Reconstructing swept surfaces from measured data

By István Kovács, Tamás Várady
Budapest University of Technology and Economics

Abstract

The goal of digital shape reconstruction is to create computer representations from measured data. The process starts with fitting a triangular mesh onto the data points, which is then segmented into disjoint regions. This is followed by region classification, where the best-suited mathematical surface types are determined and assigned to each region, thus optimizing the forthcoming fitting of the individual surfaces. In this paper our goal is to detect and reconstruct *swept surfaces*, which represent a special surface class created by sweeping a planar profile curve along a 3D spine curve. We focus on practical aspects, and attempt to eliminate difficulties due to noise and incomplete data. First an enhanced vector field of principal curvatures is created to be covered by a grid of curvature lines. Using these, we estimate a sequence of sweeping planes, and compute a curve-based representation with a best-fit profile and a spine curve. The surface reconstruction is considered successful, if the swept surface remains within a prescribed tolerance. After discussing the steps of the algorithm, real measured data sets are used to show the difficulties and the results.

1. Introduction

Digital shape reconstruction (reverse engineering) is an expanding, challenging area of Computer Aided Geometric Design [Várady & Martin (2002)]. This technology is utilized in various applications where a given physical object is scanned in 3D, and a computer representation is produced in order to perform various complex computations. A wide range of applications emerges in engineering, medical sciences, and to preserve the cultural heritage of mankind [Marks (2005)]. Examples include redesigning and remanufacturing old mechanical parts, creating surface geometries from clay models, or producing surfaces matching human body parts for hearing aids, dentures, prosthetics, etc.

1.1. *Digital shape reconstruction*

Digital shape reconstruction consists of the following technical phases: (1) 3D data acquisition (scanning), (2) filtering and merging point clouds, (3) creating triangular meshes, (4) simplifying and repairing meshes, (5) segmentation (partitioning into regions), (6) region classification, (7) fitting surfaces, (8) fitting connecting surfaces (e.g. fillets), (9) perfecting surfaces (including constrained fitting and surface fairing), (10) exporting to CAD–CAM systems for downstream applications.

The goal of *segmentation* is to partition the triangular mesh into disjoint regions based on locally estimated geometric characteristics. First, we "remove" the so-called *separator sets* that have strong curvatures, then the remaining parts — *primary regions* — are classified, i.e., the best-suited surface types for approximating the underlying data points

of the region are selected. The structure of the primary regions corresponds to the structure of the faces in the final CAD model. The literature considers segmentation the most crucial phase of digital shape reconstruction, as it fundamentally determines the quality of the final model. In this paper we assume that segmentation has already taken place, and the input of our algorithm — a primary region — is ready to be processed.

Correct *classification* is not an easy task, as we have discrete and noisy data, and we need to determine the type of a full surface from only partially available region information. Classification generally takes place in a sequential order, i.e., we set hypotheses and continue classification until the region meets the expected geometric properties of the hypothetical surface type. We start with simple surface types, and continue with more and more complex ones, as follows:

1. Planes
2a. Extruded surfaces (cylinders, or extrusions with a general profile)
2b. Drafted extrusions (cones, or drafted surfaces with a general profile)
3. Surfaces of revolution (spheres, tori, or surfaces with a general profile)
4. *General swept surfaces with constant profile*
5. Swept surfaces with a varying profile (lofting)
6a. Open free-form surfaces (disk topology)
6b. Closed (periodic) free-form surfaces (cylindrical topology)

Our goal is to confirm or reject the hypothesis that a given region can be well approximated by a swept surface. If it is true, we determine the optimal geometric parameters; if not, we step forward to the next item in the list. When only the planar sweeping property is valid, this is utilized for reconstructing sweeps with varying profiles, or for parameterizing free-form surfaces.

1.2. *Swept surfaces*

Swept surfaces are generated by moving a planar profile curve along another curve called spine or directrix. When the spine is a straight line or a circular arc, and the profile is constant, we create simple extruded or rotational surfaces. For general sweeps, the spine is a 3D curve with an associated sweeping plane and a local coordinate system defined by means of a rotational function. The profile can be constant, or constant with scaling or varying in shape.

Let us denote the continuously parameterized planar ($z = 0$) set of profile curves by $p(u, v) : \mathbb{R}^2 \to \mathbb{R}^3$; the twice differentiable, not self-intersecting spine curve by $\gamma(u) : \mathbb{R} \to \mathbb{R}^3$. Let $r : \mathbb{R} \to \mathbb{R}^3$ be the rotational function perpendicular to the spine, i.e., $\langle \dot{\gamma}(u), r(u) \rangle = 0$. Thus u is the longitudinal parameter, that aligns the positions of the sweeping plane, and v is the cross-sectional parameter that defines a point on the current profile in the given plane. $M_r(u)$ denotes the rotational matrix that turns the vector $(1, 0, 0)^\top$ into the direction $r(u)$, and $(0, 0, 1)^\top$ into $\dot{\gamma}(u)$, and let us denote the longitudinal scaling matrix by $M_{sc}(u)$. Then the equation of the swept surface is given as

$$S(u, v) = \gamma(u) + M_r(u) M_{sc}(u) p(u, v).$$

For sweeps with constant profile the equation can be simplified:

$$S(u, v) = \gamma(u) + M_r(u) p(v).$$

These sort of surfaces frequently occur in Computer Aided Design, and their curve-based, perfect representation is essential for data exchange and manufacturability. In the next sections we will work with discrete data sets, and create only polylines. To convert

