mathematics

MDPI

Article

Bijective Mapping Analysis to Extend the Theory of Functional Connections to Non-Rectangular 2-Dimensional Domains

Daniele Mortari [1,*] **and David Arnas** [2]

[1] Aerospace Engineering, Texas A&M University, College Station, TX 77845-3141, USA
[2] Aeronautics and Astronautics, Massachussetts Institute of Technology, Cambridge, MA 02139, USA; arnas@mit.edu
* Correspondence: mortari@tamu.edu

Received: 27 July 2020; Accepted: 11 September 2020; Published: 16 September 2020

Abstract: This work presents an initial analysis of using bijective mappings to extend the Theory of Functional Connections to non-rectangular two-dimensional domains. Specifically, this manuscript proposes three different mappings techniques: (a) complex mapping, (b) the projection mapping, and (c) polynomial mapping. In that respect, an accurate least-squares approximated inverse mapping is also developed for those mappings with no closed-form inverse. Advantages and disadvantages of using these mappings are highlighted and a few examples are provided. Additionally, the paper shows how to replace boundary constraints expressed in terms of a piece-wise sequence of functions with a single function, which is compatible and required by the Theory of Functional Connections already developed for rectangular domains.

Keywords: Theory of Functional Connections; domain mappings; least-squares; functional interpolation

1. Introduction

The Theory of Functional Connections (TFC) is a mathematical methodology to perform functional interpolation, i.e., the process of deriving functionals, called *constrained expressions*, which contain the constraints of the problem already embedded on their expression. To give an example, consider the following univariate functional $y(x, g(x))$,

$$\begin{aligned} y(x,g(x)) = g(x) \quad &+ \quad \frac{2g(-3) - 2g(\pi) + (9-\pi^2)(1-g'(1))}{2\pi - \pi^2 + 15}\, x + \\ &+ \quad \frac{g(\pi) - g(-3) + (\pi+3)(1-g'(1))}{2\pi - \pi^2 + 15}\, x^2, \end{aligned} \tag{1}$$

where x is the independent variable, g is a function of x, and $(')$ indicates first derivative. This functional *always* simultaneously satisfies the following two constraints,

$$y(-3) = y(\pi) \qquad \text{and} \qquad y'(1) = 1,$$

no matter what the $g(x)$ *function is*. Function $g(x)$, called *free function*, can be any kind of function, including discontinuous or the Dirac functions, as long as *it is defined where the constraints are specified*. In this example: $g(-3)$, $g(\pi)$, and $g'(1)$ must be defined. A mathematical proof [1] has shown that constrained expressions, like the one given in Equation (1), can be used to represent the whole set of functions satisfying the set of constraints they are derived for.

The TFC has been mainly developed [1–4] to better solve constraint optimization problems, such as ODEs [5–8], PDEs [4,9], or programming [10,11], with effective applications in optimal control [12,13],

as well as in machine learning [4,14,15]. In fact, a constrained expression restricts the whole space of functions to the subspace fully satisfying the constraints. This way, a constraint optimization problem can be solved using simpler, faster, more robust and accurate methods already developed and optimized to solve unconstrained optimization problems.

In this work we focus on the application mapping tools to extend the Theory of Functional Connections to non-rectangular domains in two dimensional spaces. This is done via domain mapping. In particular, three distinct domain mappings are analyzed. These are:

- Complex mapping: which is based on using the properties of complex analytical functions to generate conformal mappings.
- Projection mapping: a new method proposed in this paper to obtain simple bijective mappings that preserve the continuity through the transformation.
- Polynomial mapping: based on the interpolation of the transformation using polynomial basis functions.

These three mappings are the subjects of the first three sections of this manuscript. These transformations are studied considering the direct and inverse mappings defined as:

1. **Direct mapping:** from rectangular domain Z (coordinates $[a, b]$ if real, or $z = a + i\,b$, if complex with $a, b \in [-1, +1]$) to generic domain W (coordinates $[x, y]$ if real, or $w = x + i\,y$, if complex).
2. **Inverse mapping:** from domain W to domain Z.

The reason this paper is specifically interested in the mapping from/to the Z rectangular domain relies on the fact that the TFC has been fully developed for rectangular domains [1–3]. Therefore, the main purpose of this study consists of developing easy mappings between the unit-square domain and any other domain, where the flexibly can accommodate typical domains appearing in physics, science, and engineering problems. Examples of this include computation of structures with "C", "I", and "T" shapes, the simplification of differential equations, or its application in fluidynamics. In other words, by obtaining these mappings, the TFC framework is consequently extended to generic domains.

In addition, this work extends the application of the TFC to discontinuous boundary constraints, which appear in the projection mapping and may also appear in some other applications. This has been done by replacing the discontinuous sequence of boundary contraint functions with a single function. Finally, a least-squares method to approximate inverse transformations is provided for those mapping transformations with no analytical inverse.

2. Complex (Conformal) Mapping

In mathematics, a conformal mapping is a transformation that, locally, preserves angles, but not necessarily lengths. In particular, the conformal property is mathematically described in terms of the Jacobian of a coordinate transformation. If the Jacobian at each point is a positive scalar times a rotation matrix, then the transformation is defined as conformal. Since any function of a complex variable, $w = f(z)$, provides a complex variable, then a complex function can be seen as a complex change of variables or, from a different point of view, as a mapping in a complex Euclidean plane [16]. In particular, any complex mapping, $w = f(z)$, is mathematically proven to be conformal, i.e., it is a transformation preserving (signed) angles, meaning, local orientations.

Conformal (angle-preserving) and area/volume-preserving mappings are particularly important transformations since they are very interesting for their applications in physics, science, and engineering. For instance, these mappings are used to transform differential equations in order to simplify the system and make it easier to solve [16].

A generic complex analytical mapping can be expressed as,

$$w = w_r + i\,w_i = f(z) = f(z_r + i\,z_i) = f_r(z_r, z_i) + i\,f_i(z_r, z_i), \tag{2}$$

where $z, w \in \mathbb{C}$, $z_r, z_i, w_r, w_i \in \mathbb{R}$ represent the original and transformed points, and $f : \mathbb{C} \to \mathbb{C}$ is the transforming function, where $f_r, f_i : \mathbb{R} \to \mathbb{R}$ are the real and imaginary parts of the transformation. Examples of some basic complex mappings are: (1) Translation, $w = z + z_t$, (2) Scaling, $w = \rho\, z$, (3) Rotation, $w = e^{i\,\phi}\, z$, (4) Affine, $w = \alpha\, z + \beta$, (5) Inversion, $w = 1/z$, (6) Exponential, $w = e^z$, (7), Squaring, $w = z^2$, (8) Cayley, $w = (z-1)/(z+1)$, and (9) Möbius (linear fractional), $w = (\alpha\, z + \beta)/(\gamma\, z + \delta)$, which is subject to $\alpha\, \delta \neq \beta\, \gamma$. In particular, the set of Möbius mapping includes, as a subset, translations, scalings, affines, inversions, and Cayley mappings.

Figure 1 shows the effect of inversion ($w = 1/z$), exponential ($w = e^z$), squaring ($w = z^2$), and Möbius mappings for $z = a\ + i\, b$ within the $a, b \in [-1, +1]$ domain. In particular, the Möbius mapping was selected with $\alpha = -3.8 + 4.4\, i$, $\beta = 3.4 - 0.2\, i$, $\gamma = 1$, and $\delta = -0.6 + 3.8\, i$. These values were obtained by selecting $\gamma = 1$ and imposing the mapping

$$\begin{aligned} z_1 &= -1 - i \quad &\to \quad w_1 &= 2 - 3\, i \\ z_2 &= +1 - i \quad &\to \quad w_2 &= 3 - i \\ z_3 &= +1 + i \quad &\to \quad w_3 &= i \end{aligned}$$

allowing the computation of α, β, and δ. The Möbius mapping can be seen as the stereographic projection from the plane to a rotated and translated unit-sphere. The Möbius mapping plays a particularly important role in the conformal complex mapping because: (1) it admits a closed-form expression for the inverse mapping and (2) is the composition or the most basic transformations of the space, including similarities, orthogonal transformations, and inversions. Also and for several dimensions equal or higher than 3, Liouville's theorem states that Möbius transformations are the unique transformations that are conformal.

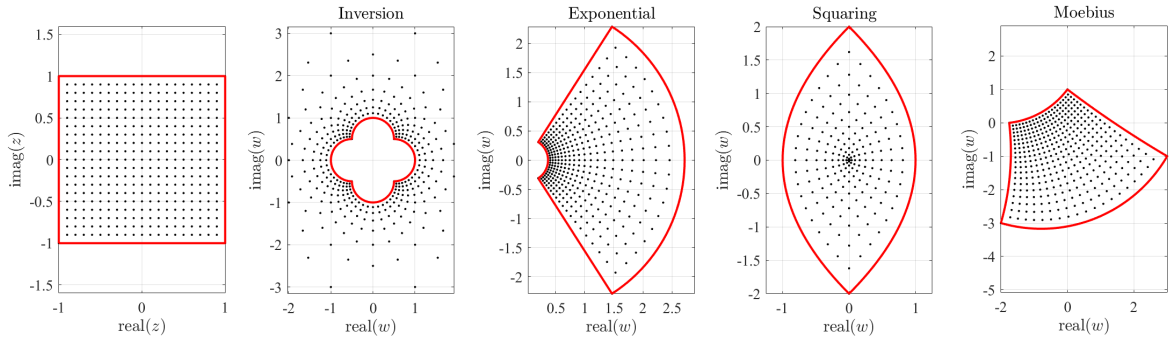

Figure 1. Examples of complex conformal mappings.

