Computers in Industry 64 (2013) 1290–1300

Contents lists available at SciVerse ScienceDirect

Computers in Industry

journal homepage: www.elsevier.com/locate/compind

3D geometry reconstruction from orthographic views: A method based on 3D image processing and data fitting

Lapo Governi *, Rocco Furferi, Matteo Palai, Yary Volpe

Department of Industrial Engineering, University of Florence, Via di Santa Marta 3, 50139 Firenze, Italy

ARTICLE INFO

Article history:
Received 31 August 2012
Accepted 1 February 2013
Available online 19 March 2013

Keywords:
3D reconstruction
Orthographic views
Voxel imaging
3D geometry fitting
Industrial design

ABSTRACT

Industrial esthetic designers typically produce hand-drawn sketches in the form of orthographic projections. A subsequent translation from 2D-drawings to 3D-models is usually necessary. This involves a considerably time consuming process, so that some automation is advisable.

Common approaches to this "reconstruction problem" start directly from "exact" 2D vector representations or try to vectorize 2D raster images prior to the reconstruction phase. These approaches, however, typically fail to deal with free form geometries like the ones commonly found in esthetic industrial design.

This work presents a new methodology suitable for free form geometries, comprising the generation and processing of a 3D voxel image obtained from a hand drawing, the creation of a set of 3D curves fitting the voxel image and the automatic generation of surface patches on the resulting curve network.

Several case studies are also presented in order to emphasize and discuss strengths and weaknesses of the proposed method.

1. Introduction

Computer Aided Design systems are deemed to be essential for all the phases characterizing the design and the development of a new industrial product. However, especially for products characterized by a strong stylistic content, handmade drawings are often preferred to CAD software packages.

In the early stage of the development of a new product, esthetic designers typically produce a rich set of sketches (usually in the form of orthographic projections) to develop and communicate their ideas. Product-managers often have to select stylistic alternatives based on a set of hand-drawings depicting possible solutions. However, it is much more effective to base the selection on a virtual 3D model which conveys far more information. In fact, some hand-drawn alternatives are even "translated" into 3D models (and possibly rendered models) capable to provide a more realistic view of the object and to allow a deeper analysis of the design intent.

The translation process, involving a close interaction of esthetic designers and CAD operators in order to produce a CAD model carefully representing the designer's intent, is known to be considerably time consuming. Accordingly, the design alternatives available in the form of three-dimensional models result to be a very small subset of those developed by the designers (usually one or, at most, two).

The common methodology (Fig. 1) used to turn a set of orthographic projections into a 3D model starts from arranging the scanned images of the hand-drawn views on the correspondent orthogonal planes in a CAD software environment. Using such images, the CAD operator (often assisted by the designer) manually redraws the style lines in order to obtain the 3D wireframe model which is, eventually, used as a support frame for the definition of the final surfaces.

The automation of this process, i.e. the 3D retrieval from 2D drawings (known as "reconstruction problem"), is a key target for commercial software houses as well as a vigorous focus from an academic outlook.

Recently a set of software tools have been released by major software houses, like Dassault Systemes®, Autodesk® and PTC®, which support the CAD operators in some of the reconstruction process phases. These tools, however, entail a strong user interaction and only marginally speed up the process. In addition, most of them require as an input an "exact" set of 2D vector drawings (e.g. DXF or IGES files).

From the academic point of view, a number of works have been proposed since the first '70s, providing a series of methodologies for solving the reconstruction problem starting from an "exact" set

* Corresponding author. Tel.: +39 0 554796396; fax: +39 0 554796400; mobile: +39 3480198491.
E-mail addresses: lapo.governi@unifi.it (L. Governi), rocco.furferi@unifi.it (R. Furferi), matteo.palai@unifi.it (M. Palai), yary.volpe@unifi.it (Y. Volpe).

0166-3615/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.compind.2013.02.003

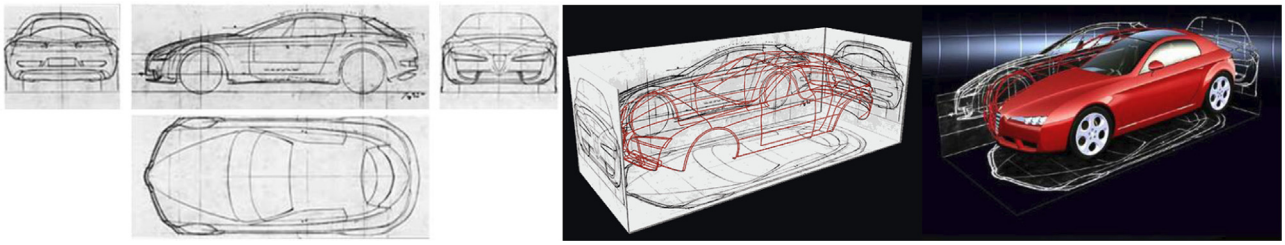


Fig. 1. Typical 2D to 3D "translation" process.
courtesy of Italdesign-Giugiaro

of 2D vector drawings. A dramatic boost to the research on this topic was provided by Wesley and Markowsky in their studies [1,2] which are probably the best known works among the researchers working on 3D reconstruction. Wesley and Markowsky provided a comprehensive set of guidelines for reconstructing 3D CAD model starting from orthographic projections. Their procedure is, still today, a milestone for almost every reconstruction study. Moving from these guidelines, many works have been developed, particularly referring to the 3D reconstruction from technical (industrial and/or architectural) vector drawings. Most of them have been conceived for retrieving 3D models starting from 2D straight edges, arcs or, at most, conics [3–7].

In addition, a few approaches have been developed dealing with the reconstruction problem based either on a set of 3D features which can be possibly found in a technical drawing (e.g. revolutions and extrusions) or on information coming from additional views, like cross-sections. The works which confront the reconstruction problem for curvilinear objects, however, usually present quite "tricky" approaches, generally involving a considerable amount of user interaction [8,9].

The most significant limitation to the approaches cited so far is that, as already mentioned, in all the cases the reconstruction is based on "exact" vector drawings and is limited to specific kinds of geometries.

Moving from these considerations, this work is meant to discuss a methodology to perform a quasi-automatic "translation" starting from a 2D scanned image of hand-drawn orthographic views.

Though this work and the ones mentioned above share the same goal (3D reconstruction from planar views), the starting point is completely different, since it consists of raster data from inherently inexact drawings like the ones, mentioned at the beginning of this section, coming from the initial design phases. Moreover, as detailed in the methodology description, the presented approach is applicable regardless of the typologies of geometries represented in the starting drawings, and is particularly well suited for free form geometries like the ones commonly found in esthetic industrial design.

The paper is organized as follows.

In Section 2 the reconstruction methodology is presented with reference to an exemplificative case study; in Section 3 additional reconstruction examples are presented in order to demonstrate the robustness of the proposed method; strengths and weaknesses of the presented approach depending on possible usage scenarios are also discussed; in Section 4 a few concluding considerations are drawn and possible options for future developments are briefly hinted at.

2. Methodology

As mentioned in Section 1, the proposed methodology is based on the processing of a single 2D raster image (e.g. bmp or jpg file) depicting the orthographic views of the object whose 3D geometry is to be retrieved.

The method is structured according to a set of phases which can be summarized in the following list.

