University of Sistan and Baluchestan

Chemical Process Design

Available online at http://cpd.usb.ac.ir/

A Method for Flattening Non-Planar Cut-Planes for Enhanced Visualization in Engineering and Medical Applications

Ali Imanparast

Department of Mechanical Engineering, University of Zabol, Zabol, Iran. E-mail: imanparast@uoz.ac.ir

ARTICLE INFO

Article type:
Research Article

Article history:
Received: 2025-04-03
Received in revised form: 2025-05-05
Accepted: 2025-05-09
Published online: 2025-05-09

Keywords: *Non-planar cut-plane 3D; Surface flattening; Engineering visualization*

ABSTRACT

A critical aspect of multidimensional data research lies in its representation. As the dimensionality of data escalates, visualizing and comprehending the underlying information becomes increasingly challenging. One effective approach is to reduce data dimensionality for presentation purposes. Slicing three-dimensional (3D) data yields a two-dimensional (2D) representation of a specific cross-section, a technique widely applied in engineering and medical sciences. Moreover, non-planar sections may offer distinct advantages in particular contexts. In this study, an approach for flattening non-planar cutting planes based on rotational transformations was introduced to enhance the visualization of complex 3D datasets. This method is suitable for flattening both scalar and vector data and offers excellent parallelization capabilities from a computational perspective. The effectiveness of this approach was demonstrated through one example: visualization of fluid flow with heat transfer in a helical pipe with multiple orifices. The results showcase the advantages of the method in providing comprehensive and accessible visualizations, which facilitate enhanced analysis and interpretation of complex geometrical data.

Cite this article: Imanparast, A., (2025), A Method for Flattening Non-Planar Cut-Planes for Enhanced Visualization in Engineering and Medical Applications, *Chemical Process Design*, 4(1), 74-85. http//doi.org/10.22111/cpd.2025.51570.1054

DOI: http//doi.org/10.22111/cpd.2025.51570.1054

Publisher: University of Sistan and Baluchestan.

1. Introduction

Visualizing simulation results is a crucial aspect of scientific research. This task becomes challenging when the geometries under investigation are complex and three-dimensional, often leading to difficulties in achieving accurate and desired representations. One of the methods for visualizing information in certain engineering disciplines (such as fluid and solid mechanics, electrical engineering, and biomedical engineering) is the use of cut-plane visualization. Accurate 2D presentation of 3D objects can be effective in understanding 3D aspects of the object and facilitates the quicker transmission of information [1].

Sometimes, in 3D geometries, selecting one proper planar cross-section that display the information desired by researchers is challenging or impossible. Currently, the cross-sectioning available in most commercial engineering software (such as Tecplot [2], ParaView [3], COMSOL [4]) is limited to planar. Certainly, within a select group of software tools, such as VTK [5], the functionality for non-planar slicing is provided.

Also, in medical applications where geometries are generally complex 3D (such as blood vessels), the ability to utilize non-planar representations can be extremely important [6]. Non-planar visualization enhances the representation and analysis of complex anatomical structures, improving the understanding of their functions and pathologies. It can significantly aid medical imaging and surgical planning by providing a clearer view of the spatial relationships between tissues and organs. Volume Clipping is a powerful tool for understanding 3D data [7] and has useful applications in medical imaging [8]. This method allows for spatial slicing of 3D images, enabling more precise visualization and analysis of complex anatomical structures. In some software, such as 3D Slicer, volume clipping (slicing) is possible [9].

Furthermore, even with the availability of non-planar sectioning, the complexity of certain geometries can pose challenges for visualization using computer monitors or paper prints, as these are inherently 2D display mediums. In such instances, achieving a comprehensive understanding often necessitates creating multiple figures from various viewing angles or interactively exploring the data with 3D visualization software.

A viable approach to address this issue is to transform the 3D cut-plane results into a 2D format, effectively flattening (or parameterizing) the 3D planes. Analyzing 2D images rather than 3D objects simplifies the process for both human perception and computational analysis. As noted previously, the flattened 2D representation of 3D data is not a novel concept, and various studies have explored general-purpose flattening techniques for 3D surfaces [10-13].

Extensive research has been conducted on the flattening of 3D surfaces for the design and manufacturing of objects using sheet metal. The primary methods employed for this purpose include geometric flattening methods, finite element methods, and energy-based methods. Each of these approaches optimizes the 2D mapping in different ways, tailored to the requirements of the manufacturing process (see, for example, [14-18]).

Several flattening techniques also exist in the medical field: Bulls Eye plots, used for displaying a variety of cardiac information in a flattened circular format; Vessel Maps, for representing 2D plots of cerebral aneurysms and vasculature; medial axis reformation (MAR), for obtaining curved cuts of blood vessels along their medial axis; and others [19]. A visualization technique that combines non-planar cutting planes with their flattened representation is known as Curved Planar Reformation (CPR), which finds extensive application in medicine, particularly for displaying tubular structures such as piping systems and blood vessels. Three primary CPR methods exist: projected CPR, stretched CPR, and straightened CPR, as outlined by Kanitsar et al. [20]. Subsequently, Kanitsar et al. [21] enhanced stretched CPR to flatten multipath vascular structures.

Saroul et al. [22] presented two methods for distance-preserving surface flattening relative to specific lines along a curve or to an arbitrary point, with the aim of visualizing non-planar medical images. Karim et al. [23] implemented a surface mesh parameterization method for flattening the complex 3D shape of the left atrium.