2.1. Control Points in Complex Conformal Mapping

Given a set of n complex control points, w_i with $i \in \{1, \ldots, n\}$, in the W domain associated with n corresponding boundary reference points, z_j with $j \in \{1, \ldots, n\}$, in the unit-square Z domain, the transformation,

$$w(z) = w_i\, \phi_i(z) \qquad \text{where} \qquad \phi_i(z_j) = \delta_{ij} \tag{3}$$

is a n-point complex conformal mapping. In this equation, $\phi_i(z)$ are a set of switching functions related to the reference points, z_j. In particular, Let select $n = 4$ control points (red markers on the top-left plot of Figure 2) defined by,

$$\text{reference points:} \begin{cases} z_1 = -1 - i \\ z_2 = +1 - i \\ z_3 = +1 + i \\ z_4 = -1 + i \end{cases} \qquad \text{control points:} \begin{cases} w_1 = 0 \\ w_2 = +5 - 2\, i \\ w_3 = +6 + 8\, i \\ w_4 = -2 + 4\, i \end{cases} . \tag{4}$$

The switching functions, $\phi_i(z)$, are expressed as a linear combination of n complex support functions. For the sake of simplicity, let the support functions be selected as complex monomials. The main reason for that is the simple closed-form expression of derivatives and integrals for this set of functions. Please note that since *any* set of n linearly independent functions can be selected as support functions, this provides additional degrees of freedom that are not analyzed in this study. Therefore, using complex monomial, the switching functions are expressed as,

$$\phi_i(z) = c_{ij}\, z^{j-1}$$

where c_{ij} are a set of constants to be found. Since the switching property states that $\phi_i(z_j) = \delta_{ij}$, we can derive the following expression,

$$\begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \\ c_{41} & c_{42} & c_{43} & c_{44} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ z_1 & z_2 & z_3 & z_4 \\ z_1^2 & z_2^2 & z_3^2 & z_4^2 \\ z_1^3 & z_2^3 & z_3^3 & z_4^3 \end{bmatrix}^{-1} = \frac{1}{16} \begin{bmatrix} 4 & -2+2i & -2i & 1+i \\ 4 & 2+2i & 2i & -1+i \\ 4 & 2-2i & -2i & -1-i \\ 4 & -2+2i & 2i & 1-i \end{bmatrix},$$

which provides the coefficients needed for the direct ($Z \to W$) mapping given in Equation (3).

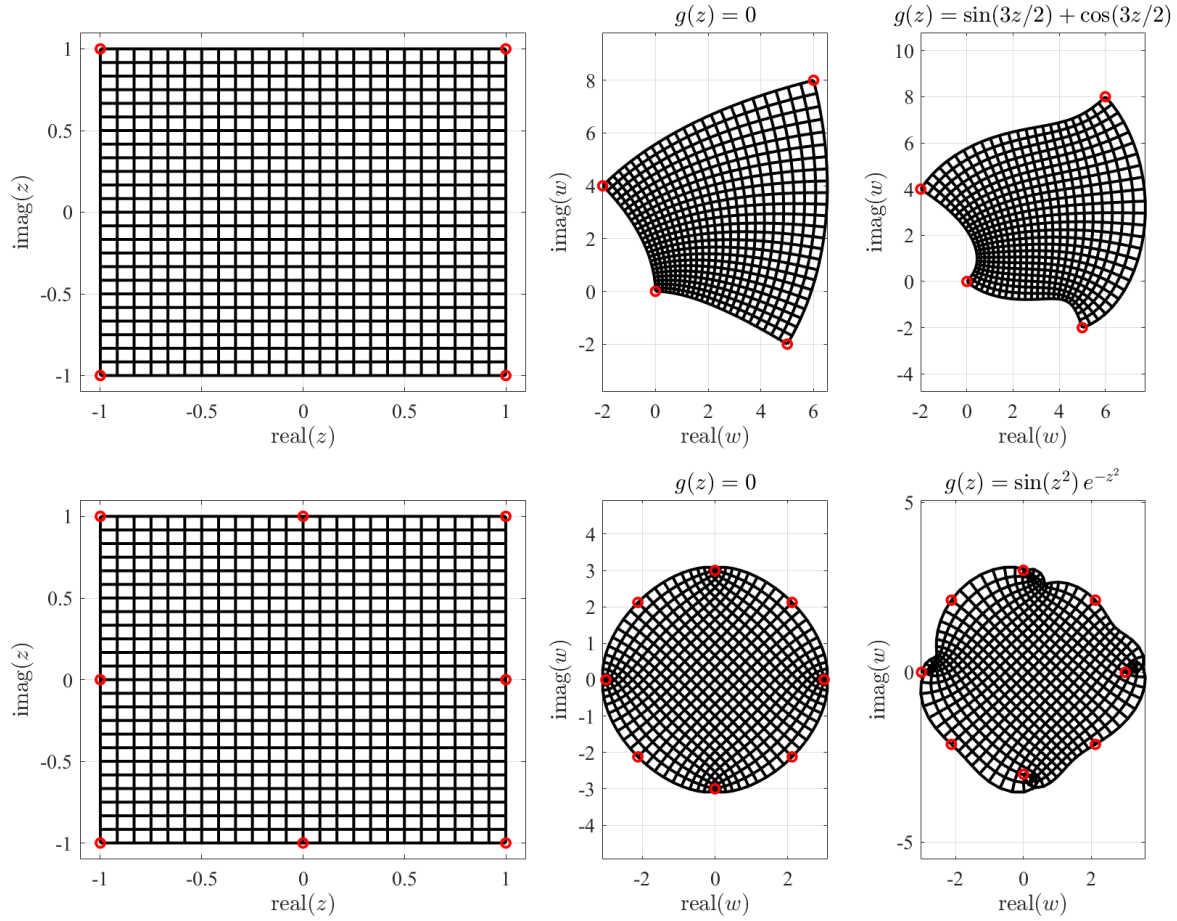

Figure 2. Rubber effect on the W domain by the TFC free function.

It is important to outline that in general complex mapping may not be bijective, which makes impossible to find an inverse transformation for the whole domain. However, due to the switching functions, the inverse mapping always exists for the control points. When the complex mapping is

bijective, then a least-squares approximated inverse mapping using a basis of complex orthogonal polynomials and presented in Section 5 can be used.

2.2. Complex TFC Mapping

We can generalize the previous result by using the Theory of Functional Connections. In particular, the functional,

$$w(z, g(z)) = g(z) + \sum_{i=1}^{n} [w_i - g(z_i)]\, \phi_i(z), \tag{5}$$

where $g(z)$ is a free complex function (where the $g(z_i)$ must be defined), can be seen as the most general expression that defines the conformal mapping between the control points in Z and W. In fact, Equation (5) represents the TFC functional generalization of Equation (3).

Figure 2 shows one effect of the free function in the complex mapping defined by the control points from Equation (4). The free function modifies the domain while the conformal property is still preserved. The top-left plots of Figure 2 shows a grid of orthogonal lines in the Z domain, while the top-center plot shows the lines mapped in the corresponding W domain using $g(z) = 0$, i.e., using Equation (3), and the control points given in Equation (4). The free function does not change the mapping of the control points. The points mapping is the constraint embedded in Equation (5). The top-right plot shows the mapped W domain obtained using the free function $g(z) = \sin(3z/2) + \cos(3z/2)$, i.e., applying Equation (5). This shows the free function "rubber" effect on the W domain. While the top figures use four control points (red dots), the bottom plots are obtained for eight control points distributed along a circle and using for the free function, the expression $g(z) = \sin(z^2)\, e^{-z^2}$. This example highlights that the selection of control points is not completely free because they often produce non-bijective mappings (points in the W domain may be associated with multiple points in the Z domain). Non-bijective mappings may occur when one or more control points fall inside the W domain.

Equation (5) highlights the fact that the free function can be used not only to derive functional satisfying constraints, but also to warp domains (rubber effect). The free function, however, is not anymore completely free if the important bijective property (see the bottom-right of Figure 2) must be preserved.

3. Projection Mapping

In this section, a transformation method based on a projection that preserves the continuity of the space and provides both the direct and the inverse mappings is presented. However, and due to its nature, the conformal property is not maintained by this mapping approach.

Projection mapping is based on the selection of a set of points (the projection points) inside the boundaries of the domain in such a way that a straight segment can be defined between any point in the domain and at least one projection point, where this segment does not cross at any moment any of the boundaries of the domain. This means that the distances from these projection points to the points at the boundary can be expressed as a continuous function, or in other words, if $\delta\Omega$ represents the boundary of the domain, the projection is well defined if and only if $\delta\Omega = \delta\Omega(\theta)$ is a bijective function where θ is the polar angle coordinate of a given point in $\delta\Omega$ with respect the projection point.

The most interesting feature of the projection mapping is that, using an identical formalism, the direct and the inverse mapping transformations can be derived (isomorphism). If the projection points are selected properly, the continuity of the transformation can be assured; however, in general, the derivative of this transformation is not defined in the whole domain and thus, the transformation is not conformal.

To show the methodology in a clearer manner, we first present the case of a single projection point. Then, we use this result to generalize the methodology for the case of multiple projection points.

3.1. Single-Point Projection Mapping

The idea of this methodology is to perform a linear transformation in polar coordinates with respect to a projection point that lies inside the domain. While the projection mapping can be applied to any dimensional domains, to be consistent with the complex conformal mapping, the analysis is here restricted to bivariate domains.

Let $\{r, \theta\}$ be the polar coordinates with respect to the projection point defined in the space Z of a given point inside the domain, and let $\{d, \phi\}$ be the polar coordinates of that point with respect to the transformed projection point in the space W. As said before, the projection points are selected in such a way that it is possible to connect univoquely and with a straight line any point in the boundary with the selected projection point. Therefore, for any point inside the square boundary of Z, the distance from the projection point to the boundary can be defined as $r_b(\theta) = f(\theta)$ where f is a continuous but not differentiable function. In the same way, if the distances from the projection point to the boundaries in W can be expressed as $d_b(\phi) = g(\phi)$, then, the following direct transformation can be defined:

$$\begin{aligned} \phi &= \theta; \\ d &= d_b(\theta)\frac{r}{r_b(\theta)}. \end{aligned} \tag{6}$$

This represents a transformation that maintains both the polar angle to the point as well as the ratio between the distances from the projection points to the point to transform and to the point in the boundary located at a polar angle θ from the projection point, i.e.,: $d/d_b(\theta) = r/r_b(\theta)$. In the same way, the inverse transformation can be defined by inverting the previous relation, or by applying the same formalism for the inverse transformation:

$$\begin{aligned} \theta &= \phi; \\ r &= r_b(\phi)\frac{d}{d_b(\phi)}. \end{aligned} \tag{7}$$

Equations (6) and (7) allow performing a the projection transformation using a simple transformation. Nevertheless, in some applications it is useful to define the boundary in W as a given polygon defined by a set of control points. Therefore, we include in this section the algorithm to generate a transformation mapping between a set of n control points. In this case, and since we are using the control points as the reference to perform the transformation, the angles in both mappings are in general not equal as in the previous equations. In particular, a linear interpolation in the angle between the control points and the projection point is performed in order to maintain the continuity of the transformation.

