Title:
A Labeling Algorithm for Trimmed Surface Fitting

Authors:
Márton Vaitkus (vaitkus@iit.bme.hu) and Tamás Várady (varady@iit.bme.hu)
Budapest University of Technology and Economics

Keywords:
Surface fitting, trimmed surfaces, constrained parameterization, labeling

DOI: 10.14733/cadconfP.2018.86-90

Introduction:
Approximating data points or triangle meshes with tensor-product (TP) parametric surfaces is a task of fundamental importance in computer-aided geometric design. Our particular interest relates to *trimmed regions*, bounded by an irregular multi-sided loop of boundary segments having no obvious tensor-product structure. There are two major applications motivating our work.

First, in reverse engineering [8], a CAD model is to be produced from a measured point cloud. A triangle mesh is generated, then segmented into disjoint regions, corresponding to the faces of the final B-Rep model. Each region is approximated by some surface type, such as simple primitives (planes, quadrics, etc.), procedural surfaces (sweeps, lofts, fillets, etc.), or else by truly free-form shapes, e.g. TP Bézier surfaces or NURBS, the representations used in commercial CAD systems and data exchange standards (IGES/STEP) [2].

Second, there are non-standard surface representations, such as transfinite [9] and control-point based [10] multi-sided patches that possess advantageous properties compared to tensor-product NURBS in certain modelling tasks. Nevertheless, for downstream CAD/CAM applications, the geometry eventually needs to be sampled and approximated by some standard surface.

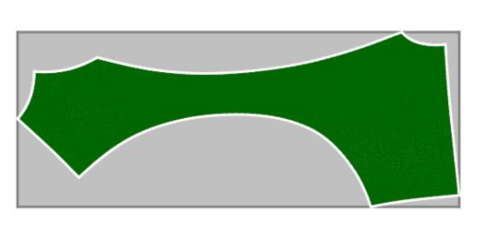
Parametric surface fitting is a difficult problem that depends on various parameters, including error tolerances, knot vectors, regularization weights, etc. [11]. In order to formulate it as a linear least-squares problem "appropriate" (u,v) parameter values must be assigned to the data points; as this will fundamentally determine the qualities of the final surface. When all four tensor-product boundaries are given, the data points usually possess a natural parameterization [1,3,4,6], however, in the trimmed case we have no 'a priori' information about the orientation of the four-sided patch, or about the surface geometry beyond the trimmed region.

In the majority of cases no single “best” TP patch exists for a given trimmed region, and prioritizing the various requirements can be challenging in practice. The fitted surface should be accurate within some tolerance, aligned with the geometric features of the shape, and have a relatively low number of control points. The curvature distribution and the surface extension for the unknown areas beyond the trim curves must be well-controlled, as well.

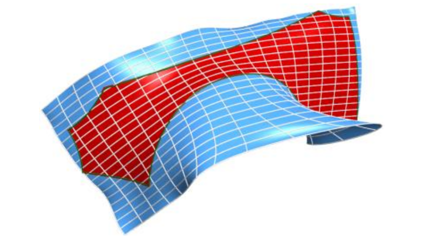
Accordingly, we wish to compute parameterizations that yield well-suited surfaces for CAGD applications. In a recent paper [7], we have introduced new techniques to support trimmed surface fitting based on the fundamental idea of *labeling*, which assigns certain boundary segments to the sides of the domain rectangle. Labels were assumed to be given as part of the input, e.g. defined manually by the user. The automatic assignment of labels was identified as the most important avenue for future research.

In this paper, we describe such an algorithm for automatic labeling. As we are not aware of any theory that characterizes an “ideal” parameterization for trimmed regions, here we propose a heuristic algorithm. We do not claim to produce the "best" possible solution in every scenario, as labeling often involves subjective and application-dependent considerations, however our results generally meet engineering expectations.