In this paper, we propose a method based on rotational transformations that enables the 2D visualization of geometric, scalar, or vector data associated with free-form, smooth cut-planes. This approach is beneficial in medical studies, biomedical engineering, and more generally, in all engineering fields for visualizing the results of numerical simulations or experimental studies, particularly for geometries with a tubular nature. This is especially valuable in

cases where a planar section is insufficient and the required information is not displayed in that section. The proposed method reflects geometric properties and proportions with reasonable accuracy. Furthermore, computational optimization of the method is discussed.

2. Materials and methods

2.1. Constructing an illustrative geometry

To better illustrate the proposed method, a representative geometry is introduced, and its various processing stages are described. A helical pipe provides a fully three-dimensional and relatively complex geometry for visualization purposes. The central axis of this helical pipe is defined by the following equation:

$$\begin{cases} x = \cos(t) \\ y = \sin(t) \\ z = t \end{cases} \tag{1}$$

where $0 \le t < 2\pi$ (Fig. 1(a)). Subsequently, a tube with a radius of 1 was generated around this central axis using a MATLAB code by J. H. Wesenberg [24] (Fig. 1(b)). Subsequently, the entire tube is centrally cross-sectioned. The resulting surface, a non-planar form, is illustrated in Fig. 1(c). This non-planar surface will serve as a demonstrative example throughout this paper. It is important to note that, for simplification in this example, the intersection of the cutting plane and each section of the tube (cross-section line) is a straight line. Consequently, for instance, points P_1, P_2 and P_3 are collinear (Fig. 1c). However, due to the fact that the normal to the plane at different points does not lie in the same plane, the resulting cutting plane exhibits a complex nature. It is evident that no single planar cut on this tube can provide comprehensive information from the tube's beginning to its end, similar to the presented curved cross-section.

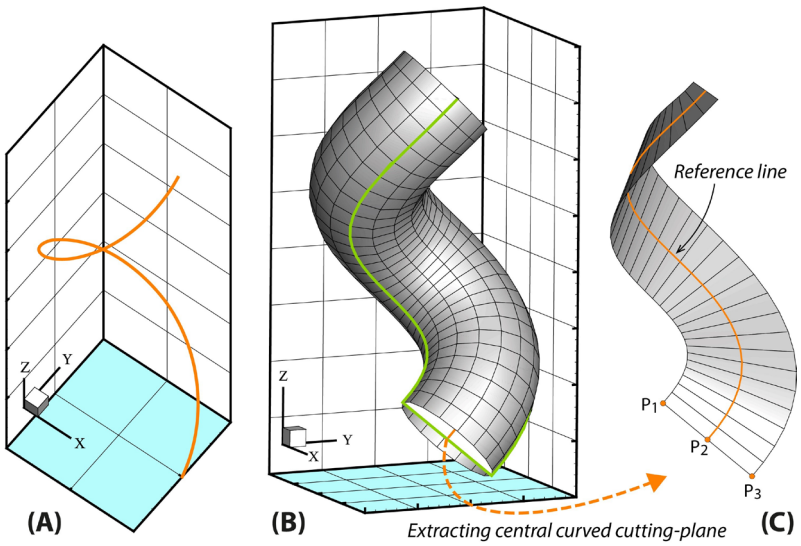

Fig. 1. (a) A helical line serving as the central or reference line for generating a helical tube. (b) The helical tube generated with a radius of 1, based on the central line in A. (c) A central cross-section of the helical tube, forming a non-planar surface

2.2. Flattening method

Consider the non-planar cutting plane derived from the previous step, characterized by three points in each cross section. To flatten this surface, we employ the following sequential steps:

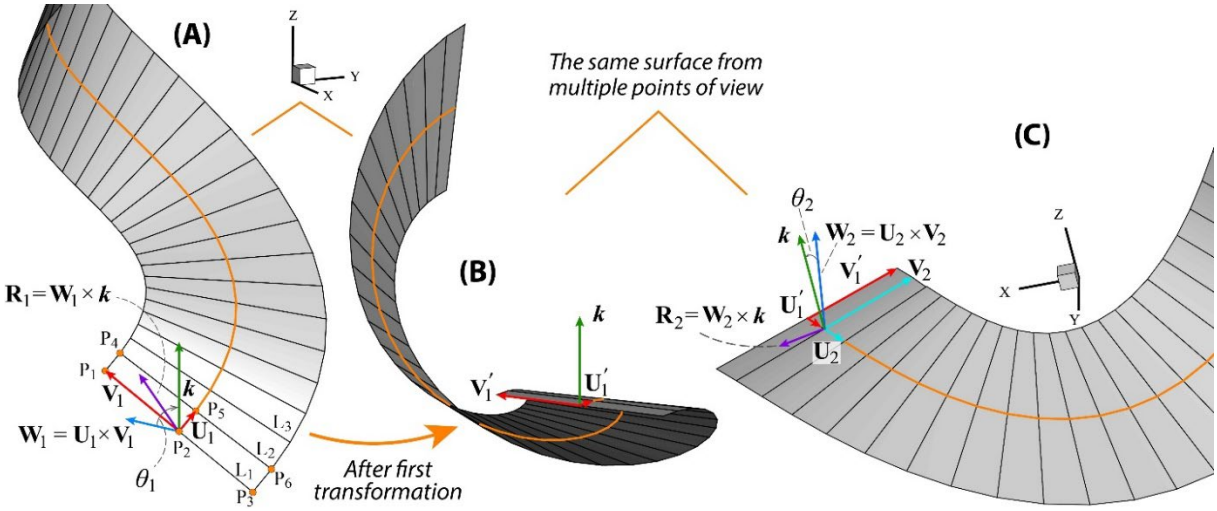

Fig. 2. (**a**) Display of vectors $\boldsymbol{U}_1$, $\boldsymbol{V}_1$, $\boldsymbol{W}_1$ and $\boldsymbol{R}_1$ necessary for the rotation and translation of the first section line of the non-planar surface to the xy-plane. (**b**) The same surface after the first rotation and translation from the same perspective as surface A. At this stage, the points on the surface have not changed relative to each other. (**c**) Display of vectors $\boldsymbol{U}_2$, $\boldsymbol{V}_2$, $\boldsymbol{W}_2$ and $\boldsymbol{R}_2$ necessary for the rotation and translation of the second section line of the non-planar surface to the xy-plane. This surface is exactly the same surface as B from different perspective

1) A point from the first section is chosen as the reference point. In this example, the middle point of the initial section is assumed to be the reference point (P_2 in Fig. 2a).

2) Vector $\boldsymbol{U}_1$ (from the middle point of the first section to the middle point of the second section, along the reference line) and $\boldsymbol{V}_1$ (from the middle point to the starting point of the first section) are calculated. Then, their cross product, $\boldsymbol{W}_1$, is obtained.

3) The coordinates of all points on the curved surface are shifted so that the reference point is positioned at the origin of the coordinates.

4) If $\boldsymbol{W}_1$ is not aligned with $\boldsymbol{k}$ (the unit vector along the z-axis), all points on the cutting plane are rotated around vector $\boldsymbol{R}_1$ (the cross product of $\boldsymbol{W}_1$ and $\boldsymbol{k}$) by the angle θ_1 (the angle between $\boldsymbol{W}_1$ and $\boldsymbol{k}$). During this step, all relevant vector quantities, such as velocity, are also rotated by the same angle and direction. Up to this stage, points P_1, P_2, P_3, P_5 and therefore, $\boldsymbol{U}_1$ and $\boldsymbol{V}_1$ have been mapped onto the xy-plane (Fig. 2b). Additionally, the processing of the data for points P_1 to P_3 (points on line 1) has been completed, and these points will not be included in subsequent calculations. As previously mentioned, it is assumed that the points in each section lie along a straight line. Therefore, points P_1, P_2 and P_3 are on a straight line, and similarly, P_4, P_5 and P_6 are also aligned.

5) The points on line 2 are now processed. Vectors $\boldsymbol{U}_2$ and $\boldsymbol{V}_2$ are formed as shown in Fig. 2c, and then the vector perpendicular to these two vectors, $\boldsymbol{W}_2$ and the rotation axis, $\boldsymbol{R}_2$, are obtained.

6) Points other than those on line 1 are temporarily shifted so that P_5 is positioned at the origin of the coordinates. It is emphasized that, at this stage, all vector quantities must also be shifted with the same rotation.

7) A rotation around $\boldsymbol{R}_2$ by the angle θ_2 is applied to all points and vector quantities (other than the previous ones) such that line 2 is positioned on the xy-plane and its vector quantities are flattened.

8) Then, the points other than those on line 1 are shifted by the negative amount that was transferred in step 6. After this step, the points on line 2 are also excluded from the calculations.

The steps from 5 to 8 are repeated iteratively until the final cross line is reached. To rotate the last line, since there are no subsequent points to form vectors as in the preceding steps, points from the penultimate line are used. Fig. 3 depicts the flattened surface of the sample that is shown in Fig. 2.

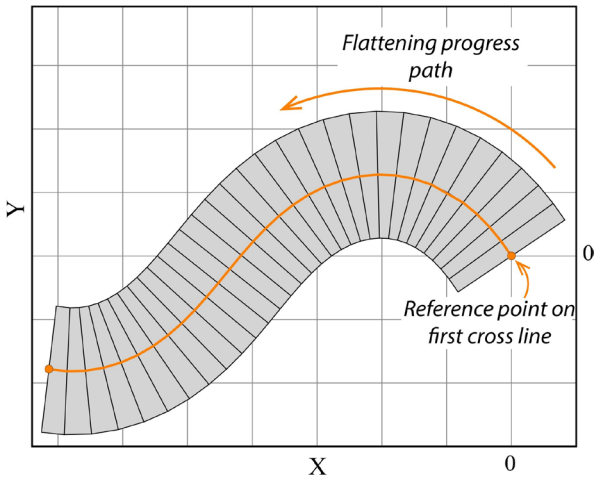

Fig. 3. Flattened Surface and Flattening Path: From Fig. 2, showing the path from the first to last cross line reference points

2.3. Computational optimization and parallelization

If the model contains a larger number of points within each cross-sectional line (as opposed to the simplified example with three points per line), the previously described method becomes inefficient. In that method, points in each cross-section undergo a number of rotations and translations that corresponds to their cross-section's position. Specifically, points on the first cross-sectional line experience one rotation, those on the second experience two, and so forth. To reduce computational complexity and enhance performance, the following approach is proposed:

2.3.1. Mapping the position of points

a) For each cross-sectional line, only two points are considered. We assume that the points within each cross-section lie on a straight line; thus, two points are sufficient to define it. The chosen points are the starting point (e.g., point P_1 on the first line) and a reference point, specifically the midpoint of that cross-sectional line (P_2). This midpoint, P_2, lies on the reference line.

b) Given that the distances between the points on each cross-sectional line remain constant, the distance of each point from the starting point of its respective line is stored.

c) After all cross-sections are projected onto the xy-plane (using the two selected points per cross-section), the remaining points of each cross-sectional line are positioned relative to the starting point of their respective cross-section, aligned with the vector from the starting point to the reference point. The inter-point distances are maintained as stored previously. This approach reduces computational complexity, as only two points in each cross-section need to be rotated instead of all points.