Let a and b be the coordinates in Z of a given point inside the original domain that is required to be transformed, and x and y be the coordinates in W of the transformed point. Moreover, $\boldsymbol{c} = \{c_a, c_b\}$, $\boldsymbol{v} = \{v_x, v_y\}$ are the coordinates of the projections points in Z and W respectively, while $\{p_a(j), p_b(j)\}$, and $\{q_x(j), q_y(j)\}$, with $j \in \{1, \ldots, n+1\}$, are the coordinates of the n control points defined in Z and W respectively plus the first one repeated in order to close the polygon. Please note that the control points have to be consecutive, i.e., it must be possible to generate the polygon by linking the control points in the order provided.

The transformation process follows these steps:

1 The polar coordinates of the point with respect to the projection point in Z are computed.
2 We determine the region from the domain in Z in which the point is located. In that regard, each region is defined by the triangle formed between the projection point and two consecutive control points in Z. This is done using the polar angle boundaries of the region.

3 We compute the distance from the projection point in Z to the intersection between the straight line passing through the projection point and the point to transform, and the boundary of the domain in Z. This provides the value r_b.

4 We determine the transformed region in W. This is done knowing that the control points $\{p_a(j), p_b(j)\}$, are the transformed versions of $\{q_x(j), q_y(j)\}$ in their respective spaces.

5 A linear interpolation in the polar angle is performed between the original point and the angle boundaries of the regions in which the point is located. This step provides the polar angle ϕ.

6 We compute the distance between the projection point in W and the intersection between the straight line with angle ϕ passing through the projection point, and the boundary of the domain in W. This provides the value d_b.

7 The distance of the transformed point to its projection point d is obtained by applying the relation seen in Equation (6).

A more in detail description of the operations required to transform $\{a, b\}$ into $\{x, y\}$ is presented in Algorithm 1. On the other hand, the inverse transformation is equivalent to the process presented so we do not repeat it in here. Finally, it is important to note that since the transformations used in the projection mapping are in fact linear transformations, the transformation is always bijective if the condition about the definition of the projection point with respect to the boundary is met.

As an example of this method, Figure 3 (right) presents the result of applying the transformation (direct and inverse) to the control points provided by Table 1 and the projecting points $\boldsymbol{c} = \{0, 0\}$ and $\boldsymbol{v} = \{0, 0\}$. As it can be seen, the polynomial in W resembles a "T" shape, where each consecutive pair of control points generates a region where the transformation is continuous and differentiable. However, at the boundaries between regions the transformation stops to be differentiable, although it maintains the property of continuity.

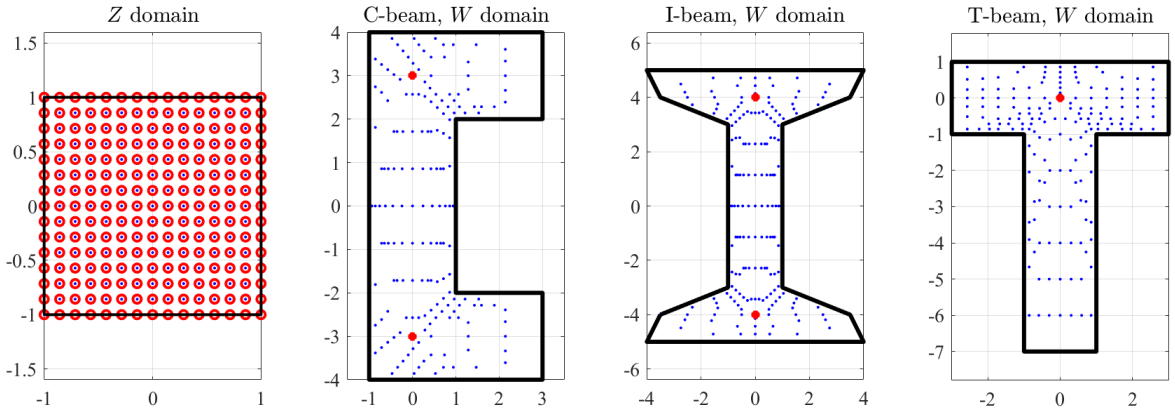

Figure 3. Examples of projection mapping for "C", "I", and "T" beam shape domains.

Table 1. Control points for the "T" beam shape transformation.

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
p_a	−1	−0.5	0	0.5	1	1	1	1	1	0.5	0	−0.5	−1	−1	−1	−1	−1
p_b	−1	−1	−1	−1	−1	−0.5	0	0.5	1	1	1	1	1	0.5	0	−0.5	−1
q_x	−2	−1	−1	−1	0	1	1	1	2	3	3	3	0	−3	−3	−3	−2
q_y	−1	−1	−4	−7	−7	−7	−4	−1	−1	−1	0	1	1	1	0	−1	−1

3.2. Multiple-Point Projection Mapping

There are cases in which it is not possible to generate a function $d_b(\phi) = g(\phi)$ as in the previous case due to the shape of the domain in W. This happens for instance when the domain boundary cannot be represented as a function in polar coordinates from the projection point. Nevertheless, in these situations it is possible to solve this problem using a set of projection points instead of just

one; however, some considerations regarding the continuity of the transformation must to be taken into account first. In that regard, and as a first approach to deal with this situation, we could think on finding the smallest number of projection points required to cover the whole domain. This represents in effect the so-called "art gallery problem". The "art gallery problem" (known also as the museum problem) is a problem in computational geometry providing the solution for the minimum number of required points to cover the complete domain. Its name comes from the real-world problem of placing the minimum number of guards in an art gallery such that, all together, can observe the whole gallery. This problem was first solved by Chvátal in 1975 [17], but Fisk, in 1978, provided such a brilliant and short proof [18] that Paul Erdős decided to include it in "The BOOK", an imaginary book in which God keeps the most elegant proof of each mathematical theorem. This book was then published after Erdős dead [19]. The solution to this problem states that $\lfloor n/3 \rfloor$ guards are always sufficient and (sometimes) necessary to guard a simple polygon with n vertices. In general these points are in the boundary of the polygon and thus, although the theorem is interesting, it is not really practical for most applications since it distributes points more densely in the boundaries of the domain in W rather than obtaining a more uniform distribution of points.

Therefore, instead of focusing on identifying the smallest number of projection points required, in this work we deal with the problem on how to separate both domains (in Z and W) in a set of regions, one for each projecting point, in such a way that the property of continuity is preserved through the transformation. Once these regions are defined, we proceed as in the case of single-point projection mapping. Please note that by doing that, each time that a point is transformed, the method requires to determine the region, from the ones defined, to which it belongs. This can be done, for instance, by using the transformation for each of these regions checking if the point is inside the region in the Z space.

Let $\delta\Omega$ be the boundary in W of a given domain. Then, connecting with a line two points in $\delta\Omega$ implies connecting two points in the boundary of Z since during the transformation the boundary in one space transforms into the boundary on the other space. For instance, from Figure 4, if we select the points A and B from W, we know that both points belong to the boundary in Z and also, both points can be connected through a curve in Z and W. Then, if we use just straight lines as curves, this means that A and B in Z must be in a different side of the square boundary in order to define a region of area different to zero. The same can be said with the points C and D. Following this procedure, it is possible to generate four regions in which each projection point can cover completely its own region. At this point, it is important to note that the boundary between regions 1 and 4 is the curve between A and B in both spaces. In the same way, regions 3 and 4 have a boundary in the curve between A and C, while in regions 2 and 4 the boundary in common is defined by the curve between B and D. This means that since the transformation is continuous inside the regions and at the boundaries of these regions, then, the whole transformation is continuous in the whole domain. This process can be continued even using the boundaries between these defined regions. This means that, for instance, if region 4 is split in half vertically, a point in the curve between A and B will be part of the new boundary created between regions 4′ and 4″, while another point in the boundary should be in $\delta\Omega$ between C and D.

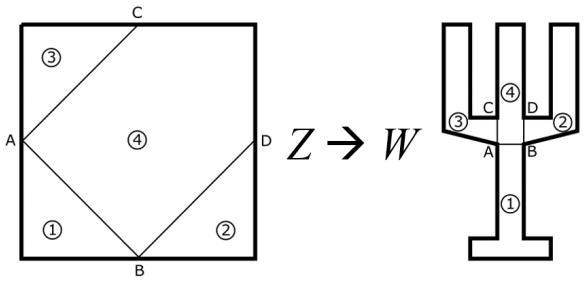

Figure 4. Definition on region for multi-point projection mapping.

To present an example of application for the case of multiple projection points, we apply this methodology to the "C" and "I" beam transformations shown in Figure 3 . To that end, we know that for these cases we can cover all the domain with two projection points. Moreover, and since both figures are symmetric with respect $b = 0$ and $y = 0$ respectively, we use this line as the boundary between the two regions in which to apply Algorithm 1. In particular, for the case of the "C" beam shape, we define the control points for the two defined regions in Table 2, where the projecting points used are $\boldsymbol{c}_1 = \{0, 0.5\}$, $\boldsymbol{c}_1 = \{0, -0.5\}$, $\boldsymbol{v}_1 = \{0, 3i\}$, and $\boldsymbol{v}_2 = \{0, -3i\}$. Figure 3 shows the result of the transformation.

Algorithm 1: Direct transformation by projection mapping using a set of n control points.

