SCIPEDIA

Integrating Vector Computation and Numerical Methods for Complex Surface Design in Engineering via MATLAB

Juan Carlos Sarango Cuenca*, Vicente García, Jacobo Vásquez and Mayra Salazar

Department of Exact Sciences, University of the Armed Forces, Sangolqui, Ecuador

INFORMATION

Keywords:
Computational simulation 1
structural engineering 2
geometric parameterization 3
surface integrals 4
three-dimensional visualization 5

DOI: 10.23967/j.rimni.2026.10.75902

SCIPEDIA Revista Internacional de Métodos Numéricos para Cálculo y Diseño en Ingeniería

Revista Internacional Métodos numéricos para cálculo y diseño en ingeniería RIMNI

In cooperation with CIMNE

Integrating Vector Computation and Numerical Methods for Complex Surface Design in Engineering via MATLAB

Juan Carlos Sarango Cuenca*, **Vicente García, Jacobo Vásquez and Mayra Salazar**

Department of Exact Sciences, University of the Armed Forces, Sangolqui, Ecuador

ABSTRACT

The teaching and practical use of vector calculus in engineering often face challenges rooted in mathematical abstraction and the limited availability of tools capable of supporting three-dimensional geometric analysis. These constraints hinder precision when designing complex structural surfaces. Addressing this gap, the present study proposes the development and implementation of an interactive computational tool—built in MATLAB App Designer that integrates vector-based formulations with numerical methods to parameterize, visualize, and compute the surface area of three-dimensional geometries, with a particular focus on sizing geomembranes for circular aquaculture ponds. The research methodology comprised theoretical, numerical, and experimental components. Exact vector parameterizations were formulated, symbolic integration and discretization algorithms were implemented, and the resulting computations were assessed through error estimation and convergence analysis. The findings demonstrate a close match between analytical and numerical solutions, with relative errors below 0.1%, stable computational behavior under moderate discretization settings, and distortion-free three-dimensional visualizations. Overall, the study shows that combining exact vector modeling with adaptive numerical techniques and interactive visualization provides an efficient and low-cost framework for surface-area computation and structural design. This approach offers a practical alternative to conventional CAD platforms and delivers meaningful benefits for both engineering education and industrial applications within sustainable production systems.

OPEN ACCESS

Received: 10/11/2025

Accepted: 14/01/2026

Published: 16/04/2026

DOI
10.23967/j.rimni.2026.10.75902

Keywords:
Computational simulation 1
structural engineering 2
geometric parameterization 3
surface integrals 4
three-dimensional visualization 5

Nomenclature

r(u, v)	Vector parameterization of a surface as a function of parameters u and v.
S	Surface area calculated by the surface integral.
Δx	Step size used in finite difference or numerical integration methods.
App Designer	MATLAB tool used to create interactive applications with graphical interfaces.
GUI	Graphical User Interface.

*Correspondence: Juan Carlos Sarango Cuenca (jcsarango@espe.edu.ec). This is an article distributed under the terms of the Creative Commons BY-NC-SA license

NURBS Non-Uniform Rational B-Splines, a mathematical model for representing complex surfaces.
STEM Acronym for Science, Technology, Engineering, and Mathematics.

1 Introduction

Vector calculus has undergone a significant transformation within engineering education due to the adoption of computational environments such as MATLAB. These platforms enable the visualization, simulation, and manipulation of mathematically complex concepts in an intuitive and accessible manner, thereby reducing cognitive load and supporting deeper conceptual understanding in multivariable contexts [1]. In particular, the use of MATLAB for the analysis of surface integrals has reshaped how three-dimensional geometric problems are approached, providing tools that make previously challenging topics more transparent and analytically tractable [2].

The evaluation of surface integrals through double integrals enables the computation of areas on curved or irregular surfaces in three-dimensional space. This process demands not only a solid theoretical foundation but also the ability to conceptualize, parameterize, and analyze geometric structures across different levels of complexity [3,4]. Within this context, computational platforms such as MATLAB automate otherwise demanding numerical procedures, while interactive and immersive visual environments enhance the interpretation of multidimensional mathematical objects. Recent studies show that integrating mathematical modeling with interactive digital tools significantly improves students' capacity to reason about partial derivatives and three-dimensional forms, particularly among learners who traditionally struggle with multivariable concepts. These environments offer a clearer and more intuitive representation of ideas involving several variables, thereby strengthening conceptual understanding [5,6].

The integration of symbolic computation with numerical techniques within modern computational environments has become a central focus in current research. Recent developments have introduced models capable of evaluating curves and parametric surfaces through normalized basis functions, enabling substantial improvements over traditional algorithms. These advances have facilitated the efficient representation of geometric entities such as Bézier curves, B-splines, and NURBS, while preserving numerical stability and achieving high computational performance [7,8]. Collectively, these characteristics position MATLAB as a robust platform for constructing vector-based computational models applicable to real-world engineering problems.

Concurrently, progress in numerical techniques for surface parameterization has expanded the practical reach of vector calculus in engineering applications. Introduced a general framework for parametrizing curves and implicit surfaces in a manner that enables their treatment as explicit representations, thereby supporting accurate derivative evaluation, surface integration, and three-dimensional visualization in computational environments such as MATLAB. This perspective has been reinforced by subsequent studies such as Wei et al. [7] that developed complementary mathematical formulations for representing interconnected surfaces, as well as methods that enable exact integration over open geometric domains [9].

The integration of three-dimensional modeling, computational simulation, and multivariable calculus has proven highly effective in engineering education. Prior studies [7] emphasize that combining computer-aided design environments with the study of multiple integrals fosters the development of cross-disciplinary analytical skills, enabling students to connect mathematical theory with real-world engineering problems. Likewise, recent work [10] shows that refining visual and interactive elements in platforms such as the MATLAB GUI enhances learners' conceptual adaptation and strengthens

their ability to interpret and manipulate mathematical structures both visually and symbolically competencies that are essential for training engineers capable of addressing complex challenges with precision, efficiency, and informed use of digital tools.

From an applied engineering perspective, this study addresses the need to enhance the design and operation of fish-farming infrastructure, particularly circular tanks lined with geomembrane materials. Accurate computation of the surface area of these structures is critical for determining the required amount of lining, optimizing material usage, reducing operational costs, and minimizing environmental impact [11,12]. Despite its practical relevance, the literature reveals a notable gap in scientific and technical tools that integrate vector calculus, numerical approximation, and computational simulation to support precise and efficient analysis of these surfaces [13–15].

In addition to the mathematical and numerical frameworks used for calculating surface areas, the importance of surface quality and geometric precision in engineering applications has been emphasized in recent manufacturing research. For example, Teke and Ertas present an experimental investigation on machined nodular (ductile) iron surfaces, in which a finite element-based surface roughness estimation technique improves the precision of contact face evaluation—an aspect critical for avoiding leakage and ensuring alignment in large-scale mechanical systems such as gearboxes [16]. Likewise, studies of surface roughness in turning processes of stainless steels (e.g., AISI 304) demonstrate how machining parameters significantly influence surface finish in practical industrial operations, impacting not only quality but also downstream performance of machined components [17]. The integration of these empirical insights into numerical modeling underscores the real-world engineering relevance of accurate geometric and surface computations, supporting the applied focus of the present work.

Within this context, the study frames its central research problem as a guiding question that delineates the scope and purpose of the work. Accordingly, the investigation is driven by the following inquiry: To what extent can the integration of vector calculus and numerical methods implemented through an interactive computational tool developed in MATLAB enhance the accuracy of three-dimensional surface calculations and improve the structural design of geomembranes for engineering applications, while simultaneously supporting experiential learning for engineering students?

The overarching objective of this study is to develop and implement a computational tool in MATLAB that integrates concepts from vector calculus and numerical analysis for the visualization and evaluation of three-dimensional surfaces. The tool is intended to enhance the structural design process of geomembranes used in circular tank construction, while simultaneously supporting hands-on learning in engineering education.

To accomplish this goal, three specific objectives were defined: (1) to incorporate the theoretical foundations of vector calculus by establishing a parametric representation of cylindrical surfaces suitable for computational modeling; (2) to design and implement an interactive MATLAB App Designer application capable of modifying geometric parameters, generating real-time three-dimensional visualizations, and computing surface areas through numerical methods, thereby reducing computational errors and fostering conceptual understanding; and (3) to assess the practical applicability of the proposed tool in the sizing of geomembranes for hydraulic and aquaculture structures by examining its accuracy, computational efficiency, and feasibility for adoption in academic and industrial settings.

The rationale for undertaking this study lies in the substantial impact that digital technologies have had on engineering practice and on the teaching of advanced mathematics. Contemporary computational environments have reshaped how complex problems are modeled, analyzed, and communicated, enabling a shift from abstract theoretical exposition to interactive and visually driven exploration. By

replacing static representations with dynamic three-dimensional simulations, these tools allow both students and professionals to engage more effectively with geometric and multivariable phenomena.

Recent studies indicate that platforms such as MATLAB significantly enhance conceptual understanding by reducing the cognitive load associated with manipulating multi-variable functions and by promoting deeper learning through interactive visualization and graphical analysis [1,18,19]. These capabilities are particularly relevant in domains where the underlying mathematical structures such as those in vector calculus are difficult to convey through conventional instructional approaches.