2.3.2. Mapping vector quantities

d) Unlike point positions, vector quantities such as nodal velocity vectors require rotation at every point. Therefore, the approach used for point positions cannot be applied directly. To minimize the number of rotation operations, the rotation matrix for each cross-sectional line ($\mathbf{M}_i$) is stored, which was determined in step c.

e) For each cross-section, the final rotation matrix is calculated as accumulative product of the rotation matrices. This means the final rotation matrix is obtained by sequentially multiplying the rotation matrix of each cross-section with those of preceding cross-sections. The calculation is as follows:

$$\begin{aligned} \mathbf{M}_1^{\mathrm{n}} &= \mathbf{M}_1^{\mathrm{o}} \\ \mathbf{M}_2^{\mathrm{n}} &= \mathbf{M}_2^{\mathrm{o}} \times \mathbf{M}_1^{\mathrm{n}} \\ &\vdots \qquad\quad \vdots \\ \mathbf{M}_i^{\mathrm{n}} &= \mathbf{M}_i^{\mathrm{o}} \times \mathbf{M}_{i-1}^{\mathrm{n}} \end{aligned} \tag{2}$$

where the superscripts n and o represent 'new' and 'old', respectively. Subsequently, in each cross-section i, the vector quantities are rotated using the final rotation matrix $\mathbf{M}_i^{\mathrm{n}}$.

Consequently, instead of each vector quantity in cross-section i undergoing i rotations, it is projected onto the flattened xy-plane with a single rotation, leading to a significant reduction in computational costs. Furthermore, this method offers a notable advantage through its effective parallelization capabilities. After the rotation matrices are determined for each cross-sectional line, the computation of mapped coordinates for the points on these lines, along with the associated vector quantities, becomes entirely independent. Therefore, these computations can be distributed across multiple processing cores without the need for shared variables. This is an optimal form of parallelization for algorithms, ensuring maximum efficiency in the utilization of processing cores.

2.4. Flattening general non-planar cut-planes

While we previously simplified the cutting surface by restricting it to a configuration where points within each cross-section lie on a straight line, there is no inherent mathematical constraint limiting us to this. The flattening process can initially be applied to the curved cross-sectional lines and subsequently along the reference line. Consider a non-planar cut-plane, as illustrated in Fig. 4(a).

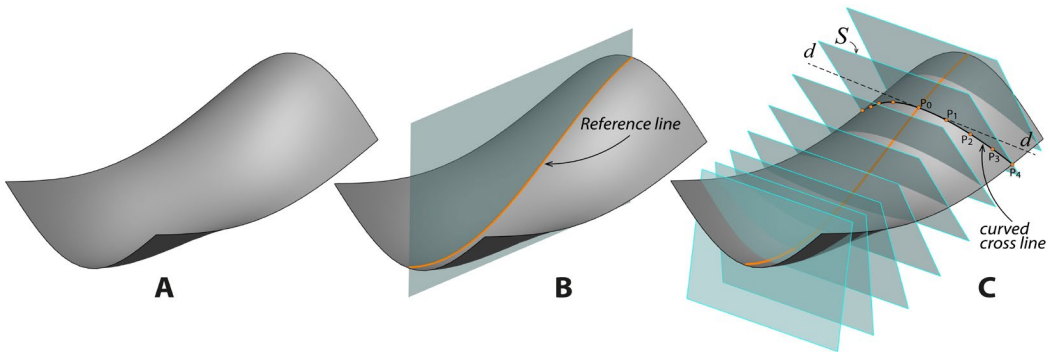

Fig. 4. (a) A typical curved cut-plane exhibiting curvature in all directions; (b) A reference line created by intersecting a flat plane with the curved cut-plane; (c) Curved cross lines formed by intersecting the curved plane with additional flat planes

This surface exhibits curvature across all three coordinate axes, meaning there is no single direction in which the cross-sectional lines remain straight. A possible solution to address this complexity is outlined below:

I. Initially, a suitable reference line is selected on the cut-plane. This is accomplished by intersecting a flat plane with the desired cut-plane, as illustrated in Fig. 4b, with the resulting intersection forming the reference line.

II. Subsequently, additional flat planes are intersected with the cut-plane, similar to the configuration depicted in Fig. 4c. These intersections create curved cross-sectional lines. For example, consider the curved line resulting from the intersection of the planar surface $\boldsymbol{S}$ and the cut-plane.

III. The tangent vector to each cross-sectional curved line at its intersection point (e.g., P_0) with the reference line is defined as the reference direction for that cross-sectional line (e.g., the direction dd in Fig. 4c). Segments along each cross-sectional line (e.g., P_0P_1) are rotated around the normal vector to $\boldsymbol{S}$, initiating from the intersection point, until they align with direction dd. Vector quantities associated with these segments undergo the same rotation, while scalar quantities remain unchanged.

