Quadrangular Parameterization for Reverse Engineering

David Bommes, Tobias Vossemer, and Leif Kobbelt

RWTH Aachen University

Abstract. The aim of Reverse Engineering is to convert an unstructured representation of a geometric object, emerging e.g. from laser scanners, into a natural, structured representation in the spirit of CAD models, which is suitable for numerical computations. Therefore we present a user-controlled, as isometric as possible parameterization technique which is able to prescribe geometric features of the input and produces high-quality quadmeshes with low distortion. Starting with a coarse, user-prescribed layout this is achieved by using affine functions for the transition between non-orthogonal quadrangular charts of a global parameterization. The shape of each chart is optimized non-linearly for isometry of the underlying parameterization to produce meshes with low edge-length distortion. To provide full control over the meshing alignment the user can additionally tag an arbitrary subset of the layout edges which are guaranteed to be represented by enforcing them to lie on iso-lines of the parameterization but still allowing the global parameterization to relax in the direction of the iso-lines.
Key words: reverse engineering, quadrangular remeshing, global parameterization

1 Introduction

The quality of a quadrangulation depends on several, often conflicting, properties. The most important ones are *regularity, alignment, orientation* and *sizing. Regularity* means that the number of irregular vertices (valence $\neq 4$) should be minimized, while *orientation* implies that the elements should be locally oriented to the principal curvature directions to best capture the geometry. For geometries with sharp features *alignment* plays an important role, i.e. feature lines should be explicitly represented through edges in the quadrangulation, which is an even stronger requirement than *orientation.* Finally *sizing* means that the edge-length should be nearly constant which can be interpreted as providing a close to isometric inherent parameterization in regular regions of the mesh. Quadrangulations optimizing these properties are perfectly suited for many computer graphics applications like Modeling, e.g. as structure aligned control meshes for Catmull-Clark subdivision, multi-level hierarchies or shape matching to name a few.

The aim of Reverse Engineering is to convert a given unstructured triangle mesh into such a structured quadrangulation (see Figure 1). Full automatic

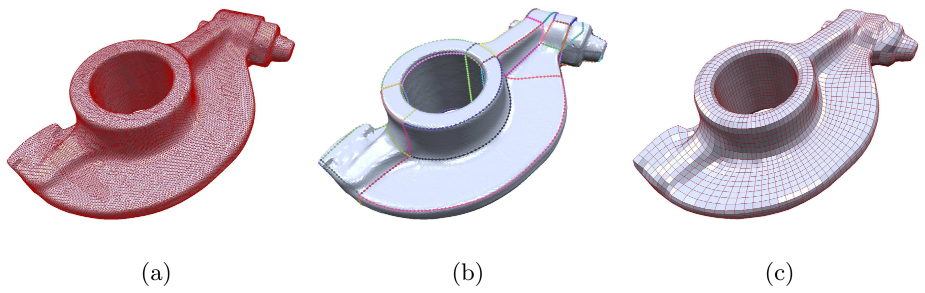


Fig. 1. Reverse Engineering Pipeline: (a) The input is a dense, unstructured triangle mesh. (b) The user provides a coarse layout controlling the quadrangulation. Singularities can only occur at nodes of this layout. (c) A distortion minimizing parameterization is computed to extract a pure quadmesh.

generation has proven to be a very hard global task and often design decisions depend on the intended usage and cannot forecast by only looking at the geometry. Therefore in Reverse Engineering full user-control is desired where the user can easily provide the topology, i.e. the number and position of singularities, and some alignment constraints for the resulting mesh. This can be achieved in a simple way by using coarse layouts which partition the surface in quadrilateral patches as illustrated in Figure 1 (b). From this layout a globally smooth parameterization can be computed by assigning a two-dimensional chart to each patch and connecting the parameterizations of neighboring charts with so called transition functions (see Figure 1 (c)).
The resulting mesh quality strongly depends on the metric distortion of the parameterization and on the alignment to sharp features. Consequently we present a new method to handle both tasks in a robust way, enabling the usage of global parameterization techniques for Reverse Engineering. Our main contribution is a chart optimization technique which minimizes the distortion of the resulting global parameterization. In contrast to other methods each chart is allowed to be an arbitrary 5 degree-of-freedom (DOF) quadrilateral with interior angles possibly differing from 90 degrees. As a result we need to specify generalized transition functions between these charts. Other important ingredients of our practical Reverse Engineering method are alignment constraints and T-Vertices, enabling simplified layout design. Figure 2 illustrates the gain of quality due to our chart optimization where chart corners are chosen to form a unit square (a), an optimized rectangle (b) and an optimized general quadrilateral (c).

2 Related Work

Remeshing of surfaces was investigated intensively in the last years and there are many good surveys on this topic such as [1] and [2]. Therefore we will only discuss the most relating works.

Direct quadrangulation methods like [3, 4] trace lines of principal curvatures to generate a quaddominant mesh. These methods have no explicit control over regularity and placement of singularities which is necessary for Reverse Engineering.