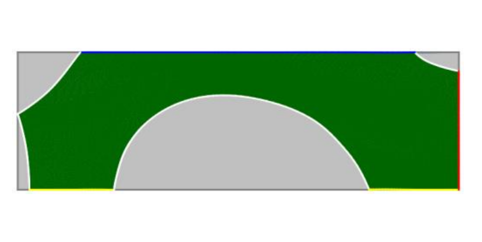
First, we will briefly summarize our work published in [7] and describe the labeling concept. Then, we present the new labeling algorithm. Finally, we demonstrate the effectiveness of our approach by a few examples.



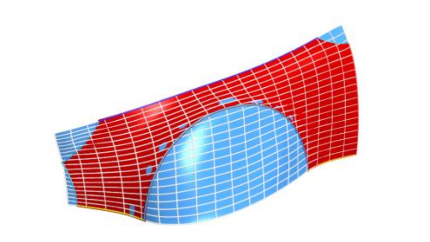
a) Parameterization without Labeling



b) Fit without Labeling



c) Parameterization with Labeling



d) Fit with Labeling

Fig. 1: Labeling Example.

<u>Labeled Parameterization and Extension for Trimmed Surface Fitting:</u>

In [7], we introduced pre-processing techniques to support TP fitting for trimmed patches. Our aim was two-fold: (i) to help orienting the quadrilateral surface based on the geometry of the patch, leading to a simpler and better aligned control grid; (ii) to avoid “weak” control points that have only a few data points in their support and thus could lead to unstable, oscillating fits [11].

Labeling is a powerful technique to orient the yet-unknown surface and it narrows down the set of possible parameterizations. We can assign labels to particular boundary segments -- using our notations -- North, West, South, East, prescribing that a segment must lie somewhere on the boundary of the TP surface to be fitted. With other words, certain segments are mapped to particular sides of the domain rectangle. Other segments may remain Unlabeled.

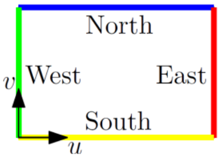


Fig 2: Labels

In [7], we assumed that labels have already been defined and accordingly, we pre-processed the data in two phases prior to fitting. First a *guiding frame* was constructed, which extrapolated the labeled segments into a four-sided virtual boundary loop, then the data was *parameterized* by a constrained optimization of the As-Rigid-As-Possible distortion energy. Second, the 2D triangle mesh in the *(u,v)* plane was *supplemented* so that the entire domain rectangle was covered, and this was inversely *mapped back to 3D*, optimizing an energy that balanced the smoothness of the extrapolated surface and the fairness of its boundary curves.

Fig.1 demonstrates the advantages of our approach; a trimmed region representing a car body panel is to be approximated by a TP surface. Compare the parameterization and the fitted surface 'without' and 'with' labeling.

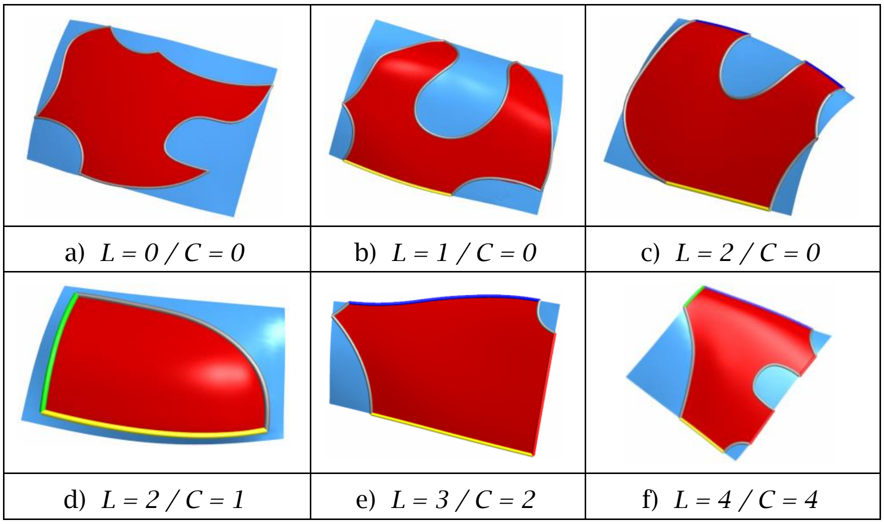


Fig. 3: Admissible label configurations.

