RigMesh: Automatic Rigging for Part-Based Shape Modeling and Deformation

DUC NGUYEN, Telecom Paris

The creation of a 3D model is only the first stage of the character animation pipeline; before animation can occur, the model must be rigged with a skeleton and corresponding skin weights. While automatic rigging techniques exist, they are typically applied as a post-processing step and assume a finalized mesh. In iterative modeling workflows, however, geometry is frequently modified, making repeated rigging both inefficient and disruptive.

In this work, I reproduce a previously proposed sketch-based modeling system that integrates 3D shape creation with automatic rig generation. The primary technical challenge lies in enabling real-time 3D reconstruction from user sketches while maintaining structural consistency as parts are added, merged, or cut. Rather than introducing a fundamentally new rigging formulation, the system couples conventional skeletal representations with dynamically updated geometry, ensuring that rig structures remain synchronized with evolving shapes.

Additional Key Words and Phrases: rigging, skeletonization, skinning, animation, sketch-based modeling

1 INTRODUCTION

In standard character animation pipelines, a rig consists of a skeleton structure composed of joints and bones, along with skin weights that bind the surface mesh to the skeleton. Although computational methods exist for automatic skeleton extraction and skinning [Nealen et al. 2007], these approaches are generally applied as post-processing steps. As a result, modeling and rigging remain sequential and loosely coupled processes. If a designer modifies the geometry after rigging, the skeleton and skin weights often need to be recomputed, making iterative design cumbersome.

In this report, I review the work of [Borosan et al. 2012], who address this fundamental challenge in the 3D animation pipeline: the separation between modeling and rigging.

The key idea is to actively maintain the skin and skeleton in a single coherent framework. Instead of treating rigging as a separate stage that follows shape creation, their system maintains a fully rigged mesh throughout the modeling process. Each sketched component immediately produces a surface, a skeleton, and corresponding skin weights. When parts are merged or cut, the system updates both the geometry and the rig in real time. This design supports incremental modeling and enables users to pose and adjust shapes interactively during creation.

The authors adopt a modeling-by-parts paradigm, which aligns with human perception of objects as compositions of components. This approach also facilitates part reuse and local editing.

2 ALGORITHM WALKTHROUGH

The proposed framework maintains a fully rigged mesh representation throughout all modeling operations. Instead of separating modeling and rigging into sequential stages, the system consistently preserves three tightly coupled components: surface geometry, skeletal structure, and skin weights. The following subsections describe the main algorithmic components of the framework.

2.1 Skin Generation

The first stage generates the data required to initialize a SkinnedMesh structure. Given an input sketch, the system constructs the surface geometry through the following steps:

(1) **Contrained Delaunay Triangulation (CDT)** [Chew 1987]. The input contour is re-parametrized and triangulated using CDT. The internal edges of the triangulation are referred to as chords. This defines the initial chord configuration.
(2) **Laplacian Smoothing** including both chord centers and orientations, following the principles of the **Chordal Axis Transform (CAT)** [Prasad 1997].
(3) **Region Classification**. Based on chord adjacency, mesh regions are classified as:
   - **Pipe**: Regions composed exclusively of chords.
   - **Junction**: Triangles incident to 3 chords
   - **Cap**: Triangles incident to only 1 chord.
(4) **Surface Generation**. Unlike the original paper, which performs additional inner-pipe processing and junction merging, the surface of each defined region is generated directly.
(5) **Topological Laplacian Smoothing**1 is applied globally to improve geometric smoothness.
(6) **Isometric Remeshing** is performed to improve triangle quality. This produces near-equilateral triangles, which significantly improves the stability and robustness of subsequent local mesh operations.

After surface generation, the mesh is guaranteed to be manifold. This assumption simplifies subsequent processing and reduces the need for additional validity checks.

2.2 Skeleton Generation

Skeleton extraction reuses the **smoothed chord configuration** computed in the previous stage. Since chord centers closely approximate the medial structure of the shape, they provide a strong initialization for the skeletal graph. However, the raw chord structure contains more joints than necessary. Therefore, we apply the following refinement steps:

(1) **Cylindrical Douglas–Peucker Simplification (CDP)**. We apply the originally proposed algorithm to remove redundant joints along nearly collinear bone chains.
(2) Short bones that are not located on leaf branches are contracted.
(3) Short leaf branches are pruned using a breadth-first search (BFS).

