A New Approach in CAD System for Designing Shoes

Simon Kolmanič and Nikola Guid

Faculty of Electrical Engineering and Computer Science, University of Maribor, Slovenia

The flattening of digitized surfaces is still very important in design of thin walled objects such as the airplane wings, parts of car bodies, textile products, and shoe uppers. Especially in shoe industry, the ability of quick respond to changing market needs is essential for successful competition. To give needed flexibility to a shoe designer, special CAD/CAM systems have been developed. Those systems are based on algorithms for surface reconstruction and surface flattening. In this article a fast algorithm for surface reconstruction and surface flattening is presented. Developable stripes are used to approximate a surface. In this way the surface can be flattened fast and without any distortions.

Keywords: surface flattening, developable surfaces, pattern engineering, digitized surfaces, shoe design.

1. Introduction

The needs on shoe market change today faster than ever. The shoe series are therefore smaller and the new shoe models have to be developed faster. The classical design of a new shoe is bound to the shoe last that is the form on which a shoe is constructed. The last gives the shoe a shape (see Figure 7a). To construct the appearance of a new shoe, the style lines are drawn on the shoe last. The style lines are geodesic curves that partition the shoe last surface in the 3D patches which have to be flattened in the plane to get cutting model for shoe uppers manufacturing. The flattening of these patches is therefore a reverse engineering process to assembling the leather parts for the shoe upper. This process is expensive and requires a well-trained developer. To make the design process faster and the production of new series cheaper, special CAD/CAM systems are used. The final result obtained by all CAD systems for the shoe design is a flat pattern based on the style lines, drawn on the shoe last. To generate flat patterns, the shoe last has to be digitized. The style lines can be drawn either on the real shoe last or later on the digitized one. If the style lines are drawn on the real shoe last, the lines have to be digitized, too. The designing of shoe uppers consists of two phases: reconstruction of digitized surfaces and flattening of these surfaces. An especially hard problem is flattening of the digitized surface since distortions like tearing and overlapping can occur in resulting flat pattern. Generally, an arbitrary surface cannot be unrolled into the plane without the distortions, since the distortions can only be completely eliminated if the surface is developable [Faux (1981)]. It is clear that the pattern designer wishes to reduce distortions to the minimum. This is very hard to do automatically, although some methods reducing the distortions significantly exist already [Parida (1993), Azariadis (1997)]. Especially serious problems in pattern generation are caused by overlaps in generated pattern [McCartney et al. (1999)]. Since deriving patterns from 3D surfaces is an old problem (first attempts to solve it automatically were made as early as in 1980 [Manning (1980)]), many methods have been developed already. We have divided them into two groups: methods for flattening the surfaces in one piece and methods for per partes surface flattening [Kolmanič (2002)]. The methods of the first group are not suitable for automatically flattening of arbitrary surfaces, since overlaps are eliminated from the generated pattern only if the surface is developable. The exceptions are some new methods [Gu et al. (2002), Zigelman et al. (2002)] that were developed for faster rendering of surfaces modelled with irregular triangle meshes and for non distorted texture map-

ping. Gu presents an interesting method to convert irregular triangular meshes into completely regular structure, called geometry images. The name is not given by accident since the surface geometry is presented by an image where standard compressing algorithms can be used. Zigelman's method uses a set of mathematical techniques called multi-dimensional scaling, to flatten the surface into plane where the texture is mapped. Both methods eliminate overlaps from the flat pattern, but the geodesic distances between points are not preserved. This is important if we want to use a generated pattern for composing 3D objects. To eliminate overlaps and preserve geodesic distances, methods of the second group are more appropriate [Bennis et al. (1991)].

