OptCuts: Joint Optimization for Seam Placement and Parameterization of 3D Surfaces

ANONYMOUS AUTHOR(S)

Parameterizing 3D surfaces to the 2D plane is a fundamental problem with applications in texture mapping, remeshing, and detail transfer, etc. For most surfaces, one has to introduce discontinuities (seams) and distortion while producing a map. Existing techniques typically decide between the two independently: first placing seams and then minimizing distortion, and thus produce sub-optimal results. We propose a joint discrete-continuous optimization framework that optimally and progressively introduce or remove seams (in topology steps) in between distortion minimizations (in descent steps). We use a linear combination of symmetric Dirichlet energy and seam length as objective, of which the stationary w.r.t. both UV topology and coordinates are guaranteed to be reached within a bounded number of alternating iterations per balancing factor, input model, and initial embedding. **Minchen: [NOTE] (Here stationary w.r.t. UV topology is only in the approximation sense, because there might still be basic topological operations that could decrease the objective but end up not chosen because it's local evaluated energy decrease is not the largest one.)**

Specifically, in descent steps, we minimize symmetric Dirichlet energy using projected Newton method given the current UV topology. In topology steps, we search for a nearby UV topology that locally decrease the objective the most by querying a filtered set of basic topological operations. To be appropriately aggressive on searching in the topological space, **Minchen: [TODO] we develop an analogous line search method as in continuous settings and allow multiple fracture initiation.** Since in application scenarios, an upper bound for distortion or seam length is more intuitive than picking a balancing factor, we also provide a constrained optimization view of this broader problem that seeks stationary w.r.t. both primal (distortion and seams) and dual (balancing factor) variables subject to user specified distortion or seam length upper bounds.

Our method automatically produces high quality UV maps without any user assistance. **Minchen: [TODO] We also show that given a UV configuration by other methods, our method can improve the distortion and seam placement**, and that our framework has the potential to handle bijectivity, seamlessness, and user preferences jointly within it as well.

CCS Concepts: • **Computing methodologies → Mesh geometry models**;

Additional Key Words and Phrases: geometry processing, mesh parameterization, seam placement, numerical optimization, ...

ACM Reference Format:
Anonymous Author(s). 2018. OptCuts: Joint Optimization for Seam Placement and Parameterization of 3D Surfaces. *ACM Trans. Graph.* 1, 1 (March 2018), 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

0730-0301/2018/3-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

2 RELATED WORKS

related methods:
AutoCuts [Poranne et al. 2017]
Seamster [Sheffer and Hart 2002]
geometry images [Gu et al. 2002]
Multi-chart geometry images [Snyder et al. 2003]
D-Chart [Julius et al. 2005]
Boundary First Flattening [Sawhney and Crane 2017]
SeamCut [Lucquin et al. 2017]

components:
Bijective parameterization with free boundaries [Smith and Schaefer 2015]
projected Newton [Teran et al. 2005]
Minchen: [TODO] MIPS [Hormann and Greiner 2000]

3 AN ALTERNATING FRAMEWORK OF CONTINUOUS AND DISCRETE OPTIMIZATION FOR MESH PARAMETERIZATION

The most basic and intuitive mesh parameterization objective regarding both seams and distortion is minimizing distortion with as-sparse-as-possible seams introduced. However, seam sparsity usually leads to discontinuous energies w.r.t. UV coordinates $U \in \mathcal{R}^{2n_v}$, (n_v is the number of vertices on the input mesh), which is non-trivial to be considered into existing distortion minimization routines. Instead of progressively approximating seam sparsity energy with a continuous counterpart applying homotopy optimization method as [Poranne et al. 2017], we handle this discrete energy in a combinatoric way - searching in the topological space.

3.1 Formulation

This topological space is a directed graph G_T with its vertices $v_T \in V_T$ being all possible UV topologies of a given 3D surface, and its edges $e_T \in E_T$ are the basic topological operations conducted on a mesh such as vertex split, edge merge, etc, that can transform one UV topology to a nearby topology.

Now, if we consider both distortion and seam in one objective E_w, we can define the value f_v of vertex $v_{T,i}$ as

$$f_v(v_{T,i}) = \min_{U_i} E_w$$

and the weights f_w of edge $e_{T,m}$ from $v_{T,i}$ to $v_{T,j}$ could just be defined as

$$f_w(e_{T,m}) = f_v(v_{T,j}) - f_v(v_{T,i})$$

Thus our problem could be written as

$$\min_{U, v_T} E_w$$

which could be stated as to search for a $v_{T,i}$ on G_T where all edges connected to it satisfies $f_w \geq 0$.