Algorithm for Automatic Labeling:
Space limitations prevent us to provide mathematical details, however we present the basic concept of our algorithm, as follows. The input is a triangle mesh (manifold-with-boundary, orientable, possibly multiply connected) with its perimeter boundary loop segmented into a sequence of oriented polylines. Such segmentation of the boundary is naturally produced in the context of reverse engineering [8] and curve-network based surfacing [9,10].

The output is a labeling of the boundary segments as South, East, North, West, or Unlabeled. It is not necessary that all four directions are used, and the same label may be attached to more than one segments. Adjacent labels can form corners – either *real* corners when two labels share a common endpoint, or *virtual* corners formed by their smooth extensions at the endpoints. Labels may terminate at a *no-corner*, when no sensible extension can be created. It can easily be deduced that only six possible label configurations exist by the number of labels (L) and corners (C), as shown by simple examples in Fig. 3. Our aim is to select the configuration best suited for the surface to be created.

The algorithm proceeds as follows. First, we identify those boundary segments which can serve as *label candidates*; more precisely, we exclude those segments that cannot serve as boundaries of the presumed TP surface. Our trimmed region can be interpreted as a remainder, after cutting off parts from a quadrilateral surface; consequently, segments meeting at concave angles cannot be label candidates. TP boundaries tend to be geometrically simple curves, while trimmed boundaries obtained by surface-surface intersections may have more complex geometry [5]. This property can be measured by various quantities, such as the geodesic curvature, and corresponding segments are removed from the set of label candidates, as well.

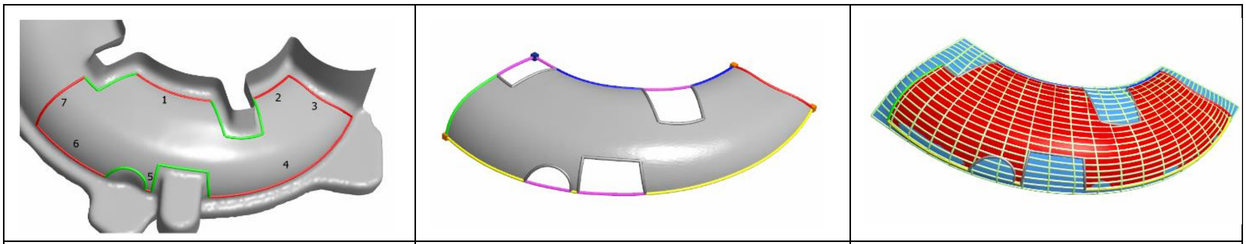
Next, we take disjoint pairs of adjacent label candidates, and decide whether they need to be concatenated into *multi-segment* labels or define a virtual corner. In the former case, a connecting curve with low curvature variation must exist between the endpoints of the labels. In the latter case, after smoothly extending the labels a virtual corner is computed. This corner may not exist, may be far away from the trimmed region or can be well-positioned. The vicinity and the spanned angle of the virtual corner qualifies it as 'weak' or 'strong', and this will help us to exclude or retain it in the next phase.

After carrying out the concatenations, the result may not correspond to any of the admissible label configurations, necessitating further merging or removal of label candidates. Fundamentally we have two operations: killing corners or killing labels. In the former case we take the weakest corner and exclude it by concatenating the related labels ($C \rightarrow C\text{-}1$, $L \rightarrow L\text{-}1$). In the latter case we carry-out hypothesis-tests: assuming we remove label k, a virtual corner is determined by the extensions of labels $k\text{-}1$ and $k\text{+}1$, and we qualify the strength of this new corner. We perform this for each potential label and select the strongest configuration ($L \rightarrow L\text{-}1$, $C \rightarrow C\text{-}1$). It may happen that we have to remove more than one label in one go. For example, take two adjacent labels *k*, *k+1*; removing either of them would yield a very weak corner, but labels *k-1* and *k+2* would create a no-corner, so the adjacent labels need to be removed ($L \rightarrow L\text{-}2$, $C \rightarrow C\text{-}3$). These steps are iterated until we arrive at one of the six admissible configurations, when all the boundary segments get labeled accordingly. Note, that for all intermediate configurations $C \le L$ always holds.