Global parameterization methods as introduced by Khodakovsky et al. [5] take a different way. In most cases a multi-chart parameterization is computed where continuity conditions between neighboring charts ensure that all mappings of the integer grid $\mathbb{Z} \times \mathbb{Z}$ stitch together compatibly and form a mesh consisting of quadrilaterals only. Some methods use a guiding vector field, often derived from the principal curvatures of a surface, for local element orientation [6–8]. However, the creation of suitable guiding fields is alone a very hard problem. Recently Ray et al. proposed a method to design smooth N-symmetric fields with prescribed singularities [9]. But the automatic placement of singularities is still unsolved.

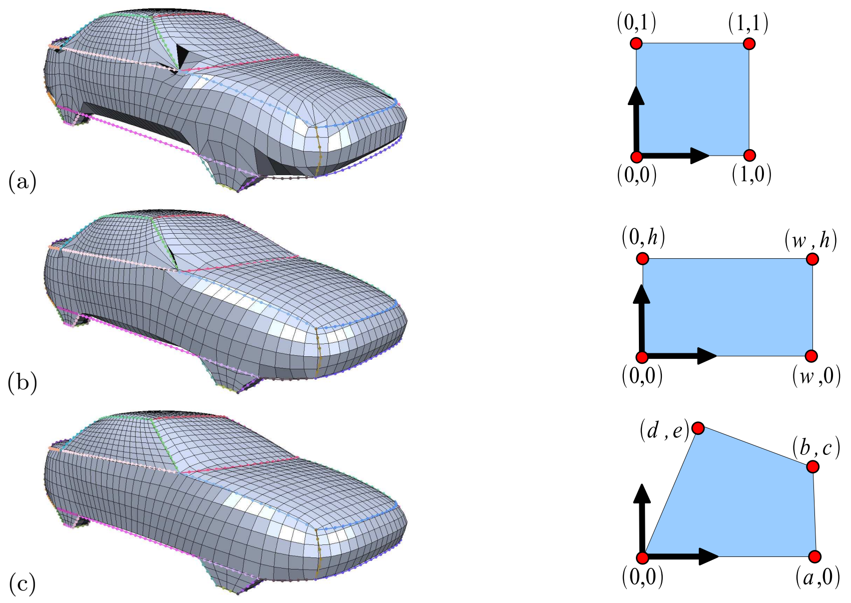


Fig. 2. (a) A global parameterizations using unit charts leads to large distortions and foldovers for a simple car model. (b) Even with optimized rectangular domains the distortions get large where the patches have a trapezoidal shape. (c) Our generalized, quadrangular parameterization leads to low distortion. Notice that in this example no alignment constraints were applied.

Instead of a vector field a coarse layout can be used to define the partitioning of the surface into patches for a multi-chart parameterization and define the rough element orientation. For user-controlled Reverse Engineering this way is very appealing and intuitive. Dong et al. used the Morse-Smale complex of Laplacian eigenfunctions to automatically compute a layout [10]. This method needs only a few singularities but they are placed at almost random positions and there is no element orientation w.r.t. geometric relevant structures. Huang et al. added orientation and alignment control to this method by incorporating a guiding vector field [8].

A user designed coarse layout, here called singularity graph, was also used in [11] to compute globally smooth harmonic parameterizations. These layout-based techniques are closely related to our method. Therefore we will discuss them in more detail.

The method of Dong et al. [10] uses simple unit squares as charts for a globally smooth parameterization. This is justified because in their layout, neighboring surface patches, originating from the Morse-Smale complex of the Laplacian eigenfunction, have similar size. Furthermore the layout vertices, representing the mesh singularities, are relaxed on the surface to prevent foldovers and large distortions. In Reverse Engineering such a relaxation technique is not reasonable since it interferes with the desired user-control. Figure 2 (a) shows the result of a globally smooth parameterization onto unit square shaped charts with a fixed user provided layout. As one can see there are large distortions and foldovers reflecting the fact that neighboring surface patches are far from being equally sized. So obviously unit square charts are not sufficient for our setting.

If one would restrict the layout to quadrilaterals and choose all free coefficients to one the globally smooth parameterization technique of Tong et al. [11] is exactly the same as the one discussed in the last paragraph. Notice that this equality is non-trivial since both papers use a different formalism to derive the final global linear system. Besides, the parameterization of Tong et al. is more general because it allows a larger class of charts. Each chart is a polygon where the vertices lie on integer positions and all edges are aligned to the coordinate axes, accordingly all interior angles of a chart are multiples of 90 degree. In our car example this means moving from unit squares to rectangular charts with two DOF's, namely the two independent edge lengths. In the original method this new DOF's are chosen manually or by using a heuristic which simply rounds the length of the corresponding layout edges to integer. Figure 2 (b) shows the result of the car example using rectangular charts. Here we already used our chart optimization technique presented in Section 3.1, instead of their heuristic, to minimize the length distortion. But still we observe large distortions, for example near the corner of the front window.

The problem is that the surface patches are far from being rectangular. Consequently we consider an even more general class of charts, i.e. we allow charts to be arbitrary quadrilaterals with five DOF's. We exploit these DOF's to minimize the distortion of the parameterization and the result can be seen in 2 (c). Even without alignment constraints the quadmesh edges follow the user prescribed layout and the length distortions are much lower. This introductory example motivates our design choices for a practical Reverse Engineering method. In the subsequent paragraphs our method will be explained in more detail.