This report is submitted as a part of project for Advanced 3D Computer Graphics (IMA904/IG3DA), Telecom Paris.

The resulting skeleton is always maintained as a single connected, cycle-free tree structure. This property ensures compatibility with standard skeletal animation frameworks.

2.3 Skin Weight Computation

Skin weights determine the influence of each joint on surface vertices. Following the Pinocchio framework [Baran and Popović 2007], joint influence is modeled via heat diffusion.

Given the generated skeleton, weights are computed as follows:

(1) Vertices near each bone are identified as constrained conditions.
(2) Solving the heat diffusion equation across the mesh surface.
(3) Normalizing per-vertex bone influences.

This diffusion-based formulation produces smooth weight transitions and stable deformations. It is robust to irregular triangulations and yields natural articulation behavior.

2.4 Mesh cutting procedure

Mesh cutting is performed using a user-defined stroke. The stroke is projected along the camera view direction to define a cutting plane. Both the mesh and its associated skeleton are divided accordingly.

The procedure is as follows:

(1) Boundary loops created by the cut are projected onto the cutting plane.
(2) The projected loops define a polygonal region.
(3) **GridMesh** [Nealen et al. 2009] is used to generate a temporary patch.
(4) Local smoothing is performed using [Sorkine and Cohen-Or 2004] to improve geometric continuity near the boundary.

For skin weight recomputation, weights are propagated inward from the boundary loop to the newly generated patch by solving the heat diffusion equation with hard constraints along the boundary 3. This localized recomputation preserves deformation consistency while maintaining computational efficiency.

2.5 Mesh merging procedure

After removing intersecting triangles between two meshes, boundary loops are obtained and stitched together.

The merging process consists of:

(1) Identifying pairs of boundary loops that are spatially close.
(2) Estimating a common plane for each loop pair.
(3) Projecting both loops onto the estimated plane and expanding one loop to enclose the other.
(4) Applying Constrained Delaunay Triangulation [Chew 1987] to generate connectivity between the two loops.
(5) Applying local Laplacian smoothing around the stitched region.

The prior isometric remeshing step ensures that triangles are close to equilateral, which improves the stability and quality of the newly generated patch.

For skin weight recomputation, a local region near the stitched boundary is selected. Only bones close to this region are included in a localized global skin solve, preserving efficiency while ensuring smooth weight transitions.

2.6 Skeleton refinement procedure

The framework allows user-driven skeletal editing. Users can:

- Collapse a bone,
- Split a bone,
- Drag joints interactively to adjust articulation.

After structural modification, a global skin recomputation is triggered to maintain consistent deformation behavior.

This interactive refinement step ensures that the skeleton remains semantically meaningful and visually aligned with the user's intended character structure.

3 MATHEMATICAL ANALYSIS

The 2 main equations which is reused intensively throughout the frame work is **Laplacian Smoothing Equation**1 and **Heat Diffusing Equation/Process**3. Both of them would turn out to be solving a linear equation $Ax = b$ in which A is sparse and positive definitve which can be solved using a sparse Cholesky solver.

But I also introduce constraints to the problem:

- Weak Constraint: It would be nice if $x_i = v$
- Hard Constraint: x_i must be v

3.1 Smoothing equation

$$x = \arg\min_{x, x_H = v_H} \|Lx\|_2^2 + \|A(x_W - v_W)\|_2^2 \tag{1}$$

with:

- L being the Laplacian matrix that encodes the smoothness prior over the solution (can be topology-based or geometric-based a.k.a Laplace-Beltrami operator (LBO)).
- H being the mask for hard constraint
- W being the mask for weak constraint
- A being a positive diagonal matrix representing weights associated with weak constraints

Since $x_H = v_H$ is a hard constraint, we can partition the vector x into known (hard-constrained) and unknown parts:

- x_H: components with hard constraints (known values)
- x_U: components without hard constraints (unknowns to solve for)

Similarly, we also partition L into 2 column matrices $[L_U, L_H]$. And now we are left with a simpler optimization problem:

$$x_U = \arg\min_{x_U} \|L_U x\|_2^2 + \|L_H x\|_2^2 + \|A(x_W - v_W)\|_2^2 \tag{2}$$

At this stage, the optimization problem resembles a standard linear least-squares formulation and can be solved efficiently due to the sparse structure of the involved matrices.