Here we could describe only a simplified version of the labeling algorithm. The sequence of the concrete geometric constructions, the computation of the various qualifiers and thresholds, and the special cases are discussed in details in the paper.

<u>Test examples:</u>
In this section we demonstrate the algorithm by means of three examples.

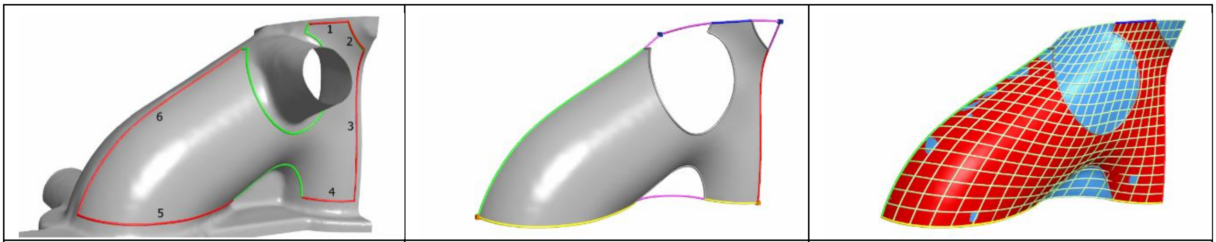
<u>Example 1:</u> this is a relatively straightforward configuration (Fig. 4). The concave parts fall out leaving seven label candidates. 1-2 and 4-5-6 are concatenated, yielding two multiple-segment labels. 2-3, 3-4 and 6-7 define real corners. 7-1 is a strong virtual corner computed by extending and snapping the related label curves. Thus we have four labels and four corners, and the algorithm terminates.



a) Mesh and boundary loop, label candidates - red	b) Label segments concatenated, virtual corners detected	c) Labelled surface fitted

Fig. 4: Example 1.

<u>Example 2</u>: this configuration is more complex (Fig. 5). The concave parts fall out, leaving six label candidates. 4-5 are concatenated yielding a multi-segment label at the bottom. Then - in the first round - there are four real corners: 1-2, 2-3, 3-4 and 5-6, and a single virtual corner 6-1. Five labels and five corners do not form an admissible configuration, thus the algorithm attempts to remove one label and one corner. In the second round, the algorithm evaluates the cost of removing each label, and measures which newly computed virtual corner would be the strongest. Deleting label 1, 3, 4-5 or 6 would yield weak virtual corners, while deleting label 2 leads to a good final solution.

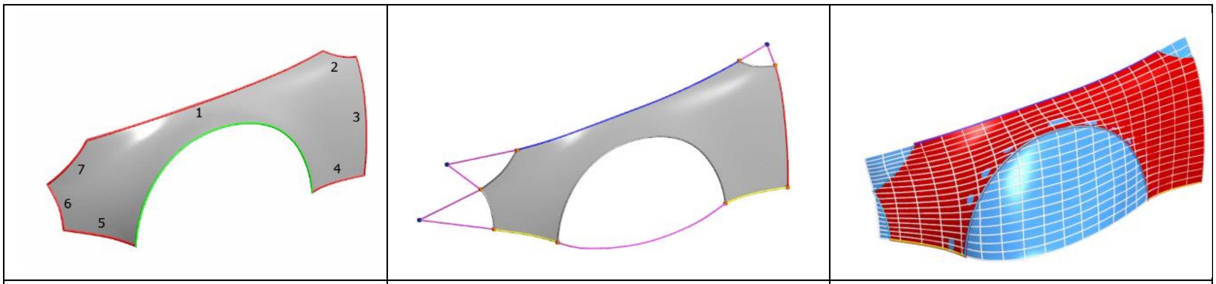


a) Mesh and boundary loop, label candidates - red	b) Labels concatenated, virtual corners detected	c) Labelled surface fitted

Fig. 5: Example 2.