1. Building a wireframe-like voxel cloud (3D image) representing a raster model of the original object.
2. Separating branches (3D labeling) of the resulting 3D image by means of approximate intersection detection.
3. Obtaining a 3D wireframe model made of spline curves starting from each labeled entity of 3D image and, successively, refining the obtained curves in the intersection zones.
4. Building a surface representation of the object by using surface patches attached to the spline curve network.

Each of these main phases will be detailed in the rest of this section, with reference to an exemplificative case study, by splitting them in procedural steps.

2.1. Building a wireframe-like voxel cloud (3D image) representing a raster model of the original object

Step 1 – The Image of the hand-drawn sheet depicting the orthographic views is acquired by means of a flatbed scanner (grayscale – 8 bit). The scanning resolution needs to be selected so that separate lines in the drawing remain separate in the resulting digital image (Fig. 2). In practice, this requires to have at least 3–4 "white" pixels separating each drawing line. Typically, an indicative resolution value can be 150 dpi for a line thickness drawing in the range 0.5/1.0 mm.

Step 2 – The orthographic views in the resulting image are identified and the sheet orientation is automatically compensated. In this step, it is assumed that the views are correctly arranged in the original drawing (according either to the first-angle or to the third-angle projection rule) and separated by means of continuous lines representing the mutual intersections of the planes on which the views are projected.

The sheet orientation is compensated by identifying the two orthogonal lines in the image characterized by maximum length (reference lines). This is done applying the well-known Hough transform to the thresholded image, where the threshold value t_s must be manually selected in order to obtain a clear binary representation of the drawing. First, a slope range of $\pm 10^\circ$ (0.1° step) with respect to the horizontal direction is selected and the best scoring line is considered (slope α_h); secondarily the best scoring line with a slope in the range $\alpha_{v^*} \pm 5^\circ$, where $\alpha_{v^*} = \alpha_h + 90^\circ$. The slope of this new line is labeled as α_v. The intersection point of the two lines is identified and is considered to be the origin O of the set of orthographic views (Fig. 3). Finally, the original grayscale image is rotated around O using bi-cubic resampling, so that the line characterized by the highest overall score results to be horizontal (either α_h or α_v is set to zero).

Afterwards, the separate views in the rotated image are identified by isolating the four quadrants delimited by the horizontal line and the origin O.

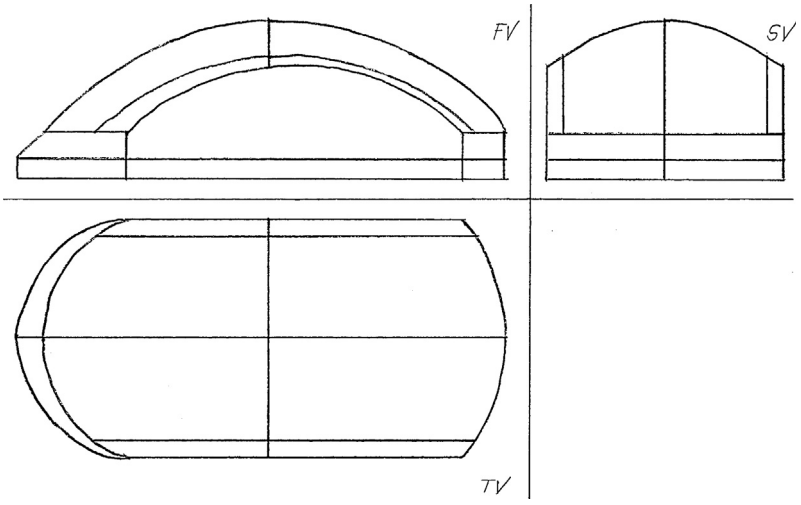


Fig. 2. Scanned image (grayscale).

Each view is labeled as front view (FV), side view (SV) or top view (TV) according to its position with respect to the origin O and to the empty quadrant.

In this step the bounding boxes dimensions and positions (in pixel) are also computed for each orthographic view. This is done by computing, for each view, the bounding box of the connected component characterized by the maximum number of pixels (Fig. 4).

Step 3 – The images contained in each single bounding box are separately considered and, since the dimensions in the different views need to be coherent, they are scaled (bi-cubic resampling is used) to compensate for possible discrepancies. Once the 2D bounding boxes have been adjusted, it is possible to identify the 3D bounding box of the object represented in the original drawing.

The set of resulting images, after a new thresholding, undergo a weighted, iterative dilation process using a 3×3 structuring element (all values set to 1).

Assuming a total of n_s dilation iterations are used, the new white pixels, obtained at each dilation iteration i, are multiplied by a weighting coefficient d_i defined as follows.

Note that all the dilation iterations are applied on the binary image obtained at the previous one. Only after all iterations are performed, the weights are applied.

The result for an exemplificative value of $n_s = 2$ is shown in Fig. 5.

The number of iterations to be used in the process is set according to the expected error in correspondence among the projections (for additional details see also the description of Step 5).

Step 4 – The new grayscale images obtained in Step 3 are extruded orthogonally to their image plane to cover the entire 3D bounding box computed in Step 2. A set of 3D images is, thereby, obtained with each voxel having a real value between 0 and 255.

Step 5 – The entire set of 3D images, after normalization, is multiplied element by element so to obtain a final 3D image G, where the object outlines are represented by a 3D wireframe-like voxel cloud. A given region of the 3D object outline is correctly reproduced in G if sufficiently good matching is found between the corresponding 2D outlines in the original views (perfect match: $G(i,j,k) = 1$; partial match: $0 < G(i,j,k) < 1$; no match: $G(i,j,k) = 0$). According to this definition, the amount of acceptable error in order to identify a 3D region by matching the 2D views, is directly related to the number of dilation steps n_s which, as a consequence, needs to be selected suitably. In Fig. 6a and b the resulting 3D images obtained respectively by setting $n_s = 0$ (insufficient) and $n_s = 2$ (sufficient) are shown.

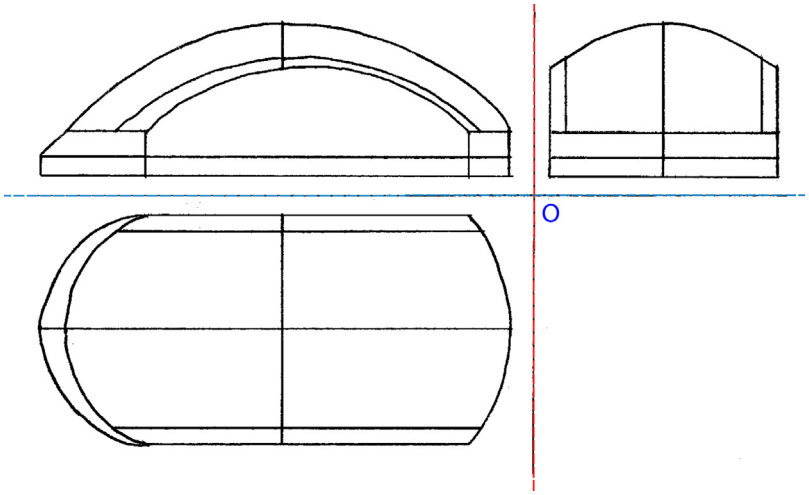


Fig. 3. Reference horizontal (light blue) and vertical (red) lines and rotation center "O". (For interpretation of the references to color in this figure legend, the reader is referred to the web version of the article.)