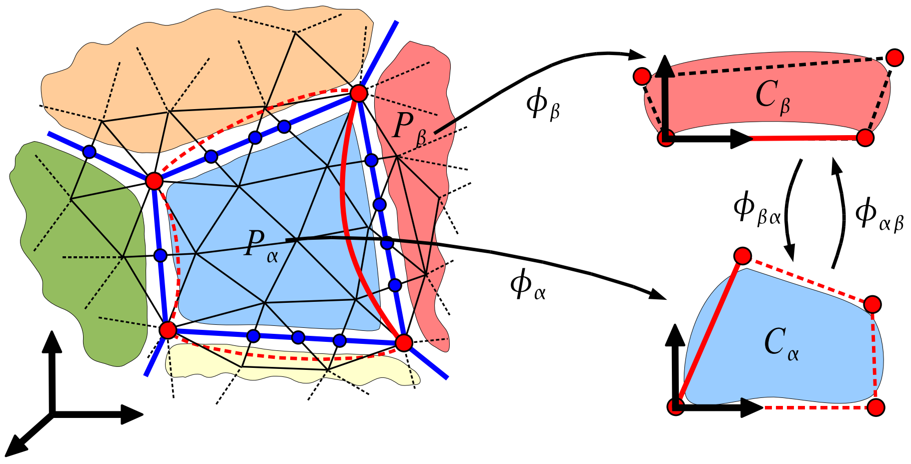


Fig. 3. The left part shows the layout of a multi-chart parameterization. Vertices of the layout graph (red) lie at mesh vertices and edges of the layout graph (blue) cut several mesh edges. Each inner vertex of a patch P_α stores its parameter coordinates w.r.t the local frame of chart C_α. For all pairs of neighboring patches transition functions $\phi_{\alpha\beta}$ exist which translate between their charts. Notice that the red quadrilateral, connecting the four corners of Chart C_α, mapped to the surface is generally not identical to the blue layout.

3 Quadrangular Global Parameterization

The input to our quadrangular multi-chart parameterization method is a triangle mesh $M = (V, E, F)$ of arbitrary genus, which is a set of vertices, edges and faces, and a layout graph $G = (\mathcal{V}, \mathcal{E}, \mathcal{F})$. For each edge of the layout graph the user can additionally set a tag which enforces the alignment of the parameterization onto this layout edge, as described in Section 3.2. The scenario of a multi-chart parameterization is depicted in Figure 3. The vertices of the layout graph (red points) lie at triangle mesh vertices and each edge of the layout graph intersects several mesh edges (blue points). In this way all mesh vertices are partitioned into several surface patches, which are disjoint except the layout vertices that belong to all neighboring patches. Each such patch P_α is equipped with a two-dimensional chart C_α. Assume for simplicity that each layout graph face has exactly four vertices, we will discuss in Section 3.3 how to incorporate more general settings. The task is now to compute a piecewise linear multi-chart parameterization, i.e. each vertex $v_i \in \mathbb{R}^3$ belonging to P_α is mapped by the function ϕ_α to the parameter coordinates $u_i^\alpha \in \mathbb{R}^2$ expressed w.r.t the frame of chart C_α. Additionally for triangles with vertices in different patches, for instance P_α and P_β, we need a transition function $\phi_{\alpha\beta}$ to translate between their charts in order to parameterize them. Obviously both directions are possible and inverse to each other $\phi_{\beta\alpha} = \phi_{\alpha\beta}^{-1}$ and the transition from one chart into itself is simply the identity $\phi_{\alpha\alpha} = Id_2$.

A discrete harmonic parameterization of a surface with disc topology mapping to a single chart is a well studied topic where typically the boundary of the surface is mapped to the boundary of a disc and each interior vertex has to fullfill the discrete harmonic equations

$$\sum_{j \in N_i} \bar{w}_{ij}(u_j - u_i) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{1}$$

where N_i are the one-ring neighbors of vertex v_i and $\bar{w}_{ij}$ are normalized edge weights which sum to one $\sum_{j \in N_i} \bar{w}_{ij} = 1$. In all our examples we used the normalized discrete harmonic weights

$$w_{ij} = \frac{1}{2}(cot\alpha_{ij} + cot\beta_{ij}) \quad with \quad \bar{w_{ij}} = w_{ij} / \sum_{j \in N_i} \bar{w}_{ij} \tag{2}$$

where α_{ij} and β_{ij} are the two angles opposite to edge e_{ij}. There are many other good choices like Floater's Mean Value Coordinates, see [2] for more details. The key observation is that in our multi-chart parameterization setting we can compute a harmonic parameterization in the same way. The only difference is that instead of fixing a whole boundary we now only fix the corner vertices of the layout graph in each chart and use the transition functions to compute the harmonic conditions in a common frame:

$$\sum_{(j,\beta) \in N_i} \bar{w}_{ij}(\phi_{\beta\alpha}(u_j^\beta) - u_i^\alpha) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{3}$$

.

In this formulation a global relaxation is achieved. If the transition functions are affine the combination of the above equations for all free vertices is a global linear system of dimension $2(n-k) \times 2(n-k)$ where n is the number of triangle mesh vertices and k is the number of layout vertices. The translational part of the affine transition function as well as known values of constrained layout corners are moved to the right-hand-side. Remember that the coordinates of layout corners cannot be unique because they belong to different charts with different frames. So we need to specify $4|\mathcal{F}|$ many corner positions.