Division of the surface into smaller patches is controlled by simple numerical parameters and does not require any additional interference by the user. The method presented in this article is based on a similar idea. Since the result of the digitization of a 3D surface is a cloud of points, the surface has to be reconstructed first. To do this, we have used developable stripes. After the surface reconstruction is completed, flattening is easy, since we only have to unroll the developable stripes into the plane. The idea for our method is not new. Originally, it was used by Elber [Elber (1995)] and by Hoschek to flatten the surfaces of revolution [Hoschek (1998)]. We have adopted the Hoschek idea to work with the surfaces obtained by digitization. The approximation process is controlled by only one numerical parameter and, therefore, it is very simple to use.

2. The View on Simplified System for the Shoe Design

The modern shoe design systems are based on the classical shoe design technique employing style lines, explained in the previous section. While drawing of style lines on the shoe last surface is simple, pattern construction is a demanding task that requires a skilled designer. Therefore the CAD/CAM systems for shoe design leave the shoemaker the freedom to draw style lines on the shoe lasts manually, but they automate the pattern engineering. To enable this, the CAD/CAM system for shoe design has to consist of a shoe last digitiser, a style line digitiser, a pattern engineering system, a cost analyser, and a shoe visualisation module. Additionally, a computerized sample cutter can be included.

Individual components of a shoe design system and their interdependence can be seen in Figure 1. Basic modules in a modern CAD/CAM system for shoe design are those for last digitising and pattern engineering. The result of the last digitising phase is a cloud of points in 3D space which has to be approximated by a 3D surface composed of the set of independent triangles [Fjalstrom (1993), Fong (1993), Hoppe et al. (1994), Oblonšek (1998)]. Before such a surface can be flattened, the neighbouring relations between those triangles have to be established. Although algorithms for the surface reconstruction are very complex, the user is not aware of it, because they are part of the last digitising

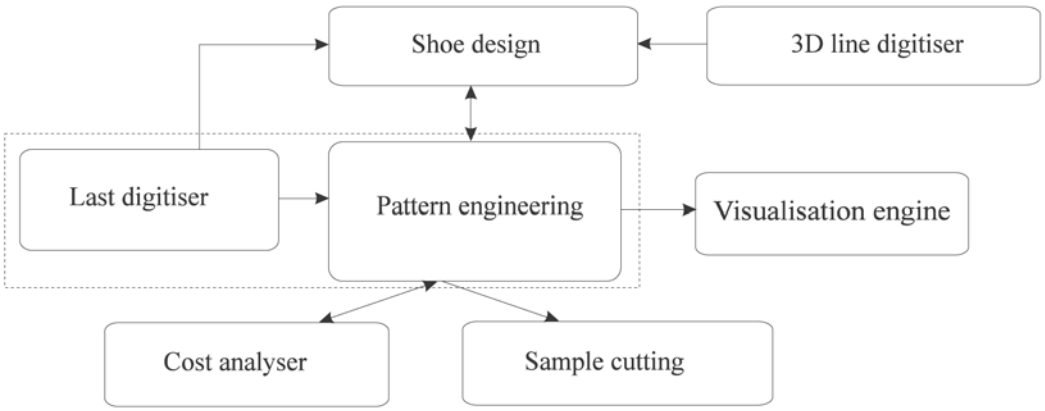
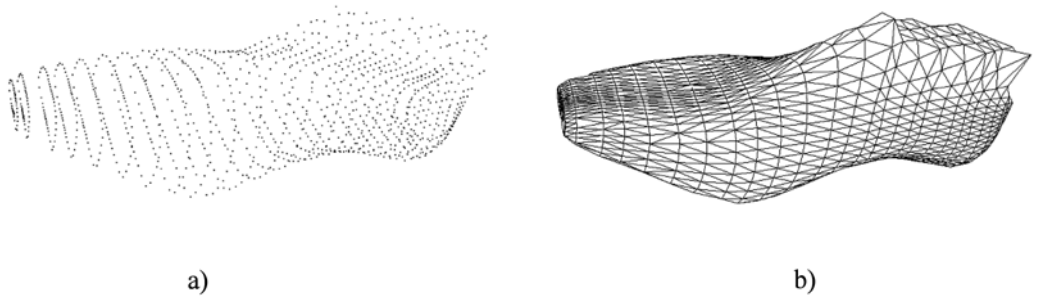


Fig. 1. Schematic view of a modern CAD/CAM system for shoe design.



a) b)

Fig. 2. a) Points obtained with the digitization device,
b) Triangulated surface as a result of an algorithm for surface reconstruction.

module. The result of the surface reconstruction, obtained by the [Oblonšek (1998)], can be seen in Figure 2.