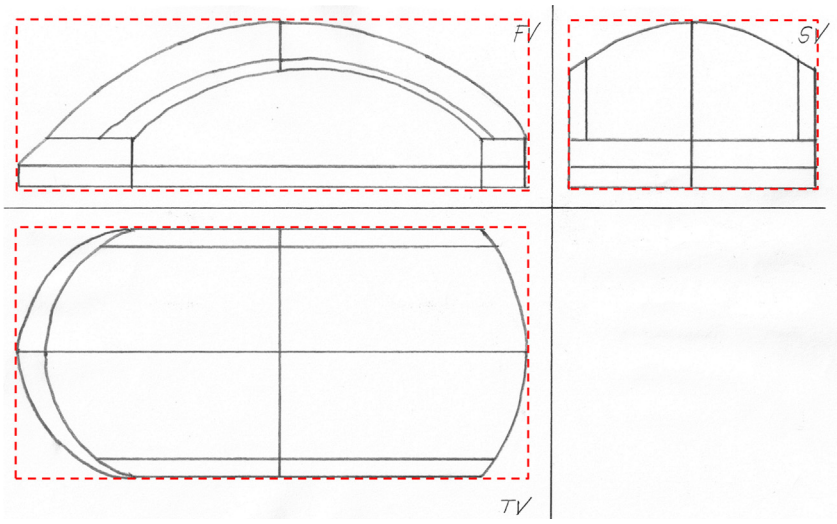


Fig. 4. Bounding boxes for the three orthographic views.

2.2. Separating branches by means of approximate intersection detection

Step 6 – Image G is converted into a binary image T ($T(i,j,k) = 1$ if $G(i,j,k) \neq 0$) which undergoes a morphological closing (with a $3 \times 3 \times 3$ unitary kernel) to remove possible small cavities. The resulting voxel cloud is transformed into a triangular mesh, which iteratively undergoes a Laplacian smoothing process according to the approach described in [10]. Due to the way the voxel model is originated, the one obtained is a closed mesh; as a consequence, the well-known problem of "disappearing" mesh regions due to mesh shrinkage, does not occur, so that no particular constraint need to be considered when performing the smoothing process.

The smoothing progress on an exemplificative region of the mesh obtained from image T is shown in Fig. 7.

Step 7 – The resulting triangular mesh is characterized by a large number of spikes (situated in the regions far from the intersections) and a few approximately equilateral triangles (situated in the regions where multiple "wires" intersect). In order to spot the approximate intersections, triangles' form factor f (i.e. the ratio between the height relative to maximum length edge and the edge itself) is analyzed and triangles with $0.5 < f < 1$ are selected. The centroids c_i of each connected group of such triangles are assumed to be a first guess for the intersection points. In Fig. 8 the detail of an intersection region is shown; the color map highlights the differences in triangles' form factor.

Step 8 – For each centroid c_i, the equation of a sphere E_i is computed so that it is centered on c_i and has a radius proportional to the estimated "wire" thickness in the dilated voxel image. In this work this parameter must be directly input and can be roughly estimated from the knowledge of the line thickness in the original 2D drawing, the scanning resolution, and the number of dilation steps n_s performed in Step 3. The spheres are assumed to represent the regions where the wires' intersections lie. Accordingly, in order to separate the "branches" composing the wireframe-like voxel model, image T is "cut" by means of spheres E_i by setting $T(i,j) = 0$ for all the voxels lying inside any of the spheres. The resulting image is characterized by the presence of a set of connected components representing the separated "branches" of T.

In some cases (Fig. 9a) two or more centroids may originate in very close positions. If the respective spheres intersect, the centroids are assumed to represent a single intersection zone. In such a case, the voxels contained in the intersecting spheres are analyzed by means of *PCA* (principal component analysis) and the principal directions are used to compute a single ellipsoid used for "cutting" image T in the place of E_i. Ellipsoid dimension is set so that the minor axis equals E_i diameter (Fig. 9b).

Step 9 – The binary 3D image resulting from Step 8 is labeled by searching for its connected components. For each labeled region, the corresponding one is extracted from image G (i.e. the set of voxels of G characterized by the same coordinates of the labeled region) and stored in a separate 3D image.

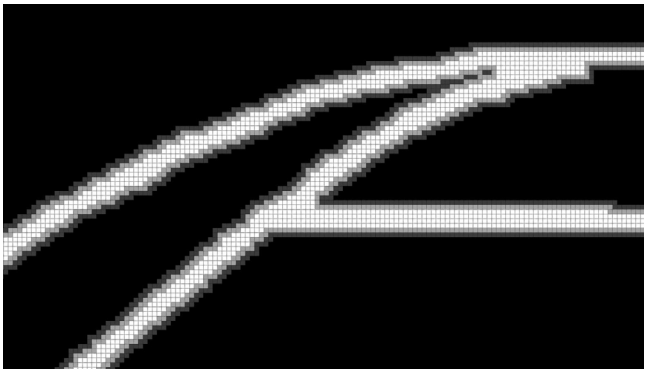


Fig. 5. Detail of the image, obtained by the iterative dilation process.

2.3. Obtaining a 3D wireframe model made of spline curves, starting from each labeled entity of 3D image

Step 10 – Each separate branch is fitted by means of a spline curve according to a process based on the use of weighted least squares. The fitting needs to be performed on an unordered voxel cloud, so a suitable approach has to be used. The basic idea is to:

- define a 3D polyline approximately following the voxel branch medial axis (Fig. 10a);
- fit the polyline knots with an interpolating spline curve (cubic) used for ordering the voxels by evaluating the order of their projection onto the spline (Fig. 10b);
- perform a final weighted least square fitting with a new approximating spline curve (cubic) using the voxel order identified in the previous step (Fig. 10c).

The weight associated to each voxel is assumed to be represented by the voxel value itself; as a consequence, the resulting spline curves tend to be well superimposed to the voxel