Inputs: control points $\{p_a(j), p_b(j)\}$, and $\{q_x(j), q_y(j)\}$; projection points $\boldsymbol{c} = \{c_a, c_b\}$, and $\boldsymbol{v} = \{v_x, v_y\}$; and point in Z to transform $\{a, b\}$.
Output: transformed point $\{x, y\}$ in W.
Angle and distance with respect to the projecting point in Z:
$\theta = \text{atan2}(b - c_b, a\ - c_a)$; $r^2 = (a - c_a)^2 + (b - c_b)^2$;
for $j = 1$ *to* $j = n$ **do**
  Angle boundaries of the region in Z:
  $\theta_{min} = \text{atan2}(p_b(j) - c_b, p_a(j) - c_a)$; $\theta_{max} = \text{atan2}(p_b(j+1) - c_b, p_a(j+1) - c_a)$;
  $\Delta\theta = \theta_{max} - \theta_{min} \mod (2\pi)$;
  if $((\theta \geq \theta_{min})$ *and* $(\theta \leq \theta_{max}))$ *or* $((\theta \geq \theta_{min} \mod (2\pi))$ *and* $(\theta \leq \theta_{max} \mod (2\pi)))$ **then**
    Line passing through the projecting point in Z:
    $m_p = (b - c_b)/(a - c_a)$; $n_p = c_b - m_p c_a$;
    Line defining the boundary in Z:
    $m_b = (p_b(j+1) - p_b(j))/(p_a(j+1) - p_a(j))$; $n_b = p_b(j) - m_b * p_a(j)$;
    Distance to the intersecting point in Z:
    $r_b^2 = ((n_p - n_b)/(m_b - m_p) - c_a)^2 + (m_b(n_p - n_b)/(m_b - m_p) + n_b - c_b)^2$;
    Angle boundaries of the region in W:
    $\phi_{min} = \text{atan2}(q_y(j) - v_y, q_x(j) - v_x)$; $\phi_{max} = \text{atan2}(q_y(j+1) - v_y, q_x(j+1) - v_x)$;
    $\Delta\phi = \phi_{max} - \phi_{min} \mod (2\pi)$;
    Transformed angle:
    $\phi = \theta - \theta_{min} \mod (2\pi)$; $\phi = \phi\Delta\phi/\Delta\theta + \phi_{min}$;
    Line passing through the projecting point in W:
    $m_p = \tan(\phi)$; $n_p = v_y - v_x m_p$;
    Line defining the boundary in W:
    $m_b = (q_y(j+1) - q_y(j))/(q_x(j+1) - q_x(j))$; $n_b = q_y(j) - m_b q_x(j)$;
    Distance to the intersecting point in W:
    $d_b^2 = ((n_p - n_b)/(m_b - m_p) - v_x)^2 + (m_b(n_p - n_b)/(m_b - m_p) + n_b - v_y)^2$;
    break;
  end
end
Transformed point:
$d = d_b r / r_b$;
$x = v_x + d\cos(\phi)$; $y = v_y + d\sin(\phi)$;

Table 2. Control points for the "C" beam shape transformation.

j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
p'_a	1	1	1	0.5	0	−0.5	−1	−1	−1	1
p'_b	0	0.5	1	1	1	1	1	0.5	0	0
p''_a	−1	−1	−1	−0.5	0	0.5	1	1	1	−1
p''_b	0	−0.5	−1	−1	−1	−1	−1	−0.5	0	0
q'_x	1	1	2	3	3	3	1	−1	−1	1
q'_y	0	2	2	2	3	4	4	4	0	0
q''_x	−1	−1	1	3	3	3	2	1	1	−1
q''_y	0	−4	−4	−4	−3	−2	−2	−2	0	0

3.3. Maintaining the Density of Points during the Transformation

In some applications it is of interest to maintain the density of points through the transformation between W and Z. This can be achieved easily using the projection mapping transformation proposed by defining appropriately the control points in the square domain in Z.

Let n be the number of uniformly distributed points in W that are inside the problem domain of area A. Let n_p be the number of points in a given region in W of area A_w that is defined by two consecutive control points and the projection point. Then, since the points are uniformly distributed: $n_p/n = A_w/A$. This argument can be also applied to the square domain in Z leading to: $n_p/n = A_z/A_s$ where A_z is the area defined by the transformed control points and the projection point in Z, and A_s is the area of the square. This means that, since n_p/n is a fixed value in the transformation, we have to impose that $A_z = A_s\, A_w/A$ in order to maintain the density of points. In other words, the proportion of the area of each region with respect to the area of the domain must be equal in both domains. Therefore, it is possible to select the two control points in the boundary of Z in such a way that this area is maintained. For instance, if the area of a given region in W is 10% of the domain area, and one of the control points in Z was already in the position $\{1, 0\}$, this means that the second control point for that region should be $\{1, 0.8\}$. It is important to note that in general, these new control points will not coincide with the corners of the square, and thus, additional control points in the square (and their transformed equivalents in W) must be generated in order to define completely the domain in Z.

3.4. Merging a Sequence of Boundary Functions into a Single Function

The strength of the projection mapping consists of providing the direct and inverse transformations with the same formalism and with no approximation. This section solves a problem arising from the application of the projection mapping when mapping the boundaries of the W-domain back to the Z-domain. This may create a sequence of functions with no C^1 continuity. This problem appears, for instance, when we perform a transformation between polygons with different number of sides. As a result of that, the continuity is lost for Newmann constraints. Because of this, there is the need for representing a sequence of functions by a single boundary function, as required by the TFC framework.

In this subsection using the Heaviside step function,

$$H(x) = \begin{cases} = 0 & \text{if} \quad x < 0 \\ = 1 & \text{if} \quad x > 0 \end{cases},$$

we show how to replace, with no approximation, a set of n contiguous piece-wise functions by a single function that can be used in the TFC framework. In particular, no C^0 continuity is also considered as it can be generated by imposing Newmann boundary constraints in the W-domain.

Consider, for example, the boundary constraints described by the following set of four piece-wise functions, each defined in contiguous ranges,

$$\begin{array}{lll} f_1(x) = -1 - 2\,x & \text{in} & x \in [-1, -1/2] \\ f_2(x) = 1 + 4(1+x)\,x & \text{in} & x \in [-1/2,\ 0] \\ f_3(x) = \sin(5\pi\,x) & \text{in} & x \in [0,\ +1/2] \\ f_4(x) = 4(1-x)\,x & \text{in} & x \in [+1/2, +1] \end{array} \tag{8}$$

This piece-wise function is shown in the top plot of Figure 5; while the equivalent single function, $f(x)$, which is derived by the general equation,

$$f(x) = f_1(x) + \sum_{k=2}^{n} H(x - x_k)\ [f_k(x) - f_{k-1}(x)] \tag{9}$$

applied for $n = 4$ piece-wise functions, is shown at the bottom plot of Figure 5. In particular, this example includes, C^0 continuity (black marker) at $x_2 = -1/2$, discontinuity (blue dashed line) at $x_3 = 0$, and C^0 and C^1 continuity (red marker) at $x_2 = 1/2$.

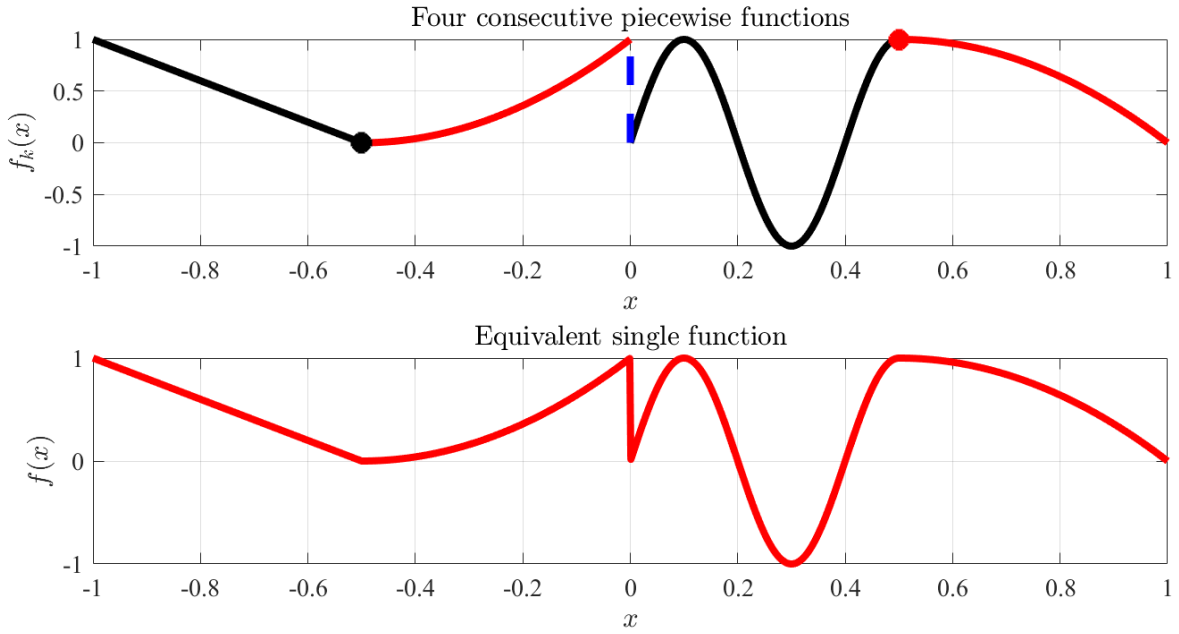

Figure 5. Four functions given in Equation (8) (**top**) and equivalent single function given by Equation (9) (**bottom**).

The function, $f(x)$, computed using Equation (9) can be used, for instance, to describe a boundary constraint at $y = -1$ in the 2-dimensional TFC matrix formulation [2,3],

$$f(x, y, g(x,y)) = g(x,y) + \boldsymbol{v}^{\mathrm{T}}(x)\ [\mathcal{M}(f(x)) - \mathcal{M}(g(x,y))]\ \boldsymbol{v}(y) \tag{10}$$

where the matrix tensor is simply,

$$\mathcal{M}(f(x)) = \begin{bmatrix} 0, & f(x) \end{bmatrix} \tag{11}$$

and the switching vectors are,

$$\boldsymbol{v}(x) = 1 \qquad \text{and} \qquad \boldsymbol{v}(y) = \frac{1}{2} \begin{Bmatrix} 2 \\ 1 - y \end{Bmatrix} \tag{12}$$

Therefore, the TFC functional is,

$$f(x,y,g(x,y)) = g(x,y) + \frac{1-y}{2}\,[f(x) - g(x,-1)] \tag{13}$$

The single function, $f(x)$, can be used in the TFC framework to represent a boundary constraint. Figure 6 shows two TFC surfaces using the single constraint boundary function, $f(x)$, shown in red and for two different expressions of the free functions: $g(x,y) = 0$ (left figure, most simple interpolating surface) and $g(x) = \sin(5\,x)\,e^{x\,\cos(5\,x)}$ (right figure).

