective skinning. In: Motion, Interaction and Games. MIG '19; 2019,.

[24] Bouaziz, S, Martin, S, Liu, T, Kavan, L, Pauly, M. Projective dynamics: Fusing constraint projections for fast simulation. ACM Trans Graph 2014;33(4). doi:10.1145/2601097.2601116.

[25] Vaillant, R, Barthe, L, Guennebaud, G, Cani, MP, Rohmer, D, Wyvill, B, et al. Implicit skinning: Real-time skin deformation with contact modeling. ACM Trans Graph 2013;32(4).

[26] Angles, B, Tarini, M, Wyvill, B, Barthe, L, Tagliasacchi, A. Sketch-based implicit blending. ACM Trans Graph 2017;36(6).

[27] Angles, B, Rebain, D, Macklin, M, Wyvill, B, Barthe, L, Lewis, J, et al. Viper: Volume invariant position-based elastic rods. Proc ACM Comput Graph Interact Tech 2019;2(2). doi:10.1145/3340260.

[28] Roussellet, V, Abu Rumman, N, Canezin, F, Mellado, N, Kavan, L, Barthe, L. Dynamic implicit muscles for character skinning. Computers & Graphics 2018;77:227–239.

[29] Wendland, H. Scattered data approximation. 2004,.

[30] Macedo, I, Gois, JP, Velho, L. Hermite interpolation of implicit surfaces with radial basis functions. 2009 XXII Brazilian Symposium on Computer Graphics and Image Processing 2009;:1–8.

[31] Wyvill, B, Guy, A, Galin, E. Extending the csg tree - warping, blending and boolean operations in an implicit surface modeling system. Comput Graph Forum 1999;18(2):149–158.

[32] Hernández, E, Salamanca, S, Merchán, P, Adan, A. A comparison of hole-filling methods in 3d. International Journal of Applied Mathematics and Computer Science 2016;26.

[33] Guo, X, Xiao, J, Wang, Y. A survey on algorithms of hole filling in 3d surface reconstruction. Vis Comput 2018;34(1):93–103.

[34] George, PL, Seveno, E. The advancing-front mesh generation method revisited. International Journal for Numerical Methods in Engineering 1994;37(21).

[35] Yang, L, Yan, Q, Xiao, C. Shape-controllable geometry completion for point cloud models. Visual Computer 2017;.

[36] Liepa, P. Filling holes in meshes. In: Eurographics Symposium on Geometry Processing. The Eurographics Association; 2003,.

[37] Zhao, W, Gao, S, Lin, H. A robust hole-filling algorithm for triangular mesh. vol. 23. 2007, p. 22.

[38] Feng, C, Liang, J, Ren, M, Qiao, G, Lu, W, Liu, S. A fast hole-filling method for triangular mesh in additive repair. Applied Sciences 2020;10(3).

[39] Curless, B, Levoy, M. A volumetric method for building complex models from range images. In: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques. 1996, p. 303–312.

[40] Davis, J, Marschner, S, Garr, M, Levoy, M. Filling holes in complex surfaces using volumetric diffusion. In: Proceedings. First International Symposium on 3D Data Processing Visualization and Transmission. 2002, p. 428–441.

[41] Argudo, O, Brunet, P, Chica, A, Vinacua, À. Biharmonic fields and mesh completion. Graphical Models 2015;.

[42] Ju, T. Robust repair of polygonal models. ACM Trans Graph 2004;23(3):888–895.

[43] Carr, JC, Beatson, RK, Cherrie, JB, Mitchell, TJ, Fright, WR, McCallum, BC, et al. Reconstruction and representation of 3d objects with radial basis functions. In: Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques. SIGGRAPH '01; 2001, p. 67–76.

[44] Guennebaud, G, Gross, M. Algebraic point set surfaces. ACM Trans Graph 2007;26(3). doi:10.1145/1276377.1276406.

[45] Centin, M, Pezzotti, N, Signoroni, A. Poisson-driven seamless completion of triangular meshes. Computer Aided Geometric Design 2015;35-36:42–55.

[46] Centin, M, Signoroni, A. Advancing mesh completion for digital modeling and manufacturing. Computer Aided Geometric Design 2018;62.

[47] White, KB, Cline, D, Egbert, PK. Poisson disk point sets by hierarchical dart throwing. In: 2007 IEEE Symposium on Interactive Ray Tracing. 2007, p. 129–132. doi:10.1109/RT.2007.4342600.

[48] Witkin, AP, Heckbert, PS. Using Particles to Sample and Control Implicit Surfaces. Computer Graphics 1994;:9.

[49] Levet, F, Granier, X, Schlick, C. Fast sampling of implicit surfaces by particle systems. In: IEEE International Conference on Shape Modeling and Applications 2006 (SMI'06). 2006, p. 39–39.

[50] Amenta, N, Choi, S, Kolluri, RK. The power crust. In: Proceedings of the Sixth ACM Symposium on Solid Modeling and Applications. 2001, p. 249–266.

[51] Meyer, M, Desbrun, M, Schröder, P, Barr, AH. Discrete differential-geometry operators for triangulated 2-manifolds. Visualization and mathematics 2002;3(7):34–57.