In this implementation, the formulation is mainly used for vertex position smoothing, and the Laplace–Beltrami operator is not explicitly employed. The second term is intentionally discarded because the unknown variable x_U represents a newly generated mesh patch, with the boundary vertices serving as hard positional constraints.

Including the second term would require extending an additional outer layer of vertices to properly define the constraints, introducing unnecessary implementation complexity. Moreover, since this term enforces smoothness across the boundary loop, removing it allows the framework to preserve sharp naturally.

3.2 Diffusing Equation

Pinocchio [Baran and Popović 2007] solves for equilibrium over the surface only, but at some vertices, it adds the heat transferred from the nearest bone. The equilibrium over the surface for bone i is given by:

$$\frac{\partial w^i}{\partial t} = \Delta w^i + H(p^i - w^i) = 0 \tag{3}$$

where Δ is the matrix represent Laplacian Beltrami Operator. $p^i_j = 1$ if the nearest bone to vertex j is i and $p^i_j = 0$ otherwise. And H is the diagonal matrix with H_{jj} being the heat contribution weight of the nearest bone to vertex j. The Pinocchio [Baran and Popović 2007] framework use $H_j j = \frac{c}{d(j)^2}$ where c is just a constant, and $d(j)$ is the distance from vertex j to closest bone not occluded by the mesh.

4 QUALITATIVE RESULT

4.1 Skinned Mesh Generation

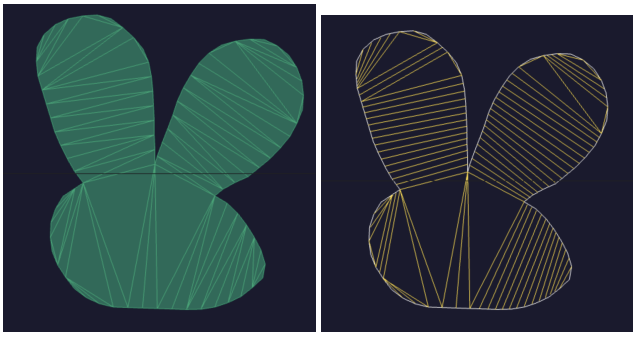

Fig. 1. Chordal Axis Transform

Figure 1 illustrates the Chordal Axis Transform (CAT) applied to the triangulated input shape. The left image shows the constrained Delaunay triangulation of the sketched contour, while the right image highlights the extracted chord configuration. These chords approximate the medial structure of the shape and serve as the foundation for subsequent skeleton generation. The smoothing of chord centers and orientations produces a stable and well-aligned medial representation.

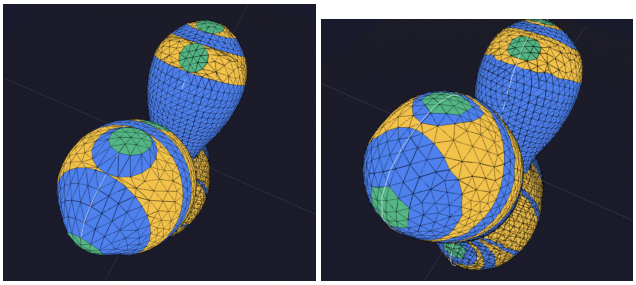

Fig. 2. Region Surface Generation

Figure 2 demonstrates the region-based surface generation process. Based on chord incidence, regions are classified into blue pipes, yellow junctions, and green caps. Each region is independently converted into surface geometry before being merged into a unified manifold mesh. The image on the left is applied Laplacian smoothing with a small smoothing factor.