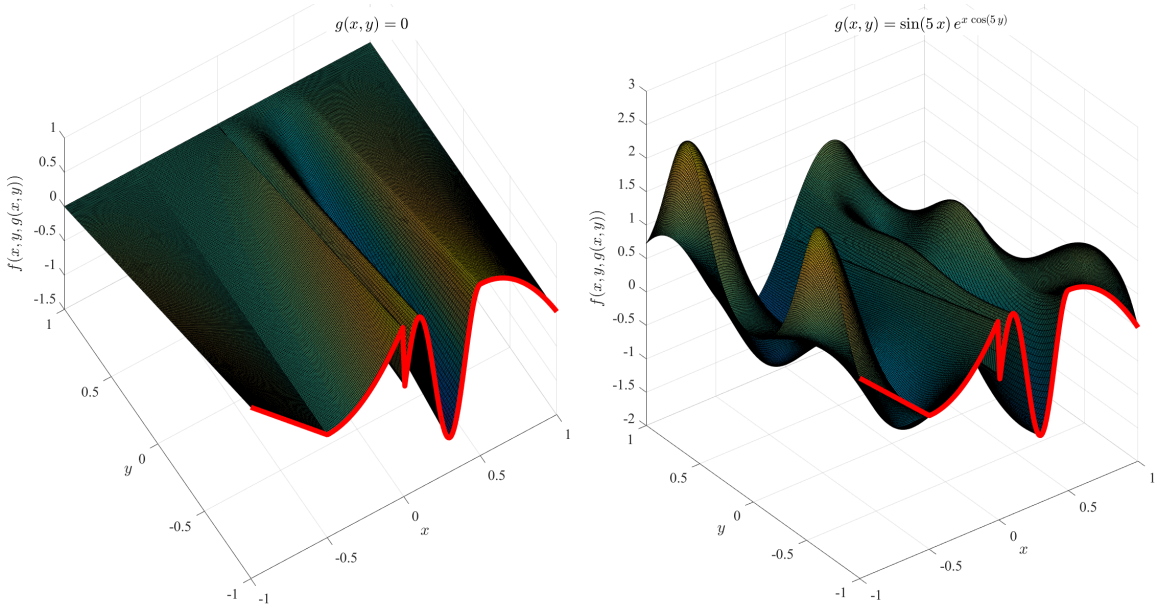

Figure 6. TFC surface examples obtained with $g(x) = 0$ (**left**) and $g(x) = \sin(5\,x)\,e^{x\,\cos(5\,x)}$ (**right**).

4. Polynomial Mapping

The polynomial mapping presented in this section is a well known mapping between two real domains. This mapping is performed using a set of n polynomial (switching) functions and a set of n boundary (control) points in the Z and W domains. While the proposed mapping can be applied to any dimensional domains, to be consistent with the complex conformal mapping, the analysis is here restricted to the bi-variate domains only.

Let the boundaries of two domains, Z and W, be identified by the following sequence of n control points,

$$\boldsymbol{z}_k = \begin{Bmatrix} a_k \\ b_k \end{Bmatrix} \qquad \text{and} \qquad \boldsymbol{w}_k = \begin{Bmatrix} x_k \\ y_k \end{Bmatrix} \qquad \text{where} \quad k \in \{1,\ldots,n\} \tag{14}$$

then, the direct mapping is provided by,

$$\begin{Bmatrix} x \\ y \end{Bmatrix} = \sum_{i=1}^{n} \begin{Bmatrix} x_i \\ y_i \end{Bmatrix} f_i(a,b), \tag{15}$$

where $f_i(a_j,b_j) = \delta_{ij}$ defines the switching property of the polynomials. This property gives the points association relationship, $[x_i,y_i] \leftrightarrow [a_i,b_i]$. The n coefficients of the switching functions are computed from the switching function property. For instance, consider the simplest example of $n = 3$ points (triangle) mapping. In this case, the mapping becomes linear, $f_i(a,b) = c_{i1} + c_{i2}\,a + c_{i3}\,b$.

The coefficients of the three switching functions ($i = 1, 2, 3$) are derived by imposing the switching functions property,

$$\begin{cases} 1 = c_{11} + c_{12}\, a_1 + c_{13}\, b_1 \\ 0 = c_{11} + c_{12}\, a_2 + c_{13}\, b_2 \\ 0 = c_{11} + c_{12}\, a_3 + c_{13}\, b_3 \end{cases} \quad \begin{cases} 0 = c_{21} + c_{22}\, a_1 + c_{23}\, b_1 \\ 1 = c_{21} + c_{22}\, a_2 + c_{23}\, b_2 \\ 0 = c_{21} + c_{22}\, a_3 + c_{23}\, b_3 \end{cases} \quad \text{and} \quad \begin{cases} 0 = c_{31} + c_{32}\, a_1 + c_{33}\, b_1 \\ 0 = c_{31} + c_{32}\, a_2 + c_{33}\, b_2 \\ 1 = c_{31} + c_{32}\, a_3 + c_{33}\, b_3 \end{cases} \tag{16}$$

or, in matrix form,

$$\begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \rightarrow \quad \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{bmatrix}^{-1} \tag{17}$$

In this example the mapping function is linear and, therefore, the inverse mapping function,

$$\begin{Bmatrix} a \\ b \end{Bmatrix} = \sum_{i=1}^{n} \begin{Bmatrix} a_i \\ b_i \end{Bmatrix} p_i(x, y) \tag{18}$$

where

$$p_i(x, y) = c_{i1} + c_{i2}\, x + c_{i3}\, y \quad \rightarrow \quad \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{bmatrix}^{-1} \tag{19}$$

is also linear. This is shown in Figure 7, where the direct mapping of a grid of points on the Z-domain (small black dots, left plot) are mapped to the W-domain (red cicles, right plot) and then mapped back to the Z-domain (red cicles, left plot).

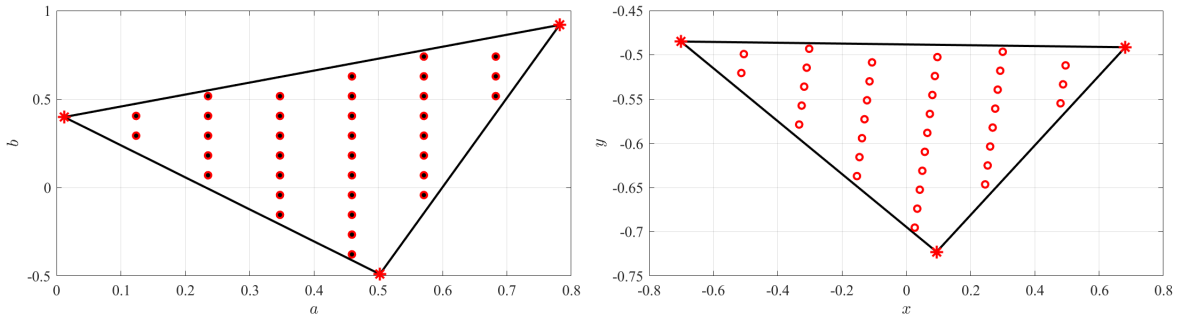

Figure 7. 3-points polynomial mapping example.

Please note that in the polynomial mapping the linearity holds in all the cases when the number of control points is equal to the dimensions of the mapping domain plus one. However, in the more interesting general case, when the number of control points is greater, then the mapping becomes nonlinear, with no closed-form inverse mapping expression. However, the mapping becomes more flexible to describe complex domains. This is shown in the next subsection where the domain is identified by four points.

4.1. 4-Points Polynomial Mapping

The polynomial mapping using $n = 4$ points on bi-variate domains (quadrangular polygons) is nonlinear. However, an inverse mapping is still possible to derive, as shown in Section 4.2. This means that this non-linear mapping is bijective.

Again, the mapping is performed between the unit-square domain, $(a, b) \in (-1, +1) \times (-1, +1)$, shown in top-left of Figure 8, and the quadrangular polygonal domain, defined by the four points, $x_k = [0,\ 5,\ 6,\ -2]$ and $y_k = [0,\ -2,\ 8,\ 4]$, shown in top-right of Figure 8. In particular, the mapping has been performed using a grid of 400 points.

The direct mapping, $(a, b) \rightarrow (x, y)$, is,

$$x = \sum_{k=1}^{4} x_k\, f_k(a, b) \qquad \text{and} \qquad y = \sum_{k=1}^{4} y_k\, f_k(a, b) \tag{20}$$

where the quadratic switching polynomial functions, f_k, (a, b), have the following expressions,

$$\begin{cases} f_1(a,b) = \dfrac{1}{4}(1 - a\ - b + a\ b) \\ f_2(a,b) = \dfrac{1}{4}(1 + a\ - b - a\ b) \\ f_3(a,b) = \dfrac{1}{4}(1 + a\ + b + a\ b) \\ f_4(a,b) = \dfrac{1}{4}(1 - a\ + b - a\ b) \end{cases} \tag{21}$$

Figure 8 shows the direct mapping to the quadrangular polygonal domain defined by the four points (center figure), $x_k = [0,\ 1,\ 0,\ -1]$ and $y_k = [-1,\ 0,\ 1,\ 0]$, and to a domain obtained by rotating clockwise by 30 deg. the Z control points (right figure). This has been done using a grid of 400 points.