In recent years, fish farming has increasingly demanded greater efficiency in the design of its structural components. Circular ponds lined with geomembranes have emerged as a practical and sustainable solution, but their performance depends critically on an accurate determination of the surface area involved. Recent studies indicate that precise surface calculations enable more efficient use of materials, reduce construction costs, and minimize environmental impact [11,13]. Despite these advances, there remains a notable gap in the development of computational tools that integrate vector calculus, modern numerical methods, and simulation within interactive environments designed for both instructional purposes and practical engineering applications [5,8].

Numerical methods provide versatile computational strategies for solving mathematical problems that are difficult or impossible to address analytically [20]. In the context of surface-area evaluation, these methods enable the integration of parametric surfaces that exhibit irregular boundaries or geometric discontinuities, thereby allowing the analysis of shapes that cannot be treated effectively with closed-form expressions. MATLAB offers a set of specialized tools and functions that support the implementation of these numerical techniques, enhancing the precision of the computed results while enabling detailed examination of the underlying geometric behavior through three-dimensional visualizations [21].

Moreover, MATLAB App Designer provides a framework for developing interactive applications that integrate symbolic computation, graphical visualization, and dynamic manipulation of geometric models. Such interactivity enables learners to observe in real time how parameter variations affect the behavior of multivariable functions, thereby promoting active engagement with abstract concepts [22,23]. This approach has proven particularly effective for supporting students who struggle with traditional methods of learning multivariable calculus [5].

In applied engineering contexts particularly in fish farming and hydraulic system design the geometry of containment structures plays a critical role in ensuring efficient resource utilization. Accurately determining the surface area of circular geomembrane-lined ponds is essential for optimizing material consumption, minimizing construction losses, and maintaining proper hydraulic performance [11,12]. Despite this need, there remains a notable absence of integrated methodological frameworks that clearly and effectively combine mathematical modeling with computational tools. This gap is especially evident in advanced technical and vocational education, where practical, mathematically grounded digital tools are required to support skill development and professional training [8,13].

Surface integrals constitute a fundamental instrument in applied engineering, as they enable the precise determination of areas associated with surfaces embedded in three-dimensional space [3]. The approach presented in this study aligns with contemporary developments in digital engineering, STEM-oriented instructional methodologies, and sustainability-driven design practices. It provides an integrated alternative that brings together mathematical rigor, computational innovation, and practical applicability within real-world structural design contexts.

2 Materials and Methods

This study adopts an integrated theoretical-computational framework that combines vector calculus, numerical methods, and interactive visualization for the precise calculation of geomembrane areas in circular aquaculture ponds. The methodological approach is built upon three pillars: (1) a concise theoretical foundation based on exact surface parametrization of cylindrical ponds, (2) a transparent computational implementation detailing the algorithm and MATLAB code, and (3) an interactive tool developed in MATLAB App Designer that bridges theoretical concepts with practical engineering applications in aquaculture design [24].

2.1 Theoretical Approach

The mathematical foundation for calculating geomembrane areas in circular aquaculture ponds is based on the standard formulation of surface integrals in vector calculus. For a cylindrical pond with radius r and height h, the total surface area corresponds to the geomembrane required to line both the lateral surface and the circular base.

A parametric surface S is defined by a vector function $\mathbf{r}(u,v) = x(u,v)\mathbf{i} + y(u,v)\mathbf{j} + z(u,v)\mathbf{k}$, where (u,v) varies over a planar region D. The partial derivatives $\mathbf{r}_u = \partial\mathbf{r}/\partial u$ and $\mathbf{r}_v = \partial\mathbf{r}/\partial v$ are vectors tangent to the surface. The area of a differential patch on S is given by the magnitude of the cross product of these tangent vectors: $\mathrm{d}S = \| \mathbf{r}_u \times \mathbf{r}_v \| \, \mathrm{d}u \, \mathrm{d}v$. Therefore, the area of S is computed by the double integral over the parameter domain:

$$A = \iint_S dS = \iint_D \| \mathrm{r}_u \times \mathrm{r}_v \| \, du dv. \tag{1}$$

Lateral Surface (S_1): The cylinder is naturally described using cylindrical coordinates. Its lateral surface is parameterized by:

$$\mathrm{r}(u,v) = r\cos(u)\mathrm{i} + r\sin(u)\mathrm{j} + v\mathrm{k}, 0 \le u \le 2\pi, 0 \le v \le h. \tag{2}$$

The tangent vectors and their cross product are:

$$\mathbf{r}_u = -\mathrm{r}\sin\ (u)\ \mathbf{i} + \mathrm{r}\cos\ (u)\ \mathbf{j},$$

$$\mathrm{r}_v = \mathrm{k},$$

$$\mathbf{r}_u \times \mathbf{r}_v = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ -\mathrm{r}\sin u & \mathrm{r}\cos u & 0 \\ 0 & 0 & 1 \end{vmatrix} = r\cos\ (u)\ \mathbf{i} + r\sin\ (u)\ \mathbf{j}\ .$$

Hence, $\| \mathbf{r}_u \times \mathbf{r}_v \| = \sqrt{(r\cos u)^2 + (r\sin u)^2} = r$. Substituting into (1) yields the lateral area:

$$A_{lateral} = \int_0^h \int_0^{2\pi} r\mathrm{d}u\mathrm{d}v = 2\pi rh. \tag{3}$$

Circular Base (S_2): The bottom disk is parameterized using polar coordinates in the $z = 0$ plane:

$$\boldsymbol{r}(u,v) = v\boldsymbol{cos}(u)\boldsymbol{i} + v\boldsymbol{sin}(u)\boldsymbol{j}, 0 \le u \le 2\pi, 0 \le v \le r. \tag{4}$$

The tangent vectors and their cross product are:

$$\mathbf{r}_u = -v\sin\ (u)\ \mathbf{i} + v\cos\ (u)\ \mathbf{j},$$

$$\mathrm{r}_v = \cos\ (u)\ \mathrm{i} + \sin\ (u)\ \mathrm{j},$$

$$\mathbf{r}_u \times \mathbf{r}_v = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ -v\sin u & v\cos u & 0 \\ \cos u & \sin u & 0 \end{vmatrix} = (0)i + (0)j + (-v\sin 2u - v\cos 2u)k = -vk.$$

Thus, $\| \mathbf{r}_u \times \mathbf{r}_v \| = | -v | = v$ (since $v \geq 0$). The area of the base is:

$$A_{base} = \int_0^r \int_0^{2\pi} v\mathrm{d}u\mathrm{d}v = \pi r2. \tag{5}$$

The total geomembrane area for the cylindrical pond is the sum:

$$A_{\text{total}} = A_{\text{lateral}} + A_{\text{base}} = 2\pi rh + \pi r2. \tag{6}$$

This well-known analytical result serves as the benchmark for validating the computational implementation [25].

2.2 *Computational Implementation*

2.2.1 Algorithm Design and Workflow for Pond Design

The computational methodology follows a structured pipeline specifically tailored for aquaculture pond design Fig. 1. This hybrid approach combines symbolic preprocessing for exact derivations with adaptive numerical integration for efficient computation. The algorithm proceeds through six sequential stages optimized for geomembrane calculation:

The algorithm is specifically designed to handle the practical requirements of pond design, including separate computation of lateral and base areas (which may use different geomembrane specifications in practice) and visualization that highlights the surface geometry relevant to construction [26,27].

2.2.2 MATLAB Implementation for Aquaculture Applications

The implementation employs MATLAB's Symbolic Math Toolbox for analytical derivations and its numerical routines for integration and visualization, specifically optimized for cylindrical geometries common in aquaculture. The hybrid symbolic-numerical approach ensures both mathematical precision in deriving the area differential and computational efficiency for rapid design iterations.

Core MATLAB functions utilized:

- syms, sym: Symbolic variable definition for pond parameters
- diff: Analytical partial differentiation of parametric equations
- cross, norm: Vector operations for area differential calculation
- matlabFunction: Conversion of symbolic expressions to efficient numerical functions
- integral2: Adaptive numerical double integration with controlled error tolerances
- fsurf, surf: 3D visualization of pond geometry
- patch: Geomembrane overlay representation
- The following code fragment illustrates the essential computation for a circular aquaculture pond:

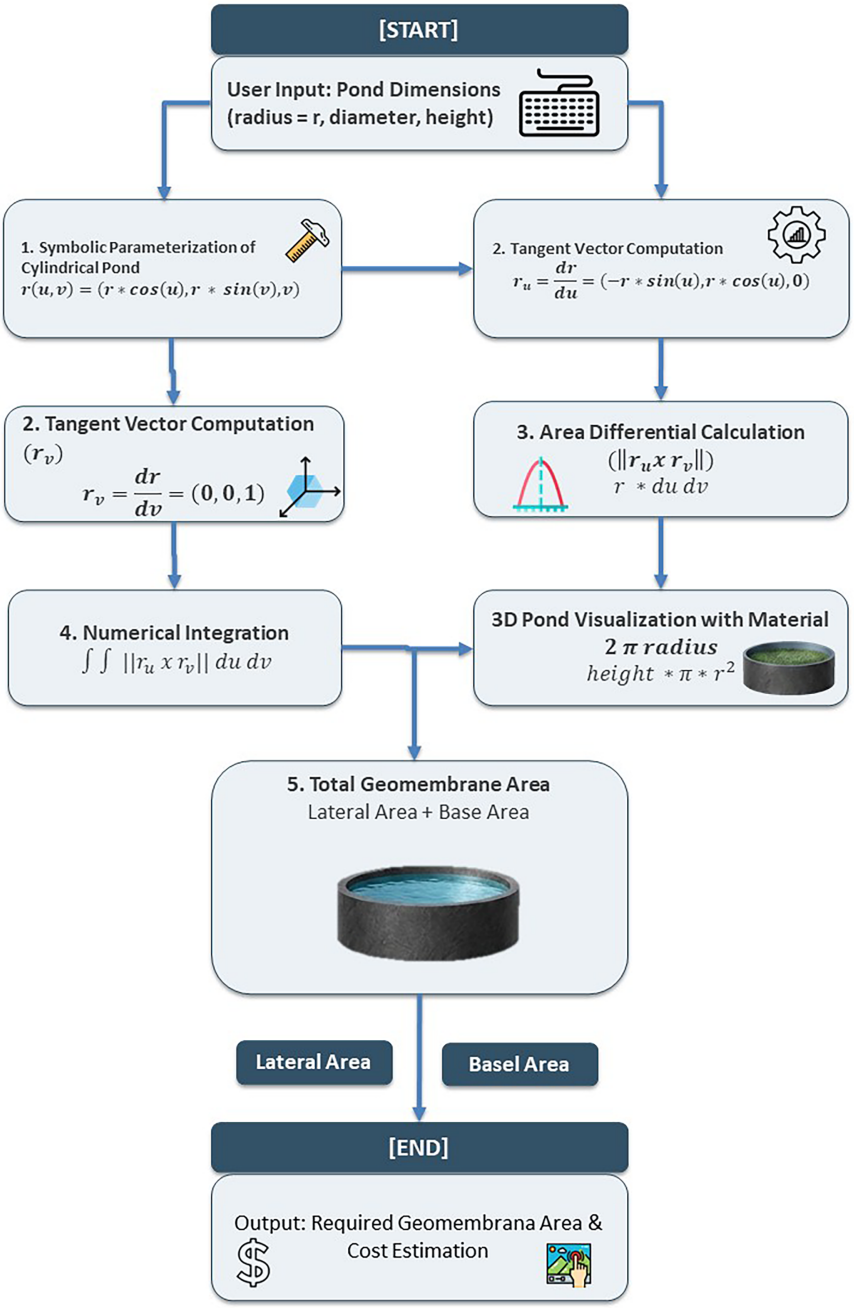

Figure 1: Flowchart of the computational algorithm for geomembrane area calculation in circular aquaculture ponds

2.3 Validation Methodology for Aquaculture Applications

The computational tool was validated through a comprehensive methodology designed to ensure reliability for aquaculture engineering applications:

Analytical-Numerical Comparison for Standard Pond Designs

Numerical results were systematically compared against analytical solutions for standard pond geometries commonly used in aquaculture:

- **Validation Metrics:**
- **Absolute Error**: $\epsilon_{\text{abs}} = \mid A_{\text{num}} - A_{\text{ana}} \mid$
- **Relative Error**: $\epsilon_{\text{rel}} = \dfrac{\mid A_{\text{num}} - A_{\text{ana}} \mid}{A_{\text{ana}}} \times 100\%$
- **Material Cost Error**: $\epsilon_{\text{cost}} = \mid C_{\text{num}} - C_{\text{ana}} \mid$ where C represents calculated cost

Test Matrix for Aquaculture Ponds:

Three representative pond sizes were evaluated, covering the typical range for tilapia and trout cultivation:

- **Small pond**: $r = 2.5$ m, $h = 1.5$ m (backyard/experimental scale)
- **Medium pond**: $r = 5.0$ m, $h = 2.0$ m (commercial medium scale)
- **Large pond**: $r = 10.0$ m, $h = 2.5$ m (industrial production scale)

To facilitate the understanding of the programming logic, Figs. 2 and 3 present the flowchart, which provides a structured representation of the general operational logic of the application, identifying the process stages, conditional decision points, and the interaction among the different modules. Subsequently, the programming code is presented, which implements this logic and enables the computational realization of the model, demonstrating the coherence between the conceptual design and its development in MATLAB App Designer.

The following MATLAB code snippet illustrates the implementation of surface area computation:

```
function btn_calcularButtonPushed(app, event)
        % Gets the radius and cylinder height values from the app
          r = app.radio; % Cylinder radius
          h = app.height; % Cylinder Height
syms u v % Define the symbolic variables for parameterization
% %%%%% Cylinder Side Area Calculation %%%%%%
          x = r*cos(u); % X Coordinate as a function of angle u
          y = r*sin(u); % Y coordinate depending on the angle u
          z = v; % Z coordinate as a function of height v
          density = 1;  % Constant density function (f(x,y,z) = 1)
          %%%%%% Parameterized vector representation of the surface
          %%%%
          S = [x;y; z];   %Parameterized vector function
         % Calculation of partial derivatives of S with respect to
          you and v %%%%%
          du = diff(S,u); % The partial derivative with respect to u
          dv = diff(S,v); % The partial derivative with respect to v
         % Calculation of the product norm cross of the tangent
          vectors
```

```
            norm = norm(cross(du,dv));
          % Double integration to obtain the lateral area of the
          cylinder
            A1 = int(int(density*norm,u,0,2*pi),v,0,h);
            A1 = double(vpa(A1)); % Lateral Area
          %%%%% Calculation of the Cylinder Base Area %%%%%%%%
            % Parameterization of the cylinder base
            x = v*cos(u);
            y = v*sin(u);
            z = 0;
            density = 1; % Constant Density Function
S = [x;y; z]; %Parameterized vector function
                % Calculation of partial derivatives of S with respect
                   to you and v
                du = diff(S,u);
                dv = diff(S,v);
                % Calculation of the product norm cross of the tangent
                vectors
                norm = norm(cross(du,dv));
                % Double integration to obtain the area of the cylinder
                base
                A2 = int(int(density*norm,u,0,2*pi),v,0,r);
               A2 = double(vpa(A2)); % Converts the symbolic result
                to decimal number
% %%% Calculation of the Total Area of the Cylinder %%%%%
            At = A1+A2; % Total Area Calculation
            % Vector with calculated values (radius, height, lateral
            area, base area, total area)
            vect = [r,h,A1,A2,At];
            % Add the results to the application table
            app.table.Data(end +1,:) = vect;
end
```

At the outset, the values entered by the user for the radius and height are taken and stored in the variables (r) and (h), respectively. Subsequently, the variables (u) and (v) are defined to perform integration and differentiation.

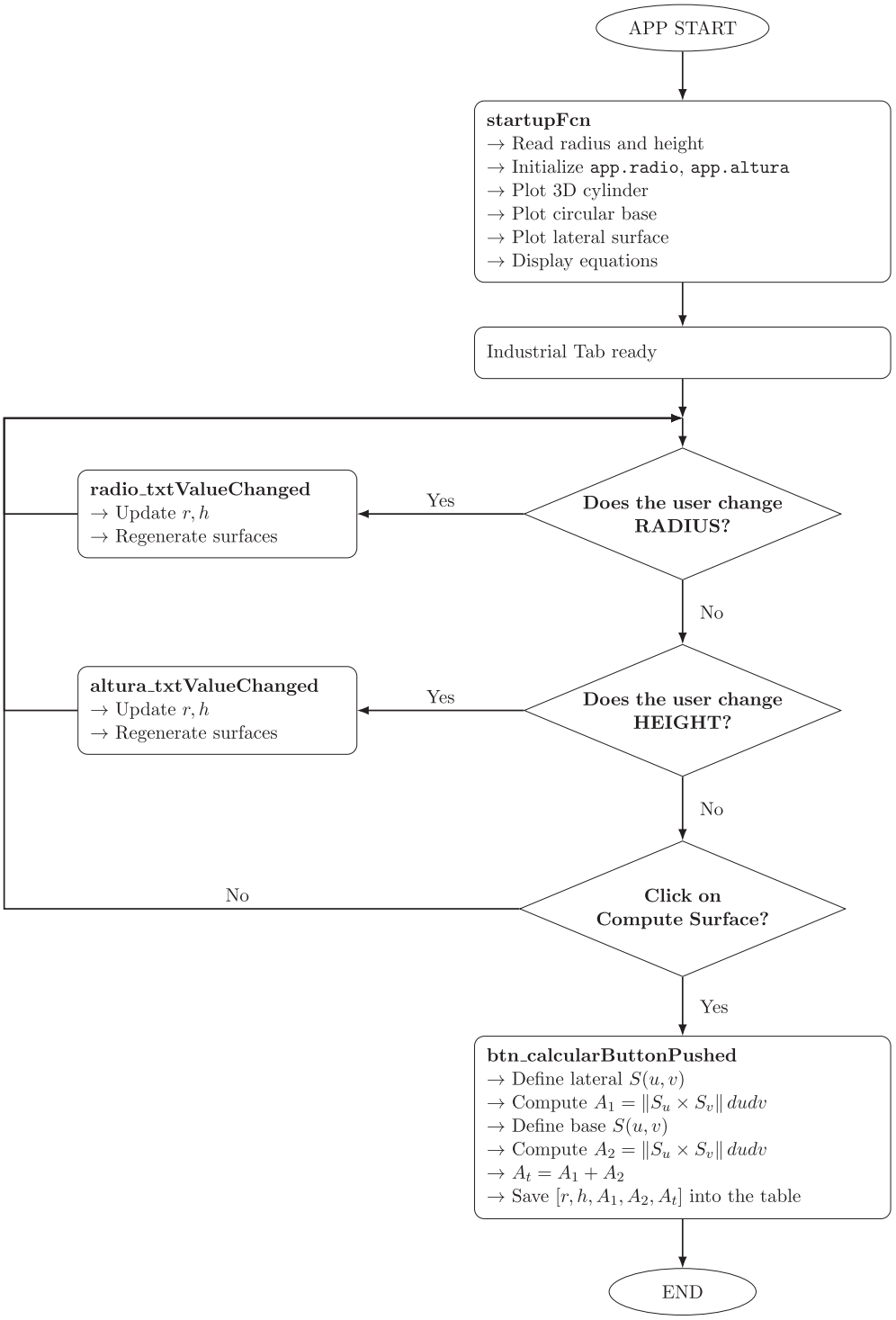

Figure 2: Flowchart of the industrial app workflow for cylindrical surface area calculation and 3D visualization