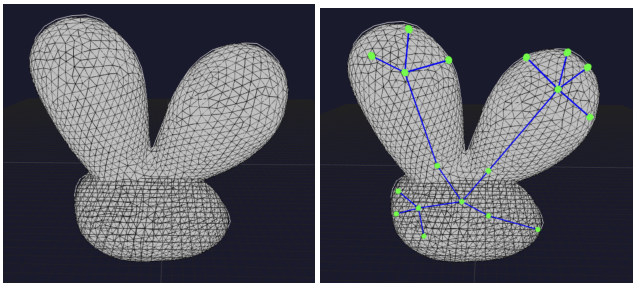

Fig. 3. Isometric Remeshing and Skeleton Generation

Figure 3 shows the mesh after isometric remeshing (left) and the corresponding generated skeleton (right). The isometric remeshing step improves triangle quality by making them close to equilateral, resulting in a more uniform vertex distribution across the surface.

The right image illustrates the skeleton extracted from the smoothed chord configuration after simplification and pruning. Redundant joints along nearly collinear segments are removed using CDP [Borosan et al. 2012]. Short bones are removed by internal contraction and branch pruning.

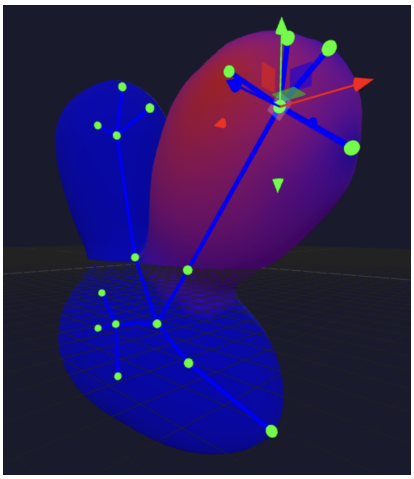

Fig. 4. Skin Weight Visualization

Figure 4 shows a visualization of the per-joint skin weights. The mesh is modeled as a heat-conductive volume, allowing heat to radiate outward from each bone toward the surrounding skin. Consequently, a bone has a strong influence on vertices within its Voronoi region, while still exerting some effect on vertices outside that region as the heat diffuses through the mesh.

4.2 Mesh Cut Process

Figure 5 illustrates the complete mesh-cutting process:

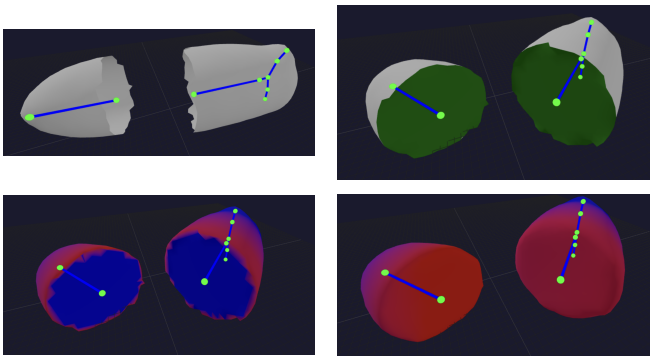

Fig. 5. Mesh Cut Process

- The bones and mesh are separated by the cut plane.
- Patches are generated to stitch the boundary loops.
- The skin around the loops is smoothed, although the patch's skin weights are not yet updated.
- Heat is diffused inward to fill the cold regions inside the patch.

4.3 Mesh Merge Process

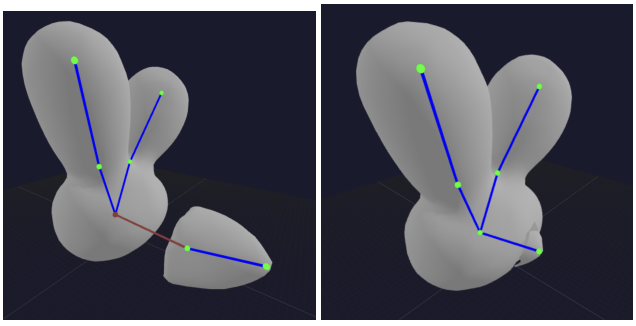

Fig. 6. Mesh Merge Snap preview

Figure 6 illustrates the snap operation during mesh merging, where the source mesh automatically aligns itself to join the snapped bones together.