To build the alternative system for shoe modelling, the problems of style lines drawing and surface flattening, which are the basic parts of the pattern engineering module, have to be solved. Currently, surface flattening is based on the definition of the Gaussian curvature on triangulated surfaces [Calladine (1984)], where overlapping of some parts of a flat pattern can occur. In [Azariadis (1997)] the method which minimizes the distortions can be found, although the distortions are not removed completely. To remove overlapping from the flat pattern, the approach of per partes surface flattening has to be considered. The cloud of 3D points is approximated by a set of developable stripes. This is possible because cross section curves can be found in the cloud of points.

Developable stripes are strained between two cross section curves. If the approximation precision is not within the tolerance limit, the developable stripe is split. It is done recursively by the divide-and-conquer strategy. The result of this procedure is a cloud of points approximated by a set of developable stripes. In the next section, the method for surface reconstruction by approximation with developable stripes is explained in more details.

3. Surface Reconstruction and Surface Flattening

We have already mentioned that the result of surface digitising is a cloud of points in 3D space, where the groups of points lie in parallel planes (the number of points in each plane is equal, see Figure 7b). If line segments connect the points in each plane in the order given by the digitising process, the cross section curves $\mathbf{c}^i$, $i = 1, 2, \ldots, N-1$, are generated. After the cross section curves are determined, the surface construction can start. Let $\mathbf{c}^s = \mathbf{c}^s(u)$ and $\mathbf{c}^f = \mathbf{c}^f(u')$ be two cross section-guiding curves, where u and u' are parameters of both curves. The equation of ruled surface defined by the curves $\mathbf{c}^s$ and $\mathbf{c}^f$ is given by [Gurunathan (1987)]:

$$\begin{aligned}\mathbf{s}\left(\mathbf{c}^s, \mathbf{c}^f\right) &= \mathbf{s}(u, u', v) \\ &= (1-v)\mathbf{c}^s(u) + v\mathbf{c}^f(u').\end{aligned} \tag{1}$$

The surface $\mathbf{s}(u, u', v)$ is defined by the set of lines connecting a point on the curve $\mathbf{c}^s$, say $\mathbf{p}^s$ with a point on the curve $\mathbf{c}^f$, say $\mathbf{p}^f$. Those lines are called linear generators and are not allowed to intersect. The surface is developable if for all linear generators the following condition is true:

$$\mathbf{n}^s \times \mathbf{n}^f = 0, \tag{2}$$

where $\mathbf{n}^s$ and $\mathbf{n}^f$ are the surface normals at points $\mathbf{p}^s$ and $\mathbf{p}^f$, which are calculated by the following equation:

$$\begin{aligned}\mathbf{n}^s &= \frac{d\mathbf{c}^s(u)}{du} \times \left(\mathbf{c}^f(u') - \mathbf{c}^s(u)\right) \text{ and} \\ \mathbf{n}^f &= \frac{d\mathbf{c}^f(u')}{du'} \times \left(\mathbf{c}^f(u') - \mathbf{c}^s(u)\right).\end{aligned} \tag{3}$$