However, computing f_v for one UV topology requires a whole continuous optimization process, and even the number of neighbors of one UV topology is in the scale of n_v. Consequently, we construct a single search path on G_T by progressively introducing or removing seams on the UV map, and we only estimate f_w on a local stencil of U for a filtered set of neighbors on G_T so that the whole process of continuous optimization is only conducted while necessary.

3.2 Method

Let's consider a simple situation, minimizing normalized symmetric Dirichlet energy [Smith and Schaefer 2015]

$$E_{SD} = \frac{1}{\sum_t |A_t|} \sum_t |A_t|(\sigma_{t,1}^2 + \sigma_{t,2}^2 + \sigma_{t,1}^{-2} + \sigma_{t,2}^{-2})$$

and normalized total seam length

$$E_{se} = \frac{1}{\sqrt{(\sum_t |A_t|)/\pi}} \sum_{i \in \mathcal{S}} |e_i|$$

where a balancing factor $\lambda \in [0, 1]$ is controlling the ratio between the two:

$$E_w = \lambda E_{se} + (1 - \lambda) E_{SD}$$

With the energy term normalization, our E_w is invariant of coordinate scale and resolution for meshes with the same shape. We minimize E_w by iteratively alternate between continuous optimization (in descent steps) and discrete optimization (in topology steps):

- In descent steps, we compute $\min_{U_i} E_{SD}$ given $v_{T,i}$ via projected Newton method [Teran et al. 2005] so that we obtain
$$f_v(v_{T,i}) = E_{se,i} + \min_{U_i} E_{SD}$$
- In topology steps, we estimate $f_v(v_{T,j})$ for a filtered set of neighbors of $v_{T,i}$ on a local stencil of U as $\hat{f}_v$ and move onto the neighbor $v_{T,i+1}$ with smallest $\hat{f}_v$.

If after a descent step, $f_v(v_{T,i}) \geq f_v(v_{T,i-1})$ is detected, we stop the process by rolling back to $v_{T,i-1}$, which is the stationary of E_w w.r.t. both UV topology (in an approximation sense) and coordinates that we are searching for.

3.3 Convergence

As our method is defined to guarantee convergence, we now analyze the convergence rate. First, it's easy to see that E_w is monotonically decreasing looking at each end of descent steps. Now we look at descent step i and $i + 1$, from $E_w^i \geq E_w^{i+1}$ we have

$$E_{SD}^i - E_{SD}^{i+1} \geq \frac{\lambda}{1-\lambda}(E_{se}^{i+1} - E_{se}^i) \geq \frac{\lambda}{1-\lambda} \frac{1}{\sqrt{(\sum_t |A_t|)/\pi}} |e|_{min}$$

if we now only consider splitting operations that keep increasing E_{se}. It's obvious that E_{SD}'s theoretical lower bound is defined to be 4, so we have

$$n_{alter} \leq \frac{(1-\lambda)\sqrt{(\sum_t |A_t|)/\pi}}{\lambda |e|_{min}} (E_{SD}^0 - 4)$$

The most important hint we can read from this is, to accelerate convergence, we can move through multiple vertices on G_T in each topology step to increase $E_{se}^{i+1} - E_{se}^i$. **Minchen: [TODO] Consequently, we build an anologous line search method and allow multiple fracture initiation to be appropriately agressive when searching in the topological space and ensure that we won't fall into bad locally optimal UV topologies.**

Minchen: [NOTE] Merge operations should be defined carefully to ensure convergence, and the proof will need updates. For example:

1. **How will we choose among merge and split? Is their energy decrease comparable?**
2. **Do we need merge to be performed on non-corner edge pairs like the sense of interior splits?**

3.4 Potential Extensions

It will be interesting to replace E_{SD} with other types of distortion energies, especially conformal energies like MIPS [Hormann and Greiner 2000] to see how seams that benefits conformality will be different from seams that benefits isometry. Besides, bijectivity could potentially be achieved by augmenting distortion energy with a penalty-based collision handling energy, possibly also assisted with air mesh method [?]. Similarly, seamless properties could also potentially be achieved by augmenting distortion energy with the correspondingly developed new differentiable objectives, and our alternating framework stays the same.