IV. Once all cross-sectional lines have been straightened, the remaining steps are executed as described previously.

For optimization, in step III., instead of having segments rotated, points can be placed along the final aligned cross line obtained in step II. such that their distances to each other mirror the corresponding distances on the initial curved cross line. However, the associated vector quantities must still undergo rotation to maintain the proper transformation of the original curved line to the aligned one. Since the axes of rotation are fixed for each cross line (the normal vector to $\boldsymbol{S}$), the vector parameters rotations are independent of each other. This characteristic of independent rotations enables a parallel implementation of these transformations.

3. Application of the method: a practical model

This section demonstrates the application of the method described in this article by presenting an example: fluid flow with heat transfer in a helical pipe having multiple orifices.

3.1. Definition of model

To create a complex geometry, we defined a helical pipe wound around a segment of a torus, with the equation below defining its centerline:

$$\begin{cases} x = \left(R + r\cos\left(nt\right)\right)\cos\left(t\right) \\ y = \left(R + r\cos\left(nt\right)\right)\sin\left(t\right) \\ z = r\sin\left(nt\right) \end{cases} \tag{3}$$

where $R = 3$, $r = 1$, $n = 5.5$ and $0 \le t < 1.2\pi$. The method for generating the tube geometry, which is centered around a specific centerline, is detailed in Section 2.1. The tubes radius was set to 0.8 m. The fluid domain was discretized using hexahedral elements with a custom-developed code. Four equally spaced orifices were integrated into the geometry. To create an orifice, three outermost layers of elements were removed from a section of the pipe. The initial temperature of all model nodes was set to 10˚C. Then, over 0.1 seconds, the wall temperature was reduced to 0˚C. The geometry and boundary conditions are depicted in Fig. 5. The fluid was assumed to be incompressible and Newtonian with mechanical properties as listed in Table 1. The k-epsilon turbulence model was chosen for this study; its characteristic coefficients are outlined in Table 2.

Table 1. Mechanical properties of fluid

Property	Value
Viscosity	0.01 Pa.s
Density	100 kg/m3
Laminar thermal conductivity	0.1 W/m.K
Spesific heat at constant pressure	150 J/kg.K

Table 2. Turbulent flow model constants

Property	Value
C_1	1.44
C_2	1.92
C_3	0.8
C_μ	0.09
σ_k	1
σ_ϵ	1.3
σ_T	0.9
Dimensionless distance from boundary	70
Von Karman constant	0.4

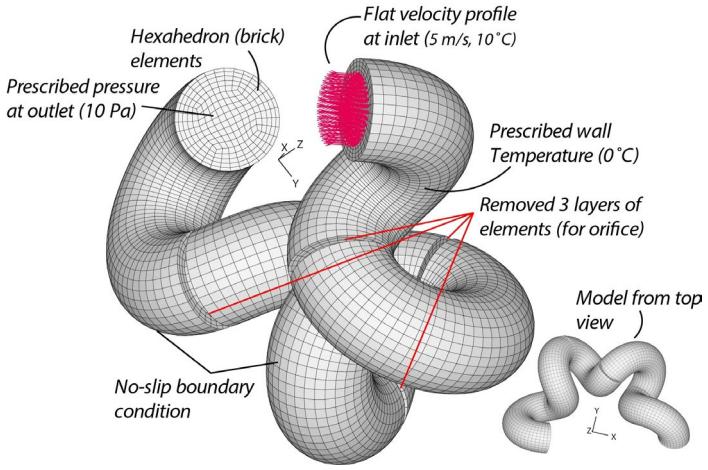

Fig. 5. The definition of the helical pipe model including boundary conditions

3.2. Solution

The numerical simulation of the problem described previously was conducted using ADINA CFD (ADINA R&D, Inc., Watertown, MA, USA), over a period of 4 seconds after achieving steady boundary conditions.

4. Results

Fig. 6 illustrates 3D and flattened cut-planes visualizing three flow characteristics: velocity vectors, velocity magnitudes with streamlines, and normalized pressure contours. Similarly, Fig. 7 presents 3D and flattened cut-planes showing temperature and heat flux vectors, along with heat flux magnitude. As is evident, accurately observing these 3D visualizations requires interactive rotation of the model to allow viewing from various angles. In other words, due to the geometric complexity, many flow details cannot be effectively depicted in a single 3D cut-plane. However, when the data are flattened, flow trends become readily apparent. For instance, flow separation and recirculation zones are clearly discernible in the flattened plots, whereas locating these zones, especially relative to the orifices, is challenging in the 3D plots.

For scalar data (such as velocity magnitude, pressure and temperature), these data points are transferred directly to the flattened nodes without any alteration. Vector data maintain their magnitudes, but their directions are converted

from 3D to 2D. Importantly, the relative orientation between adjacent data points is completely preserved, resulting in a representation that appropriately reflects the geometric characteristics and proportions at different locations.