Construction of the developable stripe is, therefore, a search for such points $\mathbf{p}^s$ and $\mathbf{p}^f$ that best

satisfy condition 2 [Kolmanič (2002)]. The surface reconstruction process starts with the generation of a developable stripe between cross sections $\mathbf{c}^1$ and $\mathbf{c}^N$. Then, the intersecting points between the generated developable stripe and other cross section planes have to be calculated (see Figure 3). If we strain, for example, the developable surface between the cross section curves $\mathbf{c}^b$ and $\mathbf{c}^f$ $(1 \leq b < f \leq N)$, the developable stripe $\mathbf{s}(\mathbf{c}^b, \mathbf{c}^f)$ intersects also the plane of cross section curve $\mathbf{c}^k$, $b < k < f$. The point of intersection $\mathbf{s}_j^k = \left[s_{x,j}^k\ s_{y,j}^k\ s_{z,j}^k\right]$ between j-th surface generator and the cross section plane can be calculated by the equations bellow:

$$\begin{aligned} s_{x,j}^k &= -\frac{Ba + Cd + D}{A + Bl + Ch}, \\ s_{y,j}^k &= ls_{x,j}^k + a, \\ s_{z,j}^k &= hs_{x,j}^k + d, \end{aligned} \tag{4}$$

where A, B, C, and D are numerical values that solve the following equation:

$$\begin{aligned} &\left((\mathbf{p} - \mathbf{c}_1^k) \times (\mathbf{c}_2^k - \mathbf{c}_1^k)\right)(\mathbf{c}_3^k - \mathbf{c}_1^k) \\ &\quad = Ax + By + Cz + D. \end{aligned} \tag{5}$$

The points $\mathbf{c}_1^k$, $\mathbf{c}_2^k$ and $\mathbf{c}_3^k$ are the three most distant points from k-th cross section curve and $\mathbf{p} = [x\ y\ z]$ is an arbitrary point in space. The values l, h, a, and b in Eq. 4 are the parameters of j-th linear generator that can be calculated by the following equation:

$$\begin{aligned} l &= \frac{s_{y,j}^f - s_{y,j}^b}{s_{x,j}^f - s_{x,j}^b}, \quad h = \frac{s_{z,j}^f - s_{z,j}^b}{s_{x,j}^f - s_{x,j}^b}, \\ a &= s_{y,j}^b - ls_{x,j}^b, \quad \text{and} \quad d = s_{z,j}^b - hs_{x,j}^b. \end{aligned} \tag{6}$$

The points $\mathbf{s}_j^b = \left[s_{x,j}^b\ s_{y,j}^b\ s_{z,j}^b\right]$ and $\mathbf{s}_j^f = \left[s_{x,j}^f\ s_{y,j}^f\ s_{z,j}^f\right]$ are j-th points of the cross section curves $\mathbf{c}^b$ and $\mathbf{c}^f$, respectively. Intersection points have to be calculated for all linear generators.

Error vector $\mathbf{e} = [e_x\ e_y\ e_z]$, calculated by the following equation, expresses the difference between the cross section curve and the intersection points:

$$\mathbf{e} = \frac{1}{n}\sum_{j=1}^{n} |\mathbf{s}_j^k - \mathbf{c}_j^k|, \tag{7}$$

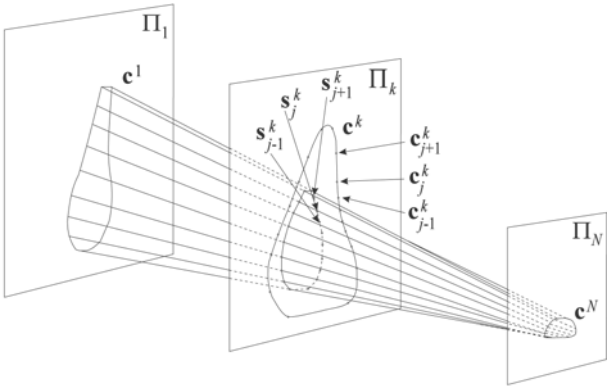


Fig. 3. Reconstruction of a surface with developable stripes.

where n is the number of points on the cross section curve.