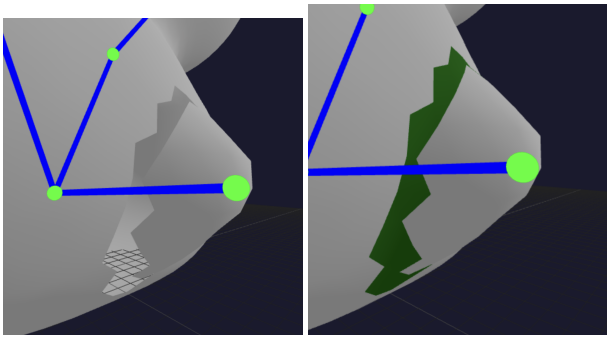

Fig. 7. Mesh Merge Patching

Figure 7 shows that triangles intersecting with the mesh are removed, and Constrained Delaunay Triangulation (CDT) is used to generate a slice that stitches the two loops together.

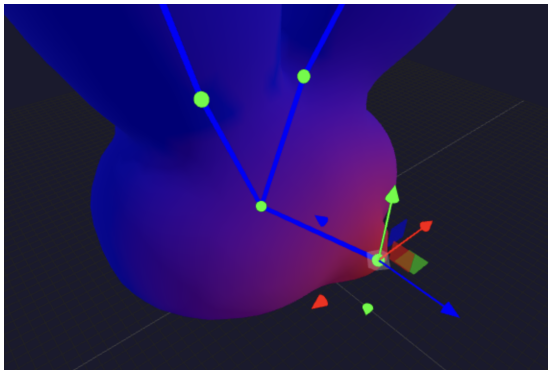

Fig. 8. Mesh Merge Finish

Figure 8 shows the final merged mesh with smoothing applied and skin weights recalculated.

4.4 Quantitative Result

The main metrics that I use to measure the quality of the mesh are the **IoU** (Intersection over Union) with the original stroke and **Laplacian** of the generated mesh at each step.

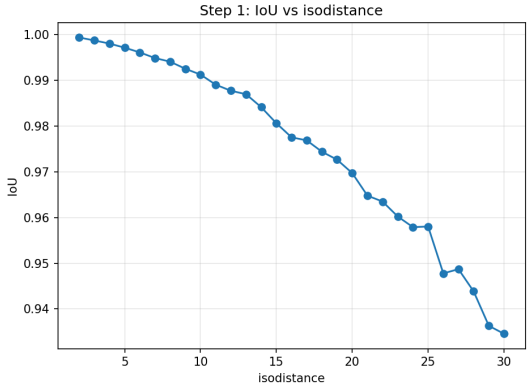

Fig. 9. **IoU** of re-parametrized polygon w.r.t. **isodistance**

After performing qualitative experiments, I have my favorite set of parameters for mesh generation:

Table 1. Surface Generation Config

Isodistance	10
Laplacian Iterations	10
Laplacian Rate	0.5
Smoothing Factor	5
Isometric Remeshing Iterations	3
Isometric Remeshing Length	10

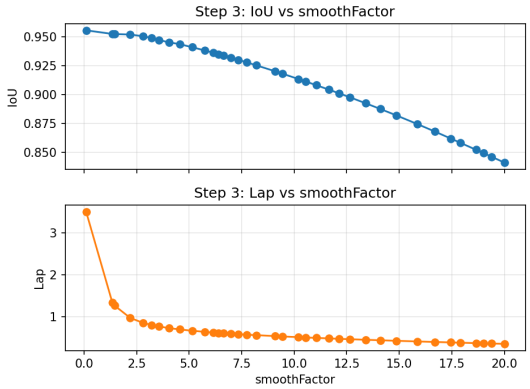

Fig. 10. **IoU** and **Lap** of 2D projection of generated mesh w.r.t. **smoothness**

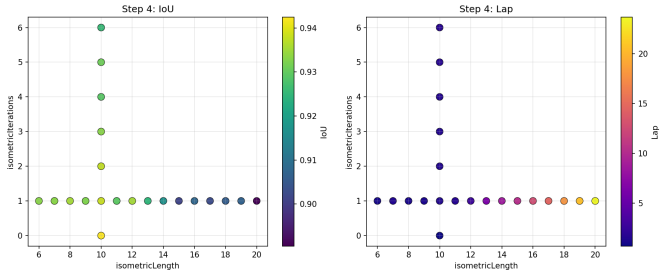

Fig. 11. **IoU** and **Lap** of 2d projection w.r.t. **No. Iterations** and **Iso-length**

It is important to note that the distance parameter depends on the scale of the user interface. The values shown in this plot correspond to the full resolution of the web canvas. Taking the scale into account allows us to dynamically adjust scale-dependent parameters in a consistent manner.