<u>Example 3:</u> finally, let us take the surface shown in the introduction (Fig. 6). Seven label candidates are determined, where 4-5 define a weak virtual corner and gets concatenated, while the other label pairs define real corners yielding six labels with six corners in the first round. The algorithm defines label 2 as to be removed, since labels 1 and 3 define a strong virtual corner. Retaining label 6 and deleting 7,

or retaining 7 and deleting 6 would yield poor configurations with weak virtual corners, so the algorithm deletes both and terminates with a 'three labels - two corners' configuration.



a) Mesh and boundary loop, label candidates - red	b) Labels concatenated, virtual corners detected	c) Labelled surface fitted

Fig. 6: Example 3.

Conclusion:
An algorithm to determine a preferred set of labels for facilitating trimmed surface fitting by tensor product parametric surfaces has been presented. Automatic labeling and the corresponding parameterization helps to fit a collection of high-quality surfaces without manual interaction, often requested in applications, such as reverse engineering or the conversion of non-standard surface representations.

Acknowledgements:
This project has been supported by the Hungarian Scientific Research Fund (OTKA, No.124727).

References:
[1] Azariadis, P. N.: Parameterization of clouds of unorganized points using dynamic base surfaces, Computer-Aided Design, 36(7), 2004, 607-623. https://doi.org/10.1016/S0010-4485(03)00138-6
[2] Farin, G.: Curves And Surfaces for CAGD, Morgan Kaufmann, 5th edition, 2002. https://doi.org/10.1016/B978-155860737-8/50014-4
[3] Lai, Y. K.; Hu, S. M.; Pottmann, H.: Surface fitting based on a feature sensitive parametrization, Computer-Aided Design, 38(7), 2006, 800-807. https://doi.org/10.1016/j.cad.2006.04.007
[4] Ma, W.; Kruth J. P.: Parameterization of randomly measured points for least squares fitting of B-spline curves and surfaces, Computer-Aided Design, 27(9), 1995, 663-675. https://doi.org/10.1016/0010-4485(94)00018-9
[5] Marussig, B.; Hughes, T. J.: A Review of Trimming in Isogeometric Analysis: Challenges, Data Exchange and Simulation Aspects, Archives of Computational Methods in Engineering, 2017, 1-69. https://doi.org/10.1007/s11831-017-9220-9
[6] Piegl, L. A.; Tiller, W.: Parametrization for surface fitting in reverse engineering, Computer-Aided Design, 33(8), 2001, 593-603. https://doi.org/10.1016/S0010-4485(00)00103-2
[7] Vaitkus, M.; Várady, T.: Parameterizing and extending trimmed regions for tensor-product surface fitting, Computer-Aided Design, 2017. https://doi.org/10.1016/j.cad.2017.11.008
[8] Várady, T.; Martin, R.: Reverse engineering, Handbook of Computer Aided Geometric Design, North-Holland, 2002, 651-681. https://doi.org/10.1016/B978-044451104-1/50027-7
[9] Várady, T.; Rockwood, A.; Salvi, P.: Transfinite surface interpolation over irregular n-sided domains, Computer-Aided Design, 43(11), 2011, 1330-1340. https://doi.org/10.1016/j.cad.2011.08.028
[10] Várady, T.; Salvi, P.; Karikó, G.: A Multi-sided Bézier Patch with a Simple Control Structure, Computer Graphics Forum, 35(2), 2016, 307-317. https://doi.org/10.1111/cgf.12833
[11] Weiss, V.; Andor, L.; Renner, G.; Várady, T.: Advanced surface fitting techniques, Computer Aided Geometric Design, 19(1), 2002, 19-42. https://doi.org/10.1016/S0167-8396(01)00086-3