Calculation of Eq. 7 is simple and, thus, it is a fast criterion for an approximation error. A developable stripe has to be narrowed if the following condition is true:

$$e = \frac{e_x + e_y + e_z}{3} > \varepsilon, \tag{8}$$

where ε is a tolerance limit. If the developable stripe is narrowed, the new cross section curve $\mathbf{c}^w$ has to be found. The index of the new guiding curve is calculated by the following equation:

$$w = \frac{b + f}{2}. \tag{9}$$

As the result, two stripes $\mathbf{s}(\mathbf{c}^b, \mathbf{c}^w)$ and $\mathbf{s}(\mathbf{c}^w, \mathbf{c}^f)$ are generated. The procedure is repeated until the developable stripes are within the tolerance limit.

The described procedure is easy to implement, which can be seen in Figure 4 where the algorithm for surface reconstruction is presented.The algorithm for surface reconstruction calls for two additional procedures. The first generates the developable stripe between two given guiding curves and it is explained in more details in [Kolmanič (2002)]. The second procedure calculates the approximation error vector.

Since the developable stripe has to be divided if the condition (Eq. 8) is true, there is no need to calculate the entire error vector if the tolerance limit is overdrawn. To reduce the error calculation time, we used the algorithm presented in Figure 5.

```
Divide(DBegin, DEnd)
  Input:  index of guiding curves
  Exit:    set of developable stripes

begin
  Generate developable stripe between cross sections DBegin and DEnd.

  if CurveDifference(DBegin, DEnd) > ε then
     begin
       if (DEnd-DBegin) == 1 then
         Save developable stripe in a set of developable stripes.
       else
         begin
            Divide(DBegin, (DEnd+DBegin)/2);
            Divide((DEnd+DBegin)/2, DEnd);
         end
     end
  else
     Save developable stripe into a set of developable stripes.
end
```

Fig. 4. Surface reconstruction algorithm.

```
double CurveDifference(DBegin, DEnd)
  Input:  indexes of guiding curves
  Exit:   difference between the cross section curves and their
          approximation

begin
  Midd = (DBegin+DEnd) div 2;
  Diff = CalculateDifference(Midd); //use eq.  8

  if Diff == 0 then
     return 0

  if Diff > ε then
     return Diff

  Diff = (Diff + CurveDifference(DBegin Midd))/2;
  Diff = (Diff + CurveDifference(Mid, DEnd))/2;

  return Diff
end
```

Fig. 5. Algorithm for optimised calculation of error vector.

After the surface is reconstructed, it has to be unrolled to the plane. Unrolling of the developable surface is, in fact, mapping of the 3D quadrangles into the plane, where geodesic distances are preserved. The mapping process must be fast and accurate. The quadrangles constructed in the plane must have the same length as the original 3D quadrangles. It turns out that the quadrangle construction is closely connected with the calculation error. Since calculation errors are accumulated and if the developable stripe consists of many quadrangles, the final error cannot be ignored. Therefore, we have to use the construction method presented in [Kolmanič (2002)]. Flattening of the developable stripe is finished after all the quadrangles of the stripe are constructed in the plane and the neighbouring relation is preserved. To do this, we distinguish two flattening directions: the primary and the secondary directions. Since the surface is controlled by two parameters, u and v, each of them can be chosen as primary or secondary flattening direction. The form of a generated flat pattern depends upon the primary direction of flattening and the starting position. In our case the primary flattening direction has been u. Each quadrangle has been constructed independently from its neighbour and then translated and rotated in order to preserve the neighbouring relation. The results of surface reconstruction and surface flattening are presented in the next section.

4. Examples of Surface Flattening

In this section we present the result of surface reconstruction and surface flattening on two test objects.