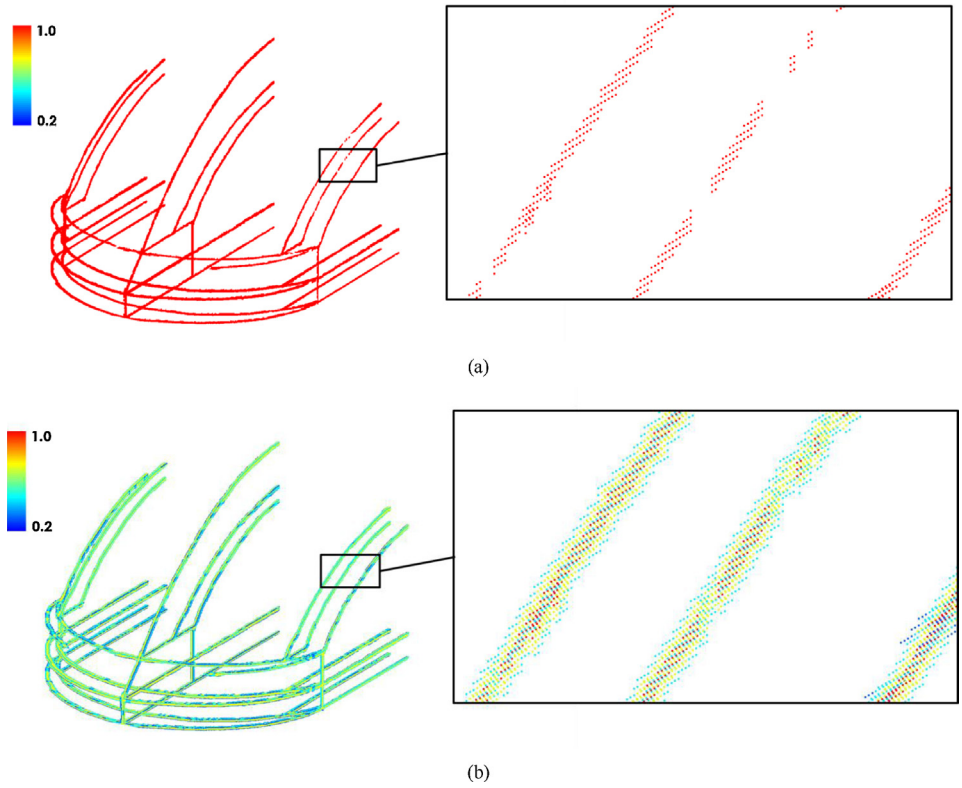


Fig. 6. Voxel images obtained with different values of n_s (colormap indicates the voxel value).

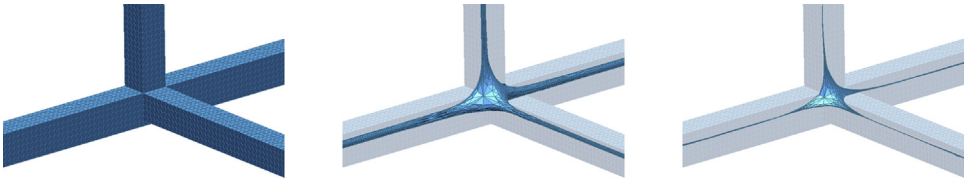


Fig. 7. Laplacian smoothing progress (detail of an intersection region).

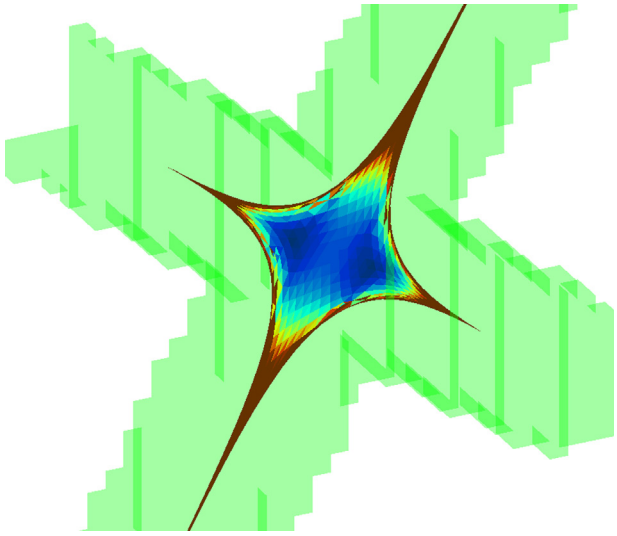


Fig. 8. Triangular mesh in the proximity of an intersection region.

cloud regions with voxel values ≈ 1. As explained in Step 5, these are the regions originated by well matching 2D projections.

The described fitting method is a 3D extension of the one described by the authors in [11], to which the reader is referred for a detailed description.

Step 11 – Once the fitting process is complete, final intersection points are computed by means of the following procedure for each cutting region (sphere or ellipsoid).

- The (spline) curves, fitting the branches which were separated by the analyzed cutting surface, are considered.
- For each curve, the point where the curve intersects the cutting entity is computed and the curve is extended from it along its tangent direction in such a way that the extension segment is delimited by the cutting surface.
- Due to their definition, the extension segments never match in a common point (which would be the intersection one); as a consequence the most likely intersection point must be identified. Since it cannot lie outside of the voxel region delimited by the cutting surface, it is possible to limit the search domain by only considering the centroids of such voxels as candidate intersection points. The voxel centroid with minimum value of the sum of square distances from extension segments is considered to be the intersection point (Fig. 11).

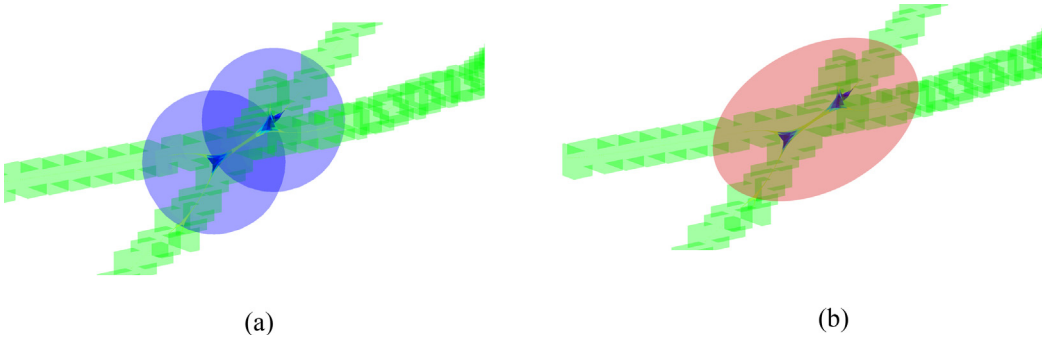


Fig. 9. Intersecting cutting spheres (a) and resulting cutting ellipse (b).

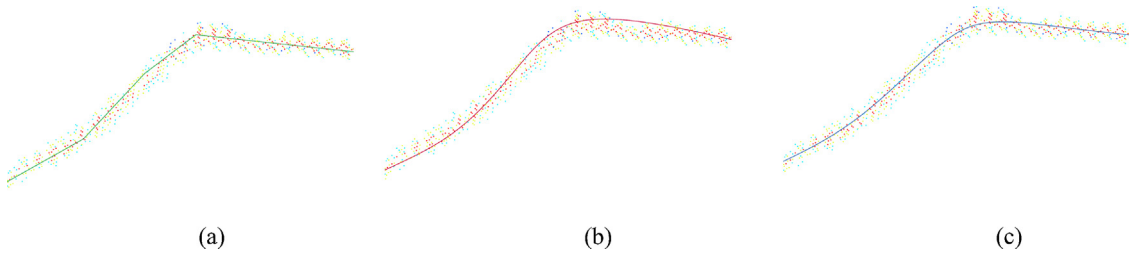


Fig. 10. Initial polyline (a); ordering curve (b); and final curve approximating the voxels (c).

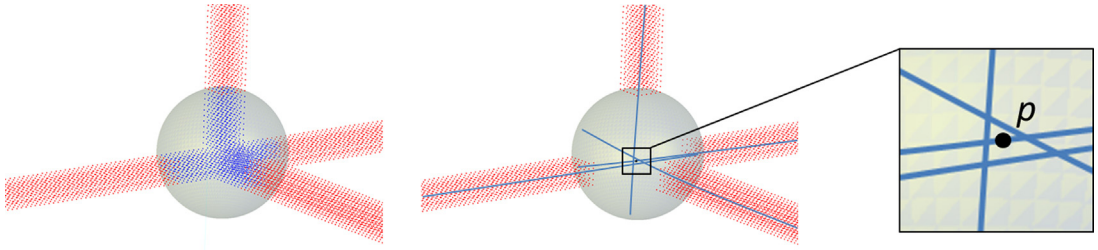


Fig. 11. Identification of the intersection point (p).