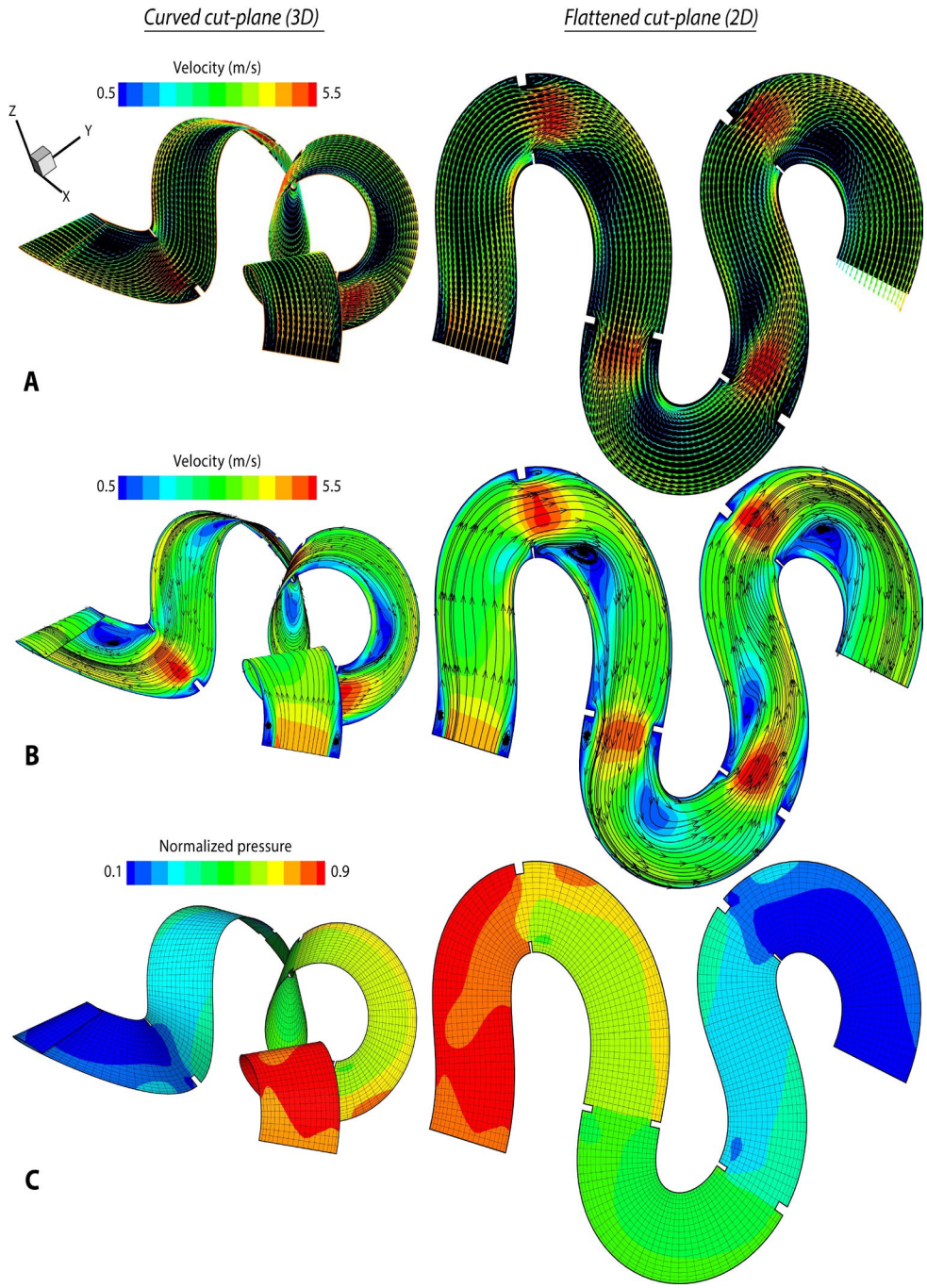

Fig. 6. Comparison of flow visualization on curved and flattened cut-planes: (A) Velocity vectors, (B) Velocity magnitude contours with streamlines, (C) Normalized pressure contours

5. Discussion

In this manuscript, the importance of non-planar cuts in visualizing numerical and experimental research results in engineering and medical fields was highlighted. This method facilitates faster information transfer to engineers, particularly when working with complex 3D tubular geometries. As previously mentioned in the literature, flattened non-planar cut-planes have important application in medicine, but the absence of this method is quite noticeable in engineering visualization software.