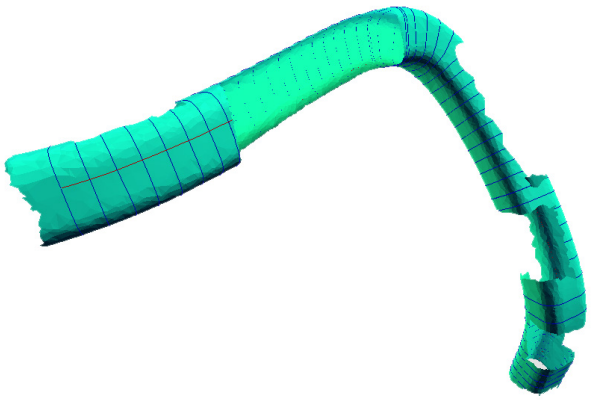

FIGURE 1. Incomplete input mesh with noise and holes in the interior.

these into standard parametric curves is a well-known task, and we do not deal with this in the paper.

1.3. *Problems*

To decide on the swept surface hypothesis, we need to estimate curvatures on noisy and discrete data sets, and we need to trace stable curvature lines over often unevenly distributed triangular regions. We compute the sweeping planes and the defining curves of the surface from noisy data, therefore filtering and smoothing the data is important in each phase of the algorithm.

Most engineering parts are bounded by *trimmed* surface portions, which have been constructed in the CAD process through surface–surface intersections. Accordingly, the corresponding regions, from which we wish to reconstruct the original surfaces, may also be incomplete, having holes in the region interior. Also, we need to pay attention to the termination of the sweeps, as the trim curves at the ends will not necessarily lie in sweep planes perpendicular to the spine, and may not contain the full profile. Consequently, the algorithm must handle incomplete grids of curvature lines, i.e., the holes must be bypassed or jumped over, and we must be able to compose the profile curve from partial sections, as well. These problems are illustrated in Figure 1.

1.4. *Previous work*

There is a wide range of publications that may be associated with our paper, including reverse engineering of shapes, segmentation over meshes, tracing curvature lines, and extracting feature curves, but — according to our best knowledge — the reconstruction of swept surfaces over incomplete, noisy data for mechanical engineering CAD has not been explored in detail. We have selected only a few interesting publications. An early paper on swept surfaces was published in Ueng et al. (1998); data points were approximated by B-spline curves and related tensor product surfaces, but no direct attempt was made to extract the feature curves of the sweeps. Lee & Kim (2004) reconstructed pipe and canal surfaces from measured data, and presented an interesting technique for thinning a set of points to generate spine curves. Huysmans et al. (2006) investigated cylindrical parameterization methods for tubular surfaces, however, they did not utilize the sweeping property and curvature-based structures. Quad-dominant meshing was also a topic of high interest, where fairly complex structures of quadrilaterals were built from a set of

numerically traced principal curvature lines (Alliez et al. (2003), Marinov & Kobbelt (2004)). Related investigations included the robustness of curvature estimations (called trusted curvatures) and overcoming meshing problems due to the occurrence of umbilical points within the global net of curvature lines.

Our paper is structured as follows. In Section 2, we deal with the numerical estimation of curvatures, and methods to unify and smooth the curvature vector field. In Section 3, we discuss how to trace curvature lines, and create consistent grids from them. In Section 4, we compute the optimal profile and spine curves, followed by the numerical evaluation of the surface representation obtained. Finally, future research topics conclude the paper.

2. Creating and repairing curvature vector fields

In this section we generate an enhanced vector field for tracing curvature lines in the forthcoming phases.

2.1. *Local curvature estimations*

It is well-known from classical differential geometry, that at each point p of a twice continuously differentiable surface S, the surface curvature is defined by two principle curvature values $\kappa_{\min}(p)$ and $\kappa_{\max}(p)$, and two associated principle directions, denoted by $t_{\min}(p)$ and $t_{\max}(p)$. The lines of curvature follow the principal directions — i.e., their tangents always coincide with one of the principal directions, and these constitute an orthogonal grid of curves on the surface. Exceptions occur at umbilical points, where the principal directions do not exist; the $\kappa_{\min}$ and $\kappa_{\max}$ values are equal, and the normal curvature is identical in every direction.

There are many published results concerning the estimation of local surface characteristics based on triangular meshes [Benkő & Várady (2004), Botsch et al. (2010)]. *Planarity* is a scalar measure, that quantifies whether a local neighborhood of a point is flat or not. The Gauss (G) and mean (H) curvatures are also important scalar measures, that can be well estimated based on triangle fans around the given points. It is also well-known that these mutually determine $\kappa_{\min}$ and $\kappa_{\max}$, i.e.,

$$G = \kappa_{\min}\kappa_{\max}, \quad H = \frac{\kappa_{\min} + \kappa_{\max}}{2},$$

and

$$\kappa_{\min,\max} = H \pm \sqrt{H^2 - G}.$$

The principal directions $t_{\min}$ and $t_{\max}$ can be estimated by means of fitting a local cylinder in least-squares sense, which leads to solving an eigenvalue equation. It is sufficient to compute the $t_{\min}$ vector, as $t_{\max}$ is orthogonal to this. The principal directions of a triangular region are shown in Figure 2; the little dashes are being set according to the stronger direction of curvature.