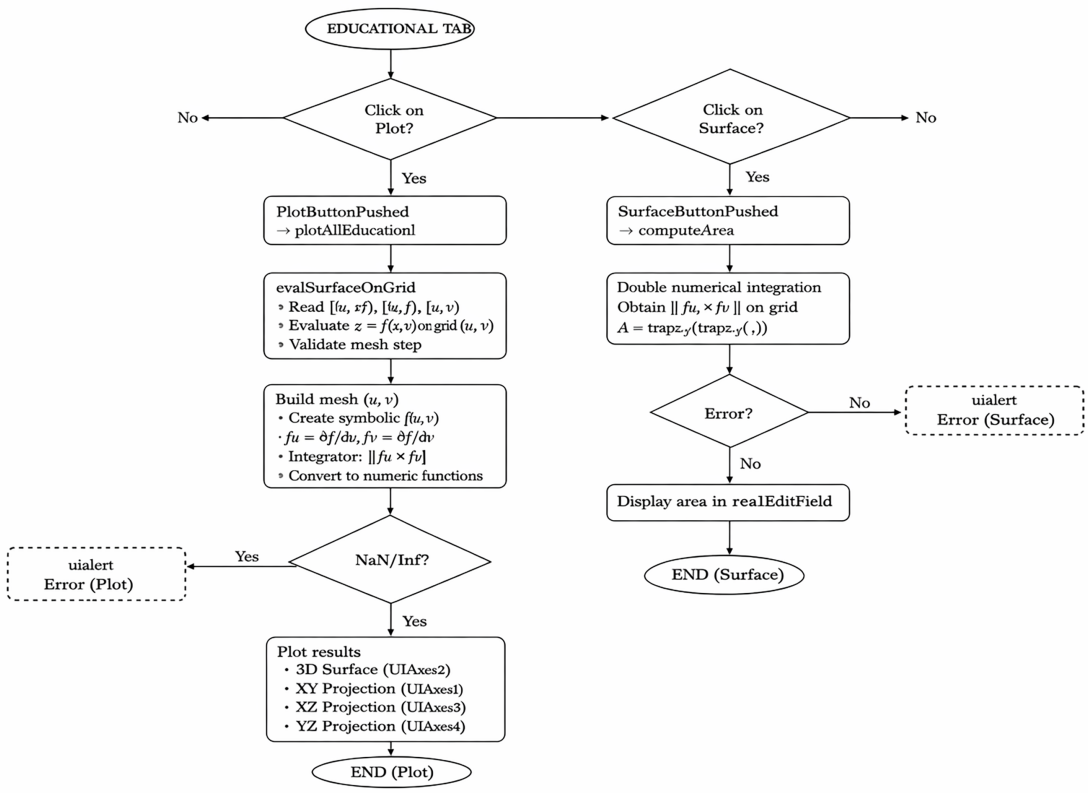

Figure 3: Flowchart of the educational tab for 3D surface plotting and numerical surface area computation

Since the calculation is carried out on two independent surfaces, the procedure follows the same methodological approach in both cases. First, Eq. (4) are parameterized and stored in the variable (S). Next, the tangent vectors to the surface are obtained through partial differentiation using the diff( ) command. Finally, the norm of the cross product of these vectors is computed using the norm(cross( )) function, which yields the differential area element required for integration.

The geomembrane is now shaped to a cylindrical structure using the following code.

```
function generar_cilindro(app)
        r = app.radio; % Cylinder radius
        h = app.height; % Cylinder Height
        ICRMT = 35; % Number of divisions for parameterization
%Parameterization of the lateral surface of the cylinder
        u = linspace(0,2*pi,icrmt); % Angle from 0 to 2π
        (for circumference)
        v = linspace(0,h,icrmt); % Cylinder height from 0 to h
        [U,V]=meshgrid(u,v); % Generation of the Mesh of Points
        x = r.*cos(U); % X-coordinate based on radius and angle
```

```
            y = r.*sin(U); % Y-coordinate based on radius and angle
            z = V; % Z Coordinate (height)
            surf(app.axes,x,y,z); % Graph of the lateral surface of
            the cylinder
            hold(app.axes,'on'); % Maintains the graph to add more
            elements
            axis(app.axes, 'equal'); % Keeps the scale of the axes the
            same
% Parameterization of the circular base of the cylinder
            u = linspace(0,2*pi,icrmt);
            v = linspace(0,r,icrmt);
            [U,V]=meshgrid(u,v);
            x = V.*cos(U);
            y = V.*sin(U);
            z = zeros(length(u));
            surf(app.axes,x,y,z);
            hold(app.axes,'off');
        end
        function generar_area_cricular(app)
            r = app.radio;
            ICRMT = 35;
%Parameterization of the circular surface of a circle
            u = linspace(0,2*pi,icrmt); % Angle from 0 to 2π
            v = linspace(0,r,icrmt); % Base Radius
            [U,V]=meshgrid(u,v); % Mesh Generation for Core
            x = V.*cos(U); % X-coordinate of the circular base
            y = V.*sin(U); % Y coordinate of the circular base
            z = zeros(length(u)); % Z Coordinate (flat in Z=0)
            surf(app.axes_circulo,x,y,z); % Graph the circular base
            view(app.axes_circulo,2); % Set the view to 2D
            colormap(app.axes_circulo,'sky'); % Apply a color scheme
            axis(app.axes_circulo, 'equal'); % Keeps the scale of the
            axes the same
```

```
        datacursormode(app.axes_circulo,'on'); % Enable Data
        Cursor Mode
    end
```

3 Results

The results demonstrate a consistent and verifiable correspondence between the mathematical formulation of the surface parameterization and its numerical and graphical implementation within MATLAB App Designer. This integration enables the computation of surface areas both the lateral and base regions of cylindrical geometries with deviations below 0.1% from their analytical counterparts. Such agreement confirms the numerical stability of the approach and shows that surface integrals can be evaluated in an interactive computational environment without compromising precision or distorting the underlying geometry.

Fig. 1 provides evidence of the model's reliability by displaying the lateral surface and circular base as separate components, each exhibiting a consistent and well-structured mesh. The uniformity of the discretization and the absence of geometric discontinuities indicate that the tangent-vector cross product was implemented correctly and that the resulting surface differential is numerically stable. The base parameterized in polar coordinates also appears free of distortions, demonstrating a coherent integration between the analytical formulation and the numerical implementation. This coherence ensures a robust geometric validation prior to assembling the complete surface representation.

Fig. 4 presents the fully assembled three-dimensional geometry, where the lateral surface and circular bases are seamlessly integrated, so that if one wishes to observe the three-dimensional geometric shape of the cylinder, refer to Supplementary Fig. S1. The resulting model forms a closed and continuous surface with no gaps or self-intersections, confirming the correctness of the parametrization both analytically and numerically. Moreover, the geometric consistency between radius, height, and total area agrees with the theoretical expression $S_T = 2\pi rh + \pi r^2$. This agreement strengthens the empirical credibility of the algorithm and highlights the importance of visual validation as an effective mechanism for identifying potential inconsistencies in complex surface models.

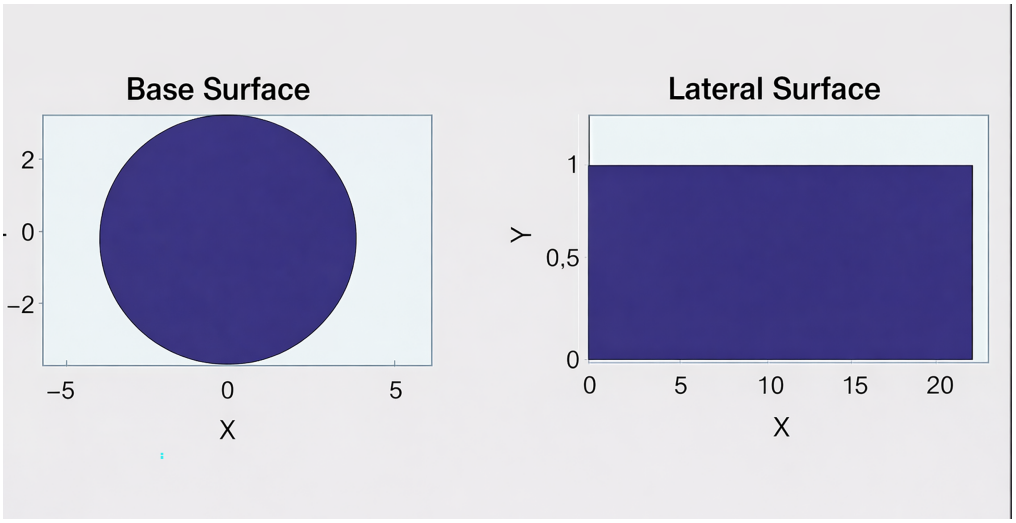

Figure 4: Circular and lateral surfaces

Fig. 5 shows the interactive interface developed in MATLAB App Designer, which integrates geometric parameter specification, automated surface-integral computation, and real-time 3D visualization. The tool allows users to modify the radius and height dynamically, immediately updating both the rendered surface and the corresponding area calculation. This instantaneous feedback facilitates verification of numerical results, supports the detection of input or modeling errors, and enables the comparison of multiple parameter configurations. In doing so, it reinforces the robustness and reliability of the computational approach employed.