Note that the above described procedure introduces a slight approximation in considering only integer coordinates, but allows to identify the intersection point by examining a finite number of candidates.

Step 12 – Each voxel branch is re-fitted using the same procedure described in Step 10 with the preliminary introduction of a set of boundary conditions so that the fitting curve endpoints coincide with the intersections found in Step 11.

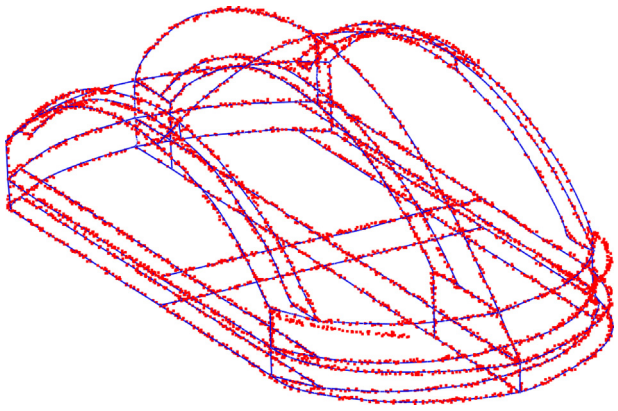


Fig. 12. Final network of spline curves superimposed on the voxel image (1% of non-zero voxels are plotted).

Step 13 – The resulting set of spline curves is reassembled in a single model which represents the vector pseudo-wire frame of the original object. The model can be translated into a DXF or IGES file and is ready to be used for surface generation in a 3D CAD environment (Fig. 12).

2.4. Building a surface representation of the object by using surface patches attached to the spline curve network

Step 14 – The edges of the wire frame model can be used as support entities to define a set of surface patches describing the geometry of the object represented in the original drawing. This phase can be performed either manually, importing the vector file (e.g. DXF and IGES) in a surface modeling software environment and selecting edge cycles defining each single surface patch, or automatically, using an approach based on the analysis of the graph made up of curves and intersections. In order to implement and test a procedure capable of performing an automatic surface generation, the work by Bagali and Waggenspack [12] has been considered. In their study they define a graph representing the wire frame model of an object, where nodes are the intersection points and edges are the curves making up the model itself. Accordingly, the problem of identifying the set of curves delimiting a single surface patch is translated into the one of determining a cycle in the original graph. In this work, while graph exploration in order to identify candidate cycles has been performed using Bagali and Waggenspack's approach (based, in its turn on the well-known

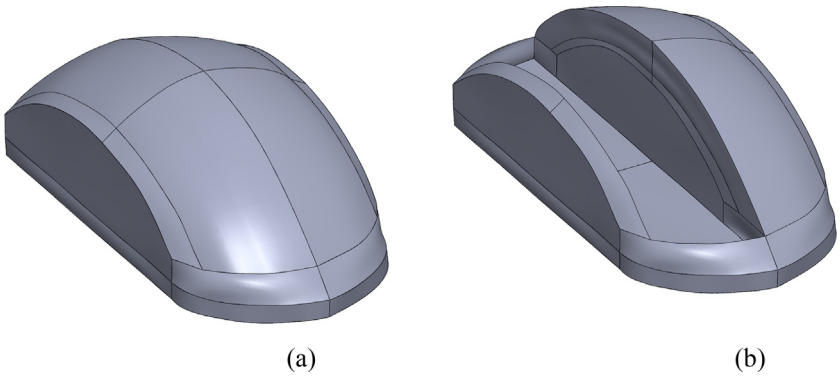


Fig. 13. Complete surface model (a); and detail of the inner surfaces (b).

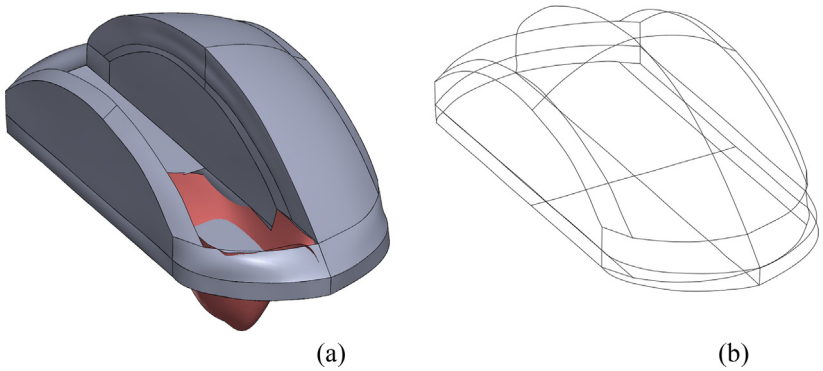


Fig. 14. Surface patch (in red) due to an incorrect cycle definition (a); and simplified curve network to be used in order to minimize the probability of wrong cycle definition (b). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of the article.)

Dijkstra's algorithm [13]), the identification of optimal cycles (i.e. the ones more likely representing a "correct" surface patch of the original object) has been performed using the approach recently described by Abbasinejad et al. in [14].

By using considerations coming from both cycles' topology and geometry, the approach determines a weighting function W, which is repeatedly evaluated by a heuristic search algorithm. In fact, the graph originated by the curves making up the pseudo-wireframe model is searched and a set of optimal cycles (i.e. the ones characterized by higher values of W) are output.

Once the cycles enclosing the surface to be created are identified, they can be automatically generated by interfacing the development environment, where the spline database has been built, with a CAD software (Fig. 13).

In case of complex 3D curves networks, the optimal cycles identification procedure may fail and lead to generate edge cycles different from the desired ones (Fig. 14a). In such cases the resulting 3D surface model needs to be manually edited in a surface modeling software environment (e.g. Rhinoceros®). Anyway, since the probability incorrect cycles are generated grows as the number of 3D curves increases in the 3D model, it is possible to decrease the frequency of this issue simply by manually simplifying the curve network prior to the cycles identification phase. More in detail, starting from the pseudo wire frame model output by Step 13, it is sufficient to delete all the curves until only the wire frame model remains (Fig. 14b). This simplification procedure proves to be very fast even for relatively complex models.

3. Results and discussion

The reconstruction methodology, implemented using Matlab® programming language (with the exception of the surface creation phase, which has been carried out interfacing Matlab® with Rhinoceros® 4.0), has been applied on a number of case studies, other than the one used to illustrate the whole procedure.

In this section, a selection of examples are presented and discussed in order to point out strengths and weaknesses of the proposed reconstruction procedure.

The first example consists of a simple set of orthogonal views depicting a cubic object. Though the reconstruction may appear to be trivial, it has to be noted that matching among the views is quite poor (Fig. 15a). In this case a low value of n_s ($n_s = 2$) generates a voxel image characterized by thin outlines, but a number of regions where the model is incomplete can be observed (Fig. 15b). To solve the problem it is sufficient to set $n_s = 6$ in order to obtain a complete, correct voxel model (Fig. 15c) which leads to the reconstruction of the curve network shown in Fig. 15d and, ultimately, to the surface model of Fig. 15e. Generally speaking, when the outlines making up the original orthographic views are well separated, it is usually possible to compensate poor matching among the views simply increasing n_s value; conversely, in case the outlines are close the one to the other, a high value of n_s produces the merging of distinct outlines into a single one, thereby corrupting the whole model.