The first object is a computer-generated vase and it is a simple surface of revolution (see Figure 6a). It consists of 888 points. The second surface is a digitized real object shoe last with 3584 points. The vase was approximated with 16 developable stripes, where the value of the tolerance limit was $\varepsilon = 1.50$. We needed 0.22s

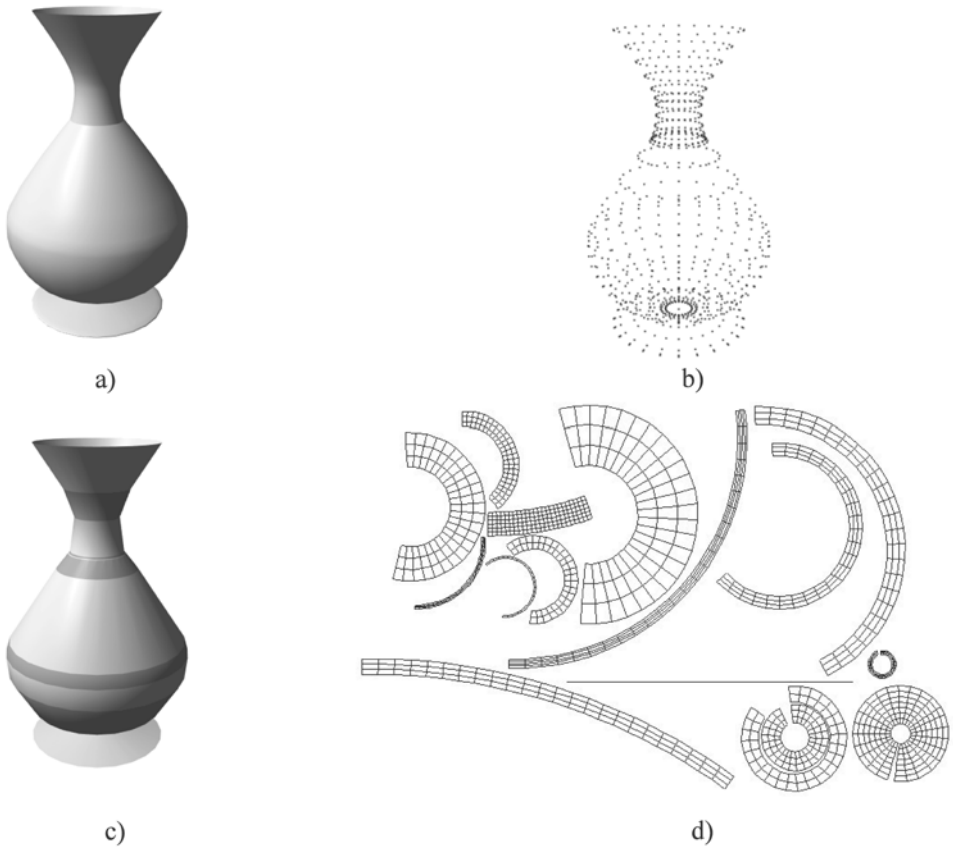


Fig. 6. a) Computer-generated test object (vase), b) Cloud of 3D points obtained with the test object, c) Reconstructed surface of the vase with the tolerance limit $\varepsilon = 1.50$, d) Result of the flattening of the vase.