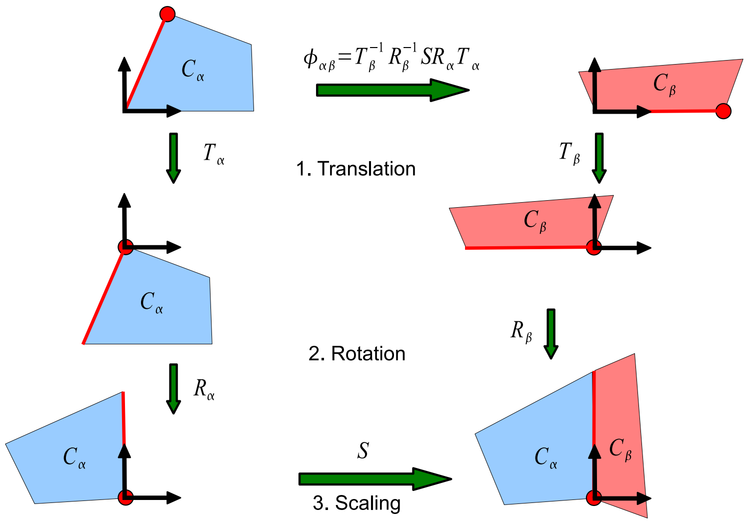


Fig. 4. A common coordinate frame of two charts C_α and C_β can be constructed by first translating a common point into the origin, then rotating the common edge to a coordinate axis and finally scaling along this axis to end up with the same edge length. The transition functions between the charts are constructed in the same way by using the inverse of either α or β functions.

These parameter coordinates of the four patch corners can be in general position (keeping the same orientation as on the surface). However we choose the first one to be the origin and the second one to lie on the first coordinate axis which makes the representation unique. So we end up with five DOF's (a, b, c, d, e) for an arbitrary quadrilateral (see Figure 2 (c)). The transition function between neighboring charts which share a common edge (red) are simple affine functions, combinations of translations, rotations and a scaling as depicted in Figure 4.

$$\phi_{\alpha\beta} = T_\beta^{-1} R_\beta^{-1} S R_\alpha T_\alpha \tag{4}$$

They can be precomputed as 3×3 matrices in extended coordinate representation before accumulating the resulting values into the global system matrix.

The only question left is how to choose adequate corner parameter coordinates (a, b, c, d, e) for a given patch. In [11] the average length of two opposing layout edges rounded to an integer was used to fix width and height of the corresponding rectangle. In the case of a five DOF chart we could do something similar by using all lengths of the patch's boundary. But as explained in the next section the available DOF's can be used to optimize the resulting parameterization in a more founded but still efficient way, which in general leads to better results.

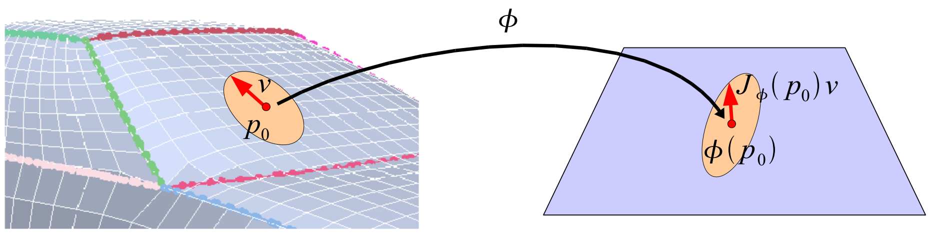


Fig. 5. Mapping a small disc from the tangent plane around a point p_0 the transformation can be approximated by the Jacobi matrix J_ϕ of the mapping ϕ. This means mapping circles into ellipses where the length of the principal axes are related to the singular values of J_ϕ.

3.1 Chart Optimization

The idea of our chart optimization algorithm is to minimize the metric distortion of the parameterization ϕ. The local distortion near a surface point p_0 in direction v (in local coordinates of the tangent plane) is described by the first order Taylor expansion

$$\phi(p_0 + v) \approx \phi(p_0) + J_\phi(p_0)v \quad \Rightarrow \quad \phi(p_0 + v) - \phi(p_0) \approx J_\phi(p_0)v \qquad (5)$$

where J_ϕ is the Jacobi matrix which can be written as two rotations and a scaling by applying the singular value decomposition

$$J_\phi = U \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix} V^T \qquad (6)$$

Mapping a unit length vector $\|v\| = 1$, lying in the tangent plane of p_0, into its chart the resulting vector has length $\|J_\phi v\| \in [\sigma_1, \sigma_2]$. Consequently a circle on the surface is mapped to an ellipse in the chart as illustrated in Figure 5. There are some well known special cases [2]:

1. $\sigma_1 = \sigma_2$ is a conformal mapping which maps circles to scaled circles
2. $\sigma_1 \cdot \sigma_2 = 1$ is an equiareal mapping
3. $\sigma_1 = \sigma_2 = 1$ is an isometric mapping with no distortion

Clearly an isometric mapping is the best we can hope for. So we try to choose our chart corners to get as isometric as possible. The desired isometry measure is $E = |\sigma_1 - 1| + |\sigma_2 - 1|$. To approximate this measure we take the quadratic Frobenius norm of the 2D strain tensor

$$E = \|J_\phi^T J_\phi - I\|_2^2 \tag{7}$$

which is 0 in the case of isometry and $(\sigma_1^2 - 1)^2 + (\sigma_2^2 - 1)^2$ when the mapping is conformal.