2.2. *Unifying and smoothing vector fields*

In order to produce a consistent grid of curvature lines, we need a unified vector field with reduced noise, and without *min-max flips*. It is also necessary that the region does not contain umbilical points, otherwise it would be impossible to parameterize the swept region by a full grid.

It often occurs that the lines of a curvature grid first follow minimal curvature lines, then at a point they continue along maximal curvatures; this is called min-max flipping. This is depicted in Figure 4.1, where apparently creating a united curvature grid is not

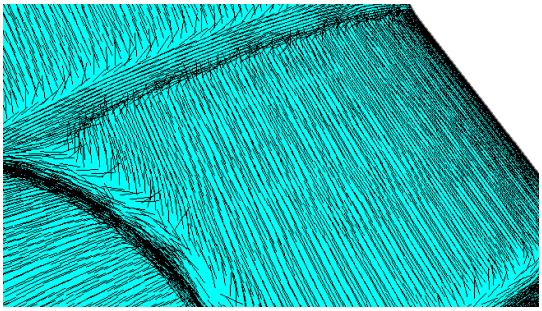

FIGURE 2. Estimating principal directions.

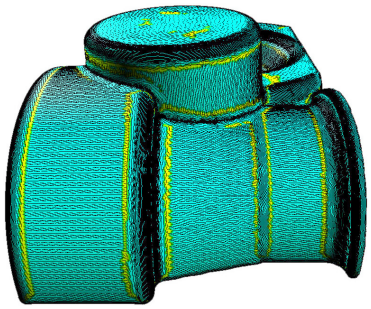

FIGURE 3. Curvature stability map.

possible. The procedures of curve tracing are likely to get terminated, as the curvature estimations are very unstable at these places (Figure 4.2). To overcome this problem, we detect the flipping areas, and swap the direction of the tracing from minimal to maximal or vice versa. A corresponding sequence is illustrated in Figure 4. The algorithm for repairing the vector fields can be split into four phases (the term "weak" will be defined in the next sections):

(a) detecting areas of unstable curvatures,
(b) discovering and avoiding min-max flips,
(c) unifying vector fields,
(d) smoothing vector fields in the weak areas.

2.3. *Curvature stability filter*

This filter selects points within a triangular mesh, where the estimated principle directions are stable, i.e., roughly identical within a given neighborhood. At these points the curvature estimation is robust, and the points have *strong* curvature. In the areas of *weak* curvature, the variance of the principal directions is high, i.e., the vectors span large angles around the points. In these areas, including areas where flipping takes place, curve tracing is likely to get stuck, or yield an incorrect result, see Figure 3.

Formally, let us denote the vector field of the minimal principal directions by $\kappa_{\min}$, and take an n-element neighborhood of a given point p, denoted by $\Omega(p,n)$ (i.e., one or two layers of triangle fans). A point p is of strong curvature, if for a prescribed angular tolerance ε

$$\forall\ p_u, p_v\ \in\ \Omega(p,n) : \langle \kappa_{\min}(p_u), \kappa_{\min}(p_v) \rangle > 1 - \varepsilon.$$

2.4. *Region classification and unification*

In order to generate a curvature grid, we are going to create a unified vector field. In the previous phase we have computed disjoint subregions by minimal principal curvatures. These subregions have vector fields perpendicular to each other, see Figure 4.2. Now our goal is to locally qualify the subregions, and decide whether the $\kappa_{\min}$ or $\kappa_{\max}$ principal curvatures will be incorporated into the final vector field. The weak points of curvature will also be repaired and/or replaced to match the curvature flow of the merged subregions.

Let us start from an initial vector field of $\kappa_{\min}$ curvature, and align the adjacent regions to match this. We qualify the subregions as type $V_{\min}$ or $V_{\max}$, indicating whether the $\kappa_{\min}$ or $\kappa_{\max}$ curvature vector field will be actually used. In the first step we pick a starting point p_0 contained in $V_{\min}$, then by parallel traversal we qualify each point p of strong curvature, by whether the previously visited neighbors of $\kappa_{\min}(p)$ are aligned with