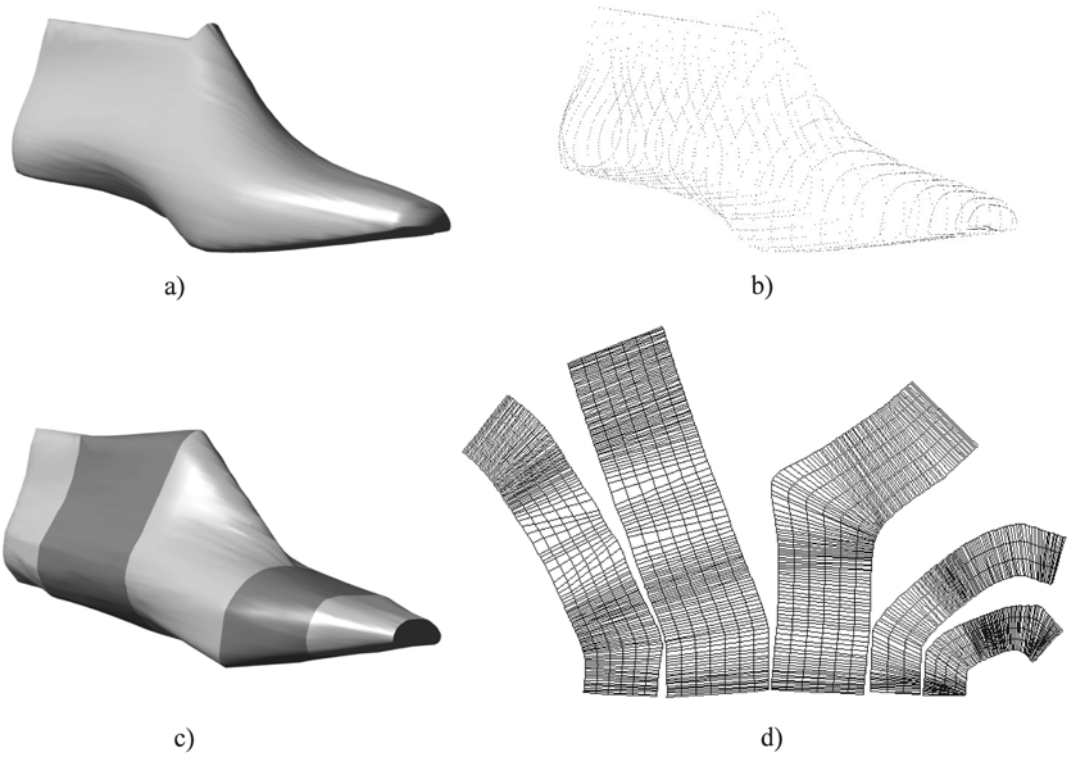


Fig. 7. a) Real test object (shoe last), b) Cloud of 3D points obtained by digitizing of the shoe last, c) Reconstructed shoe last with the tolerance limit $\varepsilon = 1.50$, d) Result of the shoe last flattening.

for its reconstruction and flattening. In Figure 6c we can see that the reconstructed surface is a good approximation of original surface.

Figure 7c shows that the approximation of the shoe last is not very good. To improve this, a smaller value for the tolerance limit must be selected.

The algorithm was tested on Athlon 900 MHz personal computer.

5. Conclusion

In this article we have presented the new method for reconstruction and flattening of digitized surfaces, which is based on the divide-and-conquer strategy by approximation of the surface with developable stripes. The method is fast but it cannot be used for reconstruction of objects with holes. To use the method in CAD systems for shoe design, the method for style lines mapping has to be developed first and the flattening method has to be modified to use those lines. In future, new methods have to be developed for pattern engineering, for elastic plane material and for the best positioning of particular stripes in order to minimize the waste of plane material. Although there is much work to be done, the described method represents a good start for an alternative flat pattern engineering system.

References

[1] P. AZARIADIS AND N. ASPARAGATHOS, Design of Plane Developments of Doubly Curved Surfaces, *Computer-Aided Design*, Vol. 29, No. 10, (1997), pp. 675–685.

[2] C. BENNIS, J.M. VEZIEN AND G. INGLESIAS, Piecewise surface flattening for non-distorted texture mapping, *Computer Graphics*, Vol. 25, No. 4, (1991), pp. 237–246.

[3] C.R. CALLADINE, Gausssian Curvature and Shell Structure, *Proceedings of the conference Mathematics of Surfaces*, (1984), pp. 179–196.

[4] G. ELBER, Model Fabrication Using Surface Layout Projection, *Computer–Aided Design*, Vol. 27, No. 4, (1995), pp. 283–291.

[5] I.D. FAUX AND M.J. PRATT, Computational geometry for design and manufacture, *Chichester: Ellis Harwood*, 1981.

[6] P.O. FJALSTROM, Evaluation of a Delaunay-based Method for Surface Approximation, *Computer-Aided Design*, Vol. 25, No. 11, (1993), pp. 711–719.