Using a triangle mesh where the mapping is piecewise-linear, the Jacobi-matrix of a triangle is constant and depends linearly on the parameter values u_0, u_1 and u_2 of the triangle,

$$J_\phi = [u_0 u_1 u_2] \begin{bmatrix} p_0 & p_1 & p_2 \\ 1 & 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \tag{8}$$

In the above equation p_0, p_1 and p_2 are the 3D triangle vertices in local 2D coordinates and u_0, u_1 and u_2 are the corresponding parameter values. Therefore $J_\phi^T J_\phi$ is quadratic and the isometry measure of a triangle t is a quartic polynomial in the parameter values, $E_t(u_0, u_1, u_2) = \|J_\phi^T J_\phi - I\|_2^2$

The aim of this section is to optimize the isometry of the harmonic parameterization by finding adequate parameter coordinates for the four corners of a chart. Consequently we need to express the isometry measure of a triangle w.r.t. these values (a, b, c, d, e). To approximate the relation between the global parameterization and the change of chart corner positions we assume that the dependency is bilinear, which is a good approximation for all interior vertices of a chart:

$$u_i = u_i(a, b, c, d, e) = s_i(1 - t_i) \begin{pmatrix} a \\ 0 \end{pmatrix} + (1 - s_i)t_i \begin{pmatrix} b \\ c \end{pmatrix} + s_i t_i \begin{pmatrix} d \\ e \end{pmatrix} \tag{9}$$

.

Since we use these bilinear coordinates s_i and t_i in the sense of freeform deformation, the parameter coordinates u_i are linear in the corner positions (a, b, c, d, e) and so the measure $E_t(a, b, c, d, e)$, now expressed in dependency of the four chart corners, is still a quartic polynomial. Finally we sum up the measures of all triangles lying completely inside the polygon formed by the chart corners that we want to optimize and weight them by the area of the corresponding surface triangle.

$$E_\alpha = \sum_{t \in C_\alpha} E_t(a, b, c, d, e) \cdot A_{\phi^{-1}(t)} \tag{10}$$

In this optimization phase all layout edges are always tagged for alignment which ensures that all vertices of patch P_α are mapped into C_α. This energy only depends on five variables and is very well conditioned because of its geometric nature. Therefore we can use a simple and efficient Newton method to find a local minimum. Since the bilinear dependency is only an approximation we have to recompute the parameterization after each chart optimization. To initialize the charts we can simply use unit charts or the heuristic of [11]. The complete algorithm works as follows:

1. tag all layout edges for alignment
2. compute an initial parameterization with unit charts
3. iterate k times
   (a) optimize all charts individually
   (b) update transition functions
   (c) recompute parameterization
4. restore user-provided alignment tags and compute final parameterization

The bilinear relation is close to the exact dependency, therefore in all our experiments three iterations were sufficient to converge. Notice that our method is similar to [10]. However, instead of relaxing the layout vertices on the surface we relax them within the charts. This is more suitable for Reverse Engineering where the user provided layout is in general not allowed to be changed. In the next section we will discuss how to incorporate layout alignment constraints into the computation.

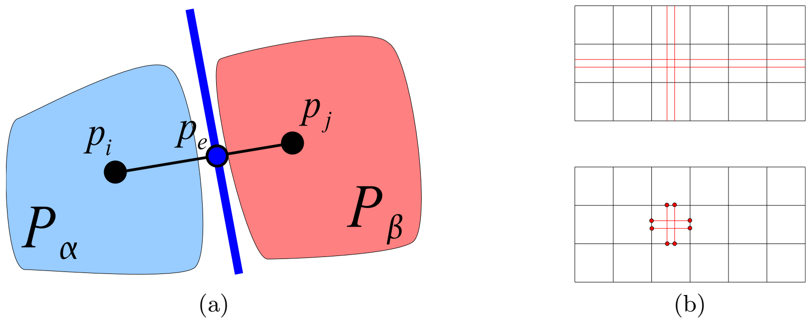


Fig. 6. (a) The parameter values at a layout edge can be computed at the intersection points of triangle mesh edges p_e by incorporating the transition function between both charts. (b) A local refinement of a quadrangular layout has global support (top). By allowing T-Vertices, the refinement of the layout remains local (bottom).

3.2 Alignment Constraints

The user can tag a subset of layout edges for alignment which ensures that it will be explicitly represented in the meshing. For the parameterization this means that the mapping of a tagged layout edge should be the straight line connecting both corresponding corners in the chart. Or in other words the parameter coordinates along the layout edge are not independent. The parameter coordinates at a point $p_e = (1-\lambda)p_i + \lambda p_j$ on the layout edge cannot be computed directly in the form $u_e = (1-\lambda)u_i + \lambda u_j$ because u_i and u_j are represented w.r.t different charts (see 6 (a)). However by employing the transition function, we can express the alignment constraint in a simple form where the image of the layout edge is constrained to have the first coordinate equal to zero. This is exactly the lower right setting in Figure 4:

$$u_e^\gamma = (1-\lambda)SR_\alpha T_\alpha u_i^\alpha + \lambda R_\beta T_\beta u_j^\beta \overset{!}{=} \begin{pmatrix} 0 \\ * \end{pmatrix} \tag{11}$$

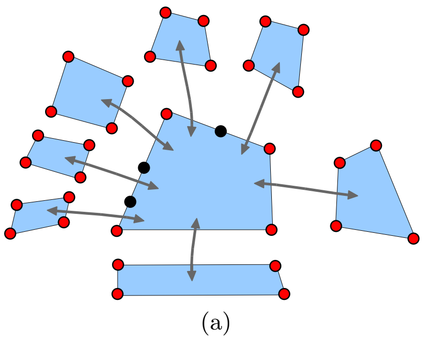
.