The second example is shown in Fig. 16a; in this case the resulting curve network features an erroneous region (topology different to the "theoretical" one) near the upper vertex (Fig. 16b). This is due to the fact that the tangent outlines in the side view originate a single voxel wire even if n_s is set to the minimum value necessary to obtain a complete reconstruction of the voxel cloud ($n_s = 1$). In this specific case, however, despite of the error in the curve network, a surface model is generated which closely resembles the theoretical one (Fig. 16c).

In case a similar example is considered (Fig. 17), though the object is slightly more complex, the procedure succeeds in the completely automatic reconstruction thanks to more separated lines.

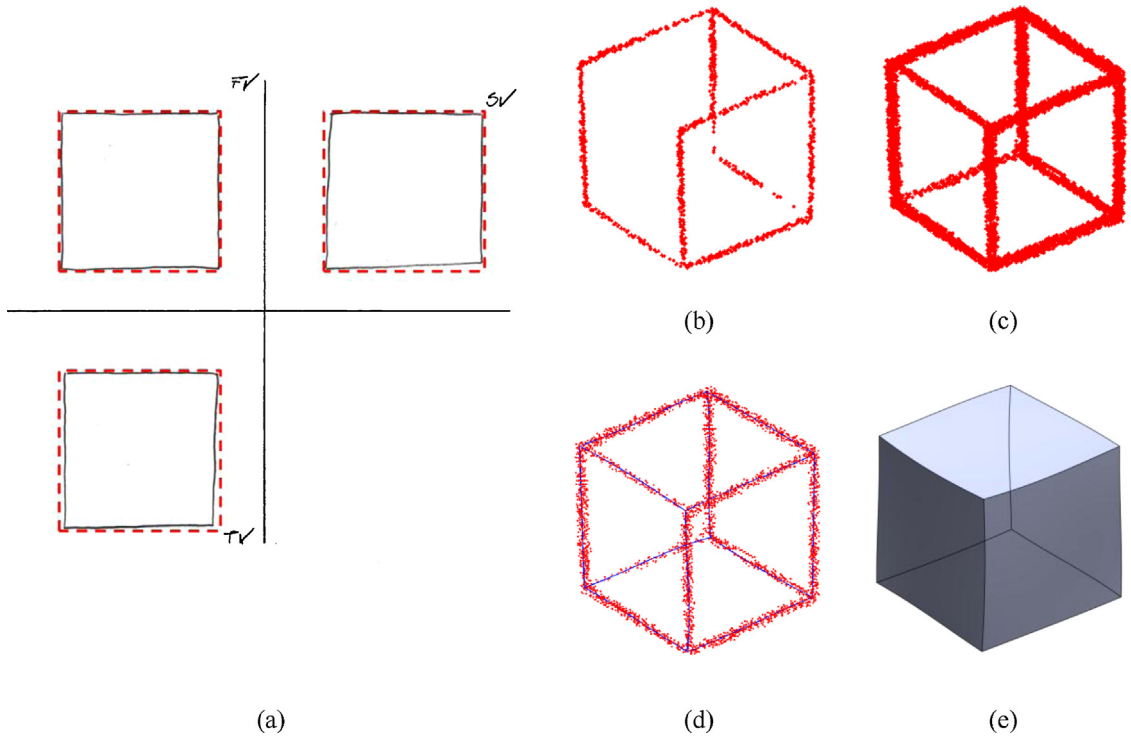


Fig. 15. Reconstruction example 1: initial drawing characterized by poor correspondence between the views (a); voxel image obtained for ns = 2 (b); voxel image obtained for ns = 6; final curve network with 10% of the non-zero voxels (d); and final surface model (e).

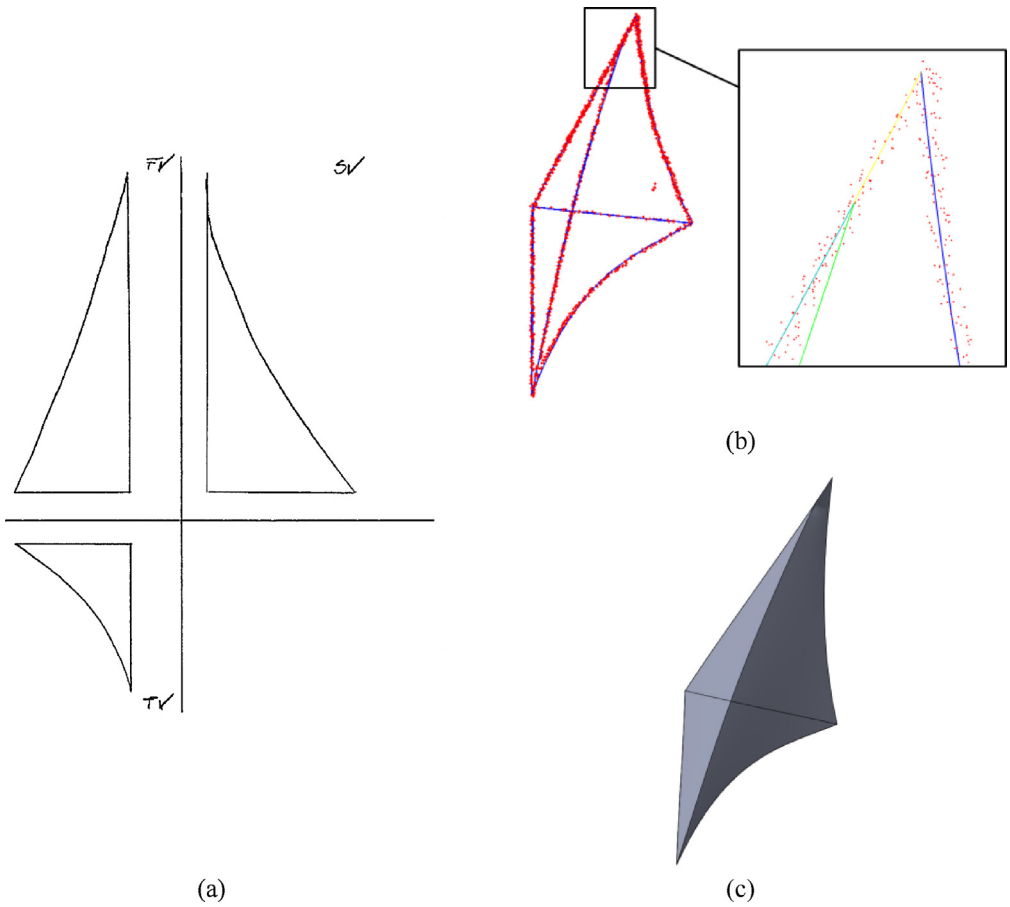


Fig. 16. Reconstruction example 2: initial drawing (a); final curve network and detail of the region characterized by wrong topology (b); and final surface model (c).