At present, I do not have a well-defined benchmark to quantitatively evaluate the "goodness" of a constructed skeleton; therefore, no quantitative results are reported. Nevertheless, we can qualitatively characterize a good skeleton using the following criteria:

- Every bone should exert a meaningful influence on the skin; no bone should have negligible or near-zero contribution.
- The influence threshold should be adaptive rather than fixed. It may depend on geometric factors such as the bone's distance from leaf joints, its length, or its structural role within the hierarchy.

5 CONCLUSIONS & FUTURE WORK

I presented an unified modeling and rigging framework that maintains a fully rigged mesh throughout the modeling process. Unlike traditional sequential pipelines that separate modeling and rigging, this approach integrates skeleton construction, skin weight computation, and deformation into a single interactive workflow. This design enables users to pose intermediate results at any stage, accelerating iteration and improving feedback during character creation.

The system supports part-based modeling and is particularly well suited for tubular organic shapes and character-like models. By dynamically updating skeletons and skin weights, the framework ensures coherent deformation behavior during mesh editing operations such as cutting, remeshing, and patch stitching.

In addition to qualitative evaluation, I provide quantitative results to measure the consistency and robustness of my incremental updates. These results demonstrate that this approach maintains deformation quality while enabling localized recomputation, making it suitable for interactive applications.

Several promising directions can further improve and extend the system:

- Exploring alternative rigging and deformation algorithms such as Dual-Quaternion Skinning (DQS), or Direct Delta Mush (DDM) [Le and Lewis 2019]...
- Customizing length kernel during isometric remeshing to balance stroke consistency and mesh quality.
- Incorporating of mesh subdivision and simplification to maintain appropriate resolution even when the mesh is scaled or undergoes significant geometric transformation.

REFERENCES

Ilya Baran and Jovan Popović. 2007. Automatic rigging and animation of 3D characters. In *ACM SIGGRAPH 2007 Papers (SIGGRAPH '07)*. Association for Computing Machinery, New York, NY, USA, 72–es. DOI: https://doi.org/10.1145/1275808.1276467

Peter Borosan, Ming Jin, Doug DeCarlo, Yotam Gingold, and Andrew Nealen. 2012. RigMesh: Automatic Rigging for Part-Based Shape Modeling and Deformation. *ACM Transactions on Graphics (TOG)* 31, 6, Article 198 (Nov. 2012), 9 pages. DOI: https://doi.org/10.1145/2366145.2366217

L. P. Chew. 1987. Constrained Delaunay triangulations. In *Proceedings of the Third Annual Symposium on Computational Geometry (SCG '87)*. Association for Computing Machinery, New York, NY, USA, 215–222. DOI: https://doi.org/10.1145/41958.41981

Binh Huy Le and J P Lewis. 2019. Direct delta mush skinning and variants. *ACM Trans. Graph.* 38, 4, Article 113 (July 2019), 13 pages. DOI: https://doi.org/10.1145/3306346.3322982

Andrew Nealen, Takeo Igarashi, Olga Sorkine, and Marc Alexa. 2007. FiberMesh: designing freeform surfaces with 3D curves. In *ACM SIGGRAPH 2007 Papers (SIGGRAPH '07)*. Association for Computing Machinery, New York, NY, USA, 41–es. DOI: https://doi.org/10.1145/1275808.1276429

Andrew Nealen, Justus Pett, Marc Alexa, and Takeo Igarashi. 2009. GridMesh: Fast and high quality 2D Mesh generation for interactive 3D shape modeling. In *2009 IEEE International Conference on Shape Modeling and Applications*. 155–162. DOI: https://doi.org/10.1109/SMI.2009.5170143

Lakshman Prasad. 1997. Morphological Analysis of Shapes. https://api.semanticscholar.org/CorpusID:16742845

O. Sorkine and D. Cohen-Or. 2004. Least-squares meshes. In *Proceedings Shape Modeling Applications, 2004.* 191–199. DOI: https://doi.org/10.1109/SMI.2004.1314506