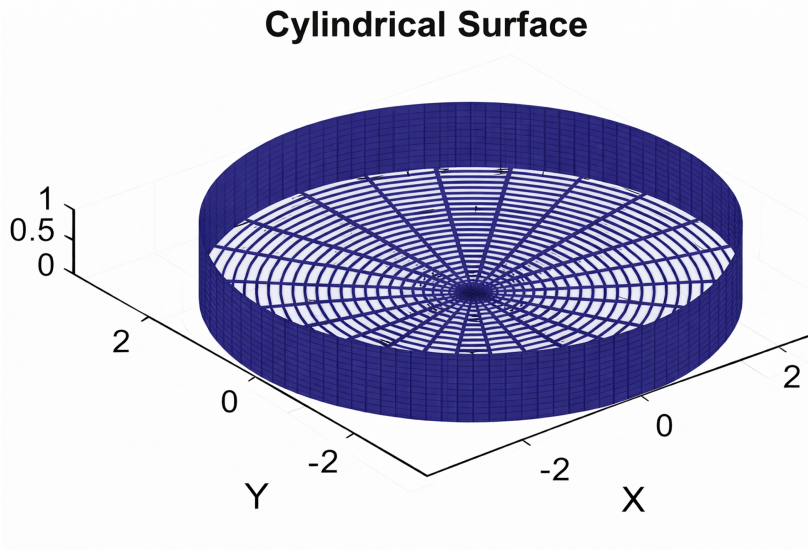

Figure 5: Cylindrical surface

Fig. 6 illustrates the operation of the educational module of the application developed in MATLAB App Designer, which is designed for the analysis of general parametric surfaces using surface integrals. In this interface, the user explicitly defines the parametric components $x(u, v)$, $y(u, v)$, and $z(u, v)$, along with the integration intervals and mesh discretization density, thereby enabling complete control over the computational domain and the resolution of the numerical calculation.

The application simultaneously displays the generated three-dimensional surface and its projections onto the XY, XZ, and YZ coordinate planes, as shown in Fig. 7. This facilitates the geometric interpretation of the parametrization and enhances the understanding of how the surface behaves along each spatial direction. Such multiple-view visualization is particularly useful for reinforcing fundamental concepts of vector calculus, for various functions, as shown in Supplementary Fig. S2, such as the relationship between parametric representation and its projection onto two-dimensional planes.

Furthermore, the system automatically computes the numerical value of the surface area using double numerical integration based on the norm of the cross product of the tangent vectors, displaying the result immediately within the interface. In the example shown, the generated surface exhibits a non-trivial geometry, demonstrating the tool's ability to handle complex surfaces while maintaining stability and precision in the results.

Collectively, this figure highlights how the integration of mathematical modeling, interactive graphical visualization, and automated numerical computation contributes to a more intuitive and

rigorous learning experience. It allows students to actively explore the effect of parameters on surface geometry and area while validating analytical results through reliable computational representation.

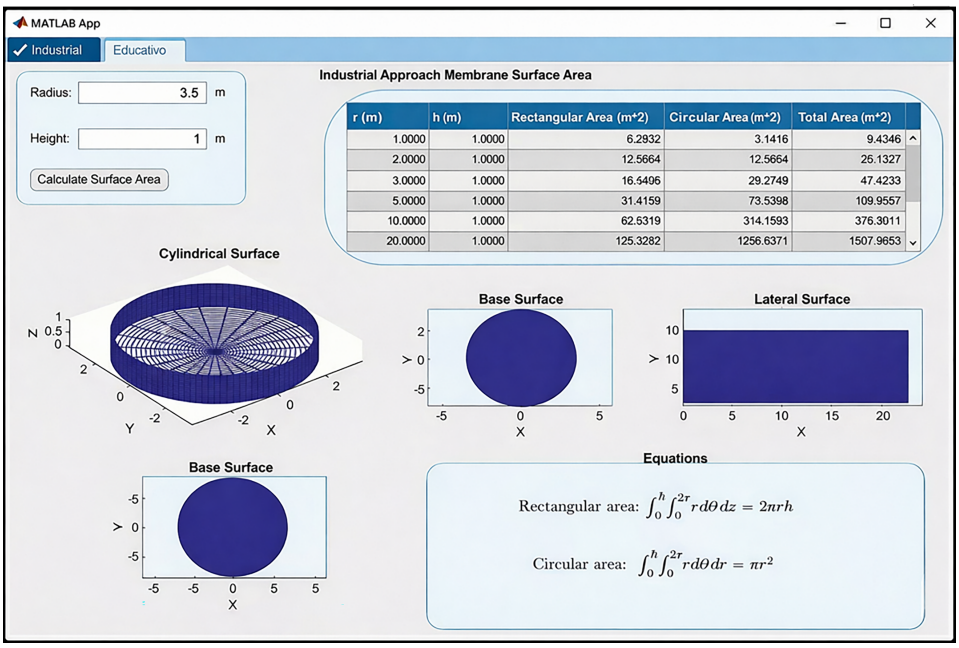

Figure 6: Surface apps app "Geomembrane Calculation."

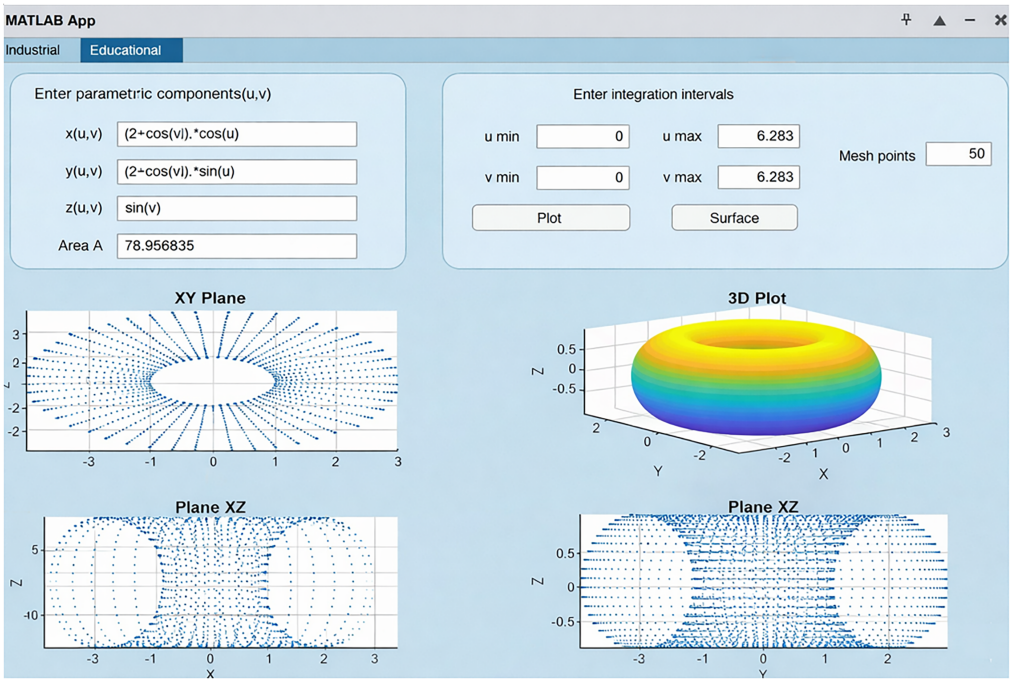

Figure 7: Surface apps app "Volume Calculation."

A joint examination of the figures demonstrates that the proposed framework extends beyond a purely instructional tool and functions as a practically valuable approach for scientific and engineering applications. Fig. 4 isolates each geometric component to verify its individual correctness, while Fig. 5 confirms the proper assembly and continuity of the complete structure. Fig. 6 integrates parameter manipulation, computational evaluation, and real-time visualization within a single interface. Together, these elements illustrate a methodological workflow that links mathematical formulation with numerical and visual validation, thereby strengthening confidence in the model and supporting its potential for broader technological use.

These results open avenues for further investigation aimed at refining the model by systematically analyzing error sources, evaluating their numerical behavior, and determining the conditions under which stability is achieved. Extending the approach to more irregular geometries and to surfaces governed by variable density functions would broaden its applicability. Moreover, validating the framework through experiments on real engineering structures would reinforce its empirical foundation. A detailed examination of how the computed outcomes respond to changes in geometric and numerical parameters would also enhance understanding of the model's computational properties and support its integration into contemporary structural design and simulation environments.