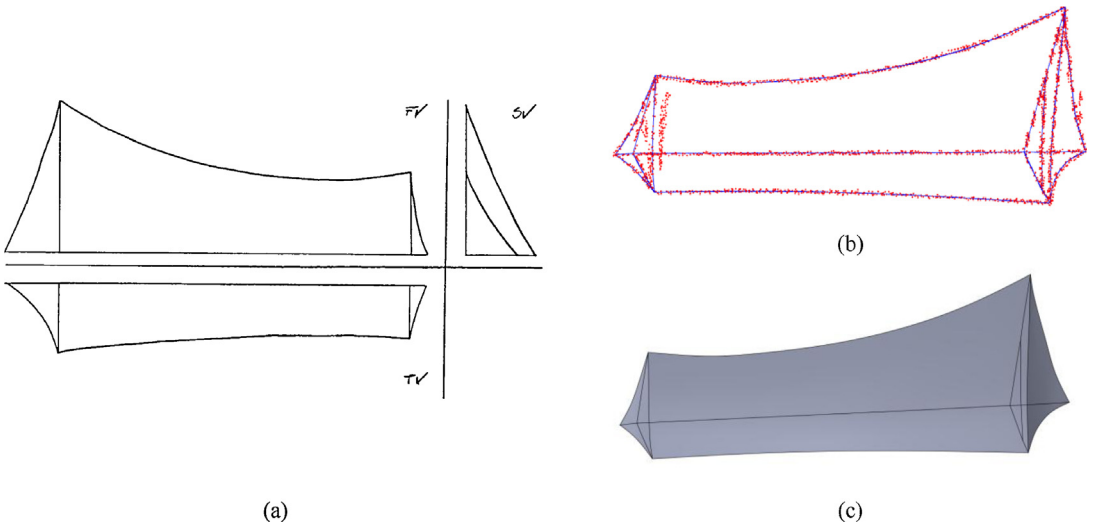


Fig. 17. Reconstruction example 3: initial drawing (a); final curve network with 5% of the non-zero voxels (b); and final surface model (c).

The resulting model proves to be correct also for the fourth example, depicting a drop-shaped object (Fig. 18). In this case, in order to avoid the erroneous reconstruction due to the tangent outlines in the upper region, a large value of *Ei* diameter has been used (40 pixels). This is a feasible solution only if there are no other outlines near the cutting region because, otherwise, these would be erased by the separation process described in Section 2 (Step 8).

The last example (Fig. 19) provides a case study similar to the one used in Section 2 to illustrate the methodology. Though the object appears to be relatively more complex with respect to the previous ones, the reconstruction process produces a completely correct model since a set of favorable conditions can be observed: good matching among the views, well separated outlines and approximately orthogonal intersections among the curves (both in the 2D drawing and in the 3D voxel image).

Moving from the results obtained so far, a few considerations can be drawn.

First, the present version of the proposed methodology can be applied only to simple 2D drawings (i.e. made of a limited number of curves), since the complexity of the pseudo wire frame voxel model rapidly grows as the number of 2D curves increases. This, in its turn, generally leads to the presence of a number of regions in 3D space where the 3D wires are very close the one to the other, so that the procedure for separating the voxel cloud branches can fail like shown in some of the examples discussed above.

This problem could possibly be overcome by introducing some additional user interaction in the first phases of the reconstruction process. More in particular it could be possible to build partial 3D

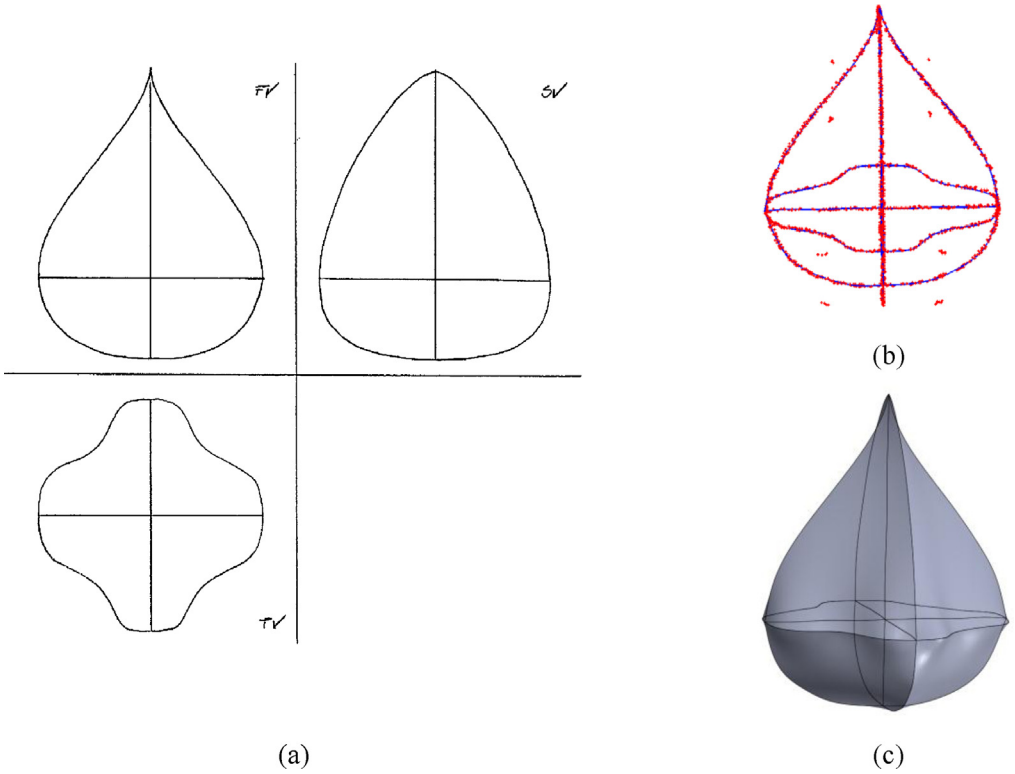


Fig. 18. Reconstruction example 4: initial drawing (a); final curve network with 5% of the non-zero voxels (b); and final surface model (c).

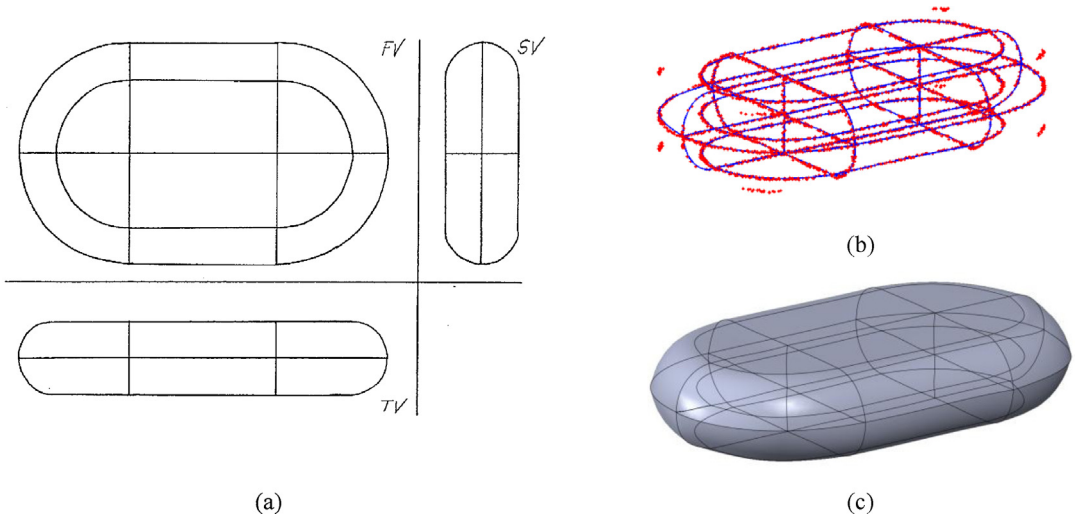


Fig. 19. Reconstruction example 5: initial drawing (a); final curve network with 1% of the non-zero voxels (b); and final surface model (c).

voxel models starting from sub-sets of 2D curves selected by the user in the original drawing, so that each corresponding voxel model would be significantly simplified. Each voxel model could be separately processed and the resulting network of 3D spline curves could be subsequently re-assembled in a single, final vector model.