Since this mapping transform lines to lines, the generic polygon side can be obtained simply by,

$$\boldsymbol{w}_{ij}(t) = \boldsymbol{w}_i + t\,(\boldsymbol{w}_j - \boldsymbol{w}_i) \qquad \text{where} \quad t \in [0, +1] \tag{22}$$

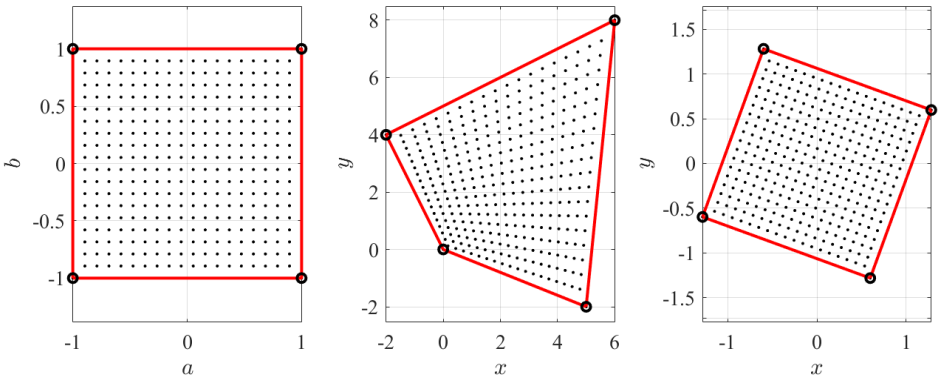

Figure 8. Polynomial mapping examples with four control points.

4.2. 4-Points Inverse Mapping

The inverse mapping, $(x, y) \rightarrow (a, b)$, can be obtained using the four boundary lines,

$$\begin{cases} \boldsymbol{w}_{12}(t_1) = \boldsymbol{w}_1 + t_1\,(\boldsymbol{w}_2 - \boldsymbol{w}_1) \\ \boldsymbol{w}_{43}(t_1) = \boldsymbol{w}_4 + t_1\,(\boldsymbol{w}_3 - \boldsymbol{w}_4) \end{cases} \qquad \text{and} \qquad \begin{cases} \boldsymbol{w}_{14}(t_2) = \boldsymbol{w}_1 + t_2\,(\boldsymbol{w}_4 - \boldsymbol{w}_1) \\ \boldsymbol{w}_{23}(t_2) = \boldsymbol{w}_2 + t_2\,(\boldsymbol{w}_3 - \boldsymbol{w}_2) \end{cases}$$

to obtain the two lines passing through the point (x, y),

$$\begin{Bmatrix} x \\ y \end{Bmatrix} = \boldsymbol{w}_{12}(t_1) + t_2[\boldsymbol{w}_{43}(t_1) - \boldsymbol{w}_{12}(t_1)] = \boldsymbol{w}_{14}(t_2) + t_1[\boldsymbol{w}_{23}(t_2) - \boldsymbol{w}_{14}(t_2)]$$

or in a scalar form,

$$t_1 \begin{Bmatrix} x_2 - x_1 \\ y_2 - y_1 \end{Bmatrix} + t_2 \begin{Bmatrix} x_4 - x_1 \\ y_4 - y_1 \end{Bmatrix} + t_1 t_2 \begin{Bmatrix} x_3 - x_4 + x_1 - x_2 \\ y_3 - y_4 + y_1 - y_2 \end{Bmatrix} = \begin{Bmatrix} x - x_1 \\ y - y_1 \end{Bmatrix}$$

which can be written as,

$$t_1 \begin{Bmatrix} C_{x1} \\ C_{y1} \end{Bmatrix} + t_2 \begin{Bmatrix} C_{x2} \\ C_{y2} \end{Bmatrix} + t_1 t_2 \begin{Bmatrix} C_{x3} \\ C_{y3} \end{Bmatrix} = \begin{Bmatrix} C_{x4} \\ C_{y4} \end{Bmatrix}$$

Setting,

$$\begin{cases} A = C_{x3}\, C_{y2} - C_{x2}\, C_{y3} \\ B = C_{x1}\, C_{y2} - C_{x2}\, C_{y1} + C_{x4}\, C_{y3} - C_{x3}\, C_{y4} \\ C = C_{x4}\, C_{y1} - C_{x1}\, C_{y4} \end{cases}$$

allows to write the solution in terms of a quadratic equation,

$$t_2 = \frac{-B + \sqrt{B^2 - 4AC}}{2\,A} \qquad \text{and} \qquad t_1 = \frac{x - x_1 - C_{y2}\, t_2}{C_{y1} + C_{y3}\, t_2}$$

and finally,

$$a \;= 2\, t_1 - 1 \qquad \text{and} \qquad b = 2\, t_2 - 1$$

Please note that if $C_{x3}\, C_{y2} = C_{x2}\, C_{y3}$, i.e., when $A = 0$, the problem becomes linear. For instance, if $C_{x3} = C_{y3} = 0$, then the solution is simply provided by,

$$\begin{bmatrix} C_{x1} & C_{x2} \\ C_{y1} & C_{y2} \end{bmatrix} \begin{Bmatrix} t_1 \\ t_2 \end{Bmatrix} = \begin{Bmatrix} C_{x4} \\ C_{y4} \end{Bmatrix}.$$

As it can be easily verified, this degeneration to a linear problem occurs when the W-domain is rectangular and rotated by any angle.

4.3. 8-Points Polynomial Mapping

Constrained expressions for cubic quadrangular domains can be obtained by mapping the domain $(a, b) \in (-1, 1) \times (-1, 1)$ unit-square domain to a cubic quadrangular domain (x, y), where each quadrangular side is identified by three points. This non-linear mapping is given by,

$$x = \sum_{k=1}^{8} x_k\, f_k(a, b) \qquad \text{and} \qquad y = \sum_{k=1}^{8} y_k\, f_k(a, b) \tag{23}$$

where the cubic polynomials functions f_k, (a, b) are the switching ($f_i(a_j, b_j) = \delta_{ij}$) functions,

$$\begin{cases} f_1(a,b) = \dfrac{1}{4}(a-1)(1-b)(a+b+1) \\ f_2(a,b) = \dfrac{1}{2}(1-a^2)(1-b) \\ f_3(a,b) = \dfrac{1}{4}(a+1)(1-b)(a-b-1) \\ f_4(a,b) = \dfrac{1}{2}(a+1)(1-b^2) \\ f_5(a,b) = \dfrac{1}{4}(a+1)(b+1)(a+b-1) \\ f_6(a,b) = \dfrac{1}{2}(1-a^2)(b+1) \\ f_7(a,b) = \dfrac{1}{4}(a-1)(b+1)(a-b+1) \\ f_8(a,b) = \dfrac{1}{2}(1-a)(1-b^2) \end{cases} \tag{24}$$

The quadrangular boundaries can be obtained by setting $a = \pm 1$ and $b = \pm 1$, respectively.

$$\begin{cases} \boldsymbol{c}_1(a) = \dfrac{1}{2}(a-1)a \begin{Bmatrix} -1 \\ -1 \end{Bmatrix} + (1-a^2) \begin{Bmatrix} 0 \\ -1 \end{Bmatrix} + \dfrac{1}{2}(a+1)a \begin{Bmatrix} 1 \\ -1 \end{Bmatrix} \\ \boldsymbol{c}_2(b) = \dfrac{1}{2}(b-1)b \begin{Bmatrix} 1 \\ -1 \end{Bmatrix} + (1-b^2) \begin{Bmatrix} 1 \\ 0 \end{Bmatrix} + \dfrac{1}{2}(b+1)b \begin{Bmatrix} 1 \\ 1 \end{Bmatrix} \\ \boldsymbol{c}_3(a) = \dfrac{1}{2}(a-1)a \begin{Bmatrix} -1 \\ 1 \end{Bmatrix} + (1-a^2) \begin{Bmatrix} 0 \\ 1 \end{Bmatrix} + \dfrac{1}{2}(a+1)a \begin{Bmatrix} 1 \\ 1 \end{Bmatrix} \\ \boldsymbol{c}_4(b) = \dfrac{1}{2}(b-1)b \begin{Bmatrix} -1 \\ -1 \end{Bmatrix} + (1-b^2) \begin{Bmatrix} -1 \\ 0 \end{Bmatrix} + \dfrac{1}{2}(b+1)b \begin{Bmatrix} -1 \\ 1 \end{Bmatrix} \end{cases} \tag{25}$$

Figure 9 shows the transformation of 900 points from the Z domain (top-left) to five different domains in W defined by the boundary control points,

$$\begin{aligned} \text{Top-center} \rightarrow & \begin{cases} \boldsymbol{x}_k = \{0, \quad 5, \quad 6, \quad -2, \quad 2, \quad 4, \quad 2, \quad 1\}^{\mathrm{T}} \\ \boldsymbol{y}_k = \{0, \quad -2, \quad 8, \quad 4, \quad -2, \quad 4, \quad 7, \quad 2\}^{\mathrm{T}} \end{cases} \\ \text{Top-right} \rightarrow & \begin{cases} \boldsymbol{x}_k = 3\cos((k-1)\,\pi/4) \\ \boldsymbol{y}_k = 3\sin((k-1)\,\pi/4) \end{cases} \quad \text{where} \quad k = 1, \cdots, 8 \\ \text{Bottom-left} \rightarrow & \begin{cases} \boldsymbol{x}_k = \{-3, \quad 0, \quad 3, \quad 3, \quad 3, \quad 0 \quad -3, \quad -3\}^{\mathrm{T}} \\ \boldsymbol{y}_k = \{-3, \quad -1, \quad -3, \quad -1, \quad 1, \quad 3, \quad 1, \quad -1\}^{\mathrm{T}} \end{cases} \\ \text{Bottom-center} \rightarrow & \text{ Bottom-left points rotated by } 5\pi/6 \text{ rad} \\ \text{Bottom-right} \rightarrow & \begin{cases} \boldsymbol{x}_k = \{-4, \quad -2, \quad 0, \quad 2, \quad 4, \quad 4\cos(\pi/4), \quad 0, \quad -4\cos(\pi/4)\}^{\mathrm{T}} \\ \boldsymbol{y}_k = \{0, \quad 0, \quad 0, \quad 0, \quad 0, \quad 4\cos(\pi/4), \quad 4, \quad 4\cos(\pi/4)\}^{\mathrm{T}} \end{cases} \end{aligned} \tag{26}$$

Specifically, the points in the top-right figure are selected as counter-clockwise points rotated by 45 deg. The (cubic) boundary is not analytically circular, but it differ from it by roughly 1%. Using more points a circular domain can be approximated with higher level of accuracy.