[7] P. FONG AND H.P. SEIDEL, An Implementation of Triangular B-spline Surfaces over Arbitrary Triangulations, *Computer–Aided Geometric Design*, Vol. 10, (1993), pp. 267–275.

[8] B. GURUNATHAN AND S.G. DHANDE, Algorithms for Development of Certain Classes of Ruled Surfaces, *Computers & Graphics*, Vol. 11, No. 2, (1987), pp. 105–112.

[9] X. GU, S. GORTLER AND H. HOPPE, Geometry images, *ACM SIGGRAPH*, (2002), pp. 355–361.

[10] H. HOPPE, et al., Piecewise Smooth Surface Reconstruction, Proceedings of SIGGRAPH'94, *ACM SIGGRAPH*, (1994), pp. 295–302.

[11] J. HOSCHEK, Approximation of Surface of Revolution by Developable Surfaces, *Computer–Aided Design*, Vol. 30, No. 10, (1998), pp. 757–763.

[12] S. KOLMANIČ AND N. GUID, The Flattening of Arbitrary Surfaces by Approximation with Developable Stripes, in: U. CUGINI AND M. WOZNY, editors, *From Geometric Modeling to Shape Modeling*, Kluwer Academic Publishers, Boston, (2002), pp. 35–46.

[13] J.R. MANNING, Computerized pattern cutting methods based on isometric tree, *Computer-Aided Design*, Vol. 12, No. 1, (1980), pp. 43–47.

[14] J. MCCARTNEY, B.K. HINDS AND B.L. SEOW, The Flattening of Triangulated Surfaces Incorporating Darts and Gussets, *Computer-Aided Design*, Vol. 31, No. 4, (1999), pp. 249–260.

[15] Č. OBLONŠEK AND N. GUID, A Fast Surface-Based Procedure for Object Reconstruction from 3D Scattered Points, *Comput. Vis. & Image Underst.*, Vol. 69, No. 2, (1998), pp. 185–195.

[16] L. PARIDA AND S.P. MUDUR, Constraint–satisfying Planar Development of Complex Surfaces, *Computer–Aided Design*, Vol. 25, No. 4, (1993), pp. 225–232.

[17] G. ZIGELMAN, R. KIMMEL AND N. KIRYATI, Texture Mapping Using Surface Flattening via Multidimensional Scaling, *IEEE Transactions on Visualization and Computer Graphics*, Vol. 8, No. 2, (2002), pp. 198–207.

Received: March, 2002
Revised: July, 2003
Accepted: October, 2003

Contact address:
Simon Kolmanič
Laboratory for Computer Graphics and Artificial Intelligence
Faculty of Electrical Engineering and Computer Science
University of Maribor
Smetanova 17
SI-2000 Maribor, Slovenia
Phone: + 386 2 2207475
e-mail: simon.kolmanic@uni-mb.si

SIMON KOLMANIČ received his BSc and MSc degrees in computer science from the Faculty of Electrical Engineering and Computer Sciences of the Maribor University in 1996 and 1999 respectively, and is currently a PhD student. He is a teaching assistant at the Faculty of Electrical Engineering and Computer Science of the Maribor University. His current research interests include computer graphics, computer-aided geometric design and computer animation.

NIKOLA GUID is presently a full professor at the Faculty of Electrical Engineering and Computer Sciences of the Maribor University (Slovenia), and the head of the Laboratory of Computer Graphics and Artificial Intelligence. His current research interests are computer graphics, computer-aided geometric design, and geometric modeling. Guid received his BSc degree in electronics and MSc degree in computer science from the University of Ljubljana, Slovenia, and PhD in technical sciences from the University of Maribor. In 1991 and 1993, each time for the period of six months, he was involved in the research project conducted at the Faculty of Engineering of the Udine University in Italy. He is a member of the IEEE and Eurographics.