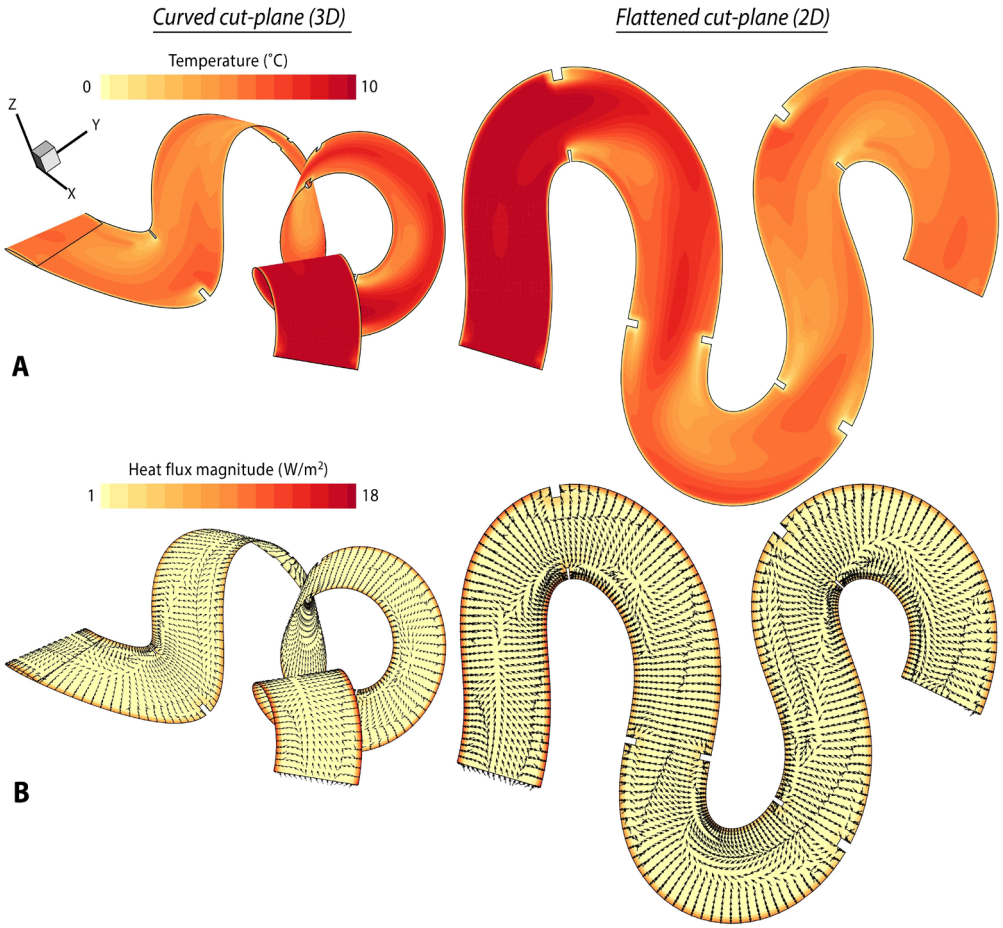

Fig. 7. Comparison of heat transfer visualization on curved and flattened cut-planes: (a) Temperature contours; (b) Heat flux vectors with magnitude contours (note that a uniform vector size has been used here)

Previously proposed methods, whether for manufacturing applications (e.g. [14-18]) or medical purposes (e.g. [20-23]), are suitable for flattening locations and ultimately mapping scalar quantities such as color, pressure, etc. In these studies, there has been no mention of how to map vector quantities such as velocity vectors onto the flattened surface. The rotational method outlined in this paper is capable of mapping vector quantities as well. Similar to the works of CPR methods [20-22], the method employed in this paper relies propagation on a reference line. While this approach might appear limiting, it remains applicable even to complex, multi-curved surfaces, as both the reference and cross lines can be curvilinear. The mapping on reference line and cross lines is isometric. Besides relatively preserving the topological aspects, the nodal values are maintained just as in the original state. Vector values are also fully preserved, and the ratio between the vectors remains logical.

In some previous works on parameterization (e.g. [10-11, 25, 17]), there is a need to generate a triangular mesh. However, in the present method, the computations for generating a triangular mesh are eliminated. If appropriate points are used during the cutting stage, there is no need to generate a triangular mesh in the flattening process. Only in the display of data in some software might a mesh be required, and a simple quadrilateral mesh can be used, which has a much lower computational cost compared to generating a triangular mesh. Furthermore, while the mesh used here made it easier to extract points for the non-planar section, if this geometry is meshed with any other type of element, data extraction on the non-planar surface can still be performed using interpolation methods.

The implementation of this method is simple and algorithmically straightforward. It does not require correction in an iterative loop and is not computationally expensive.

Another significant advantage of this method is its capability for parallelization, leveraging multiple CPU or GPU cores to enhance image processing speed, particularly in real-time displays.

A notable limitation of this methodology arises in some highly complex cut-planes, where their curvatures can lead to overlapping of different segments. This inherent constraint can be addressed through a segmentation approach, whereby the plot is partitioned into multiple sub-plots, each depicting a separate, non-overlapping region of the geometry.

The application of this data visualization method can be particularly beneficial in engineering fields, especially for problems where the geometry has a tubular nature.

Finally, as a direction for future work, advancements in cross-sectioning methods within visualization software environments such as Tec plot could be explored. Specifically, investigating how non-planar cut-planes can be easily extracted by defining specific nodes on the geometry surface and subsequently flattened would significantly enhance the practical utility of this technique for presenting research results.

6. Conclusion

This paper introduces a method for flattening non-planar cut-planes from 3D datasets using rotational transformations. The approach enables effective visualization of complex geometries, particularly where traditional planar slices are inadequate. By preserving geometric accuracy and optimizing computational efficiency, the method extends to general non-planar surfaces, making it suitable for both scalar and vector data. The example application to a complex helical pipe geometry demonstrates its utility in engineering and medical fields, offering enhanced data analysis through improved visualization of intricate 3D structures.

References

[1] Ware, C., Stevens, A.H., 2015, Visualizing 3d flow through cutting planes, 2015 ieee scientific visualization conference (scivis), 161–162. https://doi.org/10.1109/SciVis.2015.7429513

[2] Tecplot, Inc., 2024, Tecplot 360 user's manual. https://tecplot.azureedge.net/products/360/current/360-users-manual.pdf

[3] Paraview development team, 2024, Paraview user's guide, version 5.12. https://docs.paraview.org/en/latest/UsersGuide/displayingData.html

[4] Comsol ab, 2023, Comsol multiphysics reference manual, version 6.2, stockholm, sweden. https://doc.comsol.com/5.5/doc/com.comsol.help.comsol/COMSOL_ReferenceManual.pdf

[5] Schroeder, W., Martin, K., Lorensen, B., 2006, The visualization toolkit (4th ed.), kitware.