The global linear system already has full rank, therefore after adding the alignment constraints we have to relax some other equations to be solved only in least squares sense. A good choice are the harmonic constraints of all vertices which are involved in alignment constraints. This means pulling the parameterization onto the layout edge by allowing slight non-harmonicity near the constraint. Notice that our alignment constraints restrict only one coordinate of the parameterization and there is still a global relaxation in orthogonal direction. Finally the parameterization is formulated as a mixed least squares system of the form

$$\begin{bmatrix} A^TA & B^T \\ B & 0 \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} A^Tb \\ c \end{pmatrix} \tag{12}$$

where the equations $Bx = c$ are fullfilled exactly and the equations $Ax = b$ are satisfied in a least squares sense. In the next section we will describe how to simplify the layout generation by allowing T-vertices.

3.3 T-Vertices



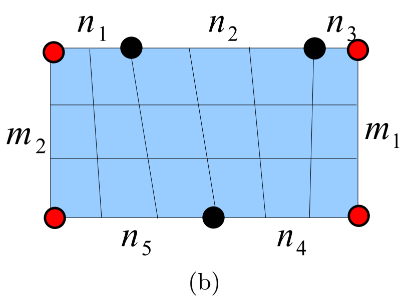


Fig. 7. (a) Allowing T-Vertices in the layout is possible in a simple way by computing a transition function per layout edge. (b) To get a closed quadrangulation the number of samples on opposite edges of a chart must be equal.

Restricting the layout graph to consist only of four sided polygons as done before is too restrictive in practice. Making a local refinement to keep more features, a global refinement would result as illustrated in Figure 6 (b). Remember that in many Reverse Engineering scenarios this layout is directly designed by a user and the effort should be as low as possible. Therefore we allow an arbitrary number of T-vertices per layout edge. This can be easily achieved by computing a separate transition function for each part of a layout edge. The parameter coordinates of T-vertices in a chart are defined by linear interpolation of the corners to keep the number of variables of a chart constant and allow to extract a mesh consisting only of quadrilaterals as explained in the next section.

4 Meshing

The meshing proceeds as follows, first a consistent quadmesh is constructed in the 2D charts of the parameterization which is then mapped to the surface. The four corners of a chart form a four sided polygon in the plane whereas each edge can be partitioned by T-vertices into several subintervals as depicted in Figure 7. By backmapping the chart polygon edge it is possible to compute the desired number of samples n_d which is the quotient of the length of the backmapped curve and the target edge-length for the meshing provided by the user. This value may be chosen differently for each layout edge. However in the case of a consistent quadmesh the number of samples cannot be chosen arbitrarily. There are the following consistency constraints:

1. The number of segments (quadmesh edges) on opposite edges of the chart polygon must be equal. In the example of Figure 7 (a) this means $n_1 + n_2 + n_3 = n_4 + n_5$ and $m_1 = m_2$.
2. Each T-vertex lies on a sampling position.
3. In each subinterval the samples are distributed linearly which guarantees that neighboring charts stitch together compatibly.

With these restrictions a consistent quadmesh can be constructed by connecting opposite sample pairs. This is always possible since condition 1 states that the number of samples is equal on opposing sides. We assemble the two equations per layout face in a common linear system $Bn = 0$ and compute free variables via Gauss elimination as done in [11]. Simply fixing the free variables by rounding the corresponding entries from the local desired number of samples n_d leads to poor results since the free variables computed by the Gauss elimination strongly depend on the numbering of the variables and can lead to strong deviations from the expected number of samples on other edges. Therefore we first compute the best continuous solution n_c which meets the constraint $Bn_c = 0$ and thus minimizes the deviation from the desired values n_d in a least squares sense. As a result we solve the linear system from equation 12 with $A = I_d$, $B = B$, $b = n_d$ and $c = 0$. Then rounding the free variables to the integer closest to the value of the continuous solution n_c leads to appealing results because the continuous solution captures the global necessary edge-length distribution.