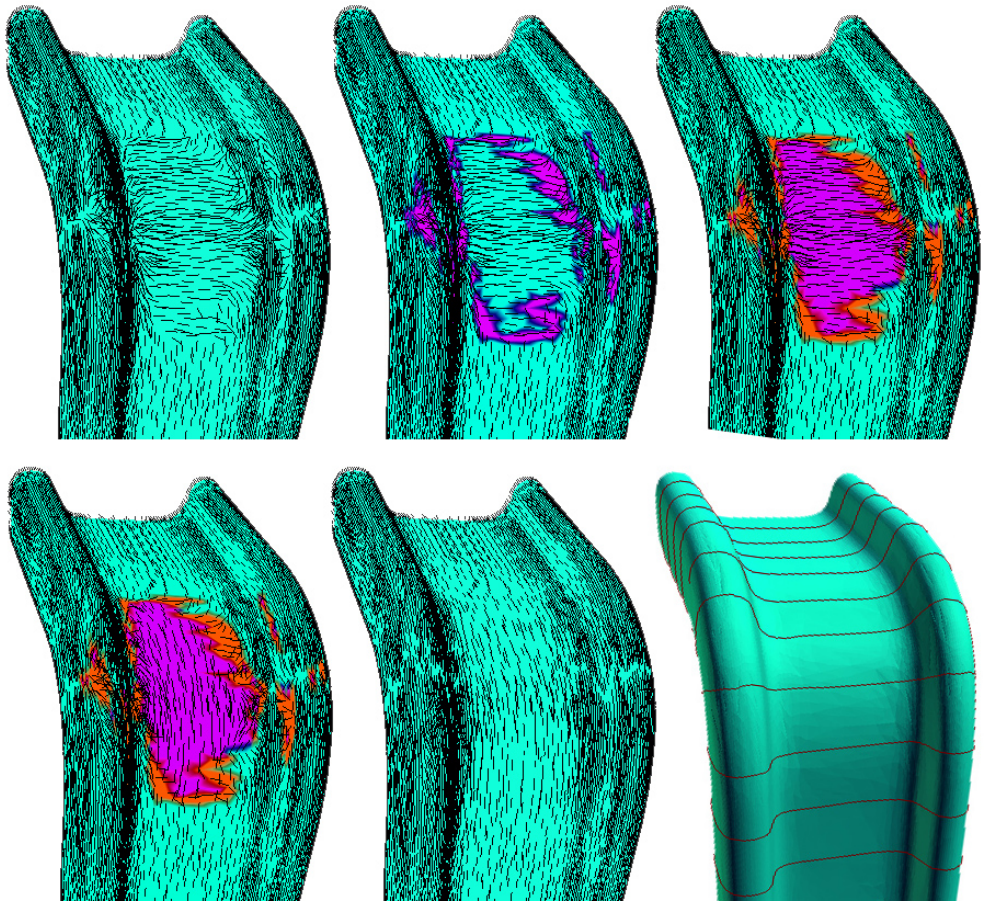

FIGURE 4. Stabilizing vector fields: 1. Local estimations, 2. Detecting areas with weak curvature, 3. Local area classification, 4. Unification, 5. Uniting and smoothing, 6. A sequence of profiles.

the vectors of $V_{\min}$ or $V_{\max}$. After traversing all points of strong curvature, these will be assigned either to $V_{\min}$ or $V_{\max}$.

2.5. *Smoothing areas of weak curvature*

Now we can create a united vector field $\kappa_{\min-\max}$ by merging $V_{\min}$ and $V_{\max}$. $\kappa_{\min-\max}$ is going to be interpreted for the areas of weak curvature. Assume p is a point with weak curvature, and the set of points with strong curvature around p is denoted by $\Omega^+(p,m)$. Using the average of these points, we replace the curvature at p by

$$A(p) = \sum_{q\in\Omega^+(p,m)} \frac{\kappa_{\min-\max}(q)}{m}.$$

To sum this up,

$$\kappa_{\min-\max}(p) = \begin{cases} \kappa_{\min}(p), & \text{for } p \in V_{\min}(p) \\ \kappa_{\max}(p), & \text{for } p \in V_{\max}(p) \\ A(p), & \text{otherwise} \end{cases}$$

Note: It makes sense to repeat the above steps once or twice, i.e., to rerun the curvature filter on the obtained vector field $\kappa_{\min-\max}$, and perform averaging again for the data

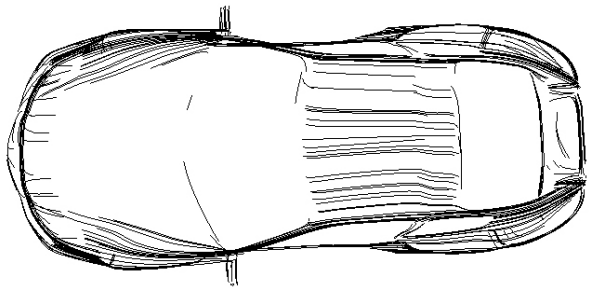

FIGURE 5. Curvature lines traced on a car model.

points of weak curvature. The enhanced vector field enables tracing curvature lines in a more robust manner.

3. Curvature lines and grids

In this section we discuss how curvature lines can be traced and merged in order to obtain an orthogonal grid.

3.1. *Curvature lines*

We have a mesh with estimated principle directions at each vertex. This discrete vector field can also be interpreted on the edges of the triangles, and at interior points, as well, if we weight the vectors at the vertices by barycentric coordinates. Our goal is to compute a surface curve (an integral curve), whose derivative always points to the direction of the vector field, i.e., we numerically solve the differential equation $\dot{\varphi} = \kappa_{\min}$ or $\dot{\varphi} = \kappa_{\max}$. We will generate a polyline, whose points lie either on the edges or in the interior of the triangles. The polyline, $G : \mathbb{R} \to \mathbb{R}^3$, can always be parameterized by approximate arc-length or by scalar curvature values. The points of G are denoted by $V(G) = \{p_0, p_1, \ldots, p_n\}$, the edges by $E(G) = \{e_1, e_2, \ldots, e_n\}$, where e_k connects the vertices p_{k-1} and p_k.