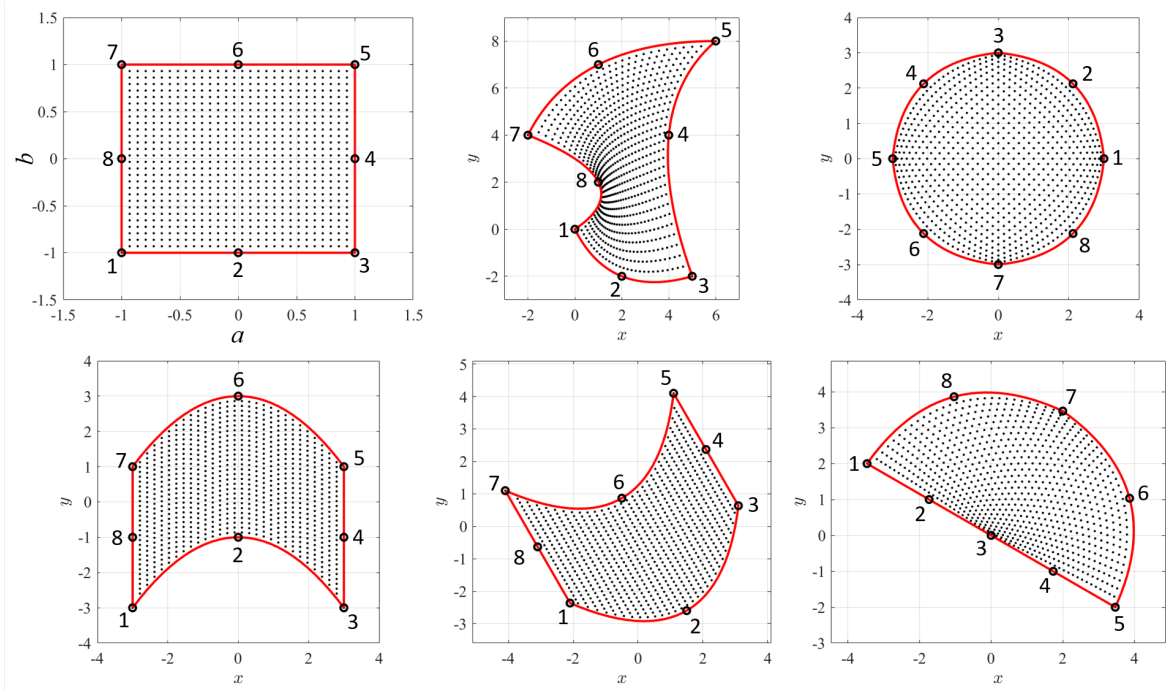

Figure 9. Polynomial mapping examples with 8 control points.

Unfortunately, the inverse of the polynomials mapping using 8 points has not been found. Because of this, the following section provides a least-squares procedure to estimate an approximated inverse mapping for the 8 (or more) points polynomial mapping as well as for complex conformal mapping with no inverse mapping.

5. Approximate Least-Squares Inverse Mapping

Inverse mappings always exist for the projection mapping, only. In fact and in general, complex and polynomial mappings do not admit closed-form inverse. However, when the direct mapping considered is bijective, then an approximated least-squares inverse mapping is proposed in this section. The inverse mapping is needed because the existing TFC methodology has been developed for any dimensional space and for a wide set of constraint types, but for rectangular domains, only. Therefore, in order to apply the TFC framework to generic domains, an inverse mapping function is required to map the problem domain into the Z rectangular domain where to apply the TFC framework.

To that end, a least-squares approximate inverse mapping is proposed as a linear combination of orthogonal polynomials,

$$\hat{z} = \xi_k \, \psi_k(w) \tag{27}$$

where $\psi_k(w)$ is the k-th orthogonal polynomial and where z and w are the complex coordinates that can be used, for real mapping, as $z = a \; + i\, b$ and $w = x + i\, y$.

Let $\boldsymbol{z}$ be the vector of N discretized points (e.g., grid or uniformly distributed) in the Z domain and $\boldsymbol{w} = f(\boldsymbol{z})$ be the direct mapping. Then, Equation (27) can be specified for all the points and a least-squares solution of the unknown coefficients vector, $\boldsymbol{\xi}$, can be estimated. It is important to outline that before applying this least-squares approach, translation and scaling might be necessary to contain all the $\boldsymbol{w}$ points in a scaled domain in which the polynomial basis functions are defined as, for instance, in the $[-1, +1]$ range for Legendre or Chebyshev orthogonal polynomials.

Figure 10 shows the accuracy results for the proposed approximate inverse mapping using the least-squares estimate. In this example, the same control points defined in Equation (4) have been used with $N = 121$ grid points in the $(-1, +1) \times (-1, +1)$ Z-domain. The left figure shows the z_i grid of

points as little black dots. The mapping points set, $[z_i, w_i]$, are then used to generate the approximate inverse mapping using Equation (27). The inverse mapping estimates, $\hat{z}_i$, were obtained using 20 Chebyshev orthogonal polynomial functions, $\psi_k(w)$. The $\hat{z}_i$ estimates are shown, as small red circles, in the left figure.

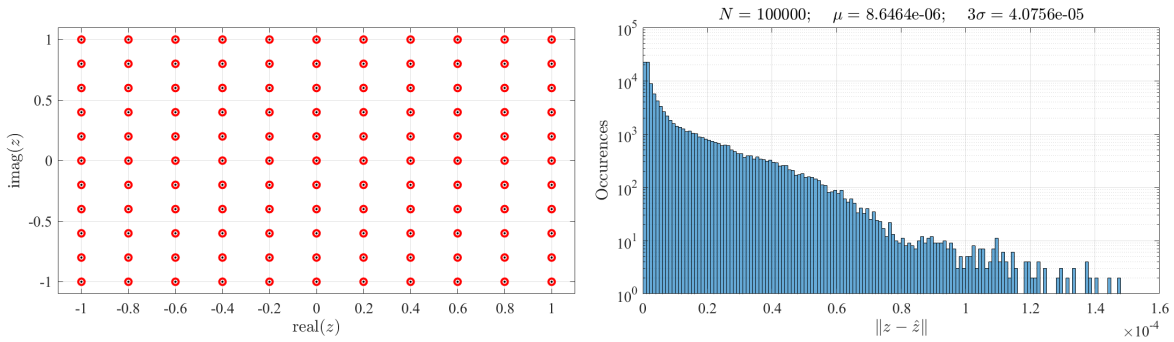

Figure 10. Approximate inverse function tests result.

To numerically validate the accuracy of this approximated inverse mapping, 100,000 Monte Carlo tests were made with (uniformly distributed) random points in the Z domain. The histogram of the error, $|z_i - \hat{z}_i|$ is shown in the right figure. The mean error, $\mu = 8.6464 \times 10^{-6}$, and $3\sigma = 4.0756 \times 10^{-5}$ error can be considered pretty good for most of the accuracy requirements in practical applications. Please note that the approximate inverse transformation must be computed just once.

Also, various distributions of points can be selected by embedding the W domain into a slightly larger rectangular domain. A grid distribution of points can be easily obtained for any convex W domains and, using a more complex algorithm, for non-convex domains. Note also that, in general, the bijection property of the transformation has to be checked for each particular mapping defined. This is due to the impossibility to derive a general proof for all possible transformations that can be generated using these approaches.

Figure 11 shows, for 57, 162, 488, and 1453 points, respectively, the selection of grid of points for the top-center domain shown in Figure 2. Using a simple algorithm it is possible to obtain collocation-type of points. This is show in the bottom plots of Figure 11 using the same number of points.

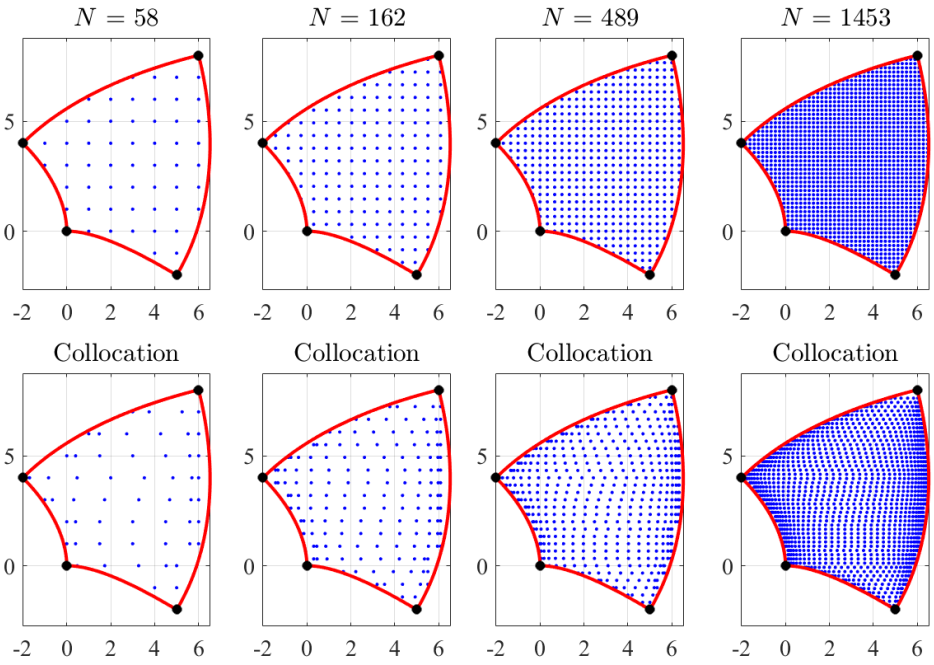

Figure 11. Points selection examples in the W domain: grid (**top**) and collocation (**bottom**).

6. Procedure to Apply the Theory of Functional Connections to Non Rectangular Domain

This section summarizes, step-by-step, how to derive a constrained expression in the W domain using the direct mapping function, $w = f(z)$. In this section, z and w indicate either, the complex or the ($z \equiv [a, b]$ and $w \equiv [x, y]$) real coordinates in the Z and W domains.