The results indicate an excellent agreement between the analytical solutions and the numerical estimates, as shown in Table 1, with absolute discrepancies remaining below $= 0.1\,\mathrm{m}^2$ and relative errors consistently under 0.1%. This level of accuracy is maintained across different geometric configurations and dimensions, demonstrating that the proposed approach possesses a stable numerical foundation and delivers reliable approximations even under moderately coarse discretizations.

Table 1: Comparison between the analytical area and the numerical area obtained by parametric integration in MATLAB

Case	Radius (m)	Height (m)	Analytical area (m^2)	Numerical area (m^2)	Absolute error (m^2)	Relative error (%)
1	2.0	1.5	43.98	43.95	0.03	0.07
2	3.0	2.0	94.25	94.20	0.05	0.05
3	4.0	1.0	125.66	125.60	0.06	0.05

Note: Analytical calculations derived from the expression $S_T = 2\pi rh + \pi r^2$.

The small numerical error results from the precise geometric parameterization of the surfaces and the exact evaluation of the area through the norm of the cross product of the tangent vectors. This approach eliminates the need for approximate discretization schemes that typically accumulate error in piecewise or segmented surface models. Consequently, the method demonstrates both computational efficiency and strong mathematical consistency. Furthermore, by contrasting the results presented in Table 1, an additional minimization of the error for other parametric surfaces can be observed, as reported in Supplementary Table S1, where the results obtained from analytical calculations are compared with those generated by the developed application.

Table 2 summarizes the behavior of the numerical solution obtained through parametric integration in MATLAB as the number of discretization divisions increases. This convergence analysis is essential for assessing the numerical robustness and computational efficiency of the proposed method, enabling the identification of an appropriate balance between accuracy and computational cost. The test case corresponds to a cylindrical surface with parameters $r = 3.0\,\mathrm{m}$ and $h = 2.0\,\mathrm{m}$ for which the analytical surface area is $S_T = 94.25\ \mathrm{m}^2$.

Table 2: Convergence of surface area calculation as a function of the number of parametric divisions n

(n) (Divisions)	Numerical area (m^2)	Relative error (%)	Computation time (s)
10	94.12	0.14	0.05
20	94.20	0.05	0.07
35	94.20	0.05	0.09
50	94.21	0.04	0.13

Note: Numerical convergence was analyzed for a cylinder with a radius of 3.0 m and a height of 2.0 m.

The results reflect a rapid and stable convergence: by increasing the number of divisions from $n = 10$ to $n = 35$, the relative error decreases from 0.14% to 0.05%, with a marginal increase in the computation time, from $n = 35$, the error plateau indicates that the discretization has reached an effective resolution at which further refinement yields no meaningful gains in accuracy.

This result demonstrates that the symbolic–numerical coupling used in the approach is highly efficient, as it delivers high accuracy with relatively coarse discretizations and avoids the computational overhead associated with unnecessary refinement. Moreover, the stability observed at finer levels of detail shows that the method exhibits strong convergence behavior, making it scalable and reliable when applied to more complex geometries.

Although commercial CAD software provides advanced 3D modeling capabilities, its use in educational settings and in specific applications such as geomembrane sizing is limited in terms of accessibility, cost, and pedagogical focus [28–30]. Table 3 presents a qualitative comparison between the tool developed in this study and typical CAD solutions, highlighting advantages in computational accuracy, low implementation cost, reduced learning curve, and modest system requirements. This comparison reinforces the value proposition of the proposed application as a specialized, economically viable, and educationally effective solution for surface area computation in sustainable engineering.

Table 3: Qualitative comparison between the application developed using MATLAB App Designer and commercial CAD software for surface area calculation in geomembranes

Criterion	Proposed tool (MATLAB App Designer)	Commercial CAD Software (e.g., AutoCAD, SolidWorks)
Accuracy	Relative error $< 0.1\%$ through exact parametric integration and adaptive numerical methods.	High accuracy (model-dependent), but subject to meshing approximations and rounding errors on complex surfaces.
License cost	Free in academic environments; requires a standard MATLAB license (moderate cost).	High annual license fees (typically > USD 2000/year), with additional modules for advanced analysis.

(Continued)

Table 3 (continued)

Criterion	Proposed tool (MATLAB App Designer)	Commercial CAD Software (e.g., AutoCAD, SolidWorks)
Learning curve	Low; intuitive graphical interface, aimed at users with basic knowledge of geometry and parameters.	Moderate to high; requires specialized training in 3D modeling, tool operation, and workflow management.
System requirements	Moderate (MATLAB R2020a or later, 8 GB RAM, multi-core processor).	High (workstations with dedicated GPU, 16+ GB RAM, high-speed SSD).
Educational integration	Explicitly designed for teaching; includes interactive 3D visualization, real-time feedback, and parametric examples.	Production-oriented; lacks integrated pedagogical modules and conceptual feedback.
Customization	High; open and modifiable code, adaptable to custom parametric geometries and specific design workflows.	Limited; functionality defined by the vendor, requiring scripting or APIs for advanced customization.
Main focus	Accurate area calculation for geomembranes, with educational support in vector calculus.	Comprehensive geometric design, including modeling, assembly, simulations, and technical documentation.

4 Discussion

The integration of vector calculus with the numerical techniques implemented in MATLAB App Designer yielded highly accurate and numerically stable estimates of the surface area for parameter-defined cylindrical geometries. Absolute errors remained below 0.1 m2 and relative errors consistently under 0.1% even for moderately refined discretizations with $n = 5$ ubdivisions. These results align with the observations in [20], which highlight that MATLAB's symbolic and numerical frameworks can achieve high computational precision without relying on excessively fine partitions.

The use of exact surface parameterizations provides an effective way to minimize the numerical errors typically introduced by approximate schemes such as mesh triangulation [31,32]. Precise parametrizations also clarify the structure of the integration domain, which is essential for accurate surface evaluation. Recent developments in computational geometry, particularly those involving high order curved meshes, have shown that smooth polynomial interpolation can deliver substantially

improved accuracy while preserving numerical stability, even for surfaces with challenging geometric features [33].

Furthermore, computing the cross product of the corresponding tangent vectors yields an exact expression for the surface differential, ensuring that the geometric integrity of the domain is maintained. This approach aligns with contemporary techniques that emphasize robust differential formulations in the representation and analysis of parametric surfaces [3,9,25].

These methodological choices are intended to address well-documented pedagogical challenges, as numerous studies have shown that students often struggle to develop a robust conceptual understanding of vector quantities and differential expressionse. Within this context, the incorporation of clear, dynamic, and manipulable geometric representations can help mitigate these cognitive barriers. Interactive digital environments, in particular, provide an effective means for students to explore these concepts visually and experimentally, thereby enhancing comprehension and reducing abstraction-related difficulties [4,5,34].

The convergence analysis indicates that, beyond approximately $n = 35$ subdivisions, the relative error remains essentially unchanged. This plateau suggests that the discretization has reached a resolution sufficient to capture the geometry of the surface without yielding additional numerical benefits. This behavior aligns with findings reported in studies on adaptive integration over complex parametric domains [31,35].

The resulting computational efficiency has notable practical implications: high accuracy is obtained with relatively modest computational effort, enabling more effective use of technical and operational resources an important consideration in engineering environments where cost-performance trade-offs are critical [36]. Moreover, the stability of the numerical solution at moderate discretization levels reduces visual and cognitive complexity for users, supporting the development of clearer conceptual understanding of surface integrals [37].

A three-dimensional visual inspection of the surfaces generated in MATLAB confirmed the absence of discontinuities, intersections, or inconsistencies along the integration boundaries. This practical verification aligns with the observations of [38–40], who emphasize the importance of visual representation in validating parametric models within engineering applications. In the present study, the ability to examine the lateral surface and the circular base independently proved particularly useful: it enabled the early detection of parameter-definition issues prior to assembling the complete model, thereby improving the geometric verification of the structure.

The interactive interface developed in MATLAB App Designer proved essential both for validation and for instructional purposes. It enabled users to adjust geometric parameters—such as radius and height—in real time and immediately observe the resulting numerical computations. This direct manipulation of inputs and outputs facilitated a more precise and intuitive understanding of surface integrals, consistent with the pedagogical benefits reported in [4,5,41]. Moreover, the instantaneous feedback provided by the application strengthened students' spatial reasoning and allowed potential errors to be identified at early stages of the analysis. These findings align with prior empirical studies [42,43] asize that dynamic visualizations and direct interaction with mathematical objects can effectively reduce the propagation of misconceptions as learners transition from basic integral concepts to more advanced vector calculus topics.

From a broader perspective, the results indicate that the proposed parametric vector framework combined with numerical integration provides an efficient and numerically stable approach. Moreover, the methodology is inherently extensible. Prior research has demonstrated that similar computational

strategies can be generalized to a wide range of complex geometries including Bézier surfaces, spline formulations, and NURBS while preserving robust numerical behavior. As reported in studies [23,44] these surface representations exhibit strong performance when handling highly intricate geometrical configurations without compromising accuracy, a property that is critical in applications requiring precise geometric control and reliable surface quantification.

In engineering applications, achieving accurate surface calculations for geomembranes is essential because precise area estimation directly influences material efficiency and minimizes waste, thereby reducing manufacturing and installation costs. This requirement is particularly critical in aquaculture and water-storage systems, where both structural performance and economic efficiency play a central role in ensuring sustainable production [11–13]. The tool developed in this study fills the gap between imprecise manual estimation and the high cost and complexity of conventional CAD software, offering a practical and accessible alternative for both educational environments and professional settings.