5 Results and Discussion

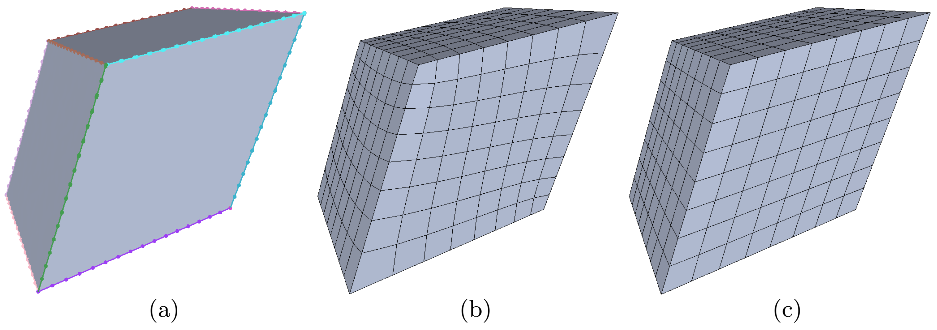


Fig. 8. (a) A sheared cube with unit edge length is segmented along the geometric edges. (b) Restricting to charts with 2 DOF's, length distortions and S-shaped isolines are unavoidable since the necessary curvature of the cone singularities is distributed over the whole geometry. (c) Our method concentrates the tangential curvature at geometric features where they don't influence the mesh quality. This approach leads to the expected result of uniformly shaped quadrilaterals.

In this section we will discuss the properties of the presented method by exploring some results. The first example is the sheared cube with unit edge length depicted in Figure 8 (a). This simple model illustrates the difference between the parameterization of [11] and our method which are displayed in (b) and (c) respectively. In (b) edge length distortions and S-shaped isolines are unavoidable because of the inherent tangential continuity of this method. This can be seen by unfolding neighboring faces of the cube where the isolines in the case of 8 (b) are smooth since the necessary curvature of the cone singularities is distributed over the whole geometry. In contrast to this result our method 8 (c) concentrates the tangential curvature at feature lines, i.e. regions of high geometric curvature where tangential continuity is not important. This example is indeed a hint how to use the presented method. Charts with five DOF's are advantageous for patches with a layout lying on geometric features while charts with two DOF's are better suited within smooth or flat regions. Typical objects consist of both types of regions, such that the user should be able to select for each patch which optimization is performed. This is possible in a straightforward way due to the fact that the optimization of individual charts is independent.

The second example is the car model depicted in Figure 2 and already discussed in the introduction. Figure 9 shows all chart polygons after 3 steps of optimization with 2 DOF's and 5 DOF's in (a) and (b) respectively. The presented optimization algorithm finds well shaped chart polygons robustly and produces almost symmetric configurations since the user-provided layout is almost symmetric. Compared to the time which is necessary for the solution of

(a)

(b)

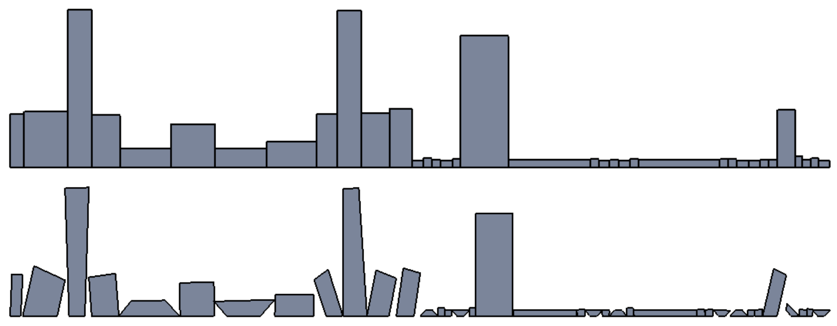


Fig. 9. (a) The corner positions of the car model's charts are optimized to lie on a rectangle. The resulting parameterization maximizes the isometry. (b) In this optimization the corners were allowed to lie in general position. Thus the resulting polygons are planar approximations of the surface patches.

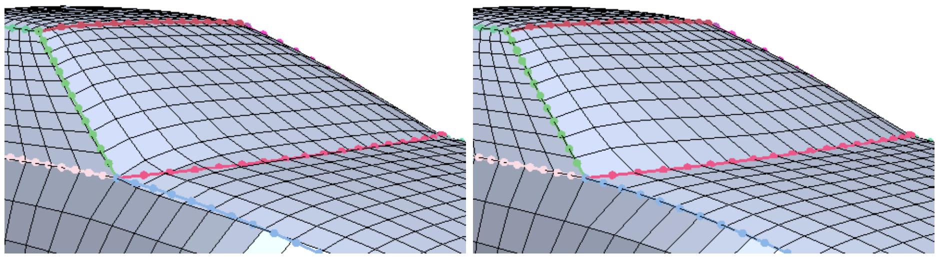
the global linear system, the optimization of charts is neglectable. Altogether the computation timings are comparable to [10] while in practice we need fewer iterations to converge. In all our examples we used the sparse direct solver SuperLU as proposed in [12].

In Figure 10 we demonstrate the usage of alignment constraints. Between the front window and the hood of the car is a sharp edge which should be represented in the final meshing to prevent sampling artifacts. Therefore the lower red layout curve is tagged for alignment. As one can see the isoline of the quadmesh connecting both endpoints of this layout curve in Figure 10 (a) is pulled onto the layout curve in Figure 10 (b) without introducing unnecessary distortion. However by using alignment constraints the computation time for solving the resulting mixed least-squares linear system is higher because of the doubled dimension.

The third example is the rockerarm model from Figure 1. For this mechanical part first a coarse layout was designed to guide the meshing. Figure 11 (a) shows a close up from the backside where large distortions appear near a geometric feature, not covered in the layout. In 11 (b) the layout was locally refined by using T-vertices. In this way the designer can control the Reverse Engineering hierarchically by starting with a rough layout which is refined until all features are captured up to the desired quality. The overall effort to design a layout is minimized by using T-vertices.