In the course of curve tracing we typically step from edge to edge, though the polyline may contain interior points, as well. Let us denote the maximal step-length by $s_{\max}$. Take a point p_i that lies on edge l_i, or in the triangle h_i. In order to determine the next point p_{i+1}, we take the surface normal n_i estimated at p_i, and intersect the edges of h_i by a plane with normal $n_i \times \kappa(p_i)$, that also contains p_i. If $|q - p_i| < s_{\max}$, this yields a new point q; otherwise the new point p_{i+1} will fall into the interior of the triangle. This process can be refined by applying the well-known Runge–Kutta method. Compute the next point q by the above algorithm and define an intermediate point $q^* = \frac{p_i+q}{2}$, then using q^* we can determine p_{i+1}.

In the case of points within the interior of the region, tracing is started by two opposite directions, and the two traced polylines will be merged later. Curves are terminated when we reach a boundary of the region, or an area of weak curvature. Of course, the boundaries of the holes are treated in the same way as the perimeter loop. A set of curvature lines are shown in Figure 5.

Tracing closed polylines requires special attention. Assume that we wish to generate a closed curvature line on a region with cylindrical topology, but the polyline initiated at point p fails to get back accurately to this point due to noise of the vector field and errors of the estimations. At the same time, it is expected that for closed curves the polyline will

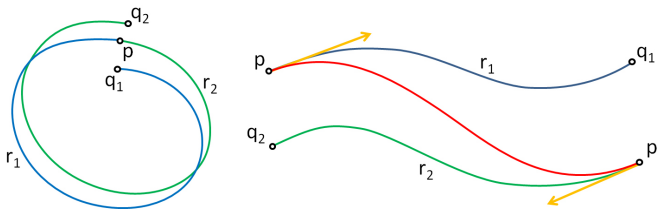

FIGURE 6. Inaccurate closed curves and Hermite blending.

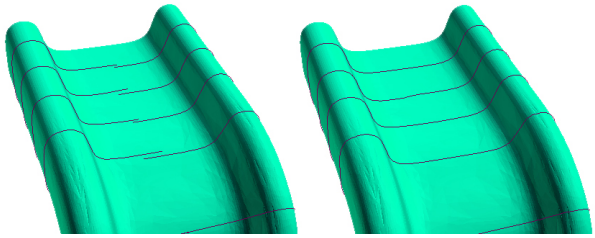

FIGURE 7. Closed curves before and after repair.

return to a neighborhood of the initial point, characterized by a radius ε. This situation can easily be recognized. We qualify a curve "almost" closed, if there exists a point q_1, for which $p \neq q_1$ and $|p - q_1| < \varepsilon$. We assume that while tracing to the opposite direction, we locate another close point q_2, as shown in Figure 6. Assume that we want to blend together two parameterized curves $r_1(u)$ and $r_2(u)$ in such a way, that the result curve $r(u)$ inherits the position and tangent of the start point of r_1, and those of the endpoint of r_2. Then we can blend by a cubic Hermite function:

$$r(u) = r_1(u)H(u) + r_2(u)H(1-u), \text{ where } H(u) = 2u^3 - 3u^2 + 1$$

We apply the same logic to merge our imperfect curves. If $r_1(u)$ runs from p to q_1, and $r_2(u)$ from q_2 to p, the blended curve $r(u)$ starts at p and arrives to p, moreover, it retains the original tangent direction of p. This method is depicted in Figure 7; the curves to be closed are shown on the left, the repaired curves on the right.

3.2. *Curvature grids*

A *curvature grid* is a collection of curvature lines defined by the vector fields of principal curvatures $R_v = \{v_1, v_2, \ldots, v_n\}$ and the orthogonal counterpart $R_h = \{h_1, h_2, \ldots, h_k\}$. We expect that the distribution of curvature lines in a grid correspond to the scalar fields of the principal curvatures, i.e., in areas of high curvature the density of the curves is higher, in flat parts it is lower. A local curvature grid exists almost everywhere, but in general a globally connected, quadrilateral curvature grid cannot always be generated due to the umbilical points on the surface.

For swept surfaces the quadrilateral curvature grid always exists, but due to holes only a certain subset of the grid can be computed. Let the curves R_v run in the cross-sectional direction along the profiles, while the curves R_h longitudinally, "in parallel" to the spine curve. Let us parameterize a selected longitudinal curve R_h by its principal curvature, and generate the orthogonal cross-sectional curves v_i in accordance with this. Let us take a section e on the longitudinal curve, and let $l(e) = |e|(c + \kappa(e))$, where $\kappa(e)$ denotes the normal curvature of the surface by direction e, and c denotes a constant for corrections to be applied in the almost flat areas, where the curvature is practically zero; without this we may obtain extremely sparse grids. This correction term also helps to stabilize

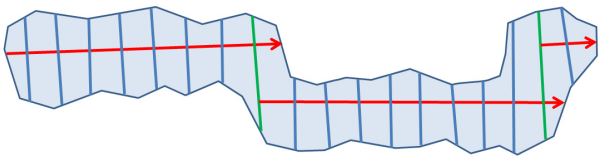

FIGURE 8. Linking cross-sectional curves in a region with multiple holes.