If an objective derived from an application is discontinuous and it could be expressed using mesh topology, then we can simply augment it into E_{se} and tackle it in the topology steps. For example, the smoothness of seams, user preferences on regional seam placement, and properties related to charts should all be able to be considered in this way.

Besides, SIMD type of parallelism could easily accelerate the $\hat{f}_v$ evaluations in the topology steps, and it also has the potential to improve the quality of the topology search by directly evaluating f_v for neighbors and track multiple branches.

4 DESCENT STEPS FOR CONTINUOUS OPTIMIZATION

4.1 Newton-type Iterations

By applying projected Newton method [Teran et al. 2005], our linear system in each iteration is symmetric and semi-definite, so we use symmetric indefinite solver on it.

Minchen: [TODO] Fix a direction for the linear system to ensure definiteness.

4.2 Potential Accelerations for Practical Use

Since our topological operations only change the mesh locally both on connectivity and coordinates, we could also update the Hessian or the decomposition locally after topology changes to save time. Besides, it's also interesting to try other Hessian approximation methods like L-BFGS [Liu and Nocedal 1989] or composite majorization [Shtengel et al. 2017] to explore further acceleration by finding a balance between computational cost and convergence rate.

For convergence tolerance of descent steps, $||\nabla E_{SD}||^2 \leq 10^{-6}$ (note that our energy is normalized) works generally well for all input models judging from the initiated fracture in the following topology step. In fact more inexact solve performs well on most of

ALGORITHM 1: Descent Steps

Data: Input model, UV coordinates U, UV topology v_T
Result: $\text{argmin}_U\ E_{SD}$ given v_T
for *each descent step inner iteration* j **do**
  compute E_{SD} gradient g^j;
  if $||g^j||^2 < 10^{-8}$ *or* $|E_{SD}^j - E_{SD}^{j-1}|/E_{SD}^{j-1} < 10^{-6}\alpha^{j-1}$ **then**
    break;
  end
  compute E_{SD} Hessian proxy P^j using projected Newton [Teran et al. 2005];
  solve for search direction p^j ($P^j p^j = -g^j$) using PARDISO [Petra et al. 2014a,b] symmetric indefinite solver;
  compute initial step size α_0^j by avoiding element inversion [Smith and Schaefer 2015];
  backtracking line search with Armijo rule [Armijo 1966] to obtain α^j;
  update $U^{j+1} = U^j + \alpha^j p^j$, $E_{SD}^{j+1} = E_{SD}(U^{j+1}, v_T)$;
end

the models with even $||\nabla E_{SD}||^2 \leq 10^{-4}$, but some may result in bad cuts, and there are also some models will result in even better cuts with $||\nabla E_{SD}||^2 \leq 10^{-8}$. Since we are conducting non-convex optimization, $||\nabla E_{SD}||^2$ is not always decreasing, which is also why we don't use Wolfe conditions for line search. The argument here for tolerance issue is that, it depends on whether we are truly in the infinitesimal region of a stationary. Some configuration with $||\nabla E_{SD}||^2 \leq 10^{-6}$ may still not inside the infinitesimal region of a stationary, where if optimization goes on, $||\nabla E_{SD}||^2$ will go up and then fall down again to the real stationary, which is understandable in non-convex optimization. Consequently, we apply an extra stopping criteria for descent steps, which depends on the relative energy decrease to resolve this issue. Combined with an appropriately small tolerance on $||\nabla E_{SD}||^2$, say 10^{-8}, this extra stopping criteria ends descent steps early appropriately only while necessary that cut initiation is not affected but projected Newton iterations and thus time needed is much less, especially for highly curved surfaces. This won't lead to bad results like simply setting larger tolerance on $||\nabla E_{SD}||^2$ since it ensures that the local optimal infinitesimal region is reached. Note that the problem of just using small tolerance on $||\nabla E_{SD}||^2$ is that it would be very unnecessary for those highly curved surfaces, where stationary needs many projected Newton iterations to reach and it won't affect cut initation.

Minchen: [NOTE] Another drawback of setting higher tolerance on $||\nabla E_{SD}||^2$ is that some initiated cuts may not increase ∇E_{SD} enough to restart descent step - it converges right after it started, which result in unwise following cut initiations. [TODO] This inspires us to try picking the topological operation that will increase the gradient the most to perform.