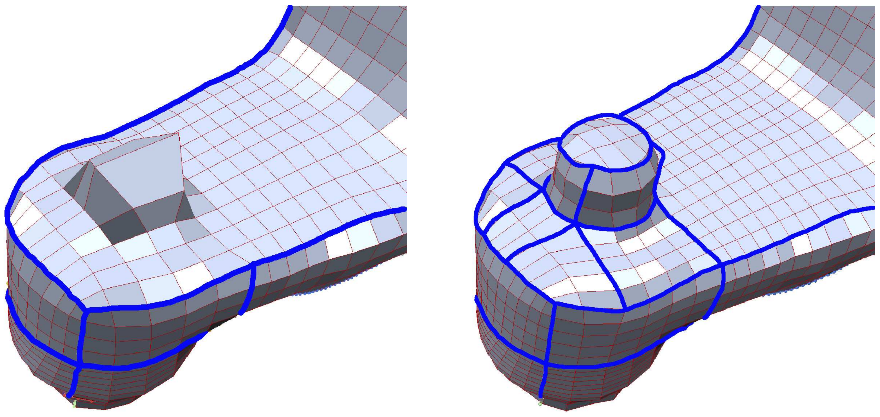
6 Conclusion

In this paper, we have presented a global parameterization technique designed for the requirements of Reverse Engineering. Our solution is a generalization of well known global parameterization methods which allows to use chart corners in general position. To find good positions automatically an iterative algorithm



(a) (b)

Fig. 10. (a) The sharp corner between front window and hood is not represented in the globally smooth parameterization without alignment constrains. This leads to sampling artifacts, i.e. triangles that cut away the sharp corner. (b) The layout curve is tagged for alignment and consequently the mesh edges are pulled onto it, leading to a better approximation of the input geometry.



(a) (b)

Fig. 11. (a) A rough layout (highlighted in blue) leads to large length distortions near a geometric feature. (b) T-vertices can be used to locally refine the layout with minimal effort. The new layout captures the geometric feature much better and avoids the length distortions.

is applied which optimizes the isometry of the resulting parameterization. This chart optimization can be used in combination with other methods as well. Additionally we described the incorporation of alignment constraints and T-vertices which are important ingredients of a Reverse Engineering solution.

Acknowledgments

This work has been supported by the UMIC Research Centre, RWTH Aachen University.

References

1. Alliez, P., Ucelli, G., Gotsman, C., Attene, M.: Recent advances in remeshing of surfaces. Research report, AIM@SHAPE Network of Excellence (2005)
2. Hormann, K., Lévy, B., Sheffer, A.: Mesh parameterization: theory and practice. In: SIGGRAPH '07: ACM SIGGRAPH 2007 courses, New York, NY, USA, ACM (2007) 1
3. Alliez, P., Cohen-Steiner, D., Devillers, O., Levy, B., Desbrun, M.: Anisotropic polygonal remeshing. ACM Transactions on Graphics. Special issue for SIGGRAPH conference (2003) 485–493
4. Marinov, M., Kobbelt, L.: Direct anisotropic quad-dominant remeshing. In: PG '04: Proceedings of the Computer Graphics and Applications, 12th Pacific Conference, Washington, DC, USA, IEEE Computer Society (2004) 207–216
5. Khodakovsky, A., Litke, N., Schröder, P.: Globally smooth parameterizations with low distortion. In: SIGGRAPH '03: ACM SIGGRAPH 2003 Papers, New York, NY, USA, ACM (2003) 350–357
6. Ray, N., Li, W.C., Lévy, B., Sheffer, A., Alliez, P.: Periodic global parameterization. ACM Trans. Graph. **25**(4) (2006) 1460–1485
7. Kälberer, F., Nieser, M., Polthier, K.: Quadcover - surface parameterization using branched coverings. Computer Graphics Forum **26**(3) (September 2007) 375–384
8. Huang, J., Zhang, M., Ma, J., Liu, X., Kobbelt, L., Bao, H.: Spectral quadrangulation with orientation and alignment control. In: SIGGRAPH Asia'08: ACM SIGGRAPH Asia 2008 Papers, to appear. (2008)
9. Ray, N., Vallet, B., Li, W.C., Lvy, B.: N-symmetry direction field design. In: ACM Transactions on Graphics. (2008) Presented at SIGGRAPH.
10. Dong, S., Bremer, P.T., Garland, M., Pascucci, V., Hart, J.C.: Spectral surface quadrangulation. In: SIGGRAPH '06: ACM SIGGRAPH 2006 Papers, New York, NY, USA, ACM (2006) 1057–1066
11. Tong, Y., Alliez, P., Cohen-Steiner, D., Desbrun, M.: Designing quadrangulations with discrete harmonic forms. In: SGP '06: Proceedings of the fourth Eurographics symposium on Geometry processing, Aire-la-Ville, Switzerland, Switzerland, Eurographics Association (2006) 201–210
12. Botsch, M., Bommes, D., Kobbelt, L.: Efficient linear system solvers for mesh processing. In Martin, R.R., Bez, H.E., Sabin, M.A., eds.: IMA Conference on the Mathematics of Surfaces. Volume 3604 of Lecture Notes in Computer Science., Springer (2005) 62–83