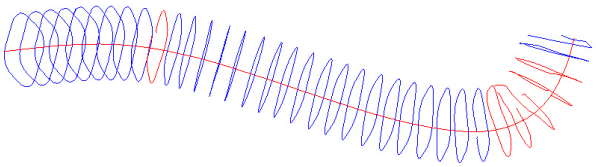

FIGURE 9. Blue curves indicate good profile candidates by a tight planarity tolerance.

uncertain values coming from the numerical curvature estimations, thus yielding a better parameterization. We apply the above formula to compute the density of curvature lines in the cross-sectional direction.

If a region does not contain holes, it is sufficient to only trace along a single longitudinal *auxiliary* curve, and then it is straightforward to link the sequence of cross-sectional curves to it. At the same time, the algorithm must be able to handle regions with holes, see Figure 1. Currently, we can handle configurations where there are no opposite holes at a given cross-section, i.e., the profiles can be partial, but must always be singly connected in the sweeping plane. In this case, we can link the cross-sectional curves by using several auxiliary curves, as it is shown in Figure 8. The first auxiliary curve is terminated as we reach the first hole. Then using the last linked cross-sectional curve, we attempt to pick new points for starting the next auxiliary curve. The one will be chosen, which produces the longest auxiliary curve for collecting the most possible cross-sections. Therefore, the whole grid structure can be characterized by an alternating "auxiliary curve — cross-sectional curve — . . . " sequence.

4. Computing the profile and the spine curve

Based on the auxiliary curve (curves) we obtain a sequence of cross-sectional curves. In the longitudinal direction we parameterize by curvature, and in the cross-sectional direction by arc-length, for the time being. In order to determine an optimal profile representation, first (i) we replace the cross-sectional curves by best-fit planar curves, then (ii) all profile candidates are projected into a common work plane. After (iii) aligning and (iv) averaging these, we compute the best approximation of the profile curve, which (v) is transformed back into the original sweep planes in a way these were sampled formerly. Finally, (vi) we slightly adjust these to find the best local fit of the ideal profile to the underlying triangles. (vii) A well-chosen reference point on the relocated profile curves define a 3D sequence of points, and that will the be the basis for fitting the final spine.

The cross-sectional curves need to be qualified by their *planarity* [Benkő & Várady (2004)]. It is easy to fit a plane onto the data points by least-squares fitting of the distances, and the sum of the squared distances can be interpreted as an indicator of planarity. This is illustrated in Figure 9. If the profiles do not satisfy the planarity criterion, the region cannot be represented as a swept surface. If the curve is nearly planar, it is projected into its best-fit plane and will be considered a profile candidate.

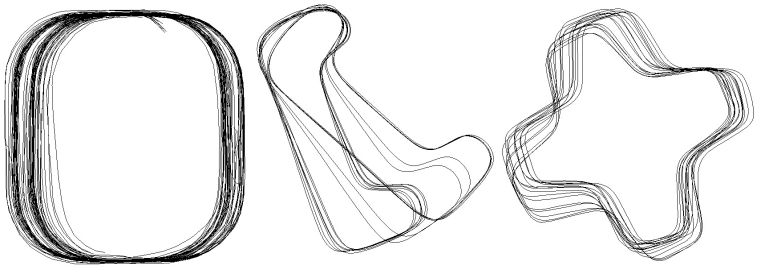

FIGURE 10. Profile candidates after preliminary fitting.

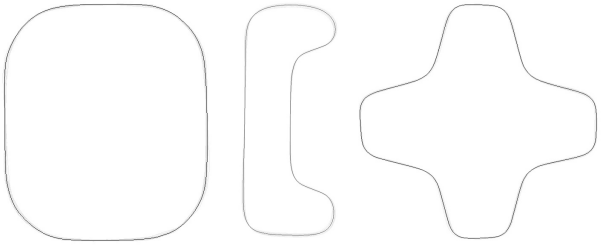

FIGURE 11. Averaged profile curves.

Preliminary fitting. The previously computed sweeping planes and profile candidates now get transformed into a common work plane by means of the auxiliary curves. Our goal is to properly align these. We assume that the consecutive profile candidates share common reference points in the work plane and we search for the best translations and rotations to get the best alignments, see Figure 10. This figure also illustrates that the profile candidates — due to the nature of measured data and numerical curve tracing — do not perfectly fit onto each other; however, this pile of curves is suitable to find an optimal profile using the well-known ICP technique, [Besl & McKay (1992)]. We can assume that the consecutive profile candidates have changed only to a moderate extent, thus we have appropriate initial alignments.