ALGORITHM 2: Candidate Filtering

Data: Input model, UV coordinates U, UV topology v_T
Result: A filtered set of UV vertices
if *boundary split* **then**
  compute divergence of local gradients for all $n_{v,b}^i$ boundary vertices;
  pick $(n_{v,b}^i)^{0.8}$ vertices with largest divergence as candidates;
else
  compute divergence of local gradients for all $n_{v,i}^i$ interior vertices that doesn't connect to boundary;
  pick $(n_{v,i}^i)^{0.8}$ vertices with largest divergence as candidates;
end

ALGORITHM 3: Local Evaluation

Data: Input model, UV coordinates U, UV topology v_T, candidate UV vertices
Result: new UV topology v_T and UV coordinates U
for *each candidate UV vertex* **do**
  if *on boundary* **then**
    for *each interior incident edge* **do**
      split and compute $\Delta E_{SD,l}$ locally;
      compute $\Delta E_{w,l} = (1-\lambda_t)\Delta E_{SD,l} + \lambda_t \Delta E_{se}$;
    end
  else
    for *each pair of incident edges* **do**
      split and compute $\Delta E_{SD,l}$ locally;
      compute $\Delta E_{w,l} = 0.5((1-\lambda_t)\Delta E_{SD,l} + \lambda_t \Delta E_{se})$;
    end
  end
end
if *!interiorSplit* **then**
  for *each fracture tail* **do**
    merge the 2 incident boundary edges with averaged position;
    if *element inversion is detected* **then**
      project the averaged position to feasible region;
      if *feasible region is empty* **then**
        continue;
      end
    end
    compute $\Delta E_{SD,l}$ locally;
    compute $\Delta E_{w,l} = (1-\lambda_t)\Delta E_{SD,l} + \lambda_t \Delta E_{se}$
  end
end
conduct the operation with largest $|\Delta E_{w,l}|$

5 TOPOLOGY STEPS FOR DISCRETE OPTIMIZATION

5.1 Evaluating Topological Operations via Optimization on Local Stencils

For boundary vertex that connects to another boundary, we free both the 2 boundary vertices while evaluating the local energy decrease.

Minchen: [NOTE] Do we need to treat compressed region and stretched region differently in order to have less overlap?

Minchen: [TODO] Since we are using the local estimation to approximate true global energy decrease, it would be necessary to find a balance between accuracy and efficiency by weighing between size of local stencils and number of optimization iterations to run.

Minchen: [NOTE] Seamster's selected high curvature vertices are set to be the leaf of the seam tree while necessary, while our interior splitting scheme seems to always allow fracture to be extended on both the 2 sides of a picked vertex, which seems suboptimal. However, instead of just trying to connect interior vertex to the boundary like geometry images, we still have the reason to preserve our current interior splits because we've seen results that holes in UV map will be needed as a better solution for resolving some interior distortion. What we need to do is to take Seamster's inspiration and improve our interior splitting scheme.

5.2 Line Search in Topological Space

Once a fracture is initiated, it almost always be extended further in the later topology steps, which justifies the robustness of our method. To speed up the process, instead of waiting for another descent step and query all the filtered candidates again, we propagate this newly initiated fracture further in between the first several inner iterations of the following descent step.

After the fracture has been initiated, we first go to descent step to run an inner iteration and record the energy decrease ΔE_w^j and energy $E_w^{j,0}$. Then we evaluate $\hat{f}_v$'s for splitting the tail vertex of the newly initiated fracture along its incident edges. If the one with largest $\hat{f}_v$ satisfies $\hat{f}_v - E_w^{j,0} \leq \Delta E_w^j$, we propagate the fracture along this edge, and run another inner iteration to do another propagation query. If there's no propagation that could benefit more than running the inner iteration, we stop querying propagation in the current descent step and run inner iterations till convergence:

ALGORITHM 4: Fracture Propagation Line Search

Data: Input model, UV coordinates U, UV topology v_T
Result: new UV topology v_T and UV coordinates U
for *each interior incident edge of current fracture tail vertex* k **do**
  split and compute $\Delta E_{SD,I}$ locally;
  compute $\Delta E_{w,I} = (1 - \lambda_t)\Delta E_{SD,I} + \lambda_t \Delta E_{se}$;
end
if $\max(|\Delta E_{w,I}|) \geq |(1 - \lambda_t)\Delta E_{SD}^j|$ **then**
  propagate fracture by splitting the vertex;
  update fracture tail record;
else
  turn off fracture propagation for the rest of the current descent step;
end