[6] Alberich-Bayarri, A., 2021, Advanced visualization basics in medical imaging, springer international publishing, cham, 55–66. https://doi.org/10.1007/978-3-030-71885-5_5

[7] Pfister, H., 2005, Hardware-accelerated volume rendering, in: Hansen, C.D., Johnson, C.R. (Eds.), visualization handbook, butterworth-heinemann, burlington, 229–258. https://doi.org/10.1016/B978-012387582-2/50013-7

[8] Zhang, Q., Eagleson, R., Peters, T.M., 2011, Volume visualization: a technical overview with a focus on medical applications, journal of digital imaging, 24 (4), 640–664. https://doi.org/10.1007/s10278-010-9321-6

[9] Lasso, A., 2017, Volumeclip, 3d slicer extension. https://www.slicer.org/w/index.php?title=Documentation/Nightly/Extensions/VolumeClip&oldid=54933

[10] Floater, M., Hormann, K., 2002, Parameterization of triangulations and unorganized points, springer berlin heidelberg, berlin, heidelberg, 287–316. https://doi.org/10.1007/978-3-662-04388-2_11

[11] Floater, M.S., Hormann, K., 2005, Surface parameterization: a tutorial and survey, in: Dodgson, N.A., Floater, M.S., Sabin, M.A. (Eds.), advances in multiresolution for geometric modelling, springer berlin heidelberg, berlin, heidelberg, 157–186. https://doi.org/10.1007/3-540-26808-1_9

[12] Sheffer, A., Praun, E., Rose, K., 2006, Mesh parameterization methods and their applications, foundations and trends in computer graphics and vision, 2 (2), 105–171. https://doi.org/10.1561/0600000011

[13] Zhang, Q., Hou, J., Wang, W., He, Y., 2024, Flatten anything: unsupervised neural surface parameterization, arXiv:2405.14633. https://doi.org/10.48550/arXiv.2405.14633

[14] Liu, Q., Xi, J., Wu, Z., 2013, An energy-based surface flattening method for flat pattern development of sheet metal components, the international journal of advanced manufacturing technology, 1155–1166. https://doi.org/10.1007/s00170-013-4908-y

[15] Liu, X., Li, S., Zheng, X., Lin, M., 2016, Development of a flattening system for sheet metal with free-form surface, advances in mechanical engineering, 8(2), 1–12. https://doi.org/10.1177/1687814016630517
[16] Wei, P., Kai, Q., Yidong, B., Chaoyang, Z., Weixi, J., 2022, Free-form surface flattening based on rigid registration and energy optimization, international journal of precision engineering and manufacturing, 23 (8), 921–927. https://doi.org/10.1007/s12541-022-00683-6
[17] Zheng, P.-F., Lou, J.-J., Lin, D.-J., An, Q., 2021, A curved surface flattening computing method combined with machining process, in: Cheng, L.-Y. (Ed.), icgg 2020 - proceedings of the 19th international conference on geometry and graphics, springer international publishing, cham, 186–198. https://doi.org/10.1007/978-3-030-63403-2_17
[18] Zheng, P., Liu, Q., Lou, J., Lian, C., Lin, D., 2022, A free-form surface flattening algorithm that minimizes geometric deformation energy, iet image processing (march), 2544–2556. https://doi.org/10.1049/ipr2.12508
[19] Kreiser, J., Meuschke, M., Mistelbauer, G., Preim, B., Ropinski, T., 2018, A survey of flattening-based medical visualization techniques, computer graphics forum, 37 (3), 597–624. https://doi.org/10.1111/cgf.13445
[20] Kanitsar, A., Fleischmann, D., Wegenkittl, R., Felkel, P., Groller, E., 2002, Cpr - curved planar reformation, IEEE visualization 2002 (vis 2002), 37–44. https://doi.org/10.1109/VISUAL.2002.1183754
[21] Kanitsar, A., Wegenkittl, R., Fleischmann, D., Groller, M., 2003, Advanced curved planar reformation: flattening of vascular structures, ieee visualization 2003 (vis 2003), 43–50. https://doi.org/10.1109/VISUAL.2003.1250353
[22] Saroul, L., Figueiredo, O., Hersch, R., 2006, Distance preserving flattening of surface sections, ieee transactions on visualization and computer graphics, 12 (1), 26–35. https://doi.org/10.1109/TVCG.2006.7
[23] Karim, R., Ma, Y., Jang, M., Housden, R.J., Williams, S.E., Chen, Z., Ataollahi, A., Althoefer, K., Rinaldi, C.A., Razavi, R., O'Neill, M.D., Schaeftter, T., Rhode, K.S., 2014, Surface flattening of the human left atrium and proof-of-concept clinical applications, computerized medical imaging and graphics, 38 (4), 251–266. https://doi.org/10.1016/j.compmedimag.2014.01.004
[24] Wesenberg, J.H., 2024, Tubeplot. https://www.mathworks.com/matlabcentral/fileexchange/5562-tubeplot
[25] Li, X., Iyengar, S.S., 2014, On computing mapping of 3d objects: a survey, acm computing surveys (csur), 47 (2), 1–45. https://doi.org/10.1145/266802