Accurate fitting with ICP. The Iterative Closest Point method is used to find the best-fit alignment of two given point sets A and B. The basic idea is that for all points $a \in A$ we locate the closest points $cl(a) \in B$. Then we search for the best transformation, that moves point set A to B by minimizing the squared distances between these point pairs. After applying this transformation to A, the algorithm is repeated or performed on B in respect to A. This makes sense, as in each new step other point pairs of A and B will be coupled for the least squares minimization. This iteration converges to the best alignment between the two point sets. Profile candidates aligned by ICP are depicted in Figure 11.

Note: In each step of the iteration we compute the best linear transformation, but we retain only the best translation and rotation. This yields less accurate results, but can be computed efficiently. This inaccuracy does not count in spite of a somewhat larger number of iterations, but in overall there is a significant gain in computational efficiency (see Eggert et al. (1997)). This method is suitable to align partial profiles, as well, see the pile of curves on the left side in Figure 11. It may also be useful to filter out the outliers using a two-pass filtering at the beginning.

The optimal transformation matrices, computed by means of the ICP alignment, need to be stored as these are going to be utilized later.

Averaging the profiles. After the ICP phase, the final profile curve can be computed

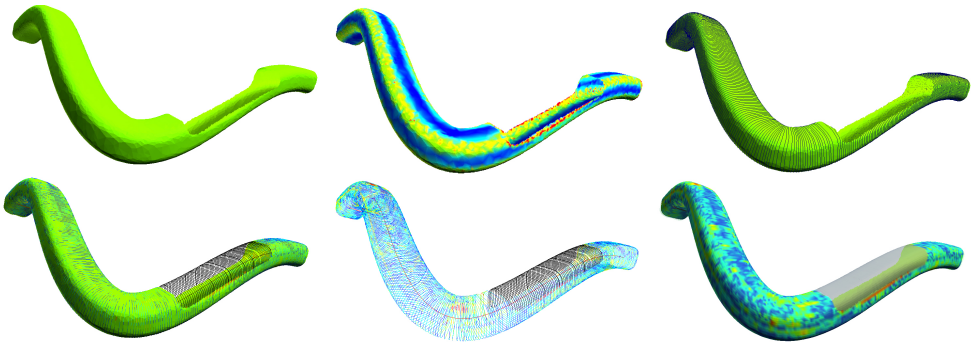

FIGURE 12. Steps of the surface reconstruction: 1. initial region, 2. mean-curvature map, 3. sweeping planes, 4. profile curves on the surface, 5. optimized profile and spine curves, 6. deviation map.

by averaging the transformed profile candidates using simple heuristics. Take the longest profile candidate (recall that we typically we work with partial curves) and parameterize the other profile candidates by this long one. Take the average of the data points that lie in the vicinity of the long profile with a thickness δ to obtain the points of the final profile. This polyline will be approximated by a B-spline curve. The result of the procedure is depicted in Figure 11.

Computing the spine. The final surface representation will be completed by computing the spine curve and the rotational function. The final profile curve P was computed by merging and averaging the profile candidates G_i, each having a transformation matrix T_i. The final profile can be put back to the surface by the inverse transformation $T_i^{-1}P$ for each G_i, which will match the original cross-sectional lines of curvatures (Figure 12). By means of these operations the points of the spine can also be computed: let $v \in P$ an arbitrary point, then the points $p_i = T_i v^{-1}$ define the points of the spine, which will be approximated by a B-spline. The discrete values of the rotational function are also given by a sequence $r_i = T_i^{-1}(1,0,0)^\top$, thus the transformation matrix can also be defined in a continuous form by fitting a multi-dimensional B-spline onto the elements of the matrix.

5. Evaluating the results

In digital shape reconstruction we search for the best possible surface representation. By means of our algorithm, one can decide — using a *planarity tolerance* — whether (i) the cross-sectional curves are planar or not. If the answer is yes, the sweeping planes can be characterized by a smooth, continuous spine curve. (ii) It can also be decided, whether the surface can be represented accurately by sweeping a single profile curve, i.e., the optimal profile curve and the profile candidates are within a *profile tolerance* or not. (iii) Finally, the representation is accepted as a swept surface, if the distances between the data points and the surface remain within a *distance tolerance*. Generally, a very small percentage of the data points are considered as outliers to be ignored, and the decision is made by the remaining points, computing either the maximal or the average deviation. If this value remains within the prescribed tolerance, the swept surface representation is considered valid.

The measured data points and the reconstructed surface model can be visually analyzed using a *deviation map*. This shows the distribution of the signed distances, and locates the most inaccurate areas. Figure 12 shows such a deviation map using rainbow color-coding; the grey areas indicate that the deviation map has no meaning over the holes.