Minchen: [TODO] Write merge propagation

Minchen: [TODO] Improve current propagation scheme by developing analogous line search method

Minchen: [NOTE] Besides fracture propagation line search, increase the depth of querying fracture initiation did the same thing but with more computational cost. However, it could improve the results because it's more global! (Inspired by Seamster)

5.3 Multiple Fracture Initiation

Minchen: [TODO]

Multiple fracture initiation would be redundant on a single cone-like region since only one fracture would be enough to release the distortion. However, initiating one fracture for each cone-like region within a single topology step would certainly accelerate the whole process.

This could be done by partitioning the UV space according to distortion or filtering measurement and initiate a valid fracture (if any) in each region in every topology step. The partition criteria could be separating the domain so that on each subdomain, the function is convex (has exactly one stationary).

6 WEIGHTING THE OBJECTIVE AUTOMATICALLY BY INTRODUCING A DUAL PROBLEM

At each inner iterate $k + 1$, we fix some λ^{k+1} and minimize the bi-objective

$$\min_{T,V} E_{SE}(V,T) + \lambda^{k+1} E_{SD}(V,T)$$

How do we get λ^{k+1}?

Our overall minimization is inequality constrained with a specified upper bound $b \in \mathbb{R}_+$ on distortion. (L2 norm on SD energy for now - pretty easy to modify to an extremal measure if we want later on.)

Our model problem minimization is then

$$\min_{T,V} E_{SE}(V,T) : b - E_{SD}(V,T) \geq 0$$

Or, equivalently,

$$\min_{T,V} \max_{\lambda \geq 0} E_{SE}(V,T) + \lambda\big(E_{SD}(V,T) - b\big)$$

Of course this is nonsmooth in λ since it does not take into account very nicely the fact that per-iteration we will start away from feasibility and want to iteratively improve both our primal variables $\{V,T\}$ and our dual variable λ. So to smoothly update to a current λ^{k+1} from a previous estimate λ^k we will add a regularizer $R(\lambda, \lambda^k)$ to make sure λ iterates behave themselves reasonably. For now lets stick with something simple: a quadratic regularizer should do the trick $R = \frac{1}{2\kappa}(\lambda - \lambda^k)^2$.

For iteration $k + 1$ this gives us

$$\min_{T,V} \max_{\lambda \geq 0} E_{SE}(V,T) + \lambda\big(E_{SD}(V,T) - b\big) - \frac{1}{2\kappa}(\lambda - \lambda^k)^2$$

And now we can first solve closed form for λ as

$$\lambda^{k+1} \leftarrow argmax_{\lambda \geq 0} E_{SE}(V,T) + \lambda\big(E_{SD}(V,T) - b\big) - \frac{1}{2\kappa}(\lambda - \lambda^k)^2$$

giving us

$$\lambda^{k+1} \leftarrow \max\big(0, \kappa\big(E_{SD}(V,T) - b\big) + \lambda^k\big)$$

We then can solve the inner iteration (with both discrete topology steps and smooth steps) with the energy

$$\min_{T,V} E_{SE}(V,T) + \lambda^{k+1} E_{SD}(V,T)$$

Followed by the next update of dual variable λ.

(Notice that throughout the above we can define a progressive λ without needing to employ subgradients to reason about nonsmoothness in our sparsity energy.)

R is to iteratively solving for λ so that it could have intermediate values between 0 and ∞. Then starting from full mesh, λ will first increase as bound is not reached, and then it will decrease when bound is reached. But currently we don't have merge to increase E_{SE} and decrease E_{SD}, so our process will stop right after it reaches the bound. The bound is obvious to be reached, how do we know the path of λ is great? If we have merge, then will the optimization converge on all T, V, λ?