1. The W domain (or subdomain, as those shown in Figure 4) is identified by four boundaries, $c_i(w) = 0$, along which the Dirichlet constraints are defined by the four functions, $v_i(w)$.
2. A set of N_g grid (or collocation-type) points within the W domain, w_k, is computed (see Figure 11),
3. If the inverse function, $z = f^{-1}(w)$, is available (as, for instance, when using the projection mapping or an invertible complex mapping) then the w_k points are mapped back to the Z domain, $z_k = f^{-1}(w_k)$. In the case the inverse mapping function is not available, then a least-squares approximated inverse function, $z = \hat{f}^{-1}(w)$, can be derived as described in Section 5.
4. A set of points, w_b, belonging to the boundary, $c_i(w_b) = 0$, are computed. At these points/coordinates the Dirichlet constraint has value $v_i(w_b)$.
5. Using the inverse (or the approximated inverse) mapping function, $z_b = f^{-1}(w_b)$, the boundary coordinates in the Z domain are computed. At these corresponding points the constraint values are $v_i(w_b)$.
6. The boundary constraints in the Z domain are obtained by interpolating the points $[z_b, v_i(w_b)]$ *and* using a constrained expression to guarantee the continuity at the four corners, as explained in [1].
7. A Coons patch [20] is then used to obtain the simplest interpolating surface satisfying the Dirichlet boundary constraints in the Z domain. This means obtaining the surface, $S_c(z)$, providing the value at coordinate z.
8. A grid of points of the $S_c(z)$ surface is then mapped to the W domain, $w_c = f(z_c)$
9. Finally, the TFC functional in the W domain is then given by,

$$S(w, g(w)) = S_c\left(f^{-1}(w)\right) + g(w)\prod_{i=1}^{4} c_i(w) \tag{28}$$

Figure 12 presents an example using the W domain shown in the top-center of Figure 10 and defined by Equation (26). To guarantee continuity at the W-domain corners, the four boundary functions, $c_i(x, y) = 0$, were derived using the surface,

$$V(x, y) = \sin(\pi(x - y)/6) + \cos(\pi(x - y)/5) + 2$$

This means that the equations, $c_i(x, y) = 0$, describing the boundaries of the W-domain defined by Equation (26) were used to derive the boundary constraints using the $V(x, y)$ surface. For this specific case, $g(x, y) = 0$, was used as free function.

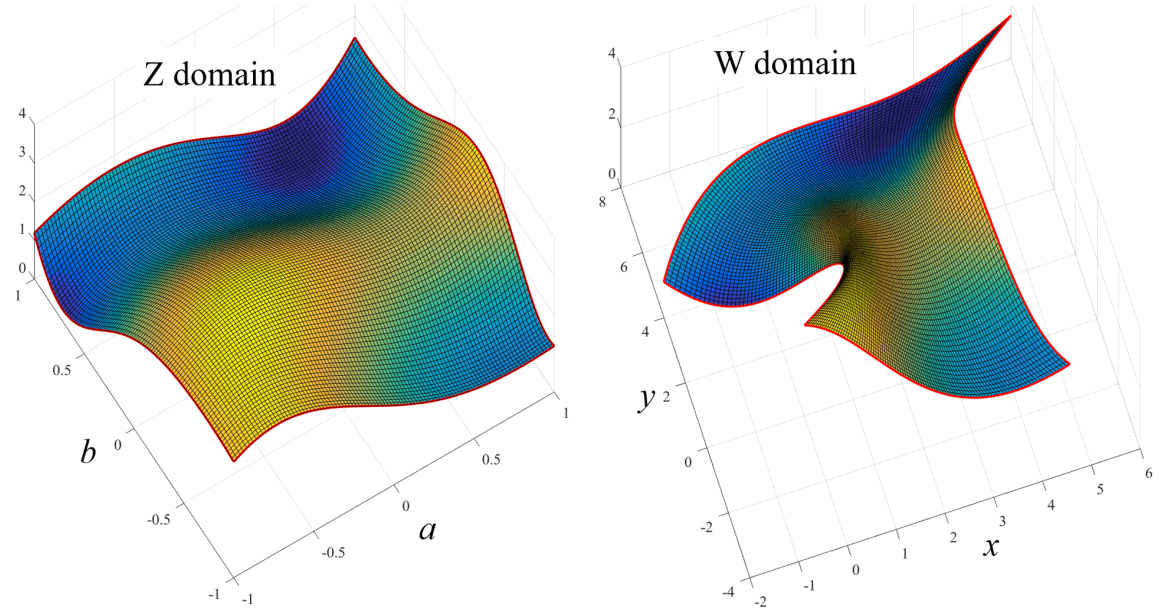

Figure 12. Example of the surfaces in the Z-domain (**left**) and in the W-domain (**right**).

7. Conclusions

This study consists of an initial investigation on how to apply the Theory of Functional Connections, which was developed for rectangular domains in any dimensional space, to generic domains in 2-dimensional spaces. This has been done by three distinct approaches: (1) complex (conformal) mapping, (2) projection mapping, and (3) polynomial mapping. Discussions and examples are provided to highlight the features of each one of these three mappings, such as, conformal property, invertible property, or flexibility to describe different domains.

For the cases of bijective mappings where no analytical inversion is known, such as for some complex mapping and for the polynomial mapping, a method to derive least-squares approximate inverse mapping is provided. In addition, this study also describes how to replace constraint boundaries defined by a sequence of different functions by a single equation. This is required to derive a Coons patch in the Z domain.

This manuscript represents the first generalization of the Theory of Functional Connections for non rectangular domains. As such, there is additional research to be performed to better clarify details and to extend the methodology even more and to study its potential applications. For instance, given a generic W domain (typically, polygonal or circular) of a real application, it would be interesting to derive a general methodology to obtain the mapping that approximates the W domain boundaries by least-squares or by some other optimization techniques. Another possible future research includes the study of optimization techniques to search for the free function, $g(w)$, which allows describing the set of four Dirichlet boundary constraints.

In other word, this incomplete study has the purpose to initiate the extension of the Theory of Functional Connections to non-rectangular domains.

Author Contributions: Conceptualization, Formal analysis, Methodology, Software, Writing: D.M. and D.A.; Supervision, D.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by NASA, grant #: 80NSSC19K1149.

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

Abbreviations

The following abbreviations are used in this manuscript:

ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
TFC	Theory of Functional Connections

References

1. Mortari, D. The Theory of Connections: Connecting Points. *Mathematics* **2017**, *5*, 57. [CrossRef]
2. Mortari, D.; Leake, C. The Multivariate Theory of Connections. *Mathematics* **2019**, *7*, 296. [CrossRef] [PubMed]
3. Leake, C.; Mortari, D. An Explanation and Implementation of Multivariate Theory of Functional Connections via Examples. In Proceedings of the AIAA/AAS Astrodynamics Specialist Conference, Portland, ME, USA, 11–15 August 2019.
4. Leake, C.; Mortari, D. Deep Theory of Functional Connections: A New Method for Estimating the Solutions of Partial Differential Equations. *Mach. Learn. Knowl. Extr.* **2020**, *2*, 37–55. [CrossRef] [PubMed]
5. Mortari, D. Least-Squares Solution of Linear Differential Equations. *Mathematics* **2017**, *5*, 48. [CrossRef]
6. Mortari, D.; Johnston, H.; Smith, L. High Accuracy Least-squares Solutions of Nonlinear Differential Equations. *J. Comput. Appl. Math.* **2019**, *352*, 293–307. [CrossRef] [PubMed]
7. Johnston, H.; Mortari, D. Least-squares Solutions of Boundary-value Problems in Hybrid Systems. *arXiv* **2019**, arXiv:1911.04390.
8. Johnston, H.; Leake, C.; Mortari, D. Least-squares Solutions of Eighth-order Boundary Value Problems using the Theory of Functional Connections. *Mathematics* **2020**, *8*, 397. [CrossRef] [PubMed]
9. Leake, C.; Johnston, H.; Mortari, D. The Multivariate Theory of Functional Connections: Theory, Proofs, and Application in Partial Differential Equations. *Mathematics* **2020**, *8*, 1303. [CrossRef]
10. Mai, T.; Mortari, D. Theory of Functional Connections Applied to Nonlinear Programming under Equality Constraints. Paper AAS 19-675. In Proceedings of the 2019 AAS/AIAA Astrodynamics Specialist Conference, Portland, ME, USA, 11–15 August 2019.
11. Johnston, H.; Leake, C.; Efendiev, Y.; Mortari, D. Selected Applications of the Theory of Connections: A Technique for Analytical Constraint Embedding. *Mathematics* **2019**, *7*, 537. [CrossRef] [PubMed]
12. Furfaro, R.; Mortari, D. Least-squares Solution of a Class of Optimal Space Guidance Problems via Theory of Connections. *Acta Astronaut.* **2020**, *168*, 92–103. [CrossRef]
13. Johnston, H.; Schiassi, E.; Furfaro, R.; Mortari, D. Fuel-Efficient Powered Descent Guidance on Large Planetary Bodies via Theory of Functional Connections. *arXiv* **2020**, arXiv:2001.03572.
14. Leake, C.; Johnston, H.; Smith, L.; Mortari, D. Analytically Embedding Differential Equation Constraints into Least-squares Support Vector Machines using the Theory of Functional Connections. *Mach. Learn. Knowl. Extr.* **2019**, *1*, 1058–1083. [CrossRef] [PubMed]
15. Schiassi, E.; Leake, C.; de Florio, M.; Johnston, H.; Furfaro, R.; Mortari, D. Extreme Theory of Functional Connections: A Physics-Informed Neural Network Method for Solving Parametric Differential Equations. *arXiv* **2005**, arXiv:2005.10632.
16. Kuliyev, S.A. Conformal Mapping Function of a Complex Domain and its Application. *Arch. Appl. Mech.* **2020**, *90*, 993–1003. [CrossRef]
17. Chvátal, V. A Combinatorial Theorem in Plane Geometry. *J. Comb. Theory* **1975**, *18*, 39–41. [CrossRef]
18. Fisk, S. A Short Proof of Chvátal's Watchman Theorem. *J. Comb. Theory* **1978**, *24*, 374. [CrossRef]
19. Aigner, M.; Ziegler, G.M. *Proofs from THE BOOK*; Springer: Berlin, Germany, 1998; ISBN 354-0-63-698-6.
20. Coons, S.A. *Surfaces for Computer Aided Design*; Technical Report; MIT: Cambridge, MA, USA, 1964.