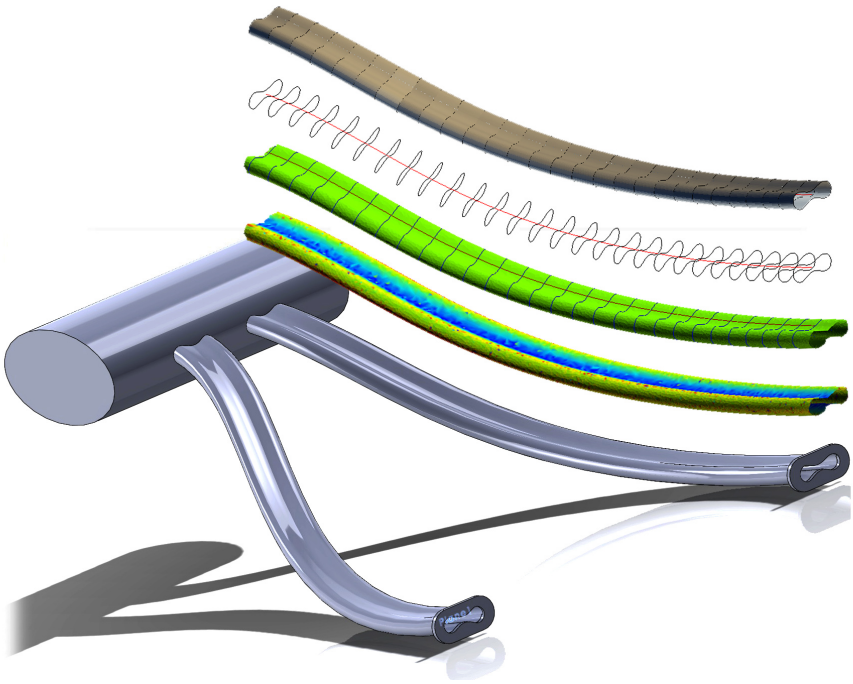

FIGURE 13. A test part: exhaust pipes and surface reconstruction.

6. Conclusion and future research

We have discussed an algorithm to detect and produce an optimal curve-based representation for swept surfaces with constant profile, see Figure 13. It is to support the classification of segmented regions with measured data points, which is an important chapter within digital shape reconstruction. The algorithm is capable of creating surface approximations over noisy and incomplete data sets, as well. A consistent, repaired and smoothed vector field is produced, by which a grid of curvature lines is created. This drives the computation of the sweeping planes, the optimal profile and the spine curve. We have analyzed the deviations from the data points, and qualified the surface whether the constant-profile sweeping property holds or not. The algorithm has been tested on real measured data sets, as well.

One of our future goals is to extend our classification, when the above hypothesis fails. Then we wish to decide whether the given region can be well-approximated by a scaled profile (including monotone shrinking or growing). If this hypothesis also fails, it is still possible that the region can be modeled by a lofted surface with variable profiles. If only the planar sweeping property remains in effect, it is beneficial to use this for parameterizing the given data points for fitting free-form surfaces.

Acknowledgements

This work was supported by the Budapest University of Technology and Economics (UMFT-TÁMOP-4.2.2.B-10/1-2010-0009), and a grant by the Hungarian Science Research Fund (OTKA, No. 101845).

REFERENCES

P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, M. Desbrun 2003 Anisotropic polygonal remeshing, *ACM Transactions on Graphics (TOG)* **22**, 485–493.

P. J. Besl, N.D. McKay 1992 A Method for Registration of 3-D Shapes, *IEEE Trans. on Pattern Analysis and Machine Intelligence (Los Alamitos, CA, USA: IEEE Computer Society)* **14** (2), 239–256.

P. Benkő, T. Várady 2004 Segmentation Methods for Smooth Point Regions of Conventional Engineering Objects, *Computer-Aided Design* **36**, 511–523. Elsevier.

M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, B. Levy 2010 Polygon Mesh Processing, AK Peters

D.W. Eggert, A. Lorusso, R.B. Fisher 1997 Estimating 3-D rigid body transformations: a comparison of four major algorithms, *Machine Vision and Applications* **9**, 272–290.

G. Farin 2002 Curves and Surfaces for Computer Aided Geometric Design. A Practical Guide, Academic Press, 5th Edition

T. Huysmans, J. Sijbers, F. Vanpoucke, B. Verdonk 2006 Improved Shape Modeling of Tubular Objects Using Cylindrical Parameterization, *Medical Imaging and Augmented Reality*, **4091**, 84–91.

I.-K. Lee, K.-J. Kim 2004 Shrinking: another method for surface reconstruction, *GMP '04: Geometric Modeling and Processing* 259–266.

M. Marinov, L. Kobbelt 2004 Direct anisotropic quad-dominant remeshing, *Proc. 12th Pacific Conf. Computer Graphics and Applications*, 207–216.

P. Marks 2005 Capturing a Competitive Edge Through Digital Shape Sampling & Processing (DSSP), *SME Blue Book Series*

W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery 1992 Numerical Recipes in C, Cambridge University Press, 2nd Edition

Zs. Terék 2010 Curve Networks to Segment Polygonal Meshes for Digital Shape Reconstruction, *Periodica Polytechnica – Electrical Engineering* **55**

W.-D. Ueng, J.-Y. Lai, J.-L. Doong 1998 Sweep-surface reconstruction from three-dimensional measured data, *Computer-Aided Design* **30**, 791–805.

T. Várady, R. Martin 2002 Reverse Engineering, *G. Farin, J. Hoschek, M. S. Kim, Handbook of Computer Aided Geometric Design, Chapter 26*, Elsevier

Z. Zhang 1994 Iterative point matching for registration of free-form curves and surfaces, *International Journal of Computer Vision* **13**, 119–152.