7 RESULTS AND DISCUSSION

click to see an index of all experiments done and all related documents

Minchen: [TODO] and [NOTE]:

- quality and timing comparison with previous methods, deal with closed surfaces by either starting from random rectangle embedding, basic heuristics like farthest points, or some learned prior (do figure out a best way for our algorithm to treat closed surfaces, potentially also higher genus surfaces?)
- show improvements starting from results given by previous methods
- how does triangulation affect our result? try same shape with different triangulation.
- given a symmetric shape, whether symmetrically triangulated or not, does our method preserve symmetry in UV space?
- there are seams being placed close to highly curved paths but not exactly aligning it, is it because our consideration is purely local or the placement is just better than human prefered results?
- do we encounter cases like there are several edges with almost equal estimated energy decrease and we pick from them by random factor?

8 CONCLUSIONS AND FUTURE WORKS

- Our method doesn't provide globally optimal solutions, the results are still locally optimal, but w.r.t. both seams and distortion, which is better than previous 2-pass methods that breaks the correlation between seams and distortion.
- take advantage of basic SIMD type of parallelism for accelerating query and improving results' quality by directly evaluating f_v for neighbors and track multiple branches, very useful for practical implementations
- if the user won't mind getting a slightly different triangulation, we could also create fractures in the interior of an element and locally remesh the stencil
- start and solve in 3D by reducing curvature so that the need for locally injective initial embedding in parameterization problems could be eliminated, and the result is only "biased" by it's 3D shape, which is the most reasonable bias
- try conformal energy like MIPS
- **Minchen: [TODO?] bijectivity**, seamless, and other augmentation of continuous energy?
- handle user preferences on seam placement
- seam smoothness, patch related discrete energy augmentation?

9 ACKNOWLEDGEMENTS

REFERENCES

Larry Armijo. 1966. Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific Journal of mathematics* 16, 1 (1966), 1–3.

Xianfeng Gu, Steven J Gortler, and Hugues Hoppe. 2002. Geometry images. *ACM Transactions on Graphics (TOG)* 21, 3 (2002).

Kai Hormann and Günther Greiner. 2000. *MIPS: An efficient global parametrization method*. Technical Report. ERLANGEN-NUERNBERG UNIV (GERMANY) COMPUTER GRAPHICS GROUP.

Dan Julius, Vladislav Kraevoy, and Alla Sheffer. 2005. D-Charts: Quasi-Developable Mesh Segmentation. In *Computer Graphics Forum*, Vol. 24.

Dong C Liu and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical programming* 45, 1-3 (1989), 503–528.

Victor Lucquin, Sebastien Deguy, and Tamy Boubekeur. 2017. SeamCut: Interactive Mesh Segmentation for Parameterization. In *ACM SIGGRAPH 2017 Technical Briefs*.

Cosmin G. Petra, Olaf Schenk, and Mihai Anitescu. 2014a. Real-time stochastic optimization of complex energy systems on high-performance computers. *IEEE Computing in Science & Engineering* 16, 5 (2014), 32–42.

Cosmin G. Petra, Olaf Schenk, Miles Lubin, and Klaus Gärtner. 2014b. An augmented incomplete factorization approach for computing the Schur complement in stochastic optimization. *SIAM Journal on Scientific Computing* 36, 2 (2014), C139–C162.

Roi Poranne, Marco Tarini, Sandro Huber, Daniele Panozzo, and Olga Sorkine-Hornung. 2017. Autocuts: Simultaneous Distortion and Cut Optimization for UV Mapping. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH ASIA)* 36, 6 (2017).

Rohan Sawhney and Keenan Crane. 2017. Boundary First Flattening. *ACM Trans. Graph.* 37, 1, Article 5 (Dec. 2017).

Alla Sheffer and John C Hart. 2002. Seamster: inconspicuous low-distortion texture seam layout. In *Proceedings of the conference on Visualization'02*.

Anna Shtengel, Roi Poranne, Olga Sorkine-Hornung, Shahar Z Kovalsky, and Yaron Lipman. 2017. Geometric optimization via composite majorization. *ACM Trans. Graph* 36, 4 (2017).

Jason Smith and Scott Schaefer. 2015. Bijective parameterization with free boundaries. *ACM Transactions on Graphics (TOG)* 34, 4 (2015).

John Snyder, Pedro V Sander, Zoe J Wood, Steven Gortler, and Hugues Hoppe. 2003. Multi-chart geometry images. (2003).

Joseph Teran, Eftychios Sifakis, Geoffrey Irving, and Ronald Fedkiw. 2005. Robust quasistatic finite elements and flesh simulation. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*.