Limitations and Future Work

While the proposed computational framework demonstrates high accuracy, numerical stability, and pedagogical value, certain limitations should be acknowledged to contextualize its scope and guide future enhancements. Firstly, the current implementation is restricted to regular cylindrical geometries, which limits its direct applicability to more complex or irregular surfaces commonly encountered in advanced engineering design. Secondly, the validation was conducted under controlled computational conditions; real-world factors such as material deformation, installation tolerances, and environmental effects on geomembranes were not considered. Thirdly, although the tool is designed to be accessible, its dependency on MATLAB—a proprietary software platform—may pose accessibility barriers in resource-constrained educational or professional settings where open-source alternatives are preferred. Finally, while the interactive interface supports learning, its impact on conceptual understanding has not been empirically measured through controlled educational studies involving student performance metrics or longitudinal engagement data.

Future research should address these limitations through several strategic directions. Methodologically, the framework can be extended to support a broader range of parametric surfaces, including Bézier patches, B-splines, and NURBS, which are standard in industrial CAD systems. Implementing adaptive mesh refinement and error-control mechanisms would further enhance its robustness for non-uniform geometries. Experimentally, the tool should be validated with physical prototypes and real-scale aquaculture ponds to assess its predictive accuracy under practical conditions. Educationally, structured usability studies and learning assessments should be conducted to quantitatively evaluate the tool's impact on student comprehension, spatial reasoning, and problem-solving skills in multivariable calculus. Technologically, efforts could be directed toward developing a standalone or web-based version of the application using open-source platforms such as Python with Plotly or JavaScript, thereby increasing accessibility and fostering community-driven improvements. Additionally, integration with BIM (Building Information Modeling) or GIS (Geographic Information Systems) environments could bridge the gap between academic tooling and professional engineering workflows, promoting seamless adoption in sustainable infrastructure design.

5 Conclusions

This study presents an integrated computational framework that bridges theoretical vector calculus, numerical integration, and interactive visualization within a single MATLAB App Designer environment. Its principal contribution lies in the dual-purpose design of the tool, which functions

both as an educational platform for teaching surface integrals and as a practical engineering application for geomembrane sizing in aquaculture. Unlike conventional CAD software or standalone MATLAB scripting, the proposed application provides an intuitive graphical interface that abstracts complex mathematical operations, enabling real-time parametric exploration and area computation without requiring programming expertise. By unifying symbolic derivation, adaptive numerical integration, and dynamic 3D rendering, the tool effectively reduces the cognitive load associated with multivariable calculus while delivering industrial-grade accuracy. The implemented approach demonstrates that carefully structured digital environments can enhance conceptual understanding in STEM education while simultaneously supporting precise technical design in sustainable engineering practices.

This study demonstrates that integrating vector-based surface modeling with numerical computation through an interactive MATLAB App Designer environment provides a robust and effective framework for accurately evaluating three-dimensional surfaces and supporting the structural design of engineering geomembranes. The findings clearly address the research objective: the combination of precise surface parameterization, rigorous vector formulations, and symbolic integration techniques yields numerical errors below 0.1% while maintaining stable convergence behavior and minimal computational cost.

In line with the central objective of the study, the developed tool enables the specification of geometric parameters, real-time three-dimensional visualization, and automatic computation of surface metrics through an interface that effectively integrates the theoretical model with its numerical implementation. This coherent linkage between mathematical formulation and computational execution represents both an educational and methodological advancement, facilitating the practical application of multivariable calculus concepts in structural design. The high level of agreement between analytical and simulated values demonstrates the robustness of the method and confirms that the algorithm maintains numerical stability across a range of geometric configurations and scales.

Regarding the first specific objective, the study reinforced the role of vector calculus in the geometric characterization of cylindrical surfaces used for surface-integral evaluation. The rigorous formulation of the differential elements and the consistent use of the norm of the cross product of the tangent vectors ensured an exact representation of the geometry, avoiding the common inaccuracies associated with triangulation or interpolation-based approaches. This methodological refinement provides a replicable framework that can be extended to more sophisticated parametric geometries—such as Bézier surfaces, B-splines, and NURBS—which are widely employed in advanced structural engineering and computer-aided geometric design.

In addressing the second specific objective, the development of the MATLAB App Designer application constitutes a meaningful contribution to STEM education and computational engineering. The interactive interface enables real-time manipulation of geometric parameters such as radius and height, providing immediate visual and numerical feedback on the resulting surface and its computed area. This real-time responsiveness supports the comprehension of multivariable relationships, enhances spatial reasoning, and reduces the cognitive load typically associated with abstract mathematical concepts. These results underscore the value of interactive computational tools in fostering analytical and technical skills within engineering education.

Regarding the third specific objective, testing the tool for calculating geomembrane surfaces in circular ponds demonstrated its practical utility. The tool enables precise surface area calculations, optimizing material usage, minimizing waste, and improving construction efficiency, thereby contributing positively to both the economic and environmental sustainability of aquaculture and water

storage systems. Compared to conventional CAD software, the proposed tool achieves comparable accuracy while requiring significantly fewer computational resources, offering a cost-effective and reliable solution for both educational and industrial applications.

Acknowledgement: The authors would like to express their gratitude to the Department of Exact Sciences at the University of the Armed Forces ESPE (Ecuador) for the academic and technical support provided during the development of this research. Special thanks are extended to the Computational Engineering Laboratory for granting access to MATLAB resources and computational infrastructure used for the experimental phase of the study.

Funding Statement: This research was carried out with institutional support from the University of the Armed Forces ESPE, Ecuador, within the framework of its academic program in Computational and Applied Engineering. The authors received no external funding for this study. All computational resources, software licenses https://la.mathworks.com/licensecenter/licenses/41181151/11593983/products, and laboratory facilities were provided by the University through its internal research infrastructure.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization: Juan Carlos Sarango Cuenca and Vicente García; Methodology: Juan Carlos Sarango Cuenca and Jacobo Vásquez; Software: Juan Carlos Sarango Cuenca; Validation: Vicente García and Jacobo Vásquez; Formal analysis: Jacobo Vásquez; Investigation: Juan Carlos Sarango Cuenca and Mayra Salazar; Resources: Mayra Salazar; Data curation: Jacobo Vásquez; Writing—original draft preparation: Juan Carlos Sarango Cuenca; Writing—review and editing: Vicente García and Mayra Salazar; Visualization: Juan Carlos Sarango Cuenca; Supervision: Mayra Salazar; Project administration: Mayra Salazar. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The authors confirm that the data supporting the findings of this study are available within the article and its supplementary materials. Additionally, the MATLAB App Designer code and computational datasets generated during the research are available from the corresponding author, Juan Carlos Sarango Cuenca (jcsarango@espe.edu.ec), upon reasonable request. Due to institutional policies at the University of the Armed Forces ESPE, full public release of the executable MATLAB application is restricted to academic use only; however, all analytical results, numerical data, and visual outputs are fully reproducible following the procedures described in this paper.

Ethics Approval: This research did not involve human participants, animal subjects, or any procedures requiring ethical approval. The study was based exclusively on computational modeling, numerical simulation, and mathematical analysis conducted within the University of the Armed Forces ESPE (Ecuador). All procedures complied with the institutional research and publication ethics policies established by the University's Research Directorate, in accordance with national academic standards and the principles of the Declaration of Helsinki (for ethical research integrity).

Conflicts of Interest: The authors declare no conflicts of interest.

Supplementary Materials: The supplementary material is available online at https://doi.org/10.23967/j.rimni.2026.10.75902. The supplementary materials for this article include: Supplementary Figure S1: Three-dimensional visualization of the cylindrical surface generated in MATLAB App Designer. Supplementary Figure S2: Example of the user interface developed for real-time parameter adjustment

and area computation. Supplementary Table S1: Comparison dataset of analytical and numerical results for different geometric configurations. Supplementary File S1: MATLAB App Designer source code used for parametric surface modeling and integration.