As can be easily realized, a major drawback of the described approach resides in the necessity for each curve of the original object to be represented at least in 3 mutually orthogonal views of the starting hand drawing. This requirement can often be obtainable simply by using more views (up to 6) than the conventional 3 views representation. In fact, the described procedure can be repeated for any set of 3 mutually orthogonal views of the original drawing and, subsequently, the resulting sets of 3D curves can be merged in a final, complete model. In case this is not sufficient (e.g. for under-cut zones) the only alternative is to represent hidden lines in the hand drawings like visible ones, which, *de facto* means to draw the orthographic representation of the object wire frame model.

4. Conclusions

The proposed reconstruction methodology, despite some limitations discussed in Section 3, proved to be quite robust and to require only a little interaction from the operator. Evidently, due to its working principle (based on voxel cloud elaboration, fitting and surface generation), the method provides a resulting 3D geometry which can be considered an approximate version of the one achievable by means of a conventional 3D modeling process. However, the manual reconstruction of the vector wireframe and the surface generation proves to be considerably speeded up even if some reworking may be necessary for correcting possible reconstruction errors and adding some kind of constraints among the splines (e.g. symmetry, tangency, orthogonality). Future work will be addressed to deal with this kind of issues and, possibly, to integrate the developed procedure with a commercial CAD software package.

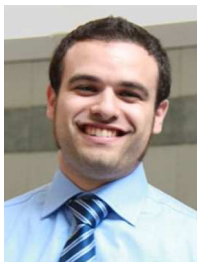
References

[1] G. Markowsky, M.A. Wesley, Fleshing out wire frames, IBM Journal of Research and Development 24 (5) (1980) 582–597.
[2] M.A. Wesley, G. Markowsky, Fleshing out projections, IBM Journal of Research and Development 25 (6) (1981) 934–954.
[3] R. Furferi, L. Governi, M. Palai, Y. Volpe, 3D model retrieval from mechanical drawings analysis, International Journal of Mechanics 5 (2) (2011) 91–99.
[4] Q.W. Yan, C.L.P. Chen, Z.S. Tang, Efficient algorithm for the reconstruction of 3d objects from orthographic projections, Computer-Aided Design 26 (9) (1994) 699–717.
[5] M.H. Kuo, Reconstruction of quadric surface solids from three-view engineering drawings, Computer-Aided Design 30 (7) (1998) 517–527.
[6] S.X. Liu, S.M. Hu, Y.J. Chen, J.G. Sun, Reconstruction of curved solids from engineering drawings, Computer-Aided Design 33 (14) (2001) 1059–1072.
[7] K. Inoue, K. Shimada, K. Chilaka, Solid model reconstruction of wireframe CAD models based on topological embeddings of planar graphs, Journal of Mechanical Design 125 (3) (2003) 434–442.
[8] P. Company, A. Piquer, M. Contero, F. Naya, A survey on geometrical reconstruction as a core technology to sketch-based modeling, Computers & Graphics 29 (6) (2005) 892–904.
[9] M.A. Fahiem, S.A. Haq, F. Saleemi, A review of 3D reconstruction techniques from 2D orthographic line drawings, in: GMAI 2007: Geometric Modeling and Imaging, Proceedings, 2007, pp. 60–63.
[10] M. Desbrun, M. Meyer, P. Schroder, A.H. Barr, Implicit fairing of irregular meshes using diffusion and curvature flow, Computer Graphic (1999) 317–324.
[11] R. Furferi, L. Governi, M. Palai, Y. Volpe, Multiple incident splines (MISs) algorithm for topological reconstruction of 2D unordered point clouds, International Journal of Mathematics and Computers in Simulation 5 (2) (2011) 171–179.
[12] S. Bagali, J. Warren, N. Waggenspack, A shortest path approach to wireframe to solid model conversion, in: Proceedings of the Third ACM Symposium on Solid Modeling and Applications, ACM, Salt Lake City, UT, 1995, pp. 339–350.
[13] T.H. Cormen, C. Stein, R.L. Rivest, C.E. Leiserson, Introduction to Algorithms, McGraw-Hill Higher Education, Boston, MA, 2001.
[14] F. Abbasinejad, P. Joshi, N. Amenta, Surface patches from unorganized space curves, Computer Graphics Forum 30 (5) (2011) 1379–1387.

Lapo Governi, Ph.D., is assistant professor at the Department of Industrial Engineering, University of Florence, Italy. Graduated M.Sc. in Mechanical Engineering at the University of Florence (Italy) in 1998. In 2002 he obtained the title of Ph.D. in Machine design and Construction. After working as a post-doctoral researcher at the Department of Mechanics and Industrial Technologies of the University of Florence, in 2005 he assumed a Faculty position as Assistant Professor for the courses of "Reverse Engineering and Rapid Prototyping" and "Design and modeling methods". His main scientific interests are: Computer Aided Design, computational geometry, reverse engineering, machine vision, tools and methods for product design and development. He is author of more than 70 scientific publications. His most recent works have dealt with 3D reconstruction from orthographic views, vision-based product and process assessment and spline-based approximation of point clouds.

Rocco Furferi, Ph.D., is assistant professor at the Department of Industrial Engineering, University of Florence, Italy. Graduated M.Sc. in Mechanical Engineering at the University of Florence (Italy) in 2001. In 2005 he obtained the title of Ph.D. in Machine design and Construction. He was employed as a post-doctoral researcher at the Department of Mechanics and Industrial Technologies of the University of Florence until 2008, when he assumed a Faculty position as Contract Assistant Professor for the courses "Mechanical Drafting" and "Computational Graphics". His main Scientific interests are: development of artificial vision systems, artificial neural networks, colorimetry, reverse engineering, computational geometry. He is author of more than 60 publications. Some latest works described methods for color assessment of textiles, algorithms

for 3D reconstruction of objects from orthographic views and ANN-based systems for industrial applications.



Matteo Palai, is Ph.D. candidate at the Department of Industrial Engineering, University of Florence, Italy. Graduated M.Sc. in Mechanical Engineering at the University of Florence in 2009. His Doctorate work deals with 2D to 3D reconstruction and computational geometry. He is author of a few scientific publications about techniques for spline-based approximation of point clouds, 2D to 3D reconstruction methods and machine vision systems.

Yary Volpe, Ph.D., is assistant professor at the Department of Industrial Engineering, University of Florence, Italy. Graduated M.Sc. in Mechanical Engineering at the University of Florence (Italy) in 2002. In 2006 he obtained the title of Ph.D. in Machine design and. Starting from 2006 he worked at the Department of Mechanics and Industrial Engineering of the University of Florence as a research grant holder; in 2011 assumed a Faculty position as Contract Assistant Professor for the course of "Computational Graphics". His main scientific interests are: Computer Aided Design, image processing, virtual prototyping, FE simulation. He is author of a number of scientific publications. His recent research work was dedicated to techniques for comfort-oriented design, machine vision-based systems and spline-based approximation of point clouds.