References

1. Gimenez Palomares F, Monsoriu Serra JA, Abraham Ibrahim S. Aprender métodos matemáticos programando con Matlab. In: Proceedings of the II Congreso Nacional de Innovación Educativa y Docencia en Red; 2016 Jul 7–8; Valencia, Spain. (In Spanish). doi:10.4995/inred2016.2016.4370.
2. Díaz JL. Integrating the anthropological theory of didactics in multivariate Calculus education: challenges, pedagogical shifts, and innovative activities. Int Electron J Math Educ. 2024;19(1):em0767. doi:10.29333/iejme/14142.
3. Larson RE, Edwards BH. Cálculo tomo II. 10th ed. Boston, MA, USA: Cengage Learning; 2016. (In Spanish).
4. Obradovic D, Mishra LN, Sharma N, Mishra VN. Matlab educational tools in mathematics teaching. Appl Math Inf Sci. 2021;15(3):241–252. doi:10.18576/amis/150301.
5. Cheong KH, Chen JS, Kang K, Yeo DJ. Supporting students' visualization of multivariable Calculus partial derivatives via virtual reality. Mathematics. 2023;11(4):831. doi:10.3390/math11040831.
6. Wong SF, Mahmud MM, Wong SS. Innovative approaches: advancing Calculus learning through hands-on modelling and Calcplot3D integration. In: Proceedings of the 2024 9th International Conference on Distance Education and Learning; 2024 Aug 8; Guangzhou China. p. 157–61. doi:10.1145/3675812.3675823.
7. Wei Z, Zhou X, Jiang J. A note on surface integrals of vector fields. OALib. 2021;8(10):1–11. doi:10.4236/oalib.1107934.
8. Yan L. Evaluation algorithms for parametric curves and surfaces. Mathematics. 2025;13(14):2248. doi:10.3390/math13142248.
9. Choi GPT, Giri A, Kumar L. Parametrización adaptativa que preserva el área de superficies anatómicas abiertas y cerradas. Comput Biol Med. 2022;148(8):105715. (In Spanish). doi:10.1016/j.compbiomed.2022.105715.
10. Sucipto L, Irpan S. Optimizing the use of Matlab GUI attributes in the creation of Calculus learning media: an effort to measure students' innovative attitudes. IJECA Int J Educ Curric Appl. 2022;5(1):79. doi:10.31764/ijeca.v5i1.7981.
11. Godoy AC, Rovigatti Chiavelli LU, Oxford JH, Rodrigues RB, de Oliveira Ferreira I, Marcondes AS, et al. Evaluation of limnological dynamics in Nile tilapia farming tank. Aquac Fish. 2021;6(5):485–94. doi:10.1016/j.aaf.2020.08.005.
12. Muñoz JFS, Bohórquez IG. Producción intensiva y automatizada de tilapia roja en estanques circulares. Rev Integr Investig Apl Desarro Tecnológico E Innovación. 2016;4:62–83. (In Spanish).
13. Acevedo J, Villamizar C. Plan de negocio para producción de tilapia roja en estanques de geomembrana bajo parámetros de ambiente controlado en el municipio de la mesa de los santos. Univ Coop De Colomb Bucaramanga. 2020:1–68. (In Spanish).
14. Romero-Cano LA. Modular simulation as a teaching tool: integrating MATLAB-Simulink into Heat Transfer courses to promote active learning and conceptual understanding. Educ Chem Eng. 2025;53:171–7. doi:10.1016/j.ece.2025.07.004.
15. Yueh MH, Huang HH, Li T, Lin WW, Yau ST. Optimized surface parameterizations with applications to Chinese virtual broadcasting. Etna. 2020;53:383–405. doi:10.1553/etna_vol53s383.
16. Teke IT, Ertas AH. An experimental study on nodular iron machined surfaces utilizing a capable 2D finite element model for precise surface roughness estimation. Processes. 2024;12(3):549. doi:10.3390/pr12030549.
17. Morejón A, Alonso JR. Surface roughness in machining of AISI 304 stainless steel. Rev De Ing Mecánica. 2024;24(2):19–25. (In Spanish).

18. Feng T. The impact of cloud technology and the MatLab app on the academic performance and cognitive load of further mathematics students. Educ Inf Technol. 2024;29(11):13577–93. doi:10.1007/s10639-023-12386-0.
19. Peñaherrera D, González E, Morales F, Coral R. Desarrollo de una aplicación didáctica para prácticas de comunicaciones ópticas mediante Matlab app designer. Rev Ibérica De Sist E Tecnol De Informação. 2023;E59:39–50.
20. Keviczky L, Bars R, Hetthéssy J, Bányász C. Introduction to MATLAB (advanced in textbooks in control and signal processing). Berlin/Heidelberg, Germany: Springer; 2019. doi:10.1007/978-981-10-8321-1_1.
21. Cao P, Zhang Y, Long X, Tan C, Chang J, Wang H. Multiphysics simulation of static water freezing process in pumped storage power stations using MATLAB. Case Stud Therm Eng. 2025;74(2):106921. doi:10.1016/j.csite.2025.106921.
22. Khedekar MD, Aher SJ, Mandale MB, Bobalade DD, Patil VM, Yadav SN. Enhancing the teaching-learning experience with the implementation of MATLAB tool: a case study. Educ Adm Theory Pract. 2024;30(5):10891–903. doi:10.53555/kuey.v30i5.4857.
23. Zhang Z, Li Y. Design and experiment of complex trajectory cam based on MATLAB. In: Appleby R, Zhang H, Liu L, editors. Proceedings of the 2nd International Conference on Intelligent Design and Innovative Technology (ICIDIT 2023). Dordrecht, The Netherlands: Atlantis Press; 2024. p. 427–36. doi:10.2991/978-94-6463-266-8_46.
24. Maldonado JJC, Macho LKG, Casallas EC. Investigación aplicada y desarrollo experimental en el fortalecimiento de las competencias de la sociedad del siglo XXI. Tecnura. 2023;27(75):140–59. (In Spanish). doi:10.14483/22487638.19171.
25. Korde A, Cengiz Ertekin R. Vector calculus-II. In: Engineering mathematics for marine applications. Cambridge, UK: Cambridge University Press; 2023. p. 19–35. doi:10.1017/9781108363235.004.
26. Mohammed AM, Ali Huneiti Z, Balachandran W, Al-Naafa MA. A study of the effects of using Matlab as a pedagogical tool for engineering mathematics students. Int J Onl Eng. 2013;9(2):27. doi:10.3991/ijoe.v9i2.2511.
27. Chyung SY, Guarino J, Scheepers M, DeLeon AR, Adams C, Williams P. The value of interactive simulations used in an undergraduate math class. In: Proceedings of the American Society for Engineering Education Annual Conference & Exposition; 2011 Jun 26–29; Vancouver, BC, Canada. p. 1–17.
28. Cai Q, Bajuri MR, Leong KE, Chen L. Multimodal learning interactions using MATLAB technology in a multinational statistical classroom. Multimodal Technol Interact. 2025;9(10):106. doi:10.3390/mti9100106.
29. Azzam NA, Al-Kayyali RAA. An integrated method for STEM education using MATLAB. In: Proceedings of the 2020 Advances in Science and Engineering Technology International Conferences (ASET); 2020 Feb 4–Apr 9; Dubai, United Arab Emirates. New York, NY, USA: IEEE; 2020. p. 1–4. doi:10.1109/ASET48392.2020.9118265.
30. Aftab S, Moghadam RH. Development of an integrated MATLAB-based GUI for geomechanical parameters determination. Results Eng. 2025;28(3):107085. doi:10.1016/j.rineng.2025.107085.
31. Grinspun E, Secord A. Introduction to discrete differential geometry: the geometry of planar curves. In: Proceedings of the SIGGRAPH Asia 2008: Cursos ACM SIGGRAPH ASIA 2008; 2008 Dec 10–13; Singapore. p. 1–4. doi:10.1145/1508044.1508053.
32. Ibhadode O, Fu YF, Qureshi A. FreeTO—freeform 3D topology optimization using a structured mesh with smooth boundaries in Matlab. Adv Eng Softw. 2024;198(1):103790. doi:10.1016/j.advengsoft.2024.103790.
33. Praetorius S, Stenger F. Dune-CurvedGrid—a dune module for surface parametrization. Arch Numer Softw. 2022;6(1):1–27. doi:10.11588/ans.2022.1.75917.
34. Ding J. Basic MATLAB operations. In: MATLAB scientific plotting and data analysis. Amsterdam, The Netherlands: Elsevier; 2025. p. 1–26. doi:10.1016/b978-0-443-36757-1.00001-2.
35. Khan IR, Ohba R. New finite difference formulas for numerical differentiation. J Comput Appl Math. 2000;126(1–2):269–76. doi:10.1016/S0377-0427(99)00358-1.

36. Wang B, Bai J, Lu S, Zuo W. An open source MATLAB solver for contact finite element analysis. Adv Eng Softw. 2025;199:103798. doi:10.1016/j.advengsoft.2024.103798.
37. Jones SR. Scalar and vector line integrals: a conceptual analysis and an initial investigation of student understanding. J Math Behav. 2020;59(2):100801. doi:10.1016/j.jmathb.2020.100801.
38. Gálvez A, Iglesias A. Matlab-based problem-solving environment for geometric processing of surfaces. In: Mathematical software—ICMS 2006. Berlin, Heidelberg: Springer; 2006. p. 35–46. doi:10.1007/11832225_4.
39. Houcque D. Introduction to Matlab for engineering students. Sch Eng Appl Sci. 2005;1:1–75. doi:10.1017/cbo9781316341599.003.
40. Kumar LA. Introduction to predictive analytics and MATLAB®. In: Predictive analytics using MATLAB® for biomedical applications. Amsterdam, The Netherlands: Elsevier; 2025. p. 1–30. doi:10.1016/b978-0-443-29888-2.00001-5.
41. Legland D. MatGeom: a toolbox for geometry processing with MATLAB. SoftwareX. 2025;29(22):101984. doi:10.1016/j.softx.2024.101984.
42. Bloor MIG, Wilson MJ, Sobey AJ. Geometric design of functional surfaces. Comput Aided Des. 1996;28(9):741–52. doi:10.1016/0010-4485(95)00080-1.
43. Heinze T, Frank S, Wohnlich S. FSAT-A fracture surface analysis toolbox in MATLAB to compare 2D and 3D surface measures. Comput Geotech. 2021;132:103997. doi:10.1016/j.compgeo.2020.103997.
44. Alsaidi RAM. New numerical solution for two parametric surfaces intersection dragging problem. Int J Anal Appl. 2021;19(5):773–83. doi:10.28924/2291-8639-19-2021-773.